NASA Contractor Report 198519
AMI Report No. 9408

Development of Three-Dimensional Flow Code Package to Predict Performance and Stability of Aircraft With Leading Edge Ice Contamination

D.J. Strash and J.M. Summa
Analytical Methods, Inc.
Redmond, Washington

September 1996

Prepared for
Lewis Research Center
Under Contract NAS3–26310

National Aeronautics and
Space Administration

Analytical Methods, Inc. Report No. 9408

DEVELOPMENT OF THREE-DIMENSIONAL FLOW CODE PACKAGE TO PREDICT PERFORMANCE AND STABILITY OF AIRCRAFT WITH LEADING EDGE ICE CONTAMINATION

FINAL REPORT, Option 2

CONTRACT NAS3-26310

Submitted to:

Cryogenics Branch
NASA Lewis Research Center
21000 Brookpark Road
Cleveland, OH 44135

By:

Analytical Methods, Inc.
2133 - 152nd Avenue N.E.
Redmond, Washington 98052

(206) 643-9090

ABSTRACT

In the work reported herein, a simplified, uncoupled, zonal procedure is utilized to assess the capability of numerically simulating icing effects on a Boeing 727-200 aircraft. The computational approach combines potential flow plus boundary layer simulations by VSAERO for the un-iced aircraft forces and moments with Navier-Stokes simulations by NPARC for the incremental forces and moments due to iced components. These are compared with wind tunnel force and moment data, supplied by the Boeing Company, examining longitudinal flight characteristics. Grid refinement improved the local flow features over previously reported work with no appreciable difference in the incremental ice effect. The computed lift curve slope with and without empennage ice matches the experimental value to within 1%, and the zero lift angle agrees to within 0.2 of a degree. The computed slope of the uniced and iced aircraft longitudinal stability curve is within about 2% of the test data. This work demonstrates the feasibility of a zonal method for the icing analysis of complete aircraft or isolated components within the linear angle of attack range. In fact, this zonal technique has allowed for the viscous analysis of a complete aircraft with ice which is currently not otherwise considered tractable.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Cont'd.)

1.0 INTRODUCTION

The flow fields associated with complete aircraft configurations are extremely complicated and remain a challenge for applied computational aerodynamics. Linear methods, i.e., pure potential flow solutions that may include nonlinear wake shape effects, have become commonly used techniques for aerodynamic predictions and analysis; however, research scientists are now developing and applying nonlinear methods, thereby incorporating more accurate mathematical models of the actual flow physics, such as compressibility and viscosity. Although much progress has been made in solving full potential, Euler, and various forms of the approximate Navier-Stokes equations (NS), a zonal method of analysis is generally projected for the entire aircraft flow-field calculation because of the wide variation of physical scales of characteristic fluid phenomena. This method is necessary not only to achieve a closed calculation on the next generation of computers, but also to obtain the required numerical accuracy over large regions that must include fine meshes to resolve the viscous layers as well as large flow gradients. Computational requirements for simulating the effects of ice contamination are even more severe since the method must predict aircraft performance *and* stability degradation caused by the local formation of complex ice shapes. Furthermore, the method should also be practical and fast enough to be utilized in preliminary design studies to ascertain component sensitivity to icing.

In this report, the work described involves a simplified, uncoupled zonal procedure which has been utilized to assess the capability of individual methods for simulating icing effects on a Boeing 727-200 aircraft. The zonal procedure includes the combined potential flow/boundary layer flow method (Program VSAERO[1]) for the un-iced aircraft simulation, while a Navier-Stokes method (Program NPARC[2]) is utilized for the iced components of the aircraft. Wind tunnel data[3] used in this study for the un-iced and iced configurations were provided by the Boeing Company. The calculations described here include comparison with experiment for the longitudinal forces and moments. The grid resolution utilized in this study represents a refinement over previously reported work. Further results including lateral force and moment calculations may be found in an earlier report.[4]

The general method is described in Section 2.0, the geometric considerations are presented in Section 3.0, the results are detailed in Section 4.0, and the conclusions and recommendations are described in Section 5.0. A description of the routines added to NPARC along with a printout is included in Appendix A, listing of the utility program, PRPAMI, developed under this contract is included in Appendix B, a sample NPARC input file may be found in Appendix C, and VSAERO and NPARC manual updates are in Appendix D. The contracted work to enhance the panel model of the DHC-6-200 Twin Otter for future tail-plane ice calculations is included in Appendix E.

2.0 METHODOLOGY

Unlike the previous fully-coupled zonal procedure which provided for an iterative step,[5,6,7] the zonal aspect of this method is simplified to include an initial outer boundary flow condition only, which is provided by the potential flow code and utilized by the viscous flow component. At the viscous domain outer boundary (Fig. 1), velocities and pressure are specified and density is calculated from the pressure and free-stream stagnation enthalpy. Since velocities, pressure and density are known, energy at the boundary can be determined. The viscous flow code uses these initial conditions in combination with the characteristic method to treat the outer surface boundary conditions. Further details pertaining to the implementation of flow boundary conditions may be found in the excellent discussion by Raj, et. al.[8].

The initial conditions generally involve effects from other aircraft features that are not included in the viscous domain. These may include, for example, lifting surface wake influence and fuselage blockage or upwash effects as well as the local surface boundary layer profile. In this manner, complications arising from an attempt to include other geometric features in the grid generation process are avoided. The final iced aircraft force and moment characteristics are estimated by superposition. The NPARC predicted incremental force and moment changes due to ice on individual aircraft components are combined with the VSAERO data for the un-iced aircraft. This process yields complete aircraft force and moment quantities under iced conditions. A zonal technique of this nature can allow for the viscous analysis of a complete aircraft with ice which is currently not otherwise considered tractable.

2.1 VSAERO Technical Description

VSAERO solves the Neumann problem of potential flow by converting Laplaces partial differential equation into a boundary integral equation, as described by Morino[9]. The body and wake (i.e., boundary of the fluid) are then broken into finite size elements, or panels, each with an unknown potential value. Whereas, the body shape is known, the wake shape—determined by streamline trajectories from the downstream separation lines on the body—is not. An informed guess for the wake shape normally generates a useful first solution upon which VSAERO can improve iteratively. The displacement of the inviscid fluid boundary from solid surfaces caused by viscosity (namely, the boundary layer) can also be included by an iterative process.

The panel method leads to a matrix equation of the order of the total number of surface panels. The matrix is dense and non-symmetric. The main features of the technique include generating the coefficients of the matrix, solving the matrix, and determining new wake shapes and viscous displacements.

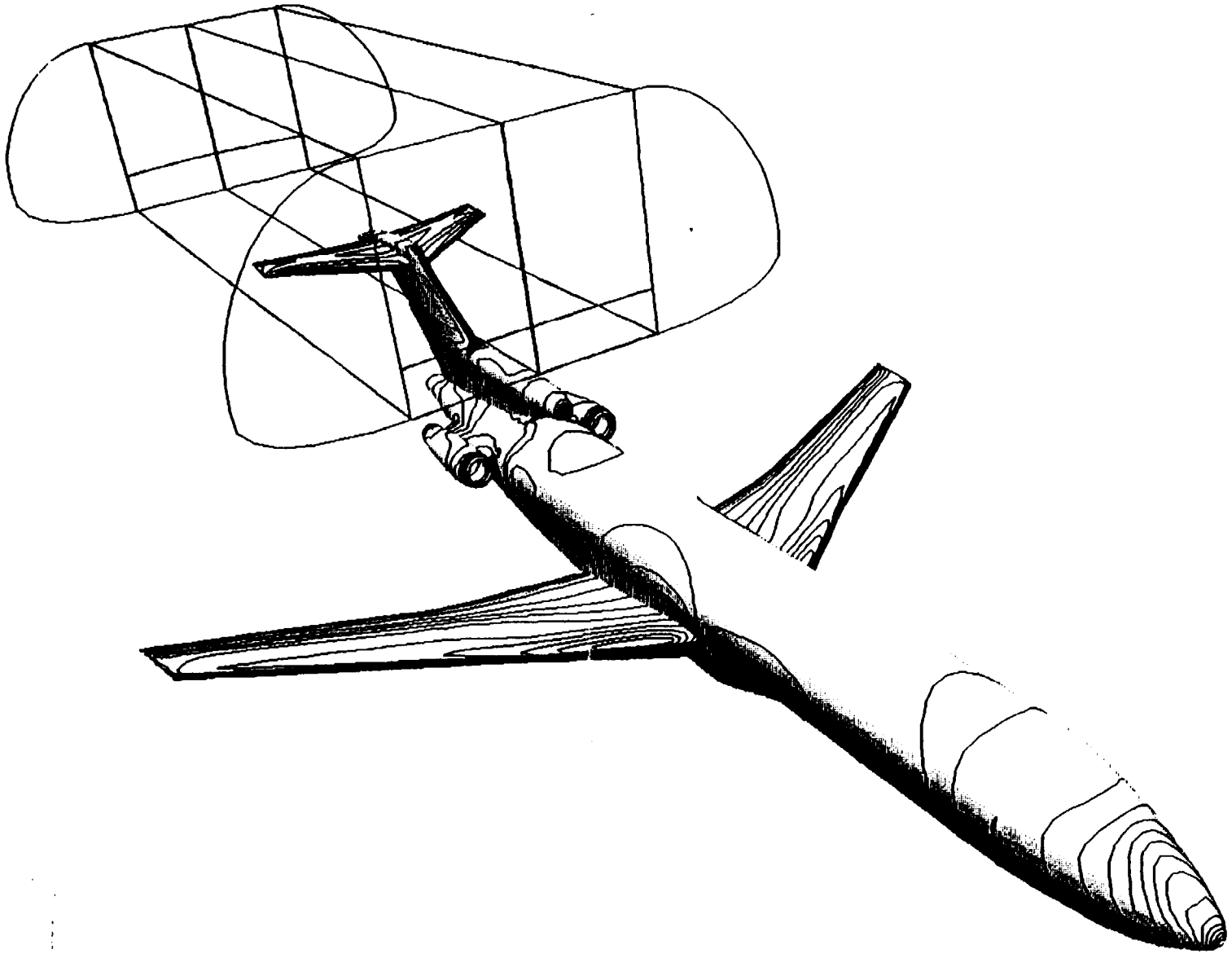


Fig. 1. Boeing 727-200 Wind Tunnel Model with Empennage Zonal Boundary Block Edges and VSAERO Computed Surface Pressure Contours at Zero Alpha.

2.2 NPARC Technical Description

NPARC Version 1.0 (formerly PARC3D), is a multi-block code developed by Sverdrup Technology, Inc.[2], and is based upon ARC3D[10] which was originally developed by Pulliam and Steger at NASA Ames Research Center. The method is based on the implicit approximate factorization algorithm of Beam and Warming[11] for unsteady compressible flow. For turbulent flow, the well-known Baldwin-Lomax model[12] is used for turbulence closure. This turbulence model also has been quite successful in the calculation of transonic flow, unless there is massive separation in the streamwise direction.

NPARC contains several improvements over the original ARC3D code, the most notable being the multi-block capability. Also, the user has the option of specifying flow boundary conditions by means of namelist input. Generally, the modifications are directed toward improvement in user productivity.

Several routines were modified or added to NPARC to accomplish the zonal component of this work. Generally the changes involved input/output of pertinent flow-field data and proper treatment of the far-field boundary condition. The outer domain zonal boundary condition implemented in this work was patterned after the freestream option available in NPARC (subroutine BCFAR, boundary condition type: 7)[2]. As previously mentioned, this boundary condition implementation is based upon the method of characteristics. The BCFAR1 routine (derived from BCFAR), was developed to take advantage of the VSAERO flow information. Also, the contiguous block interface boundary condition (subroutine INTER70, boundary condition type: 70) was modified to remove the restriction that the computational coordinates must be increasing in the same physical direction for two overlapping grid blocks.[2] The topological features of the t-tail grid-block structure did not conform to this requirement. A listing of the specific routines added to NPARC including a brief description are presented in Appendix A.

2.3 Computational Procedure

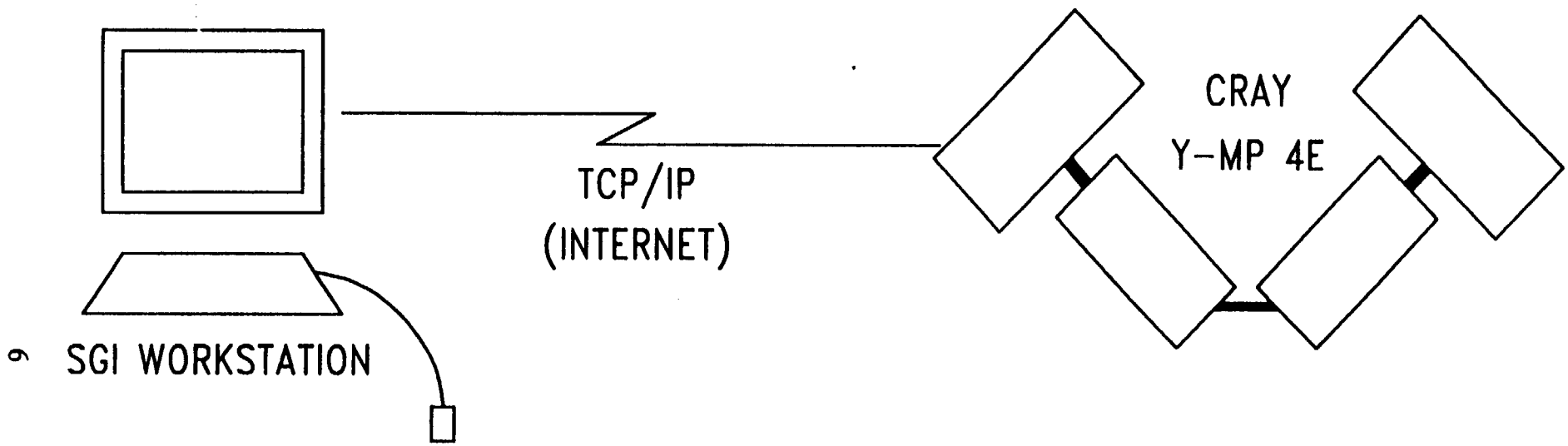
The step-by-step zonal procedure is outlined in Fig. 2. The user must first provide the basic model geometry in the form required by the particular grid generation package being used. Generally this process may be accomplished via a CAD database. The simple geometric tools available in VSAERO may be used for basic point re-distribution but in some cases the bi-quadratic interpolation has proven inadequate. Step two involves preparation of the volume grid within the flow domain of interest. In this work the grid generation process utilized the GRIDGEN system which results in the generation of a volume-grid data file in binary form.

For projects of this computational magnitude, the hardware utilization generally follows that described in Fig. 3. The user conducts certain elements of the work on a local workstation and via network access, is able to perform the more computationally demanding tasks on a super

TASK	SOFTWARE	OUTPUT
(1) PREPARE SURFACE GRID	CAD AND/OR VSAERO	VSAERO AND GRIDGEN SURFACE GEOMETRY DATA FILES
(2) PREPARE VOLUME GRID	GRIDGEN	BINARY VOLUME GRID FILE
(3) GENERATE NPARC RESTART DATA FILE	PRPAMI	NPARC RESTART DATA FILE
(4) PREPARE VSAERO SCAN INPUT DATA	PRPAMI	VELOCITY SCAN INPUT DATA TO BE APPENDED TO VSAERO INPUT FILE
(5) GENERATE OUTER FLOW B.C.'S AND UN–ICED CL, CM	VSAERO	VELOCITY SCAN OUTPUT DATA FILE
(6) GENERATE VISCOUS FLOW SOLUTION INCLUDING CL, CM	NPARC (AMI VERSION)	NPARC RESTART DATA FILE
(7) PREPARE GRAPHICS POST–PROCESSING DATA FILE	PRPAMI	OMNI3D GRAPHICS DATA FILE

Fig. 2. Task Summary of Zonal Ice Analysis Method.

HARDWARE UTILIZATION:



* GEOMETRY PREP (GRIDGEN)
* VSAERO CALCULATION
* POST-PROCESSING
* NPARC DATA FILE PREP
* NPARC CALCULATION

Fig. 3. Schematic of Hardware Utilization.

computer. For investigations requiring the memory and computational speed of a main frame super computer, the transfer of the volume grid file should proceed in binary form. This will help preserve the numerical accuracy of the model inherent in the binary format. Subsequent use of the binary data on the super computer (usually 64 bit), may require translation from 32 bit to 64 bit format which may be accomplished with the following command:

assign -F -f77 -N ieee 'filename'

This specific command is for a Cray computer running Unicos and should be used prior to each access of the binary data.

The next task is normally conducted on the main frame computer and is involved with generation of the NPARC restart data file utilizing the utility program PRPAMI, which was developed under this contract. A copy of the code together with a brief description may be found in Appendix B. The GRIDGEN volume grid file is required at this stage. The entire flow domain is initialized with freestream flow conditions which are provided to PRPAMI interactively.

The fourth step is the preparation of the VSAERO velocity scan input data. Again, PRPAMI is used in conjunction with the NPARC input file (INON5), and the restart file (REST.OLD). The user is responsible for specifying the flow boundary conditions in INON5, including the Type 8 zonal boundary condition developed for this work. (An example INON5 data input file may be found in Appendix C). The PRPAMI code will assemble the grid point coordinates associated with all of the Type 8 regions and create a data file called SCAN.DAT in a format suitable for the VSAERO velocity scan input format. The user has the option of appending this data to the end of the VSAERO input files or using a separate data file for the velocity scan Mold 9 input. Modifications to the VSAERO and NPARC user's manuals pertaining to this task may be found in Appendix D.

The fifth task outlined in Fig. 2 involves the analysis of the complete model with VSAERO. The recommended approach is to conduct an incremental study of the model starting with rigid, specified wake trajectories and potential flow only. After verifying the solution integrity, the wakes may be relaxed and the viscous correction with the streamline-based boundary layer may then be added. This process will eventually provide the force and moment coefficient of the un-iced configuration of interest and yield the off-body velocity-scan flow calculation based upon the SCAN.DAT data file. The local surface boundary layer profile may be included as part of the NPARC flow initialization by setting the variable MEET to -3 in the VSAERO input file, (refer to Appendix D).

The viscous flow solution for the isolated un-iced and iced model component is then determined as outlined under task six. As previously mentioned, the multi-block Reynold's-averaged NS method NPARC was utilized for this purpose. Depending upon the particular application, the user may have to pay especially close attention to the convergence characteristics

of the NPARC solution procedure. This topic is treated in more detail in Section 3.0. A procedure to compute the force and moment coefficients has been added to NPARC as a requirement of this work. Based upon the user-specified boundary conditions, this module determines the force and moment quantities associated with each solid surface region in both body and wind axis systems. Total aerodynamic coefficients are included at the end of the region by region force and moment summary written to the NPARC output data file. A listing of the routines that comprise this force and moment module along with a brief description is included in Appendix A.

Task seven should actually be utilized throughout the NPARC flow solution process initiated in task 6 to verify proper specification of flow boundary conditions and solution convergence characteristics. This step involves the generation of an OMNI3D compatible graphics data file with the utility code PRPAMI. The end result is a formatted data file which can be transferred to the OMNI3D platform for graphical processing.

3.0 GEOMETRY MODELING

This section details the work involved in the preparation of the Boeing 727-200 data case which was selected for validation of the proposed zonal concept. The current effort involves the analysis of the model at multiple angles of attack and zero sideslip in an effort to estimate the aerodynamic degradation due to the presence of leading-edge ice on the horizontal and vertical tail.

3.1 Panel Model

A VSAERO model of a complete Boeing 727-200 was prepared for this study. The wing was constructed with a jig twist distribution, which was consistent with the Boeing wind tunnel model (Fig. 1). The horizontal tail setting was -4 degrees with respect to the fuselage reference line. The center engine inlet was modified by removing the ducting inside the highlight and adding an elliptical faring from the highlight forward. The outboard engine was changed to a flow-through nacelle by extending the inlet duct aft to the trailing edge of the nacelle. Unfortunately, Boeing was not able to provide pictures of the support sting that was attached to the model at the aft end of the fuselage. In lieu of this information, a wake was attached to the trailing edge of the center engine exhaust to simulate the effects of the sting; therefore the accuracy of the modeling of the sting compared with the actual experiment is unknown at this time.

The VSAERO panelization of the starboard side of the aircraft included 4,105 body and 3,899 wake panels in order to simulate the symmetric flight conditions described in this report.

3.2 Grid Generation

Structured multi-block grids were developed individually about the isolated t-tail for both iced and un-iced flight conditions. Because of the zonal approach employed here, the NS domain for both grids was bounded by the region described in Fig. 1. The outer domain boundary shown here is about one semi-span from the horizontal tail. Further, the lower domain boundary includes a section of the vertical tail above the center engine. As previously described, the flow information on the outer faces of these volumes are provided by the VSAERO code including an initial estimation of the boundary layer growth on the vertical tail section.

The minimum grid spacing normal to the body surface is consistent with a y+ of order 10 (minimum $\Delta Z/C = 0.0001$). Due to the presence of the vertical tail, an H-O grid topology was used to discretize the flow field (Fig. 4). This representation generally does not resolve the leading-edge region as well as a C-type topology but the t-tail geometric constraints limit the topological choices. The three-block grids about the un-iced and iced t-tail consisted of over 690,000 grid points (194x73x49) and 815,000 (228x73x49) grid points, respectively (See Appendix C). This represents an increase of nearly 90,000 grid points over the grid resolution

Schematic of T-tail H-O Grid Topology

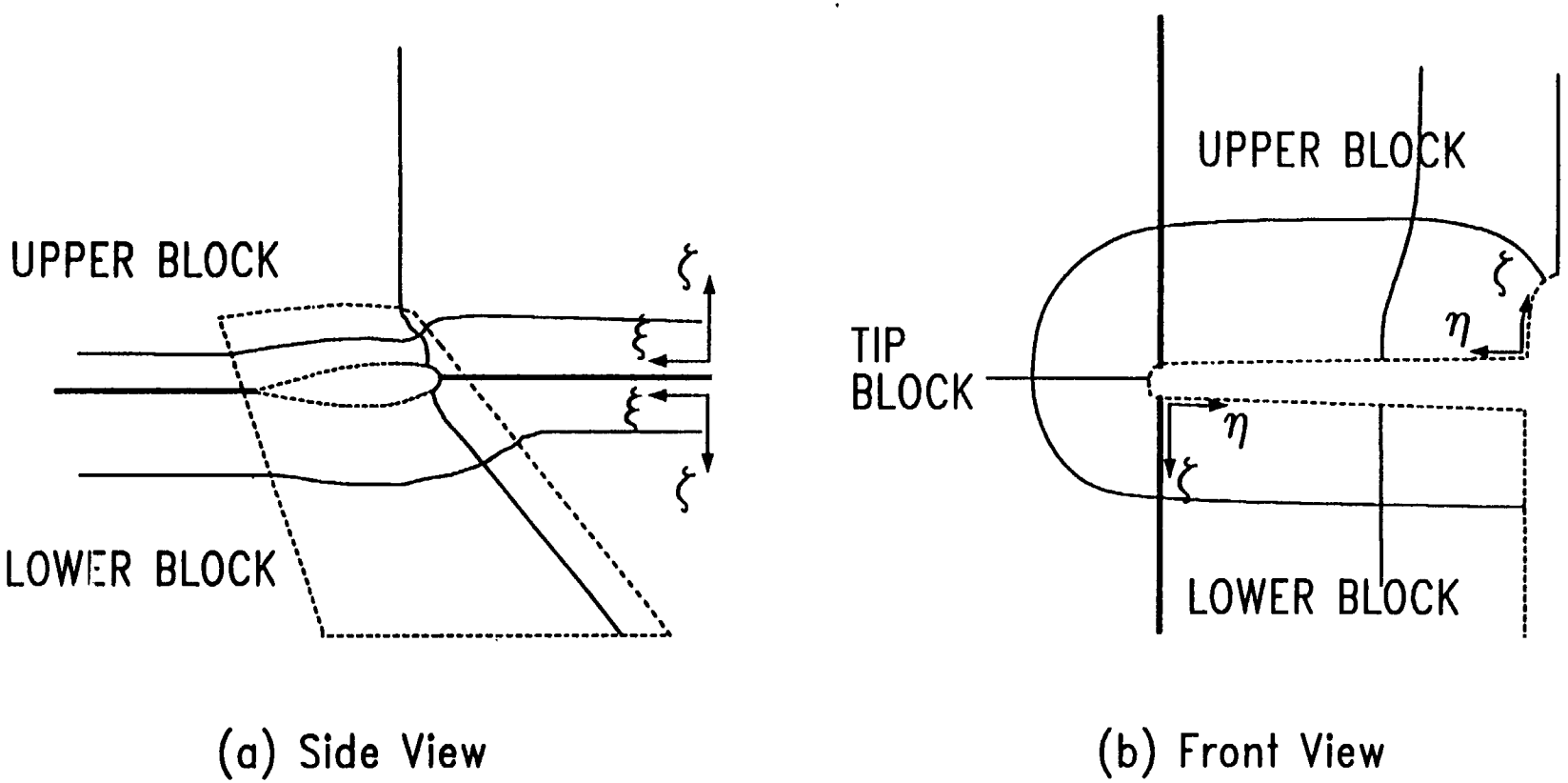


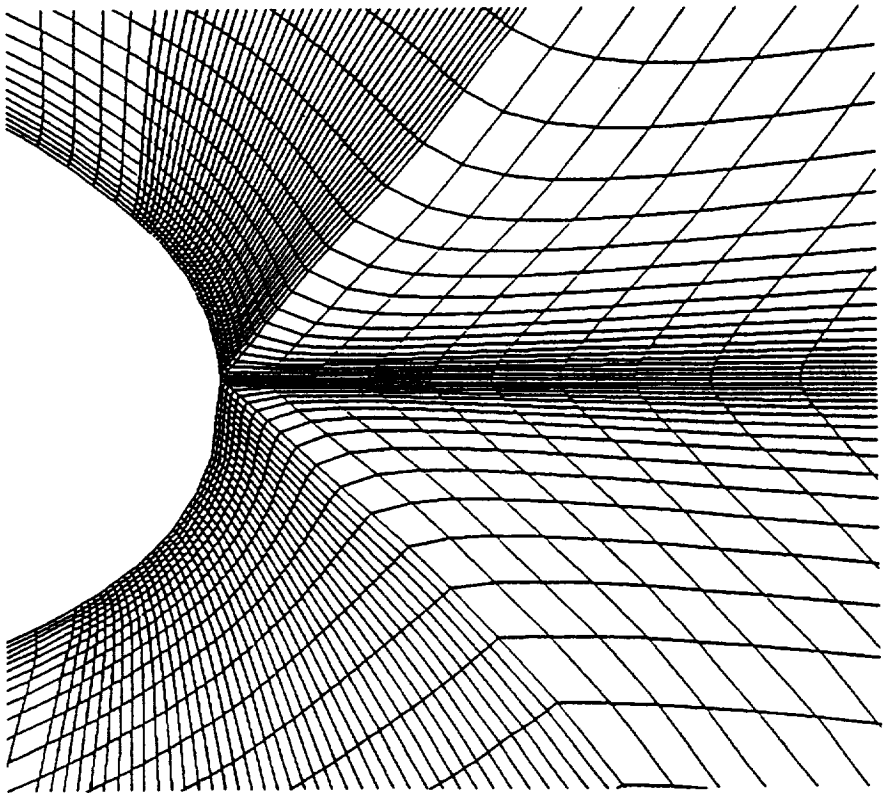
Fig. 4. Schematic of T-tail H-O Grid Topology.

used in previously reported work[4,13]. The region in the vicinity of the horizontal tail leading-edge has been refined in the work presented here as shown in Fig. 5. Geometry details of the leading-edge ice shape for the horizontal tail are presented in Section 4.0.

Volume grids for both the un-iced and iced t-tail configurations were generated using GRIDGEN3D Version 8.5[14]. Special modifications were made by MDA Engineering, Inc. to improve the control of orthogonality on selected faces. Specifically, the control functions were averaged across the leading edge of the horizontal tail (ξ direction), the leading edge of the vertical tail (ξ direction), and the tip of the vertical tail (ζ direction). Also, the foreground control functions were only calculated on the region of the tip block that corresponded to the tip surface of the horizontal tail. It is anticipated that these features will be included in version 9 of GRIDGEN3D[15].

The geometry manipulation required in this work was continually plagued with questions regarding accurate representation of the basic surface. The initial surface definition, based upon tabular data supplied by Boeing, as well as the single precision software components of the surface and grid generation processes, contributed to these difficulties. Clearly, grids that achieve y^+ values of O(1) are desirable for the viscous simulations reported here; however, this level of exactness requires surface definition to a minimum of eight places of accuracy and software tools capable of double precision computations.

(a) Original



(b) Revised

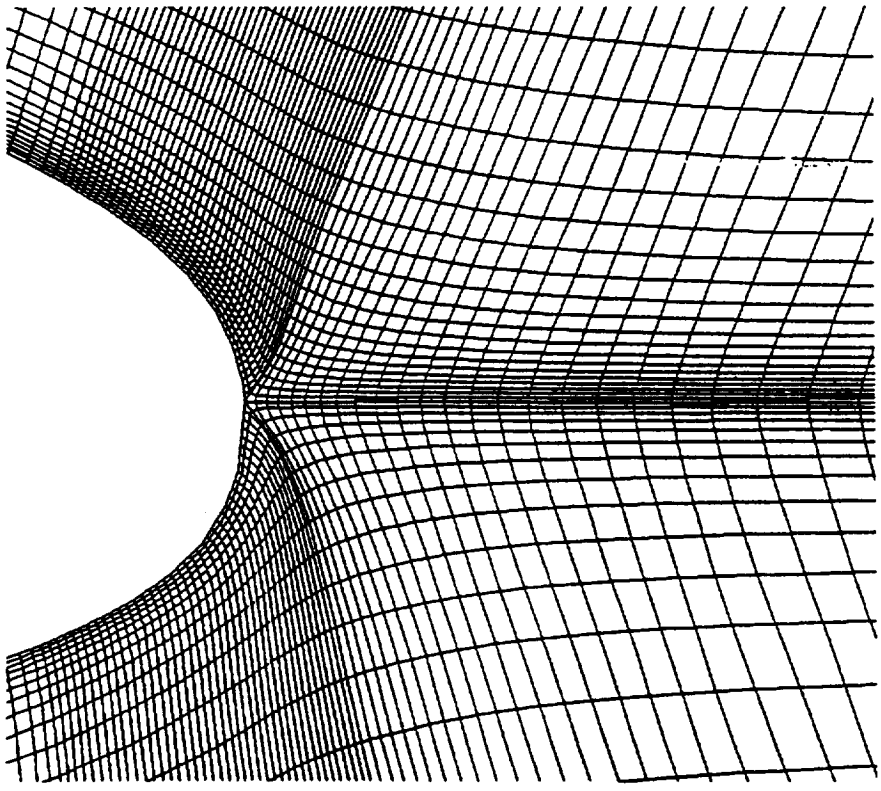


Fig. 5. Comparison of Original and Revised Grid in the Vicinity of the Horizontal Tail Tip Section.

4.0 RESULTS

A 0.075-scale model of the Boeing 727-200 was tested in January of 1981 in the Boeing VERTOL Tunnel. Wind tunnel data was obtained at a Mach number of approximately 0.12 and a Reynolds number of 1.24 million, based on wing reference chord. Longitudinal forces and moments were measured for the un-iced cruise configuration and for various combinations of iced leading edges on the horizontal and vertical tail. A summary of the specific flight conditions reported here is included in Table 1 below. This section presents aircraft force and moment comparisons to experiment which have been previously reported[4,13], as well as detailed flow features for the iced t-tail at zero degrees angle of attack.

CASE DESCRIPTION	CODE	ALPHA (deg)	YAW (deg)	MACH	Re
Un-iced Aircraft	VSAERO	-4,0,2,4	0.0	0.12	1.2 x E06
Un-iced T-Tail	NPARC	0,4	0.0	0.12	1.2 x E06
Iced T-Tail	NPARC	0,4	0.0	0.12	1.2 x E06

Table 1. Summary of Flow Calculations.

In previous work[16], it was not possible to generate an acceptable t-tail grid which would meet the single-block assumption and thin layer approximation of ARC3D. Conse-quently, longitudinal icing calculations included ice on the horizontal tail *only*. Icing effects, created by ice on the vertical tail and ice on the outboard Kruegers, were included in the test data, but were neglected in the longitudinal force and moment calculations.

In the current work, the multi-block capability of NPARC removes these earlier restrictions and allows for an improved representation of the empennage. The longitudinal characteristics are examined with leading-edge ice on both horizontal and vertical tail control surfaces as required. Based on practical considerations, the ice on the Krueger flap has been ignored in the present calculations.

4.1 Un-Iced Aircraft Simulation

Since the Mach number in the test was essentially negligible, all VSAERO runs were made without compressibility corrections. Longitudinal calculations included angle-of-attack settings of -4, 0, 2 and 4 degrees. Computed and measured force coefficients were reduced with a reference chord of 180.7 inches, a reference area of 244,800 square inches, and a half span of 648 inches. Moments were referenced to fuselage station 905.3 and waterline 184.2 inches.

The viscous correction for the un-iced VSAERO analysis was accounted for through the coupled, streamline-based integral boundary layer procedure. In the boundary layer calculations, transition was set to occur as soon as the local Reynolds number, based upon momentum thickness, reached a value of 200. Without other test information regarding transition behavior, this method has been found to provide an appropriate boundary layer simulation for such wind tunnel test conditions. Three viscid-inviscid correction iterations were computed for each flight condition. Computed contours of constant pressure coefficient are presented in Fig. 1 for zero degrees angle of attack.

Figs. 6 and 7 compare the computed lift and pitching moment characteristics with the experimental data. The VSAERO inviscid computation is included here for qualitative verification of the boundary layer effects. Within the linear range, the VSAERO boundary layer simulation matches the slope of the experimentally measured lift curve to within 1% and the zero lift angle agrees to within about one-tenth of a degree. This comparison is greatly improved over that previously reported in Ref. 16, which exhibited an error of approximately three-fourths of a degree. After further consultation with Boeing, it was determined that the defined wing twist distribution utilized in the previous work did not match the model specification; consequently, the current results represent the correct wing twist distribution. The calculated pitching moment slope of -0.032 compares well to the test value of -0.029 per degree and the computed zero pitching moment angle of attack is within 0.33 degree of the test value. Discussion of the un-iced aircraft longitudinal stability curve is included in the next section.

4.2 Iced Aircraft Simulation

In order to compute the force and moment increments of the aircraft due to ice, the viscous flow characteristics of the Boeing 727-200 empennage with and without leading-edge glaze ice was examined. Table 1 provides a detailed summary of the NPARC computations. Each of the NPARC computations required approximately 2.4xE-0.05 sec./grid point/iteration and 23 Mw of memory on a CRAY-YMP Model-E.

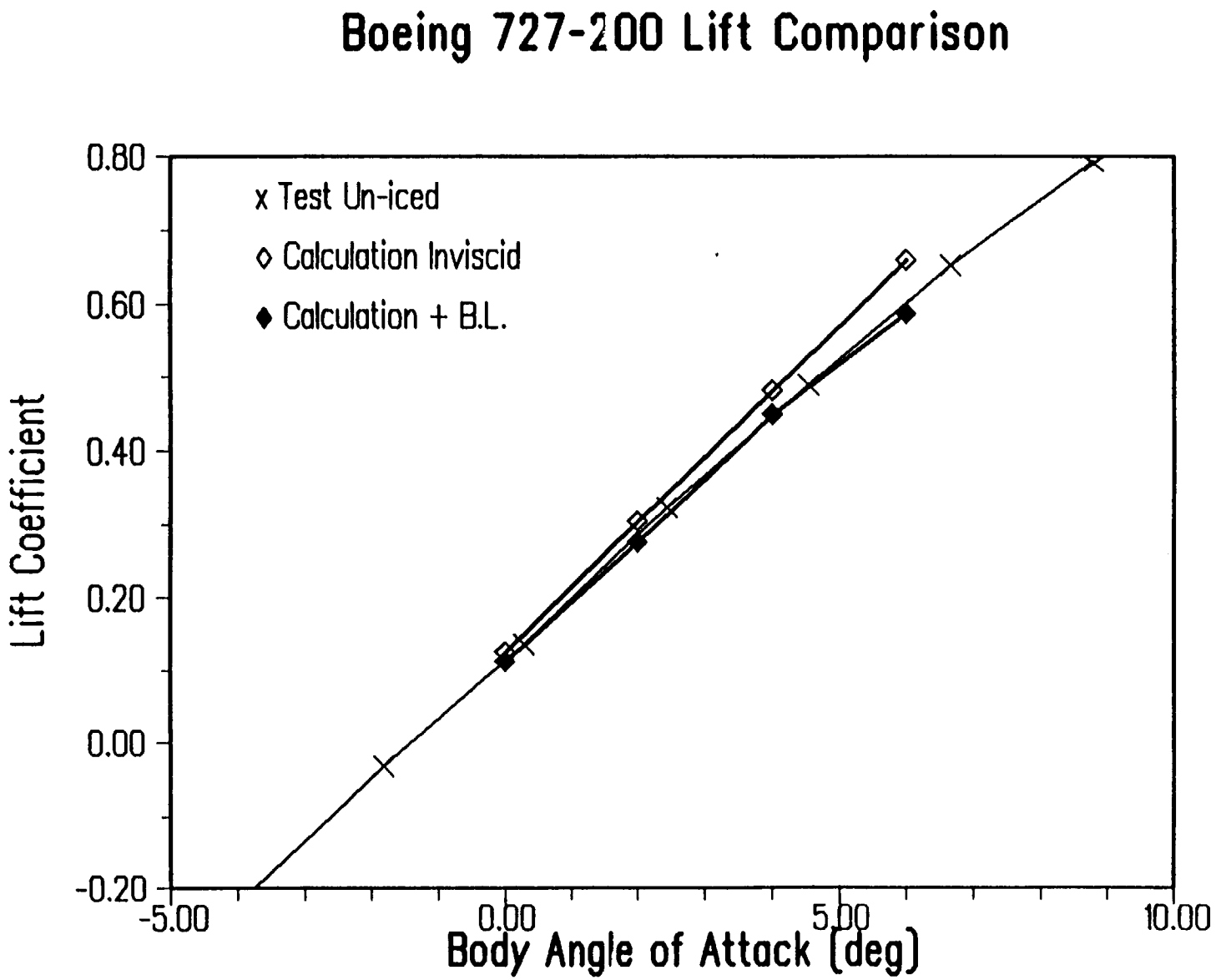


Fig. 6. Comparison of Measured to Calculated Aircraft Lift.

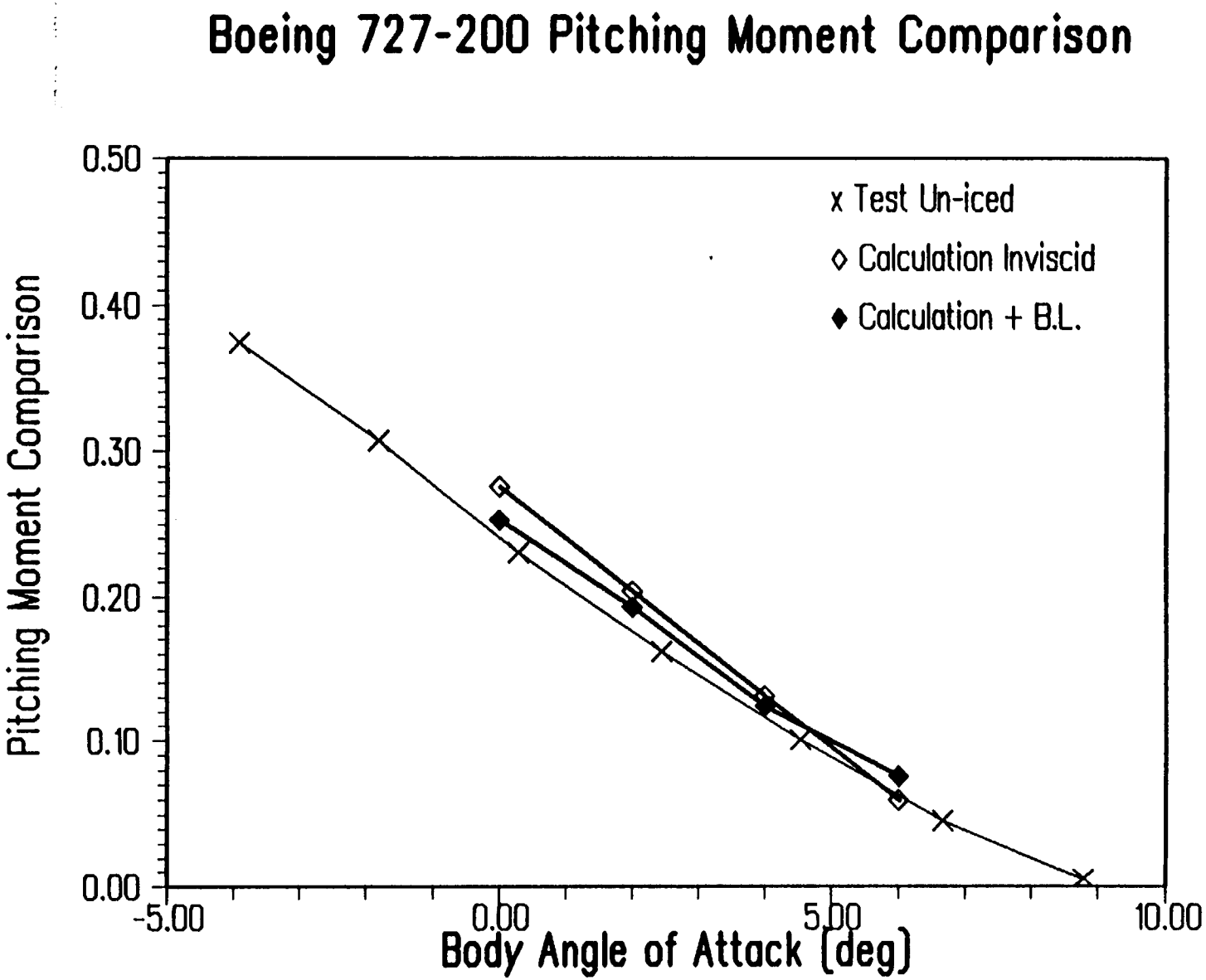


Fig. 7. Comparison of Measured to Calculated Aircraft Pitching Moment.

In all of the NPARC simulations, the flow was assumed to be fully turbulent at a model aircraft Reynolds number of 1.24 million. In these simulations, NPARC required from 10,000 to 15,000 iterations to achieve residual reductions on the order of 2 to 3 for a CFL number of 2 (see for example, Fig. 8). In practice, residual reduction of 3 to 4 orders is the generally accepted convergence criterion. In this study however, various factors have contributed to the poor residual convergence behavior. First, the fine grid resolution tends to restrict the maximum CFL number or time-step size which results in an increased number of iterations.[17] Second, the multi-block nature of the domain decomposition adversely affects solution convergence due to the time lag of the boundary information.[18] Third, the low subsonic Mach number is at or near the operational limit of the compressible flow equation set.[17] Further testing is required to quantify the relative contribution of each factor and to suggest means by which the residual convergence can be accelerated.

In addition to residual history, the integrated lift and pitching moment coefficients were monitored as an aid in determining the convergence level. The history of lift and pitching moment coefficient is shown for the iced and un-iced t-tail configuration in Fig. 9 for an angle of attack of zero degrees. The unsteady behavior previously noted[4,13], is not obvious in the current calculations. It is apparent that the grid refinement and/or code differences between PARC3D and NPARC have contributed to this improvement in computed flow characteristics. As shown by the time histories of lift and pitching moment, the iced t-tail required more iterations to insure adequate convergence levels in comparison to the un-iced geometry.

A comparison of the un-iced and iced horizontal tail mid-span sectional pressure distribution at 0° angle of attack is presented in Fig. 10. As indicated, the presence of leading-edge ice generates additional down force resulting in an increased aircraft nose-up pitching moment. A close examination of the iced airfoil indicates that the ice shape produces a negative droop of the leading-edge. Thus the ice shape modifies, locally, the airfoil camber to produce the additional loading. There is also an apparent contribution from the effect of leading-edge ice on the boundary layer which would require further study to identify.

The horizontal tail reversed flow region in the vicinity of the upper and lower ice horn is presented in Fig. 11 for both original and current grid resolution. It is apparent that the flow solution on the refined grid is improved at least in a qualitative sense over the result on the original grid. A close-up view of the lower horn clarifies this point further (Fig. 12). The solid horizontal line in both figures marks the boundary between the upper and lower grid blocks. Note especially the flow vectors in the vicinity of the ice horn leading-edge and block interface. The refined grid result shows marked improvement over the original grid calculation.

The computed increment in lift and moment coefficients due to the presence of ice at zero degrees angle of attack is -0.0025 and 0.0122 respectively; while the corresponding increments based upon the original grid density are -0.0017 and 0.0095. This does not appreciably modify the previously reported force and moment results. Furthermore, this result tends to alleviate the earlier concerns regarding repeatability due to the relatively small force and moment increments

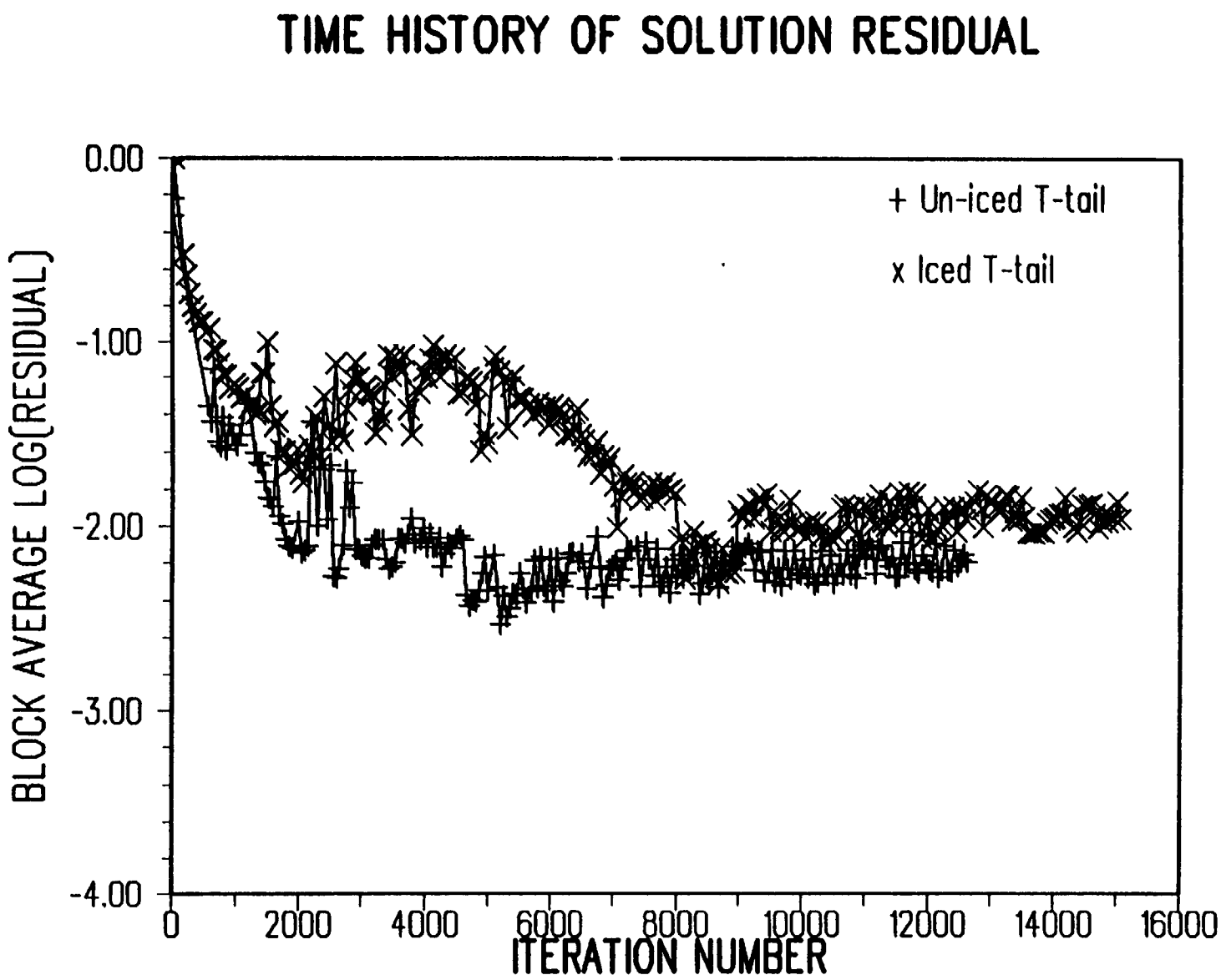


Fig. 8. History of the Block Average of Solution Residual.

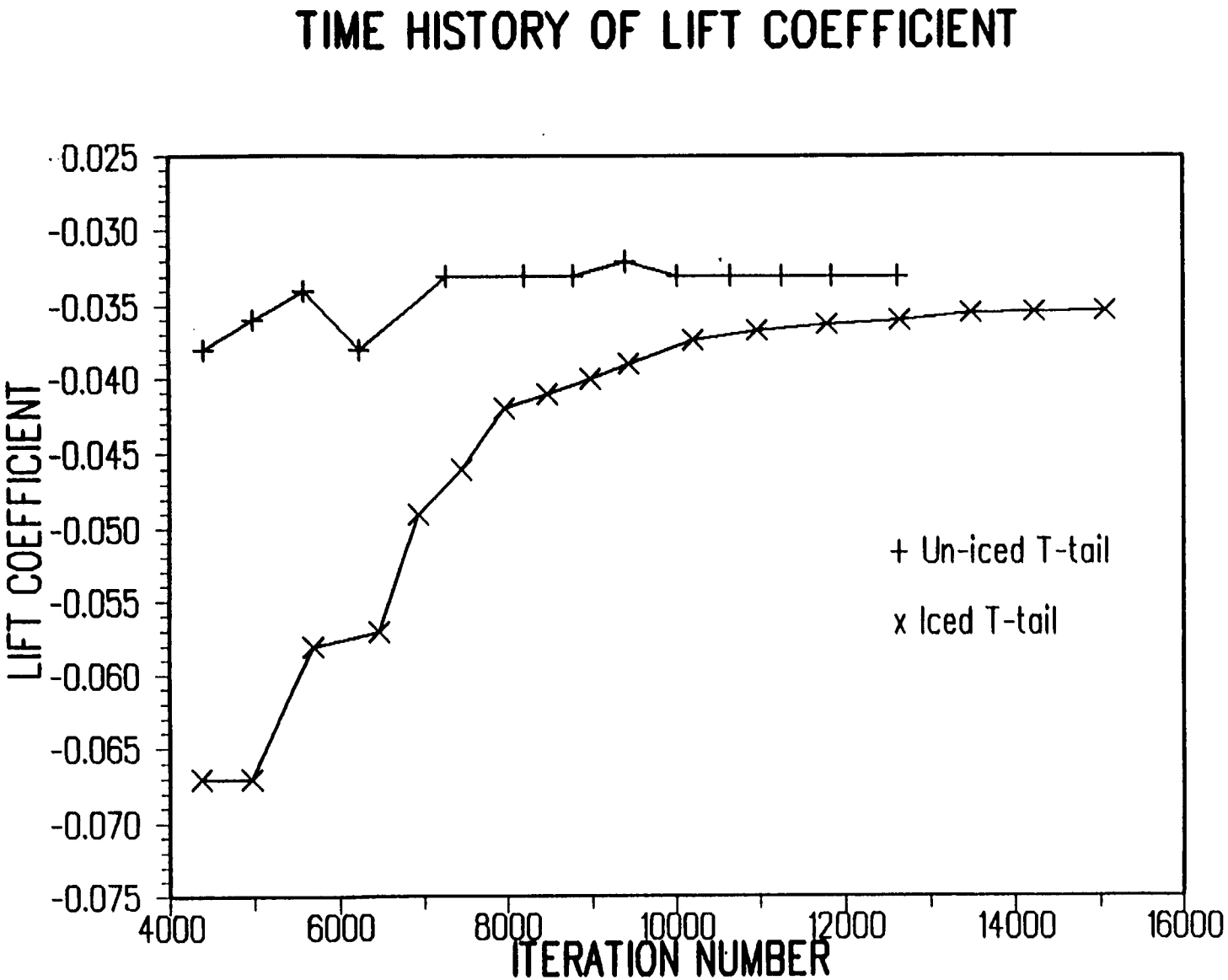


Fig. 9(a). History of Empennage Computed Lift Coefficient.

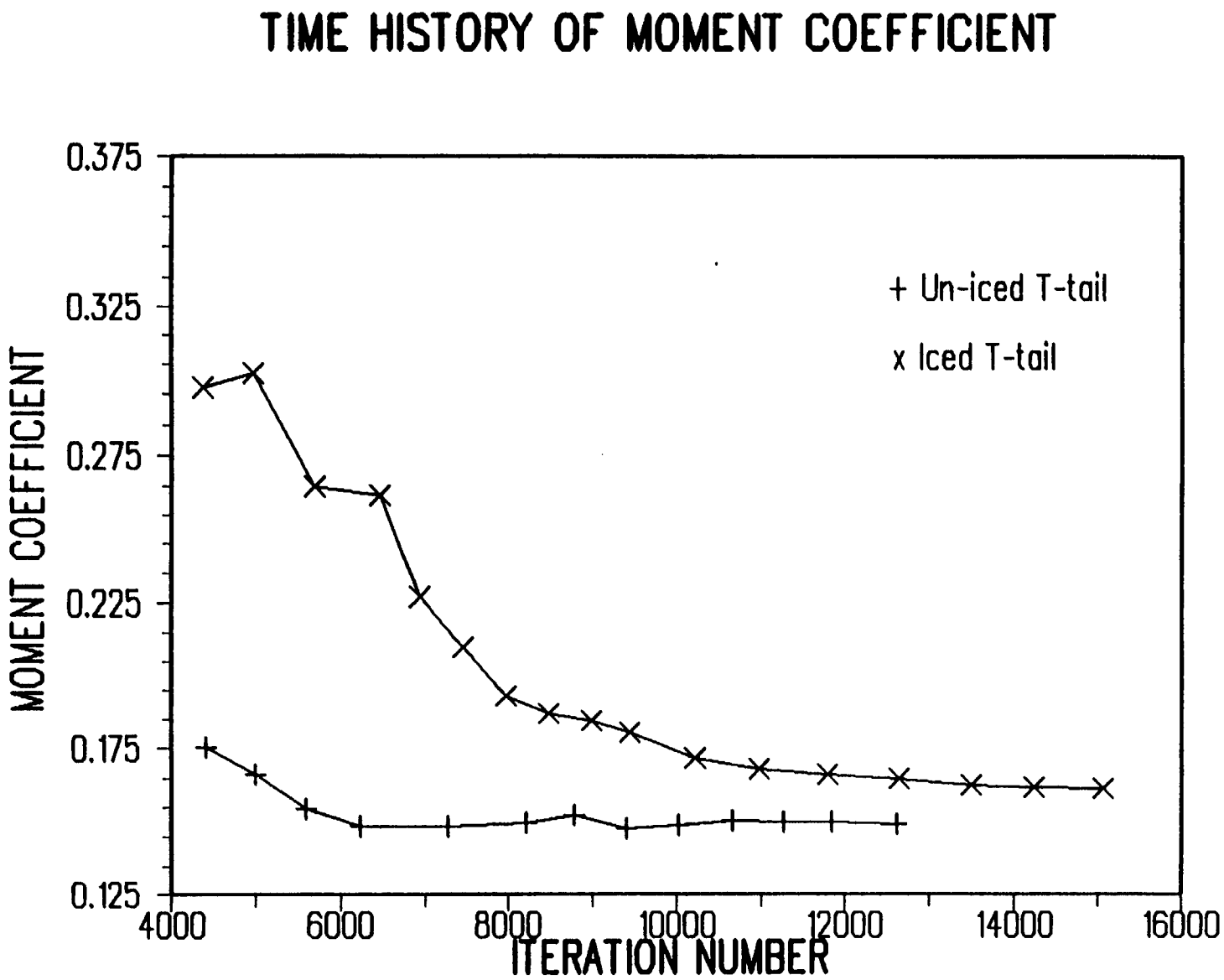


Fig. 9(b). History of Empennage Computed Pitching Moment Coefficient.

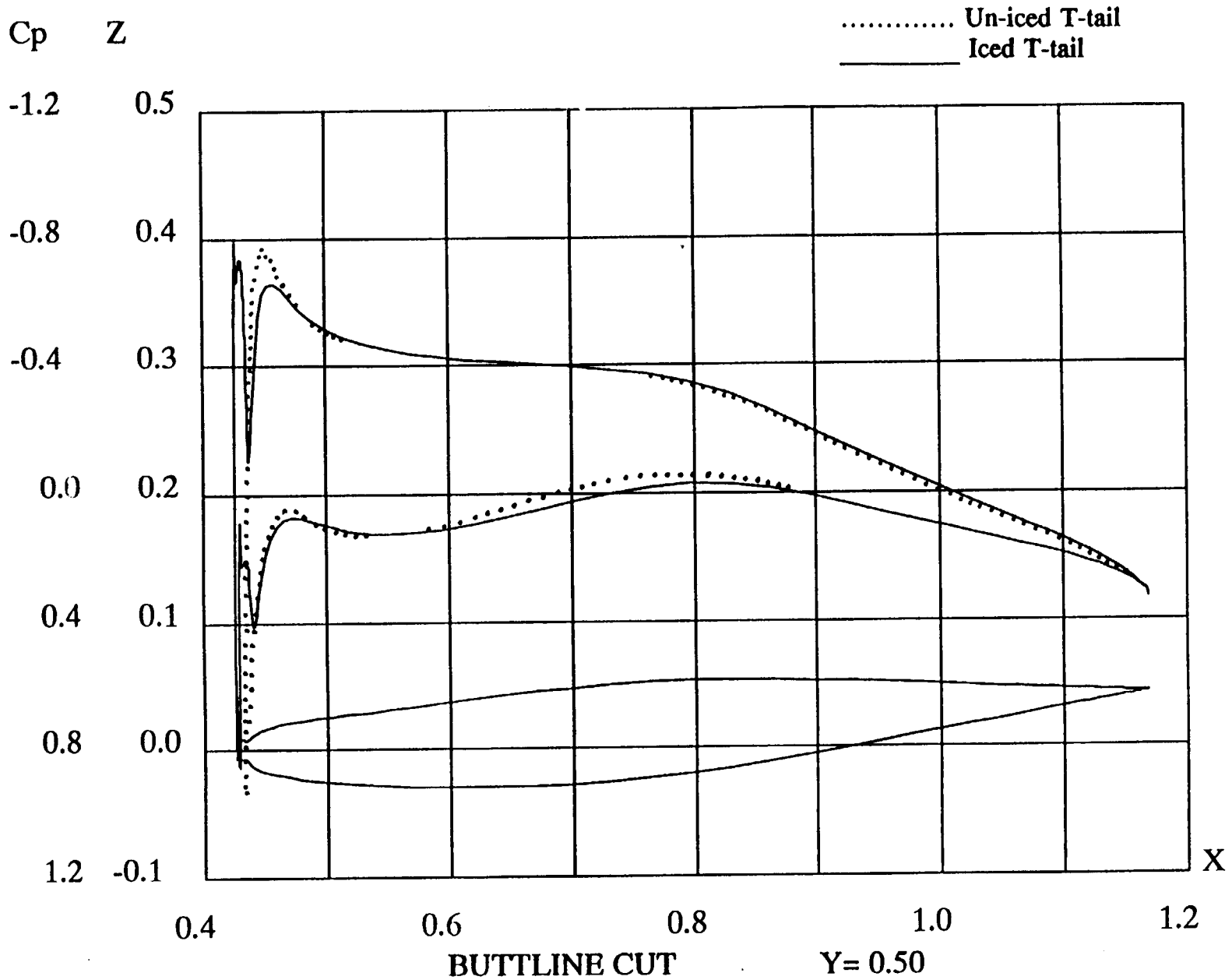


Fig. 10. NPARC Computed Surface Pressure Distribution at Mid-Span of the Horizontal Tail.

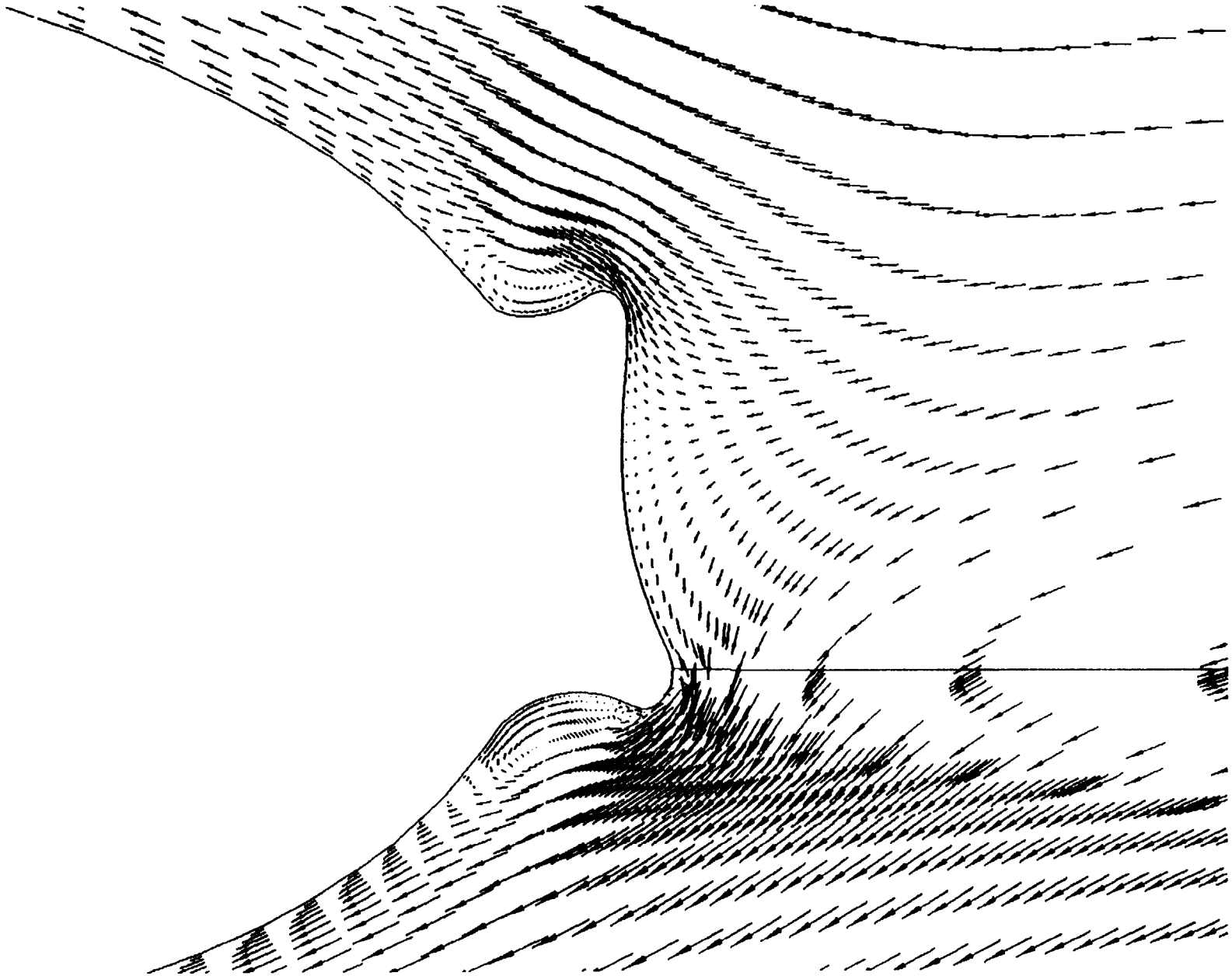


Fig. 11(a). Flow Velocity Vectors in the Vicinity of the Iced Horizontal Tail Leading Edge for the Original Grid. Flow is from Right to Left.

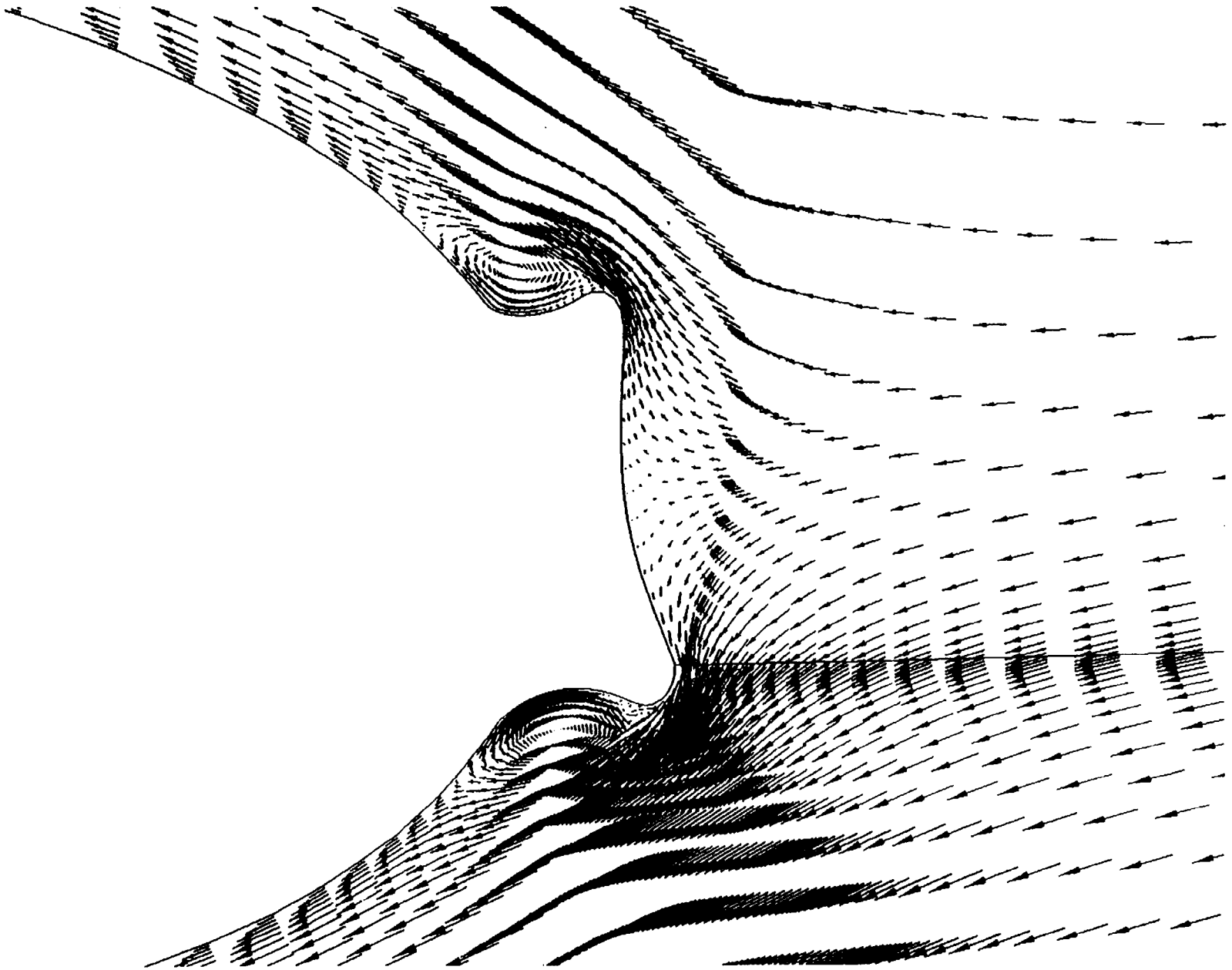


Fig. 11(b). Flow Velocity Vectors in the Vicinity of the Iced Horizontal Tail Leading Edge for the Revised Grid. Flow is from Right to Left.

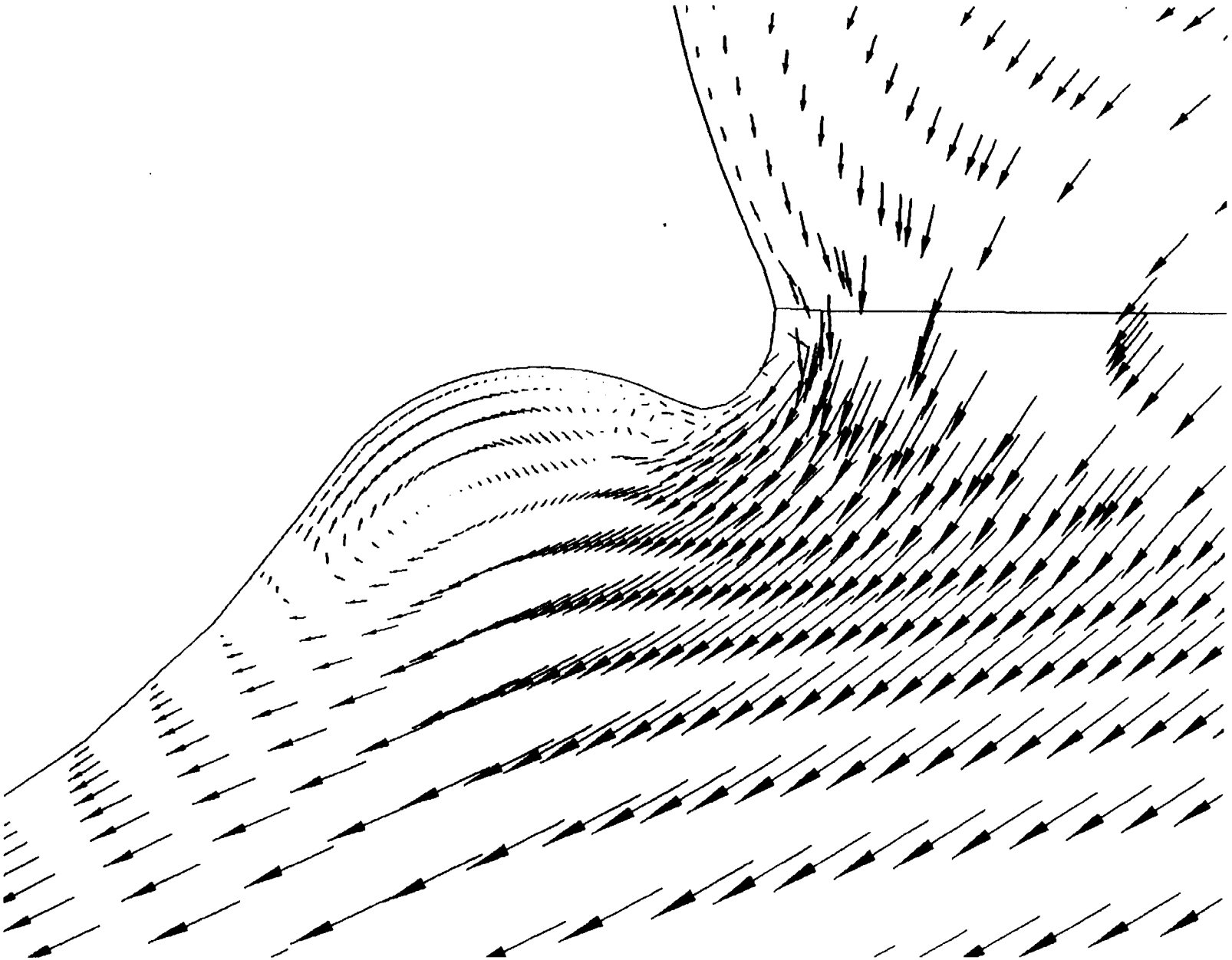


Fig. 12(a). Close-up View of Flow Velocity Vectors in the Vicinity of the Lower Lobe of Iced Horizontal Tail for the Original Grid. Flow is from Right to Left.

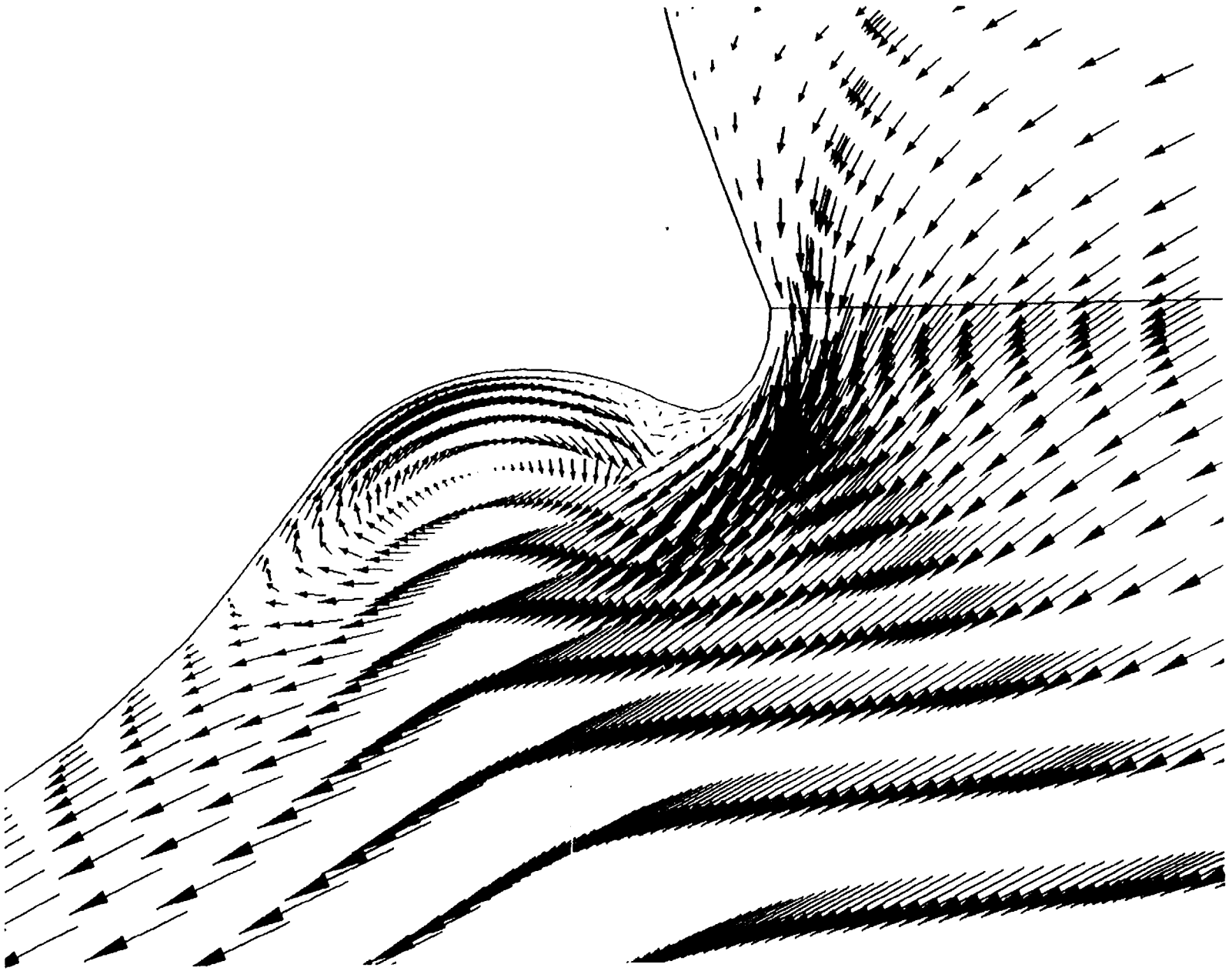


Fig. 12(b). Close-up View of Flow Velocity Vectors in the Vicinity of the Lower Lobe of Iced Horizontal Tail for the Revised Grid. Flow is from Right to Left.

due to ice[4,13]. In fact, the detailed force and moment comparisons presented in this report are based on calculations utilizing the original grid resolution.

The computed aircraft lift and pitching moment characteristics are compared with experimental data in Figs. 13 and 14 and a quantitative breakdown is presented below in Tables 2 and 3. For comparison purposes, the test data characteristics were estimated by restricting the region of interest to the linear angle-of-attack range. The computed lift curve slope with ice matches the experimental value to within 1%, and the zero lift angle agrees to within two-tenths of a degree. In addition, the computed pitching moment curve slope of -0.035 correlates with the test value of -0.032 per degree.

TEST	CALCULATION	UN-ICED	ICED	$dC_L/d\alpha$ (per deg)	α_o (deg)	C_{L_o}
X		X		.077	-1.48	.114
X			X	.077	-1.44	.111
	X	X		.084	-1.36	.114
	X		X	.085	-1.29	.110

Table 2. Detailed Comparison of the Aircraft Longitudinal Characteristics:C_L vs, α.

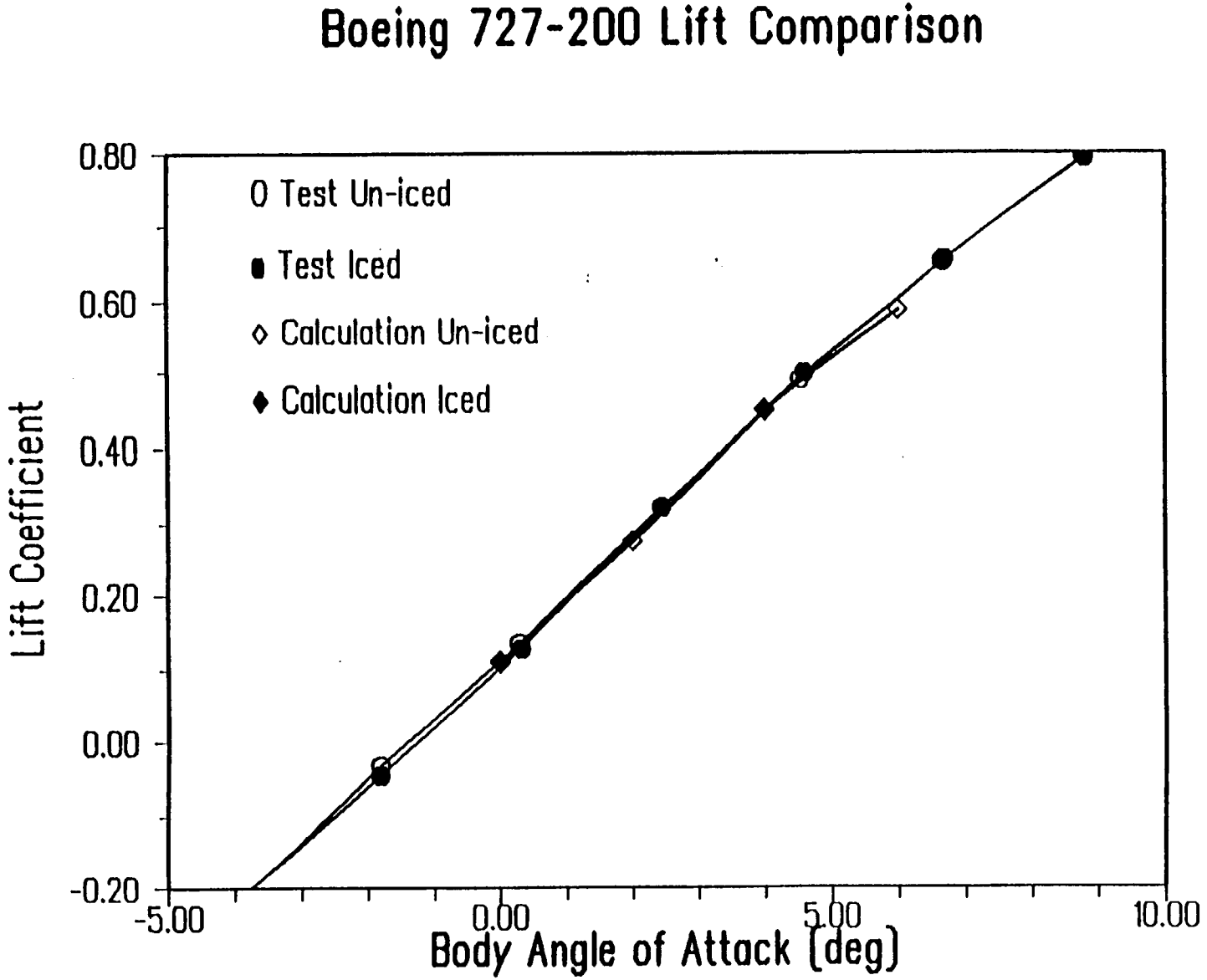


Fig. 13. Comparison of Measured to Calculated Impact of Ice on Aircraft Lift Coefficient.

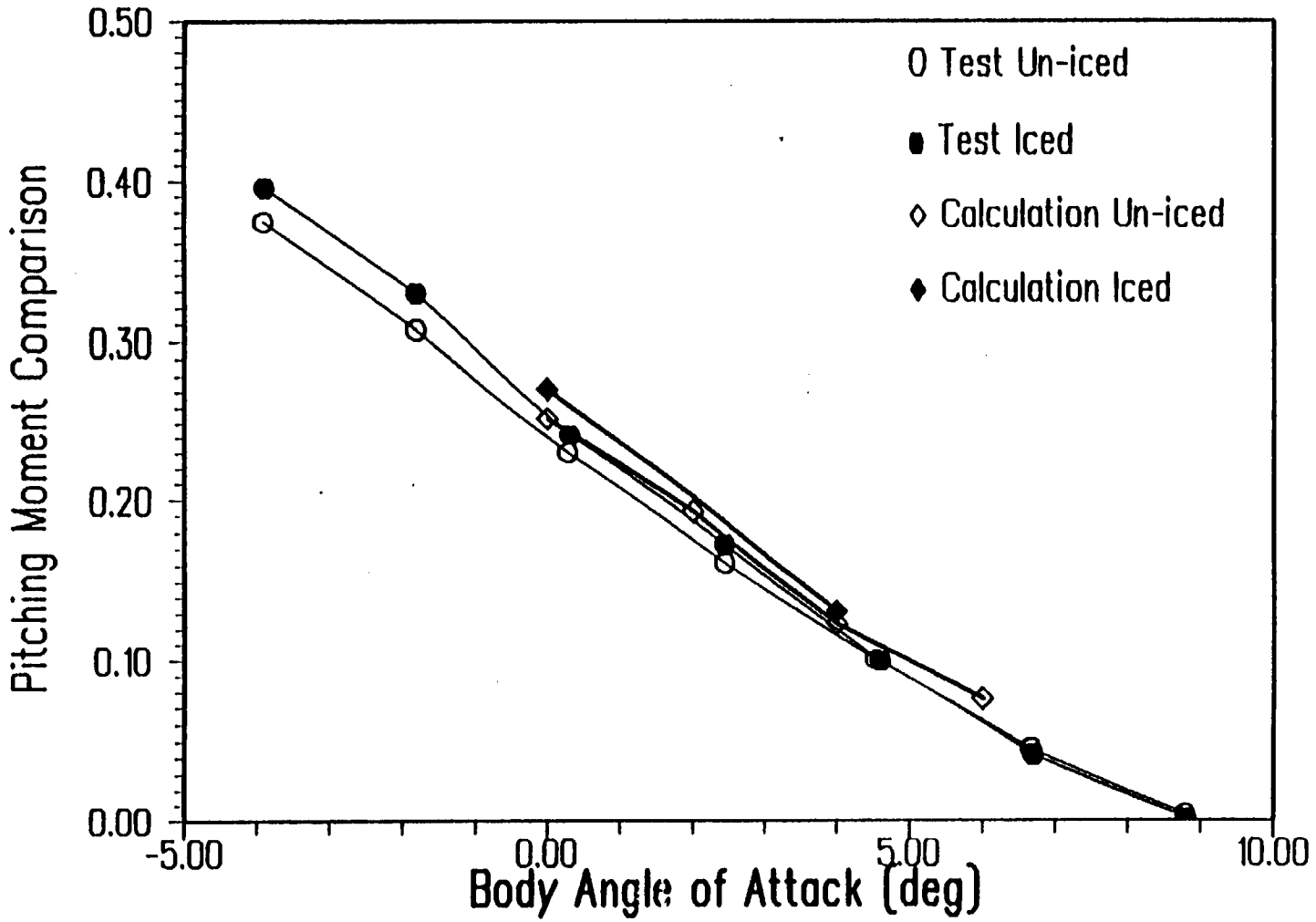


Fig. 14. Comparison of Measured to Calculated Impact of Ice on Pitching Moment Coefficient.

TEST	CALCULATION	UN-ICED	ICED	$dC_M\gamma/d\alpha$ (per deg)	ABSCISSA INTERCEPT (deg)
X		X		-.029	8.15
X			X	-.032	7.94
	X	X		-.032	7.84
	X		X	-.035	7.80

Table 3. Detailed Comparison of the Aircraft Longitudinal Characteristics:C_{My} vs. α.

As indicated, the computed absolute slope and intercepts compare well with the test data as well as the increments due to ice. In particular, several incremental quantities may be estimated from the tabulated data. The experimental increment due to ice in lift curve slope, (ΔC_{L_α}), zero lift angle of attack, (α_o), and lift at zero angle of attack, (ΔC_{L_o}), of 0.000, 0.04 and -0.003 compares very well with their respective computed values of 0.001, 0.07 and -0.004 (see Table 2). The experimental increment due to ice in pitching moment curve slope, (ΔC_{M_α}), and abscissa intercept, ($\Delta\alpha$), of -0.003 and -0.21 correlates well with their respective computed values of -0.003 and -0.04 (see Table 3). Evidently, for the longitudinal case, the additional effects of the ice on the outboard Kruegers that was included in the experiment are indeed small for the examined angle of attack range.

The aircraft longitudinal stability curve is graphically presented in Fig. 15 and a detailed comparison of the computed and test stability data are presented below in Table 4. As shown, the computed slopes for the un-iced and iced aircraft are within about 2% of the test data. Further, based on estimates derived from Table 4, the experimental increment due to ice in stability curve slope,(dC_{My}/dC_L) and pitching moment at zero lift (ΔC_{M_o}), of 0.16 and -0.01 compares favorably with the computed values of 0.18 and -0.003 respectively. In this case, the addition of ice to the vertical and horizontal tail leading-edge is a destabilizing influence.

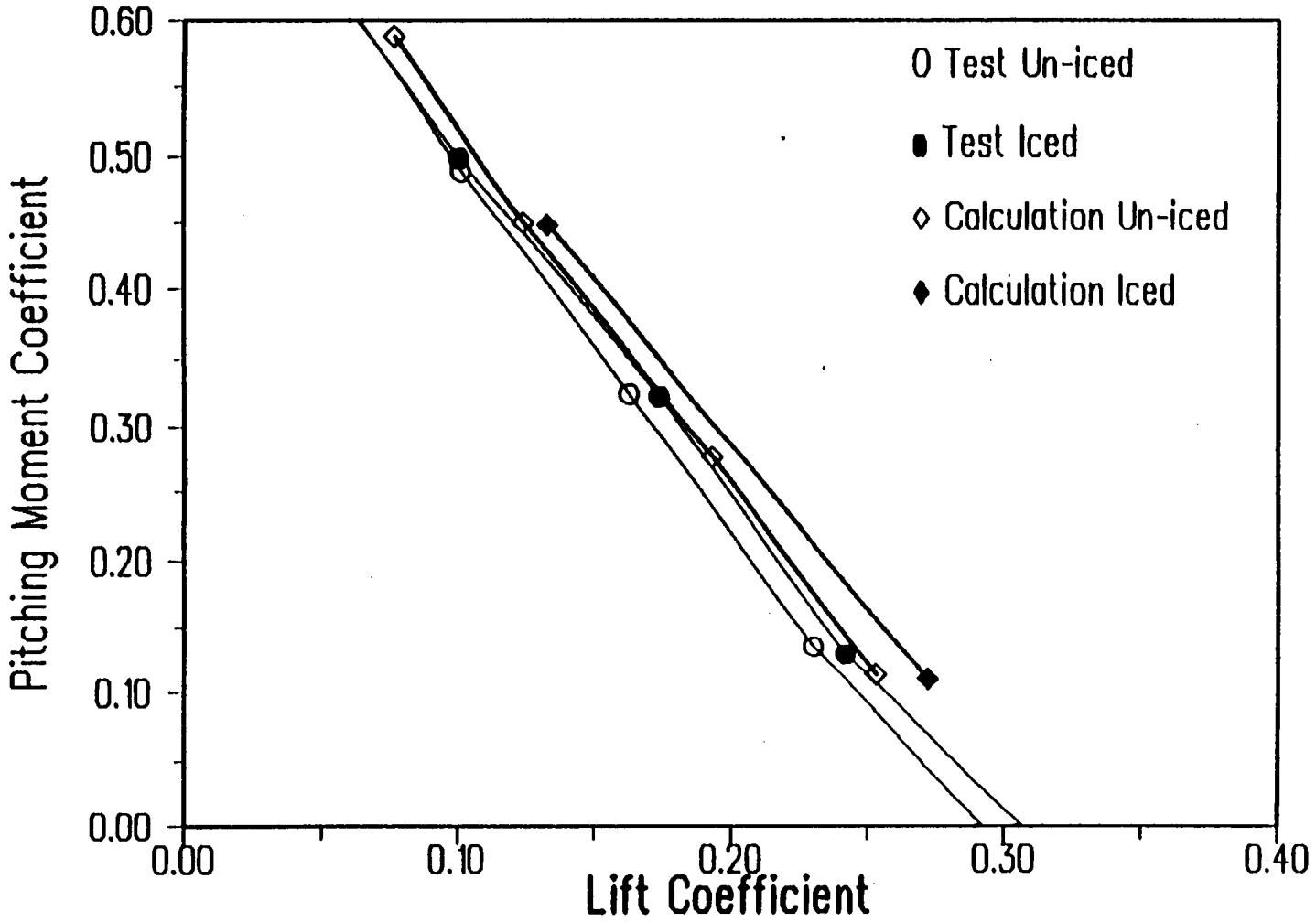


Fig. 15. Comparison of Measured to Calculated Impact of Ice on Aircraft Stability.

TEST	CALCULATION	UN-ICED	ICED	dCm*/d* (per deg)	C_{My}
X		X		-2.63	.761
X			X	-2.47	.751
	X	X		-2.60	.773
	X		X	-2.42	.770

Table 4. Detailed Comparison of the Aircraft Longitudinal Characteristics: C_{My} vs. C_L.

Generally, computed increments due to ice are in the proper direction and are of the same magnitude as the test data for the aircraft longitudinal characteristics. The small relative magnitude of these increments, though, makes it difficult to discuss error in terms of percent.

5.0 CONCLUSIONS AND RECOMMENDATIONS

In the work reported here, a simplified, uncoupled, zonal procedure is utilized to assess the capability of numerically simulating icing effects on a Boeing 727-200 aircraft. The computational approach combines potential flow plus boundary layer simulations by VSAERO for the un-iced aircraft forces and moments, with NS simulations by NPARC for the incremental forces and moments due to iced components. The computed lift curve slope with and without empennage ice matches the experimental value to within 1%, and the zero lift angle agrees to within two-tenths of a degree. Furthermore, the computed slope of the un-iced and iced aircraft longitudinal stability curve is within about 2% of the test data. These results demonstrate the feasibility of a zonal method for the icing analysis of complete aircraft or isolated components within the linear angle of attack range. In fact, this zonal technique has allowed for the viscous analysis of a complete aircraft with ice which is currently not otherwise considered tractable.

Future work should include investigation of solution convergence acceleration methods in order to reduce the current CPU requirements, and double precision software in order to improve the geometric accuracy. Additionally, an experimental data base that includes surface definition to a minimum of eight places of accuracy, and an incremental buildup of the iced components as well as surface pressure information should be available in order to thoroughly evaluate the CFD prediction capability.

Acknowledgements

The authors would like to thank NASA Lewis Research Center for its support of this work under Contract NAS3-26310, The Boeing Company for providing the model geometry and wind tunnel data, Jef Dawson of Cray Research, Inc. for providing valuable computer time, and John Chawner of MDA Engineering, Inc. for providing the development version of GRIDGEN as well as expertise in grid generation. In addition the authors would like to thank their colleagues at AMI, David A. Lednicer and Deana Nelson-Wilson for their invaluable help with the surface geometry preparation and the VSAERO calculations.

6.0 REFERENCES

1. Maskew, B., "Prediction of Subsonic Aerodynamic Characteristics: A Case for Low-Order Panel Methods," *J. Aircraft*, Vol. 19, No. 2, February 1982.

2. Cooper, G.K. and Sirbaugh, J.R., PARC Code: Theory and Usage, AEDC-TR-89-15, Arnold Engineering Development Center Report, Arnold Air Force Base, Tennessee, December 1989.

3. Private Correspondence, Boeing Commercial Airplane Group, January, 1992. 727-200 Wind Tunnel Test Data (tested January 1981).

4. Summa, J.M. and Strash, D.J., "Development of Three-Dimensional Flow Code Package to Predict Performance and Stability of Aircraft with Leading-Edge Ice Contamination," Final Report, Option 1, NASA Lewis Contract NAS3-26310, May 1993.

5. Summa, J.M., Strash, D.J. and Yoo, S., "A Zonal Flow Analysis Method for Two-Dimensional Airfoils," AIAA Paper 90-0571, January 1990.

6. Yoo, S., Summa, J.M. and Strash, D.J., "Angle-of-Attack Validation of a New Zonal CFD Method For Airfoil Simulations," AIAA Paper 90-3077, presented at 8th Applied Aerodynamics Meeting, Portland, Oregon, August 1990.

7. Strash, D.J., Summa, J.M. and Yoo, S., "Mach Number Validation of a New Zonal CFD Method (ZAP2D) for Airfoil Simulations," AIAA Paper 91-0185, presented at the 29th Aerospace Sciences Meeting, Reno, Nevada, January 1991.

8. Raj, et al, "Three-Dimensional Euler Aerodynamic Method (TEAM), Vol. 1: Computational Method," Interim Report, Lockheed, December 1987.

9. Morino, L. and Kuo, C.-C., "Subsonic Potential Aerodynamics for Complex Configurations: A General Theory," *AIAA Journal*, Vol. 12, February 1974.

10. Pulliam, T.H., and Steger, J.L., "Implicit Finite Difference Simulations of Three-Dimensional Compressible Flow," AIAA Paper 78-10, 1978.

11. Beam, R., and Warming, R.F., "An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law Form," *J. Comp. Physics*, Vol. 22, September 1976.

12. Baldwin, B.S. and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, 1978.

13. Strash, D.J. and Summa, J.M., "Improved CFD Zonal Modeling of Leading-Edge Ice Effects for a Complete Aircraft," AIAA Paper 94-0488, presented at the 32nd Aerospace Sciences Meeting & Exhibit, Reno, NV, January 1994.

14. Chawner, J.R. and Steinbrenner, J.P., "Demonstration of the Use of GRIDGEN to Generate a 3D, Multiple-Block, Structured Grid, AIAA-92-0069, January 1992.

15. Telecon with Mr. J.R. Chawner of MDA Engineering, Inc., January 1993.

16. Summa, J.M., Strash, D.J., Yoo, S., and Lednicer, D.A., "CFD Zonal Modeling of Leading-Edge Ice Effects for a Complete Aircraft," AIAA-93-0167, January 1993.

17. Anderson, D.A., Tannehill, J.C. and Pletcher, R.H., **Computational Fluid Mechanics and Heat Transfer**, published by McGraw-Hill, 1984.

18. Telecon with Mr. J.R. Sirbaugh of Sverdrup Technology, Inc., March 1993.

APPENDIX A

The following subroutines have been added to the NS code NPARC version 1.0 and were developed under NAS3-26310. These modifications pertain to the zonal aerodynamic analysis of aircraft or aircraft components with ice.

Subroutine Name	Description
RFIXED	Opens outer boundary condition data file PARC.SCN as fortran unit 32. Loads uo, vo, wo, cpo arrays. Called from INITIA if type 8 boundary condition specified in input.
BCFAR1	Patterned after routine BCFAR. Sets zonal boundary condition based upon uo, vo, wo, cpo information. Called from routine BC.
SETVEL	Sets local flow parameters from uo, vo, wo, cpo arrays. Called from BCFAR1.
FORMOM	Computes the force and moment coefficients for all type 60 and 61 regions. Called from main routine.
PATCH	Forms surface panel element information such as, area and normal vector. Called from FORMOM.
CPCALC	Computes local surface panel element pressure coefficient. Called from FORMOM.
CORNR	Assembles flow variables at each corner of a surface panel element. Called from CPCALC.
ANLIZ	Performs loads summation over each surface region and prints out result. Called from FORMOM.

```
      SUBROUTINE RFIXED
C*************************************************************************
cdan New routine added in December 1992 for VSAERO Vscan data.
C*************************************************************************
      include 'npami.inc'
      PARAMETER (NSCNMX=JMX*KMX*LMX)
      COMMON/SCAN/UO(NBX,JMX,KMX,LMX),VO(NBX,JMX,KMX,LMX),
     &            WO(NBX,JMX,KMX,LMX),CPO(NBX,JMX,KMX,LMX)

      OPEN(UNIT=32,FILE='PARC.SCN',FORM='FORMATTED',ERR=802,
     &            STATUS='OLD')
      REWIND 32
      NPT=0
C  SET DEFAULTS
      DO 4 N=1,NBX
      DO 4 J=1,JMX
      DO 4 K=1,KMX
      DO 4 L=1,LMX
      UO(N,J,K,L)=501.
    4 CONTINUE

C READ FIRST TWO DUMMY RECORDS...
      READ(32,*)
      READ(32,*)
      DO 5 J=1,NSCNMX
      READ(32,*,END=801,ERR=5)U,V,W,CP,NB,JP,KP,LP
      IF(JP.EQ.0)GO TO 5
      NPT=NPT+1
      UO(NB,JP,KP,LP)=U
      VO(NB,JP,KP,LP)=V
      WO(NB,JP,KP,LP)=W
      CPO(NB,JP,KP,LP)=CP
    5 CONTINUE
      WRITE(6,*)' ERROR IN RFIXED, ALL BC POINTS HAVE NOT BEEN READ'
      WRITE(6,*)' INCREASE NSCNMX PARAMETER'
      STOP

  801 CONTINUE
      WRITE(*,*)'*** TOTAL NUMBER OF FIXED BC POINTS READ...',NPT
      RETURN
  802 WRITE(*,*)' *** ERROR OPENING UNIT 32 ***'
      STOP
      END

      SUBROUTINE BCFAR1(ID,J,KA,KB,LA,LB,ISGN,PINF,TINF,
     *                  XX,XY,XZ, YX,YY,YZ, ZX,ZY,ZZ, Q )
C*************************************************************************
cdan New routine added in December 1992 for VSAERO Vscan data.
C*************************************************************************
      include 'npami.inc'
      COMMON/INTEG/ IBF(NM+1),FNMACH(NM), fill(58*NM-1+12*NDIF)
      COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
      COMMON/WAYOUT/ A0,R0,U0,V0,W0,E0,P0
      COMMON/MBLOCK/NBLOCK,MB,MBNEXT
      DIMENSION Q(JMAX,KMAX,LMAX,6),
     *          XX(Jmax,Kmax,Lmax),
     *          XY(Jmax,Kmax,Lmax),
     *          XZ(Jmax,Kmax,Lmax),
     *          YX(Jmax,Kmax,Lmax),
     *          YY(Jmax,Kmax,Lmax),
```

```
     *          YZ(Jmax,Kmax,Lmax),
     *          ZX(Jmax,Kmax,Lmax),
     *          ZY(Jmax,Kmax,Lmax),
     *          ZZ(Jmax,Kmax,Lmax)
\
C    OUTER BOUNDARY FLOW CONDITIONS
C    CAUTION: ONLY SATISFACTORY FOR INCOMPRESSIBLE FLOW CONDITIONS
      A0 = SQRT( TINF )
      R0 = GAMMA*PINF/TINF

C
      JP=J+ISGN
      SGN=ISGN
      DO 310 L=LA,LB
      DO 301 I=1,NM+1
       IBF(I)=0
  301 CONTINUE
C
      IF (ID.EQ.1) THEN
       DO 303 K=KA,KB
        ZNORM = SGN * SQRT(((XX(J,K,L)**2)+XY(J,K,L)**2+XZ(J,K,L)**2))
        DCX   = XX(J,K,L)/ZNORM
        DCY   = XY(J,K,L)/ZNORM
        DCZ   = XZ(J,K,L)/ZNORM
C    EXTRAPOLATE TO BOUNDARY
        RI    = Q(JP,K,L,1)*Q(JP,K,L,6)
        UI    = Q(JP,K,L,2)/Q(JP,K,L,1)
        VI    = Q(JP,K,L,3)/Q(JP,K,L,1)
        WI    = Q(JP,K,L,4)/Q(JP,K,L,1)
        EI    = Q(JP,K,L,5)*Q(JP,K,L,6)
        PI    = GAMI*( EI -0.5*RI*(UI**2 + VI**2 + WI**2) )
        UNI   = DCX*UI +DCY*VI +DCZ*WI
        AI    = SQRT( GAMMA*PI/RI )
C    OUTER BOUNDARY NORMAL VELOCITY
        CALL SETVEL(MB,ID,J,K,L,U0,V0,W0,CP0)
        U0=U0*XMACH
        V0=V0*XMACH
        W0=W0*XMACH
        P0=PINF+.5*CP0*XMACH
        E0 = PINF*GM1R + .5*R0*(U0**2 + V0**2 + W0**2)
ctmp
c       write(6,*)'*** bcfar1:',ID,j,jp,k,l,u0,v0,w0,p0
        UN0   = DCX*U0 +DCY*V0 +DCZ*W0
C    COMPUTE SPEED OF SOUND AND NORMAL VELOCITY COMPONENT
C    FROM RIEMANN INVARIANTS
        R1    = UN0 + (2.0*GM1R)*A0
        R2    = UNI - (2.0*GM1R)*AI
        A     = 0.25*(R1-R2)*GAMI
        UN    = 0.50*(R1+R2)
        XMN   = UN/A

        FNMACH(K)=XMN

  303  CONTINUE
      ELSE IF (ID.EQ.2) THEN
       DO 304 K=KA,KB
        ZNORM = SGN * SQRT(((YX(K,J,L)**2)+YY(K,J,L)**2+YZ(K,J,L)**2))
        DCX   = YX(K,J,L)/ZNORM
        DCY   = YY(K,J,L)/ZNORM
        DCZ   = YZ(K,J,L)/ZNORM
```

```
C     EXTRAPOLATE TO BOUNDARY
         RI     = Q(K,JP,L,1)*Q(K,JP,L,6)
         UI     = Q(K,JP,L,2)/Q(K,JP,L,1)
         VI     = Q(K,JP,L,3)/Q(K,JP,L,1)
         WI     = Q(K,JP,L,4)/Q(K,JP,L,1)
         EI     = Q(K,JP,L,5)*Q(K,JP,L,6)
         PI     = GAMI*( EI -0.5*RI*(UI**2 + VI**2 + WI**2) )
         UNI    = DCX*UI +DCY*VI +DCZ*WI
         AI     = SQRT( GAMMA*PI/RI )
C     OUTER BOUNDARY NORMAL VELOCITY
         CALL SETVEL(MB,ID,J,K,L,U0,V0,W0,CP0)
         U0=U0*XMACH
         V0=V0*XMACH
         W0=W0*XMACH
         P0=PINF+.5*CP0*XMACH
         E0 = PINF*GM1R + .5*R0*(U0**2 + V0**2 + W0**2)
ctmp
c        write(6,*)'*** bcfar1:',ID,j,jp,k,l,u0,v0,w0,p0
         UN0    = DCX*U0 +DCY*V0 +DCZ*W0
C     COMPUTE SPEED OF SOUND AND NORMAL VELOCITY COMPONENT
C     FROM RIEMANN INVARIANTS
         R1     = UN0 + (2.0*GM1R)*A0
         R2     = UNI - (2.0*GM1R)*AI
         A      = 0.25*(R1-R2)*GAMI
         UN     = 0.50*(R1+R2)
         XMN    = UN/A

         FNMACH(K)=XMN

  304  CONTINUE
       ELSE IF (ID.EQ.3) THEN
        DO 305 K=KA,KB
         ZNORM = SGN * SQRT(((ZX(K,L,J)**2)+ZY(K,L,J)**2+ZZ(K,L,J)**2))
         DCX    = ZX(K,L,J)/ZNORM
         DCY    = ZY(K,L,J)/ZNORM
         DCZ    = ZZ(K,L,J)/ZNORM
C     EXTRAPOLATE TO BOUNDARY
         RI     = Q(K,L,JP,1)*Q(K,L,JP,6)
         UI     = Q(K,L,JP,2)/Q(K,L,JP,1)
         VI     = Q(K,L,JP,3)/Q(K,L,JP,1)
         WI     = Q(K,L,JP,4)/Q(K,L,JP,1)
         EI     = Q(K,L,JP,5)*Q(K,L,JP,6)
         PI     = GAMI*( EI -0.5*RI*(UI**2 + VI**2 + WI**2) )
         UNI    = DCX*UI +DCY*VI +DCZ*WI
         AI     = SQRT( GAMMA*PI/RI )
C     OUTER BOUNDARY NORMAL VELOCITY
         CALL SETVEL(MB,ID,J,K,L,U0,V0,W0,CP0)
         U0=U0*XMACH
         V0=V0*XMACH
         W0=W0*XMACH
         P0=PINF+.5*CP0*XMACH
         E0 = PINF*GM1R + .5*R0*(U0**2 + V0**2 + W0**2)
ctmp
c        write(6,*)'*** bcfar1:',ID,j,jp,k,l,u0,v0,w0,p0
         UN0    = DCX*U0 +DCY*V0 +DCZ*W0
C     COMPUTE SPEED OF SOUND AND NORMAL VELOCITY COMPONENT
C     FROM RIEMANN INVARIANTS
         R1     = UN0 + (2.0*GM1R)*A0
         R2     = UNI - (2.0*GM1R)*AI
         A      = 0.25*(R1-R2)*GAMI
```

```
        UN     = 0.50*(R1+R2)
        XMN    = UN/A

        FNMACH(K)=XMN

  305  CONTINUE
      END IF
      DO 307 K=KA,KB
       IF (FNMACH(K).GT.1.0) THEN
        IBF(K)=1       !SUPERSONIC INFLOW
       ELSE IF (FNMACH(K).GT.0.0) THEN
        IBF(K)=2       !  SUBSONIC INFLOW
       ELSE IF (FNMACH(K).GT.-1.0) THEN
        IBF(K)=3       !  SUBSONIC OUTFLOW
       ELSE
        IBF(K)=4       !SUPERSONIC OUTFLOW
       END IF
  307 CONTINUE
      K1=KA
      IBFF=IBF(KA)
      DO 309 K=KA+1,KB+1
       IF (IBF(K).NE.IBFF) THEN
        K2=K-1
        IF (IBFF.EQ.1) THEN
         CALL FARFIX (ID,J,K1,K2,L, BIGA(IPWR) )
        ELSE IF (IBFF.EQ.2) THEN
         CALL FARIN (ID,J,JP,K1,K2,L,SGN,
     *              BIGA(IPWXX),BIGA(IPWXY),BIGA(IPWXZ),
     *              BIGA(IPWYX),BIGA(IPWYY),BIGA(IPWYZ),
     *              BIGA(IPWZX),BIGA(IPWZY),BIGA(IPWZZ),
     *              BIGA(IPWR )                          )
        ELSE IF (IBFF.EQ.3) THEN
         CALL FAROUT (ID,J,JP,K1,K2,L,SGN,
     *              BIGA(IPWXX),BIGA(IPWXY),BIGA(IPWXZ),
     *              BIGA(IPWYX),BIGA(IPWYY),BIGA(IPWYZ),
     *              BIGA(IPWZX),BIGA(IPWZY),BIGA(IPWZZ),
     *              BIGA(IPWR )                          )
        ELSE IF (IBFF.EQ.4) THEN
         CALL OUTSUP (ID,J,JP,K1,K2,L, BIGA(IPWR) )
        END IF
        K1=K
        IBFF=IBF(K)
       END IF
  309 CONTINUE
  310 CONTINUE
      END

      SUBROUTINE SETVEL(MB,ID,JPP,K,L,U0,V0,W0,CP0)
C***********************************************************************
cdan New routine added in December 1992 for VSAERO Vscan data.
C***********************************************************************
      include 'npami.inc'
      COMMON/SCAN/UO(NBX,JMX,KMX,LMX),VO(NBX,JMX,KMX,LMX),
     &            WO(NBX,JMX,KMX,LMX),CPO(NBX,JMX,KMX,LMX)

      U0=1001.
      IF(ID.EQ.1)THEN
        U0=UO(MB,JPP,K,L)
        V0=VO(MB,JPP,K,L)
        W0=WO(MB,JPP,K,L)
```

```
        CP0=CPO(MB,JPP,K,L)
      ELSEIF(ID.EQ.2)THEN
        U0=UO(MB,K,JPP,L)
        V0=VO(MB,K,JPP,L)
        W0=WO(MB,K,JPP,L)
        CP0=CPO(MB,K,JPP,L)
      ELSEIF(ID.EQ.3)THEN
        U0=UO(MB,K,L,JPP)
        V0=VO(MB,K,L,JPP)
        W0=WO(MB,K,L,JPP)
        CP0=CPO(MB,K,L,JPP)
      ENDIF
      IF(U0.GT.1000.)THEN
        WRITE(6,*)'*** ERROR... VELOCITY NOT SET ***'
        WRITE(6,*)' INDICES DO NOT MATCH SCAN DATA'
        WRITE(6,*)' MB, ID, JPP, K, L, ',MB,ID,JPP,K,L
        STOP
      ELSEIF(U0.GT.500.)THEN
        WRITE(6,*)'*** ERROR... VELOCITY NOT SET ***'
        WRITE(6,*)' SCAN VELOCITY NOT AVAILABLE'
        WRITE(6,*)' MB, ID, JPP, K, L, ',MB,ID,JPP,K,L
        STOP
      ELSE
        RETURN
      ENDIF
      END

      SUBROUTINE FORMOM
C*****************************************************************************
C     New Routine added for force & moment calc.                        7/93
C*****************************************************************************
C
      include 'npami.inc'
      PARAMETER (MAXB=4*NIP)
      COMMON/MBLOCK/NBLOCK,MB,MBNEXT
      COMMON/TIME/ DT,IVARDT
      COMMON/FORCE/CBAR,SSPAN,SREF,REFMX,REFMY,REFMZ
      COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
      COMMON/KEPS  / NRLX,NTURB,ORDER,SIGK,SIGEPS,CK1,CK2,CK3,CK4,CK5
      COMMON/INTEG/JL(MAXB),KL(MAXB),LL(MAXB)
      DIMENSION Q(JMX,KMX,LMX,5),X(JMX,KMX,LMX),
     1  Y(JMX,KMX,LMX),Z(JMX,KMX,LMX),XC(NIP),YC(NIP),ZC(NIP),
     2  CPP(NIP),AREA(NIP),ANT(NIP,3)
      NAMELIST/BOUNDS/ NJSEG,JLINE,JKLOW,JKHIGH,JLLOW,JLHIGH,
     *                 NKSEG,KLINE,KJLOW,KJHIGH,KLLOW,KLHIGH,
     *                 NLSEG,LLINE,LJLOW,LJHIGH,LKLOW,LKHIGH,
     *                 JTYPE,JSIGN,PRESSJ,TEMPJ,INTERJ,
     *                 KTYPE,KSIGN,PRESSK,TEMPK,INTERK,
     *                 LTYPE,LSIGN,PRESSL,TEMPL,INTERL,
     *                 JEDGE,KEDGE,LEDGE,JDIR,KDIR,LDIR,
     *                 NORMJ,NORMK,NORML
      NTI = 5
      REWIND NTI
      NBPT = 0
      NPAT = 0

      WRITE(6,800)
      WRITE(6,801)CBAR,SREF,SSPAN,REFMX,REFMY,REFMZ,XMACH,ALPHA,BETA
C
C     RETRIEVE THE 3D SOLUTION DATA
```

```
C
      REWIND 4
      READ(4)NC,GAMMA
      DO 50 NB=1,NBLOCK
      READ(4)JL(NB),KL(NB),LL(NB)
      READ(4)
     * (((X(J,K,L),
     *  J=1,JL(NB)),K=1,KL(NB)),L=1,LL(NB)),
     * (((Y(J,K,L),
     *  J=1,JL(NB)),K=1,KL(NB)),L=1,LL(NB)),
     * (((Z(J,K,L),
     *  J=1,JL(NB)),K=1,KL(NB)),L=1,LL(NB))

      READ(4)((((Q(J,K,L,N), J=1,JL(NB)),K=1,KL(NB)),
     *              L=1,LL(NB)),N=1,5)
      IF(NC.ge.NTURB)
     *      READ(4) (BIGA(J),J=IPWAK ,IPWAK +NXYZ-1),
     *              (BIGA(J),J=IPWEPS,IPWEPS+NXYZ-1),
     *              (BIGA(J),J=IPWTMU,IPWTMU+NXYZ-1)

      READ(NTI,BOUNDS)        ! Cray version of NAMELIST
c     READ(NTI,NML=BOUNDS)    ! IRIS version of NAMELIST
      DO 26 J=1,NJSEG
      IF(JTYPE(J).EQ.60.OR.JTYPE(J).EQ.61)THEN
        IORDR=1
        NPAT=NPAT+1
        CALL PATCH(NORMJ(J),IORDR,JKLOW(J),JKHIGH(J),JLLOW(J),
     1                    JLHIGH(J),JLINE(J),X,Y,Z,AREA,ANT,XC,YC,ZC)
        CALL CPCALC(IORDR,JKLOW(J),JKHIGH(J),JLLOW(J),JLHIGH(J),
     1                              JLINE(J),Q,NBP,CPP)
        NBPT = NBPT + NBP
        CALL ANLIZ(NPAT,NBP,NBPT,AREA,ANT,CPP,XC,YC,ZC)
      ENDIF
   26 CONTINUE

      DO 27 K=1,NKSEG
      IF(KTYPE(K).EQ.60.OR.KTYPE(K).EQ.61)THEN
        IORDR=2
        NPAT=NPAT+1
        CALL PATCH(NORMK(K),IORDR,KJLOW(K),KJHIGH(K),KLLOW(K),
     1                    KLHIGH(K),KLINE(K),X,Y,Z,AREA,ANT,XC,YC,ZC)
        CALL CPCALC(IORDR,KJLOW(K),KJHIGH(K),KLLOW(K),KLHIGH(K),
     1                              KLINE(K),Q,NBP,CPP)
        NBPT = NBPT + NBP
        CALL ANLIZ(NPAT,NBP,NBPT,AREA,ANT,CPP,XC,YC,ZC)
      ENDIF
   27 CONTINUE

      DO 28 L=1,NLSEG
      IF(LTYPE(L).EQ.60.OR.LTYPE(L).EQ.61)THEN
        IORDR=3
        NPAT=NPAT+1
        CALL PATCH(NORMI(L),IORDR,LJLOW(L),LJHIGH(L),LKLOW(L),
     1                    LKHIGH(L),LLINE(L),X,Y,Z,AREA,ANT,XC,YC,ZC)
        CALL CPCALC(IORDR,LJLOW(L),LJHIGH(L),LKLOW(L),LKHIGH(L),
     1                              LLINE(L),Q,NBP,CPP)
        NBPT = NBPT + NBP
        CALL ANLIZ(NPAT,NBP,NBPT,AREA,ANT,CPP,XC,YC,ZC)
      ENDIF
   28 CONTINUE
```

```
   50 CONTINUE
        CALL ANLIZ(-NPAT,NBP,NBPT,AREA,ANT,CPP,XC,YC,ZC)
C
  700 FORMAT(1H1,//,5X,'CURRENT SCALING PARAMETERS',
     1 ' FOR FORCE AND MOMENT SUMMARY',/,5X,'BY PRESSURE INTEGRATION:')
  801 FORMAT(//,5X,'      CBAR       SREF      SSPAN      REFMX      REFMY',
     1 '      REFMZ',/,5X,6F10.3,/,5X, '      RMACH      ALDEG    YAWDEG',/,
     2   5X,3F10.3)
      RETURN
      END

      SUBROUTINE PATCH(NRML,IORDR,JF,JL,KF,KL,LVL,X,Y,Z,AREA,
     *                                                 ANT,XC,YC,ZC)
C*************************************************************************
cdan Routine added for force & moment calc.                      7/93
C*************************************************************************
C
      include 'npami.inc'
      DIMENSION X(JMX,KMX,LMX),Y(JMX,KMX,LMX),Z(JMX,KMX,LMX),
     1  XC(NIP),YC(NIP),ZC(NIP),CPP(NIP),AREA(NIP),ANT(NIP,3)
      EPSS = 1.E-15

C
C  PATCH GEOMETRY

      NP=0
      DO 100 K=KF,KL-1
      DO 100 J=JF,JL-1
      NP=NP+1
      IF(IORDR.EQ.1)THEN
        X1 = X(LVL,J,K)
        Y1 = Y(LVL,J,K)
        Z1 = Z(LVL,J,K)
        X2 = X(LVL,J+1,K)
        Y2 = Y(LVL,J+1,K)
        Z2 = Z(LVL,J+1,K)
        X3 = X(LVL,J+1,K+1)
        Y3 = Y(LVL,J+1,K+1)
        Z3 = Z(LVL,J+1,K+1)
        X4 = X(LVL,J,K+1)
        Y4 = Y(LVL,J,K+1)
        Z4 = Z(LVL,J,K+1)
      ELSEIF(IORDR.EQ.2)THEN
        X1 = X(J,LVL,K)
        Y1 = Y(J,LVL,K)
        Z1 = Z(J,LVL,K)
        X2 = X(J+1,LVL,K)
        Y2 = Y(J+1,LVL,K)
        Z2 = Z(J+1,LVL,K)
        X3 = X(J+1,LVL,K+1)
        Y3 = Y(J+1,LVL,K+1)
        Z3 = Z(J+1,LVL,K+1)
        X4 = X(J,LVL,K+1)
        Y4 = Y(J,LVL,K+1)
        Z4 = Z(J,LVL,K+1)
      ELSEIF(IORDR.EQ.3)THEN
        X1 = X(J,K,LVL)
        Y1 = Y(J,K,LVL)
        Z1 = Z(J,K,LVL)
        X2 = X(J+1,K,LVL)
```

```
        Y2 = Y(J+1,K,LVL)
        Z2 = Z(J+1,K,LVL)
        X3 = X(J+1,K+1,LVL)
        Y3 = Y(J+1,K+1,LVL)
        Z3 = Z(J+1,K+1,LVL)
        X4 = X(J,K+1,LVL)
        Y4 = Y(J,K+1,LVL)
        Z4 = Z(J,K+1,LVL)
      ENDIF

      XC(NP)=.25*(X1+X2+X3+X4)
      YC(NP)=.25*(Y1+Y2+Y3+Y4)
      ZC(NP)=.25*(Z1+Z2+Z3+Z4)
      XP=.25*(X2+X3-X4-X1)
      YP=.25*(Y2+Y3-Y4-Y1)
      ZP=.25*(Z2+Z3-Z4-Z1)
      SMP=SQRT(XP*XP+YP*YP+ZP*ZP)
      XQ=.25*(X3+X4-X1-X2)
      YQ=.25*(Y3+Y4-Y1-Y2)
      ZQ=.25*(Z3+Z4-Z1-Z2)
      SMQ=SQRT(XQ*XQ+YQ*YQ+ZQ*ZQ)
      PVQX=YP*ZQ-ZP*YQ
      PVQY=ZP*XQ-XP*ZQ
      PVQZ=XP*YQ-YP*XQ
      AR=SQRT(PVQX*PVQX+PVQY*PVQY+PVQZ*PVQZ)
      IF(AR.LT.EPSS)AR=EPSS
      AREA(NP)=AR*4.0
      ANT(NP,1)=PVQX/AR
      ANT(NP,2)=PVQY/AR
      ANT(NP,3)=PVQZ/AR
      IF(NRML.LT.0)THEN
        ANT(NP,1)=-PVQX/AR
        ANT(NP,2)=-PVQY/AR
        ANT(NP,3)=-PVQZ/AR
      ENDIF
  100 CONTINUE

      RETURN
      END

      SUBROUTINE CPCALC(IORDR,JF,JL,KF,KL,LVL,Q,NP,CPP)
C***********************************************************************
C
C
C***********************************************************************
C
      include 'npami.inc'
      COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
      DIMENSION Q(JMX,KMX,LMX,5),CPP(NIP)

      GD     = GAMMA*GAMI
      FSMACH = XMACH
      CPS    = 1./(.5*GAMMA*FSMACH**2)
      NP     = 0

   COMPUTE CP AT 4 'PANEL' CORNERS AND AVERAGE

      DO  5 K=KF,KL-1
      DO  5 J=JF,JL-1
      IF(IORDR.EQ.1)THEN
```

```
        CALL CORNR(LVL,J,K,CPS,GD,Q,CP1)
        CALL CORNR(LVL,J+1,K,CPS,GD,Q,CP2)
        CALL CORNR(LVL,J+1,K+1,CPS,GD,Q,CP3)
        CALL CORNR(LVL,J,K+1,CPS,GD,Q,CP3)
      ELSEIF(IORDR.EQ.2)THEN
        CALL CORNR(J,LVL,K,CPS,GD,Q,CP1)
        CALL CORNR(J+1,LVL,K,CPS,GD,Q,CP2)
        CALL CORNR(J+1,LVL,K+1,CPS,GD,Q,CP3)
        CALL CORNR(J,LVL,K+1,CPS,GD,Q,CP4)
      ELSEIF(IORDR.EQ.3)THEN
        CALL CORNR(J,K,LVL,CPS,GD,Q,CP1)
        CALL CORNR(J+1,K,LVL,CPS,GD,Q,CP2)
        CALL CORNR(J+1,K+1,LVL,CPS,GD,Q,CP3)
        CALL CORNR(J,K+1,LVL,CPS,GD,Q,CP4)
      ENDIF
      NP          = NP + 1
      CPP(NP)     = 0.25*(CP1+CP2+CP3+CP4)
    5 CONTINUE

      RETURN
      END

      SUBROUTINE CORNR(J,K,L,CPS,GD,Q,CPP)
      include 'npami.inc'
      DIMENSION Q(JMX,KMX,LMX,5)
      RR          = 1./Q(J,K,L,1)
      Q1          = Q(J,K,L,1)
      Q2          = Q(J,K,L,2)*RR
      Q3          = Q(J,K,L,3)*RR
      Q4          = Q(J,K,L,4)*RR
      Q5          = Q(J,K,L,5)/GAMMA
      PP          =          GD*(Q5-.5*(Q2**2+Q3**2+Q4**2)*Q1)
      CPP         = (PP-1.)*CPS

      RETURN
      END

      SUBROUTINE ANLIZ(JP,NBP,NBPT,AREA,ANT,CPP,XC,YC,ZC)
      include 'npami.inc'
      PARAMETER (MAXB=4*NIP)
      COMMON/FORCE/CBAR,SSPAN,SREF,REFMX,REFMY,REFMZ
      COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
      COMMON/SV/ WIND(3,3),SUMFX(MBC),SUMFY(MBC),SUMFZ(MBC),
     1 SUMMX(MBC),SUMMY(MBC),SUMMZ(MBC)
      DIMENSION CPP(NIP),AREA(NIP),ANT(NIP,3),XC(NIP),YC(NIP),ZC(NIP)

      EPS = 1.E-05
      CPMIN = -10.
      CPMAX = 1.20
      JOUT = 6
      MODUP = 0
      MODWN = 0

      IF(JP.GT.0)THEN
        WRITE(JOUT,620)JP,NBPT-NBP+1,NBPT
      ENDIF
      IF(JP.EQ.1)THEN
C
C  SET UP WIND AXIS SYSTEM
C
```

```
        YAWR=BETA/57.29577951
        ALR=(ALPHA)/57.29577951
        CB=COS(YAWR)
        CA=CB*COS(ALR)
        SA=CB*SIN(ALR)
        UXR=CA
        UYR=-SIN(YAWR)
        UZR=SA
        WIND(1,1)=UXR
        WIND(2,1)=UYR
        WIND(3,1)=UZR
        WIND(1,2)=0.0
        WIND(2,2)=1.0
        WIND(3,2)=0.0
        WIND(1,3)=-1.0
        WIND(2,3)=0.0
        WIND(3,3)=0.0
        IF(UZR.LT.0.0)WIND(1,3)=1.0
        US=SQRT(UXR*UXR+UYR*UYR)
        IF(US.GE.EPS)THEN
          WIND(1,2)=-UYR/US
          WIND(2,2)=UXR/US
          WIND(1,3)=-UXR*UZR/US
          WIND(2,3)=-UYR*UZR/US
          WIND(3,3)=US
        ENDIF
      ELSEIF(JP.LT.0)THEN
        WRITE(JOUT,621)1,NBPT
        JP=-JP
        SFX=0.0
        SFY=0.0
        SFZ=0.0
        SMX=0.0
        SMY=0.0
        SMZ=0.0
        DO 100 J=1,JP
        SFX=SFX+SUMFX(J)/SREF
        SFY=SFY+SUMFY(J)/SREF
        SFZ=SFZ+SUMFZ(J)/SREF
        SMX=SMX+SUMMX(J)/SREF
        SMY=SMY+SUMMY(J)/SREF
        SMZ=SMZ+SUMMZ(J)/SREF
  100   CONTINUE
        CD=SFX*WIND(1,1)+SFY*WIND(2,1)+SFZ*WIND(3,1)
        CS=SFX*WIND(1,2)+SFY*WIND(2,2)+SFZ*WIND(3,2)
        CL=SFX*WIND(1,3)+SFY*WIND(2,3)+SFZ*WIND(3,3)
        CLMX=(SMX*WIND(1,1)+SMY*WIND(2,1)+SMZ*WIND(3,1))/SSPAN
        CLMY=(SMX*WIND(1,2)+SMY*WIND(2,2)+SMZ*WIND(3,2))/CBAR
        CLMZ=(SMX*WIND(1,3)+SMY*WIND(2,3)+SMZ*WIND(3,3))/SSPAN
        WRITE(JOUT,629) CD,CS,CL,CLMX,CLMY,CLMZ
        SMX=SMX/SSPAN
        SMY=SMY/CBAR
        SMZ=SMZ/SSPAN
        WRITE(JOUT,613)SFX,SFY,SFZ,SMX,SMY,SMZ
        RETURN
      ENDIF
C
C  LOADS SUMMATION
C
      SUMFX(JP)=0.0
```

```
      SUMFY(JP)=0.0
      SUMFZ(JP)=0.0
      SUMMX(JP)=0.0
      SUMMY(JP)=0.0
      SUMMZ(JP)=0.0

      DO 200 I=1,NBP
      XCM=XC(I)-REFMX
      YCM=YC(I)-REFMY
      ZCM=ZC(I)-REFMZ
      CPA=CPP(I)

      IF(CPP(I).LT.CPMIN)THEN
       MODWN=MODWN+1
       CPA=CPMIN
      ELSEIF(CPP(I).GT.CPMAX)THEN
       MODUP=MODUP+1
       CPA=CPMAX
      ENDIF

      DFX=-CPA*AREA(I)*ANT(I,1)
      DFY=-CPA*AREA(I)*ANT(I,2)
      DFZ=-CPA*AREA(I)*ANT(I,3)

      SUMFX(JP)=SUMFX(JP)+DFX
      SUMFY(JP)=SUMFY(JP)+DFY
      SUMFZ(JP)=SUMFZ(JP)+DFZ
      SUMMX(JP)=SUMMX(JP)+(YCM*DFZ-ZCM*DFY)
      SUMMY(JP)=SUMMY(JP)+(ZCM*DFX-XCM*DFZ)
      SUMMZ(JP)=SUMMZ(JP)+(XCM*DFY-YCM*DFX)
.00   CONTINUE
      WRITE(JOUT,699) CPMAX,MODUP,CPMIN,MODWN
      SFX=SUMFX(JP)/SREF
      SFY=SUMFY(JP)/SREF
      SFZ=SUMFZ(JP)/SREF
      SMX=SUMMX(JP)/SREF
      SMY=SUMMY(JP)/SREF
      SMZ=SUMMZ(JP)/SREF
      CD=SFX*WIND(1,1)+SFY*WIND(2,1)+SFZ*WIND(3,1)
      CS=SFX*WIND(1,2)+SFY*WIND(2,2)+SFZ*WIND(3,2)
      CL=SFX*WIND(1,3)+SFY*WIND(2,3)+SFZ*WIND(3,3)
      CLMX=(SMX*WIND(1,1)+SMY*WIND(2,1)+SMZ*WIND(3,1))/SSPAN
      CLMY=(SMX*WIND(1,2)+SMY*WIND(2,2)+SMZ*WIND(3,2))/CBAR
      CLMZ=(SMX*WIND(1,3)+SMY*WIND(2,3)+SMZ*WIND(3,3))/SSPAN
      WRITE(JOUT,629) CD,CS,CL,CLMX,CLMY,CLMZ
      SMX=SMX/SSPAN
      SMY=SMY/CBAR
      SMZ=SMZ/SSPAN
      WRITE(JOUT,613)SFX,SFY,SFZ,SMX,SMY,SMZ

      RETURN

605   FORMAT(//,5X,'INVALID WIND DIRECTION, PLEASE RE-CHECK')
613   FORMAT(1H ,5X,20HIN BODY AXIS SYSTEM//9X,3HCFX,10X,3HCFY,10X,
     13HCFZ,10X,3HCMX,10X,3HCMY,10X,3HCMZ/1H ,1X,6G13.4/,
     21H ,79(1H*)//)
620   FORMAT(1H0,70(1H*)/1H0,
     142H FORCE AND MOMENT FOR SPECIFIED PANEL SET,I2,
     2 ' ( PANELS ',I5,' - ',I5')',/,
     347H(NOTE: THE COEFFICIENTS HERE DO NOT INCLUDE THE,
```

```
   412H IMAGE SET.)//)
621 FORMAT(1H0,50(1H*)/1H0,
   123H TOTAL FORCE AND MOMENT,
   2 ' ( PANELS ',I5,' - ',I5')',//,
   247H(NOTE: THE COEFFICIENTS HERE DO NOT INCLUDE THE,
   312H IMAGE SET.)//)
629 FORMAT(1H0,
   118H FORCE AND MOMENT//,
   21H ,8X,2HCD,11X,2HCS,11X,2HCL,10X,3HCMX,10X,3HCMY,10X,3HCMZ/
   31H ,1X,6G13.4/)
699 FORMAT(8H CPMAX= ,F6.1,18H PANELS MODIFIED= ,I5,/,8H CPMIN= ,F6.1,
   1            18H PANELS MODIFIED= ,I5)
    END
```

APPENDIX B

Program PRPAMI was developed to aid the user in pre and post-processing of data required by the zonal aerodynamic procedure developed under NAS3-26310. This is an interactive program written in FORTRAN with the specific application in mind. When practical, generality in the code logic was preserved, but future problems with different topological features may require software modification. Following is a brief description of the code features taken from the self-documentation included in the code. The variable IOPT is defined by the user.

IOPT	FEATURE	FILES REQUIRED	OUTPUT FILE
1	GENERATE NPARC RESTART DATA FILE	'_____.VOL'	REST.OLD
2	GENERATE VSAERO SCAN INPUT DATA	INON5,REST.OLD	SCAN.DAT
3	GENERATE OMNI3D GRAPHICS DATA FILE	INON5, REST.OLD	PARC.FMT
4	REORDER GRID BLOCKS	'_____.VOL'	'_____.VOL2'

Note: Filenames in single quotes are provided by user.

```
      PROGRAM PRPAMI
C*********************************************************************
C  MARCH 1994
C
   JTILITY PROGRAM FOR DATA FILE MANIPULATION DEVELOPED BY
C  D.J. STRASH AT ANALYTICAL METHODS, INC. UNDER OPTION 2 OF
C  NASA LEWIS CONTRACT NUMBER NAS3-26310.
C
C  ????QUESTIONS????       CALL DAN STRASH @ (206)643-9090
C                          OR E-MAIL          dan@amiwest.com
C*********************************************************************
C
C  IOPT   FEATURE                      FILES REQUIRED     OUTPUT FILE
C---------------------------------------------------------------------
C   1     GENERATE NPARC RESTART       '_____.VOL'        REST.OLD
C          DATA FILE
C   2     GENERATE VSAERO SCAN         INON5, REST.OLD    SCAN.DAT
C          INPUT DATA
C   3     GENERATE OMNI3D GRAPHICS     INON5, REST.OLD    PARC.FMT
C          DATA FILE
C   4     REORDER GRID BLOCKS          '_____.VOL'        '_____.VOL2'
C
C  (NOTE: FILENAMES IN SINGLE QUOTES ARE PROVIDED BY USER.)
C
C*********************************************************************
C
      CHARACTER *2 GRD
    5 CONTINUE
      WRITE(*,*)
      WRITE(*,*)
      WRITE(*,*)' YOU HAVE THE FOLLOWING OPTIONS...'
      WRITE(*,*)
      WRITE(*,*)'  1. PREPARE PARC3D RESTART FILE'
      WRITE(*,*)'  2. PREPARE VSAERO SCAN INPUT FILE'
      WRITE(*,*)'  3. PREPARE OMNI3D PLOT FILE'
      WRITE(*,*)'  4. REORDER BLOCKS (.VOL FILE)'
      WRITE(*,*)
      WRITE(*,*)' PLEASE TYPE NUMBER OF DESIRED OPTION...'
      READ(*,*)IOPT

      IF(IOPT.EQ.1)THEN
        CALL OPNRES
        CALL RVOL
        CALL PAERO
        WRITE(*,*)
        WRITE(*,*)' WHAT IS THE GRID TOPOLOGY?'
        WRITE(*,*)' (CO - FOR C-O TYPE, HO - FOR H-O TYPE)'
        WRITE(*,*)
        READ(*,80)GRD
        IF(GRD(1:1).EQ.'C')THEN
          CALL COGRD
        ELSE
          CALL HOGRD
        ENDIF
      ELSEIF(IOPT.EQ.2)THEN
        CALL OPNSCN
        CALL READRES
        CALL SCNGEN
      ELSEIF(IOPT.EQ.3)THEN
        CALL OPNPLT
```

```
        CALL READRES
        CALL PLTGEN
      ELSEIF(IOPT.EQ.4)THEN
        CALL OPNRDR
        CALL REORDR
      ELSEIF(IOPT.LT.0)THEN
        STOP
      ELSE
        WRITE(*,*)' UNSUPPORTED OPTION, TRY AGAIN'
        GO TO 5
      ENDIF
      STOP
   80 FORMAT(A2)
      END

      SUBROUTINE OPNRES
      CHARACTER*80 FILUN, FILOT, FILIN
      DATA FILOT/'REST.OLD'/

      WRITE(*,*)' INPUT NAME OF GRIDGEN VOLUME GRID FILE.'
      READ(*,80)FILIN
      OPEN(UNIT=8,FILE=FILIN,FORM='UNFORMATTED',STATUS='OLD',err=9001)
      OPEN(UNIT=9,FILE=FILOT,FORM='UNFORMATTED',STATUS='UNKNOWN',
     1  err=9002)

      REWIND 8
      REWIND 9
      RETURN
 9001 WRITE(*,*)' UNABLE TO OPEN ',FILIN
      STOP
 9002 WRITE(*,*)' UNABLE TO OPEN REST.OLD'
      STOP
   80 FORMAT(A80)
      END

      SUBROUTINE OPNSCN
      CHARACTER*80 FILUN, FILOT, FILIN
      DATA FILIN/'INON5'/, FILUN/'REST.OLD'/, FILOT/'SCAN.DAT'/

      OPEN(UNIT=7,FILE=FILIN,FORM='FORMATTED',STATUS='OLD',err=9001)
      OPEN(UNIT=8,FILE=FILUN,FORM='UNFORMATTED',STATUS='OLD',err=9002)
      OPEN(UNIT=9,FILE=FILOT,FORM='FORMATTED',STATUS='UNKNOWN',
     1  err=9003)

      REWIND 7
      REWIND 8
      REWIND 9
      RETURN
 9001 WRITE(*,*)' UNABLE TO OPEN INON5'
      STOP
 9002 WRITE(*,*)' UNABLE TO OPEN REST.OLD'
      STOP
 9003 WRITE(*,*)' UNABLE TO OPEN SCAN.DAT'
      STOP
      END

      SUBROUTINE OPNPLT
      CHARACTER*80 FILUN, FILOT, FILIN
      DATA FILIN/'INON5'/, FILUN/'REST.OLD'/, FILOT/'PARC.FMT'/
```

```
      OPEN(UNIT=7,FILE=FILIN,FORM='FORMATTED',STATUS='OLD',err=9001)
      OPEN(UNIT=8,FILE=FILUN,FORM='UNFORMATTED',STATUS='OLD',err=9002)
      OPEN(UNIT=9,FILE=FILOT,FORM='FORMATTED',STATUS='UNKNOWN',err=9003)

      REWIND 7
      REWIND 8
      REWIND 9
      RETURN
 9001 WRITE(*,*)' UNABLE TO OPEN INON5'
      STOP
 9002 WRITE(*,*)' UNABLE TO OPEN REST.OLD'
      STOP
 9003 WRITE(*,*)' UNABLE TO OPEN PARC.FMT'
      STOP
      END

      SUBROUTINE OPNRDR
      CHARACTER*80 FILUN, FILOT, FILIN
      DATA FILOT/'REST.OLD'/

      WRITE(*,*)' INPUT NAME OF GRIDGEN VOLUME GRID FILE.'
      READ(*,80)FILIN
      OPEN(UNIT=8,FILE=FILIN,FORM='UNFORMATTED',STATUS='OLD',err=9001)
      WRITE(*,*)' INPUT NAME OF MODIFIED GRIDGEN VOLUME GRID FILE.'
      READ(*,80)FILOT
      OPEN(UNIT=9,FILE=FILOT,FORM='UNFORMATTED',STATUS='UNKNOWN',
     1  err=9002)

      REWIND 8
      REWIND 9
      RETURN
 9001 WRITE(*,*)' UNABLE TO OPEN ',FILIN
      STOP
 9002 WRITE(*,*)' UNABLE TO OPEN ',FILOT
      STOP
   80 FORMAT(A80)
      END

      subroutine rvol
c.....Read the VOLGRID file.

      PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
      common /grid/ x(jmx,kmx,lmx,NTB),y(jmx,kmx,lmx,NTB),
     &  z(jmx,kmx,lmx,NTB),jm(NTB),km(NTB),lm(NTB),NBLOCKS,NBB,SYM

      io3di = 8

c.....Read the volgrid file in gridgen3d format
c.....WITHOUT the ghost cells.

         read(io3di,end=9010,err=9020) nblocks
         IF(NBLOCKS.GT.NTB)THEN
           write(*,1031) nblocks
         ELSE
           write(*,1030) nblocks
         ENDIF

         do 100 mb = 1, nblocks

         read(io3di) jm(mb), km(mb), lm(mb)
```

```
          write(*,1040) mb,jm(mb),km(mb),lm(mb)

          read(io3di,end=9010,err=9020)  ( ( ( x(i,j,k,mb),
     &                                        i = 1, jm(mb) ),
     &                                     j = 1, km(mb) ),
     &                                   k = 1, lm(mb) ),
     &                                  ( ( ( y(i,j,k,mb),
     &                                        i = 1, jm(mb) ),
     &                                     j = 1, km(mb) ),
     &                                   k = 1, lm(mb) ),
     &                                  ( ( ( z(i,j,k,mb),
     &                                        i = 1, jm(mb) ),
     &                                     j = 1, km(mb) ),
     &                                   k = 1, lm(mb) )

  100 continue


      close(unit=io3di)
      return

 9010 write(*,1010)
      stop

 9020 write(*,1020)
      stop

 1010 format(/,5X,'ERROR: END OF FILE READING VOLGRID FILE',
     &       /)
  `20 format(/,5X,'ERROR: ERROR READING VOLGRID FILE',
     &       /)
 1030 format(/,5X,' NUMBER OF BLOCKS IN VOLGRID FILE...',i3,
     &       /)
 1031 format(/,5X,' ERROR READING VOLGRID FILE...',/,
     & 5X,' INCREASE BLOCK DIMENSIONS (NTB) TO ',i3,
     &       /)
 1040 format(/,5X,' NO. OF PTS IN BLOCK ',i3,' OF VOLGRID FILE',
     &       /,5X,'       VOLGRID - ',i3,' X ',i3,' X ',i3,
     &       /)
      end

      subroutine reordr
c.....Read the VOLGRID file.

      PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
      common /grid/ x(jmx,kmx,lmx,NTB),y(jmx,kmx,lmx,NTB),
     &  z(jmx,kmx,lmx,NTB),jm(NTB),km(NTB),lm(NTB),NBLOCKS,NBB,SYM
      DIMENSION IB(NTB)

      io3di = 8
      io3do = 9

c.....Read the volgrid file in gridgen3d format
c.....WITHOUT the ghost cells.

          read(io3di,end=9010,err=9020) nblocks
          write(io3do) nblocks
          write(*,1030) nblocks

          do 100 mb = 1, nblocks
```

```
      read(io3di) jm(mb), km(mb), lm(mb)
      write(*,1040) mb,jm(mb),km(mb),lm(mb)

      read(io3di,end=9010,err=9020)  ( ( ( x(i,j,k,mb),
     &                                     i = 1, jm(mb) ),
     &                                   j = 1, km(mb) ),
     &                                 k = 1, lm(mb) ),
     &                                ( ( ( y(i,j,k,mb),
     &                                     i = 1, jm(mb) ),
     &                                   j = 1, km(mb) ),
     &                                 k = 1, lm(mb) ),
     &                                ( ( ( z(i,j,k,mb),
     &                                     i = 1, jm(mb) ),
     &                                   j = 1, km(mb) ),
     &                                 k = 1, lm(mb) )

100 continue

    write(*,*)
    write(*,*)' WOULD YOU LIKE TO SCALE AND TRANSLATE GRID?'
    write(*,*)'  (1 = YES, 0 = NO)'
    read(*,*) iopt
    if(iopt.ne.0)then
  5   write(*,*)
      write(*,*)' INPUT REFERENCE LENGTH, XT, YT, ZT'
      write(*,*)
      read(*,*) refl,xt,yt,zt
      if(abs(refl).lt.1.e-07)then
        write(*,*)
        write(*,*)' TRY AGAIN...'
        go to 5
      endif
      do 150 mm=1,nblocks
      do 155 j=1,jm(mm)
      do 155 k=1,km(mm)
      do 155 l=1,lm(mm)
      x(j,k,l,mm)=x(j,k,l,mm)/refl + xt
      y(j,k,l,mm)=y(j,k,l,mm)/refl + yt
      z(j,k,l,mm)=z(j,k,l,mm)/refl + zt
155   continue
150   continue
    endif
    write(*,*)
    write(*,*)' PLEASE TYPE IN NEW BLOCK ORDER, '
    write(*,*)'  (e.g. 1,3,2).'
    write(*,*)
    read(*,*)(ib(n),n=1,nblocks)

      do 200 mm = 1, nblocks

      mb = ib(mm)
      write(io3do) jm(mb), km(mb), lm(mb)
      write(*,1040) mb,jm(mb),km(mb),lm(mb)

      write(io3do)                    ( ( ( x(i,j,k,mb),
     &                                     i = 1, jm(mb) ),
     &                                   j = 1, km(mb) ),
     &                                 k = 1, lm(mb) ),
     &                                ( ( ( y(i,j,k,mb),
```

```
     &                                          i = 1, jm(mb) ),
     &                                        j = 1, km(mb) ),
     &                                      k = 1, lm(mb) ),
     &                                    ( ( ( z(i,j,k,mb),
     &                                          i = 1, jm(mb) ),
     &                                        j = 1, km(mb) ),
     &                                      k = 1, lm(mb) )

  200 continue


      close(unit=io3di)
      close(unit=io3do)
      return

 9010 write(*,1010)
      stop

 9020 write(*,1020)
      stop

 1010 format(/,5X,'ERROR: END OF FILE READING VOLGRID FILE',
     &       /)
 1020 format(/,5X,'ERROR: ERROR READING VOLGRID FILE',
     &       /)
 1030 format(/,5X,' NUMBER OF BLOCKS IN VOLGRID FILE...',i3,
     &       /)
 1040 format(/,5X,' NO. OF PTS IN BLOCK ',i3,' OF VOLGRID FILE',
     &       /,5X,'       VOLGRID - ',i3,' X ',i3,' X ',i3,
     &       /)
      end

      SUBROUTINE PAERO
      PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
      common /grid/ x(jmx,kmx,lmx,NTB),y(jmx,kmx,lmx,NTB),
     &  z(jmx,kmx,lmx,NTB),jm(NTB),km(NTB),lm(NTB),NBLOCKS,NBB,SYM
      COMMON/AERO/ R(jmx,kmx,lmx),RU(jmx,kmx,lmx),RV(jmx,kmx,lmx),
     1 RW(jmx,kmx,lmx),E(jmx,kmx,lmx),ALPHA,BETA,FMACH,CBAR
      CHARACTER *1 GOM
      LOGICAL SYM
      DATA G/1.4/, GM1/0.4/, SYM/.FALSE./

      WRITE(*,*)
      WRITE(*,*)' INPUT ALPHA, BETA, MACH NUMBER AND CBAR...'
      WRITE(*,*)' (CBAR USED TO SCALE GEOMETRY)'
      WRITE(*,*)
      READ(*,*)ALDEG,YAWDEG,FMACH,CBAR
      IF(ABS(YAWDEG).GE.0.1)THEN
        WRITE(*,*)
        WRITE(*,*)' SHOULD THE GEOMETRY BE MIRRORED ACROSS X-Z PLANE?'
        WRITE(*,*)'  (T-TRUE, F-FALSE)'
        WRITE(*,*)
        READ(*,80)GOM
        IF(GOM(1:1).EQ.'T')SYM=.TRUE.
      ENDIF
      IF(FMACH.LT.0.1)FMACH=0.1
      NBB=NBLOCKS
      IF(SYM)NBB=2*NBLOCKS
      IF(NBLOCKS.GT.NTB)THEN
```

```
        write(*,1031) nblocks
      ENDIF

C   FORM THE ARRAYS OF NON-DIMENSIONAL CONSERVATION VARIABLES
C   CONSISTENT WITH THE FREE-STREAM MACH NUMBER

      YAWR=YAWDEG/57.29577951
      ALR = ALDEG/57.29577951
      CB=COS(YAWR)
      CA=CB*COS(ALR)
      SA=CB*SIN(ALR)
      FACT=(1.+.2*FMACH**2)
      PBAR=FACT**(-3.5)/G
      RHOB=FACT**(-2.5)
      RHOU=RHOB*FMACH*SQRT(1./FACT)
      EBAR=PBAR/GM1+.5*(RHOU**2)/RHOB
      RHOU=RHOU*CA
      RHOV=-RHOU*SIN(YAWR)
      RHOW=RHOU*SA

      DO 4 L=1,LMX
      DO 4 K=1,KMX
      DO 4 J=1,JMX
      R (J,K,L)=RHOB
      RU(J,K,L)=RHOU
      RV(J,K,L)=RHOV
      RW(J,K,L)=RHOW
      E (J,K,L)=EBAR
/     CONTINUE
      return
   80 FORMAT(A1)
   81 FORMAT(F10.0)
 1031 format(/,5X,' ERROR READING VOLGRID FILE...',/,
     & 5X,' INCREASE BLOCK DIMENSIONS (NTB) TO ',i3,
     &       /)
      END

      SUBROUTINE HOGRD
C*****************************************************************
C ASSUMES THE FOLLOWING:
C  1. 3-BLOCK H-O TOPOLOGY
C  2. BLOCKS AND INDICES ARE IN PROPER ORDER (upper-tip-lower)
C*****************************************************************
C
      PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
      common /grid/ x(jmx,kmx,lmx,NTB),y(jmx,kmx,lmx,NTB),
     &  z(jmx,kmx,lmx,NTB),jm(NTB),km(NTB),lm(NTB),NBLOCKS,NBB,SYM
      COMMON/AERO/ R(jmx,kmx,lmx),RU(jmx,kmx,lmx),RV(jmx,kmx,lmx),
     1 RW(jmx,kmx,lmx),E(jmx,kmx,lmx),ALPHA,BETA,FMACH,CBAR
      DIMENSION XX(JMX,KMX,LMX),YY(JMX,KMX,LMX),ZZ(JMX,KMX,LMX)
      LOGICAL SYM
      DATA G/1.4/, GM1/0.4/

      jto = 9
   WRITE RESTART TAPE
      NC1=0
      WRITE(jto)NC1,G
      WRITE(*,*)'JMAX KMAX LMAX'
```

```
      DO 2 MB=1,NBLOCKS
C  BLOCK #1: STARBOARD UPPER
      IF(MB.EQ.1)THEN

      JMAX=JM(MB)
      KMAX=KM(MB)
      LMAX=LM(MB)+1
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 5 J=1,JMAX
      DO 5 K=1,KMAX
      DO 5 L=1,LMAX
        IF(L.NE.1)THEN
          XX(J,K,L)=X(J,K,L-1,MB)/CBAR
          YY(J,K,L)=Y(J,K,L-1,MB)/CBAR
          ZZ(J,K,L)=Z(J,K,L-1,MB)/CBAR
        ELSE
          XX(J,K,L)=X(J,KM(NBLOCKS)-K+1,L+1,NBLOCKS)/CBAR
          YY(J,K,L)=Y(J,KM(NBLOCKS)-K+1,L+1,NBLOCKS)/CBAR
          ZZ(J,K,L)=Z(J,KM(NBLOCKS)-K+1,L+1,NBLOCKS)/CBAR
        ENDIF
    5 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSEIF(MB.EQ.NBLOCKS)THEN

C  BLOCK 3: STARBOARD LOWER
      JMAX=JM(MB)
      KMAX=KM(MB)
      LMAX=LM(MB)
       WRITE(*,*)JMAX,KMAX,LMAX
      DO 15 J=1,JMAX
      DO 15 K=1,KMAX
      DO 15 L=1,LMAX
      XX(J,K,L)=X(J,K,L,MB)/CBAR
      YY(J,K,L)=Y(J,K,L,MB)/CBAR
      ZZ(J,K,L)=Z(J,K,L,MB)/CBAR
   15 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSE

C  BLOCK #2: STARBOARD TIP
      JMAX=JM(MB)
      KMAX=KM(MB)+2
      LMAX=LM(MB)
       WRITE(*,*)JMAX,KMAX,LMAX
      DO 10 J=1,JMAX
      DO 10 K=1,KMAX
      DO 10 L=1,LMAX
      IF(K.EQ.1)THEN
        XX(J,K,L)=X(J,KM(MB-1)-1,L,MB-1)/CBAR
        YY(J,K,L)=Y(J,KM(MB-1)-1,L,MB-1)/CBAR
        ZZ(J,K,L)=Z(J,KM(MB-1)-1,L,MB-1)/CBAR
      ELSEIF(K.EQ.KMAX)THEN
        XX(J,K,L)=X(J,2,L,MB+1)/CBAR
        YY(J,K,L)=Y(J,2,L,MB+1)/CBAR
        ZZ(J,K,L)=Z(J,2,L,MB+1)/CBAR
      ELSE
```

```
          XX(J,K,L)=X(J,K-1,L,MB)/CBAR
          YY(J,K,L)=Y(J,K-1,L,MB)/CBAR
          ZZ(J,K,L)=Z(J,K-1,L,MB)/CBAR
        ENDIF
   10 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ENDIF
    2 CONTINUE
      IF(.NOT.SYM)GO TO 99

      DO 3 MB=NBLOCKS,1,-1

      IF(MB.EQ.NBLOCKS)THEN

C  BLOCK 4: PORT LOWER
      JMAX=JM(MB)
      KMAX=KM(MB)+1
      LMAX=LM(MB)
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 20 J=1,JMAX
      DO 20 K=1,KMAX
      DO 20 L=1,LMAX
      IF(K.GT.1)THEN
        XX(J,K,L)= X(J,KM(MB)-K+2,L,MB)/CBAR
        YY(J,K,L)=-Y(J,KM(MB)-K+2,L,MB)/CBAR
        ZZ(J,K,L)= Z(J,KM(MB)-K+2,L,MB)/CBAR
      ELSE
        XX(J,K,L)= X(J,KM(MB)-1,L,MB)/CBAR
        YY(J,K,L)= Y(J,KM(MB)-1,L,MB)/CBAR
        ZZ(J,K,L)= Z(J,KM(MB)-1,L,MB)/CBAR
      ENDIF
   20 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSEIF(MB.EQ.1)THEN

C  BLOCK #6: PORT UPPER
      JMAX=JM(MB)
      KMAX=KM(MB)+1
      LMAX=LM(MB)+1
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 30 J=1,JMAX
      DO 30 K=1,KMAX
      DO 30 L=1,LMAX
      IF(K.NE.KMAX)THEN
        IF(L.NE.1)THEN
          XX(J,K,L)= X(J,KM(MB)-K+1,L-1,MB)/CBAR
          YY(J,K,L)=-Y(J,KM(MB)-K+1,L-1,MB)/CBAR
          ZZ(J,K,L)= Z(J,KM(MB)-K+1,L-1,MB)/CBAR
        ELSE
          XX(J,K,L)= X(J,K,L+1,NBLOCKS)/CBAR
          YY(J,K,L)=-Y(J,K,L+1,NBLOCKS)/CBAR
          ZZ(J,K,L)= Z(J,K,L+1,NBLOCKS)/CBAR
        ENDIF
      ELSE
        IF(L.NE.1)THEN
          XX(J,K,L)= X(J,2,L-1,MB)/CBAR
```

```
          YY(J,K,L)= Y(J,2,L-1,MB)/CBAR
          ZZ(J,K,L)= Z(J,2,L-1,MB)/CBAR
        ELSE
          XX(J,K,L)= X(J,KM(NBLOCKS)-1,L+1,NBLOCKS)/CBAR
          YY(J,K,L)= Y(J,KM(NBLOCKS)-1,L+1,NBLOCKS)/CBAR
          ZZ(J,K,L)= Z(J,KM(NBLOCKS)-1,L+1,NBLOCKS)/CBAR
        ENDIF
      ENDIF
   30 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSE

C  BLOCK #5: PORT TIP
      JMAX=JM(MB)
      KMAX=KM(MB)+2
      LMAX=LM(MB)
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 25 J=1,JMAX
      DO 25 K=1,KMAX
      DO 25 L=1,LMAX
      IF(K.EQ.KMAX)THEN
        XX(J,K,L)= X(J,KM(1)-1,L,1)/CBAR
        YY(J,K,L)=-Y(J,KM(1)-1,L,1)/CBAR
        ZZ(J,K,L)= Z(J,KM(1)-1,L,1)/CBAR
      ELSEIF(K.EQ.1)THEN
        XX(J,K,L)= X(J,2,L,NBLOCKS)/CBAR
        YY(J,K,L)=-Y(J,2,L,NBLOCKS)/CBAR
        ZZ(J,K,L)= Z(J,2,L,NBLOCKS)/CBAR
      ELSE
        XX(J,K,L)= X(J,KM(MB)-K+2,L,MB)/CBAR
        YY(J,K,L)=-Y(J,KM(MB)-K+2,L,MB)/CBAR
        ZZ(J,K,L)= Z(J,KM(MB)-K+2,L,MB)/CBAR
      ENDIF
   25 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ENDIF
    3 CONTINUE

   99 RETURN

 9001 write(*,1050)
      stop

 1050 format(/,5X,'ERROR OPENING OUTPUT FILE ',
     &       /)

      END

      SUBROUTINE COGRD
C*****************************************************************
C ASSUMES THE FOLLOWING:
C   1. 3-BLOCK C-O TOPOLOGY
C   2. BLOCKS AND INDICES ARE IN PROPER ORDER
C*****************************************************************
C
      PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
```

```
      common /grid/ x(jmx,kmx,lmx,NTB),y(jmx,kmx,lmx,NTB),
     &  z(jmx,kmx,lmx,NTB),jm(NTB),km(NTB),lm(NTB),NBLOCKS,NBB,SYM
      COMMON/AERO/ R(jmx,kmx,lmx),RU(jmx,kmx,lmx),RV(jmx,kmx,lmx),
     1 RW(jmx,kmx,lmx),E(jmx,kmx,lmx),ALPHA,BETA,FMACH,CBAR
      DIMENSION XX(JMX,KMX,LMX),YY(JMX,KMX,LMX),ZZ(JMX,KMX,LMX)
      LOGICAL SYM
      DATA G/1.4/, GM1/0.4/

      jto = 9
C  WRITE RESTART TAPE
      NC1=0
      WRITE(jto)NC1,G
      WRITE(*,*)'JMAX KMAX LMAX'

      DO 2 MB=1,NBLOCKS
C  BLOCK #1: STARBOARD UPPER
      IF(MB.EQ.1)THEN

      JMAX=JM(MB)+1
      KMAX=KM(MB)
      LMAX=LM(MB)+1
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 5 J=2,JMAX
      DO 5 K=1,KMAX
      DO 5 L=1,LMAX
        IF(L.NE.1)THEN
          XX(J,K,L)=X(J-1,K,L-1,MB)/CBAR
          YY(J,K,L)=Y(J-1,K,L-1,MB)/CBAR
          ZZ(J,K,L)=Z(J-1,K,L-1,MB)/CBAR
        ELSE
          XX(J,K,L)=X(J-1,KM(NBLOCKS)-K+1,L+1,NBLOCKS)/CBAR
          YY(J,K,L)=Y(J-1,KM(NBLOCKS)-K+1,L+1,NBLOCKS)/CBAR
          ZZ(J,K,L)=Z(J-1,KM(NBLOCKS)-K+1,L+1,NBLOCKS)/CBAR
        ENDIF
    5 CONTINUE

      DO 8 K=1,KMAX
      DO 8 L=1,LMAX
        IF(L.NE.1)THEN
          XX(1,K,L)=X(2,KM(NBLOCKS)-K+1,L-1,NBLOCKS)/CBAR
          YY(1,K,L)=Y(2,KM(NBLOCKS)-K+1,L-1,NBLOCKS)/CBAR
          ZZ(1,K,L)=Z(2,KM(NBLOCKS)-K+1,L-1,NBLOCKS)/CBAR
        ELSE
          XX(1,K,L)=X(3,KM(NBLOCKS)-K+1,L,NBLOCKS)/CBAR
          YY(1,K,L)=Y(3,KM(NBLOCKS)-K+1,L,NBLOCKS)/CBAR
          ZZ(1,K,L)=Z(3,KM(NBLOCKS)-K+1,L,NBLOCKS)/CBAR
        ENDIF
    8 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSEIF(MB.EQ.NBLOCKS)THEN

C  BLOCK 3: STARBOARD LOWER
      JMAX=JM(MB)
      KMAX=KM(MB)
      LMAX=LM(MB)
       WRITE(*,*)JMAX,KMAX,LMAX
      DO 15 J=1,JMAX
      DO 15 K=1,KMAX
```

```
      DO 15 L=1,LMAX
      XX(J,K,L)=X(J,K,L,MB)/CBAR
      YY(J,K,L)=Y(J,K,L,MB)/CBAR
      ZZ(J,K,L)=Z(J,K,L,MB)/CBAR
   15 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSE

C  BLOCK #2: STARBOARD TIP
      JMAX=JM(MB)
      KMAX=KM(MB)+2
      LMAX=LM(MB)
       WRITE(*,*)JMAX,KMAX,LMAX
      DO 10 J=1,JMAX
      DO 10 K=1,KMAX
      DO 10 L=1,LMAX
      IF(K.EQ.1)THEN
        XX(J,K,L)=X(J,KM(MB-1)-1,L,MB-1)/CBAR
        YY(J,K,L)=Y(J,KM(MB-1)-1,L,MB-1)/CBAR
        ZZ(J,K,L)=Z(J,KM(MB-1)-1,L,MB-1)/CBAR
      ELSEIF(K.EQ.KMAX)THEN
        XX(J,K,L)=X(J,2,L,MB+1)/CBAR
        YY(J,K,L)=Y(J,2,L,MB+1)/CBAR
        ZZ(J,K,L)=Z(J,2,L,MB+1)/CBAR
      ELSE
        XX(J,K,L)=X(J,K-1,L,MB)/CBAR
        YY(J,K,L)=Y(J,K-1,L,MB)/CBAR
        ZZ(J,K,L)=Z(J,K-1,L,MB)/CBAR
      ENDIF
   10 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ENDIF
    2 CONTINUE
      IF(.NOT.SYM)GO TO 99

      DO 3 MB=NBLOCKS,1,-1

      IF(MB.EQ.NBLOCKS)THEN

C  BLOCK 4: PORT LOWER
      JMAX=JM(MB)
      KMAX=KM(MB)+1
      LMAX=LM(MB)
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 20 J=1,JMAX
      DO 20 K=1,KMAX
      DO 20 L=1,LMAX
      IF(K.GT.1)THEN
        XX(J,K,L)= X(J,KM(MB)-K+2,L,MB)/CBAR
        YY(J,K,L)=-Y(J,KM(MB)-K+2,L,MB)/CBAR
        ZZ(J,K,L)= Z(J,KM(MB)-K+2,L,MB)/CBAR
      ELSE
        XX(J,K,L)= X(J,KM(MB)-1,L,MB)/CBAR
        YY(J,K,L)= Y(J,KM(MB)-1,L,MB)/CBAR
        ZZ(J,K,L)= Z(J,KM(MB)-1,L,MB)/CBAR
      ENDIF
```

```
   20 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSEIF(MB.EQ.1)THEN

C  BLOCK #6: PORT UPPER
      JMAX=JM(MB)+1
      KMAX=KM(MB)+1
      LMAX=LM(MB)+1
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 30 J=2,JMAX
      DO 30 K=1,KMAX
      DO 30 L=1,LMAX
      IF(K.LT.KMAX)THEN
        IF(L.NE.1)THEN
          XX(J,K,L)=X(J-1,KM(MB)-K+1,L-1,MB)/CBAR
          YY(J,K,L)=-Y(J-1,KM(MB)-K+1,L-1,MB)/CBAR
          ZZ(J,K,L)=Z(J-1,KM(MB)-K+1,L-1,MB)/CBAR
        ELSE
          XX(J,K,L)=X(J-1,K,L+1,NBLOCKS)/CBAR
          YY(J,K,L)=-Y(J-1,K,L+1,NBLOCKS)/CBAR
          ZZ(J,K,L)=Z(J-1,K,L+1,NBLOCKS)/CBAR
        ENDIF
      ELSE
        IF(L.NE.1)THEN
          XX(J,K,L)=X(J-1,2,L-1,MB)/CBAR
          YY(J,K,L)=Y(J-1,2,L-1,MB)/CBAR
          ZZ(J,K,L)=Z(J-1,2,L-1,MB)/CBAR
        ELSE
          XX(J,K,L)=X(J-1,KM(NBLOCKS)-1,L+1,NBLOCKS)/CBAR
          YY(J,K,L)=Y(J-1,KM(NBLOCKS)-1,L+1,NBLOCKS)/CBAR
          ZZ(J,K,L)=Z(J-1,KM(NBLOCKS)-1,L+1,NBLOCKS)/CBAR
        ENDIF
      ENDIF
   30 CONTINUE

      DO 32 K=1,KMAX
      DO 32 L=1,LMAX
      IF(K.LT.KMAX)THEN
        IF(L.NE.1)THEN
          XX(1,K,L)=X(2,K,L-1,NBLOCKS)/CBAR
          YY(1,K,L)=-Y(2,K,L-1,NBLOCKS)/CBAR
          ZZ(1,K,L)=Z(2,K,L-1,NBLOCKS)/CBAR
        ELSE
          XX(1,K,L)=X(3,K,L,NBLOCKS)/CBAR
          YY(1,K,L)=-Y(3,K,L,NBLOCKS)/CBAR
          ZZ(1,K,L)=Z(3,K,L,NBLOCKS)/CBAR
        ENDIF
      ELSE
        IF(L.NE.1)THEN
          XX(1,K,L)=X(2,KM(NBLOCKS)-1,L-1,NBLOCKS)/CBAR
          YY(1,K,L)=Y(2,KM(NBLOCKS)-1,L-1,NBLOCKS)/CBAR
          ZZ(1,K,L)=Z(2,KM(NBLOCKS)-1,L-1,NBLOCKS)/CBAR
        ELSE
          XX(1,K,L)=X(3,KM(NBLOCKS)-1,L,NBLOCKS)/CBAR
          YY(1,K,L)=Y(3,KM(NBLOCKS)-1,L,NBLOCKS)/CBAR
          ZZ(1,K,L)=Z(3,KM(NBLOCKS)-1,L,NBLOCKS)/CBAR
        ENDIF
      ENDIF
```

```
   32 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ELSE

C  BLOCK #5: PORT TIP
      JMAX=JM(MB)
      KMAX=KM(MB)+2
      LMAX=LM(MB)
      WRITE(*,*)JMAX,KMAX,LMAX
      DO 25 J=1,JMAX
      DO 25 K=1,KMAX
      DO 25 L=1,LMAX
      IF(K.EQ.KMAX)THEN
        XX(J,K,L)= X(J,KM(1)-1,L,1)/CBAR
        YY(J,K,L)=-Y(J,KM(1)-1,L,1)/CBAR
        ZZ(J,K,L)= Z(J,KM(1)-1,L,1)/CBAR
      ELSEIF(K.EQ.1)THEN
        XX(J,K,L)= X(J,2,L,NBLOCKS)/CBAR
        YY(J,K,L)=-Y(J,2,L,NBLOCKS)/CBAR
        ZZ(J,K,L)= Z(J,2,L,NBLOCKS)/CBAR
      ELSE
        XX(J,K,L)= X(J,KM(MB)-K+2,L,MB)/CBAR
        YY(J,K,L)=-Y(J,KM(MB)-K+2,L,MB)/CBAR
        ZZ(J,K,L)= Z(J,KM(MB)-K+2,L,MB)/CBAR
      ENDIF
   25 CONTINUE

      CALL WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)

      ENDIF
    3 CONTINUE

   99 RETURN

 9001 write(*,1050)
      stop

 1050 format(/,5X,'ERROR OPENING OUTPUT FILE ',
     &       /)

      END

      SUBROUTINE WRDAT(JTO,JMAX,KMAX,LMAX,XX,YY,ZZ,R,RU,RV,RW,E)
      PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
      DIMENSION XX(JMX,KMX,LMX),YY(JMX,KMX,LMX),ZZ(JMX,KMX,LMX),
     1 R(jmx,kmx,lmx),RU(jmx,kmx,lmx),RV(jmx,kmx,lmx),
     2 RW(jmx,kmx,lmx),E(jmx,kmx,lmx)

      WRITE(jto)JMAX,KMAX,LMAX
      WRITE(jto)((( XX(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX),
     *          ((( YY(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX),
     *          ((( ZZ(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX)
      WRITE(jto)((( R(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX),
     *          (((RU(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX),
     *          (((RV(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX),
     *          (((RW(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX),
     *          ((( E(J,K,L),J= 1,JMAX),K=1, KMAX),L=1, LMAX)
```

```
      RETURN
      END

      SUBROUTINE READRES
      PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
      DIMENSION R(JMX,KMX,LMX),RU(JMX,KMX,LMX),RV(JMX,KMX,LMX),
     1  RW(JMX,KMX,LMX),E(JMX,KMX,LMX),X(JMX,KMX,LMX),Y(JMX,KMX,LMX),
     2  Z(JMX,KMX,LMX)
      COMMON/DATA/ NBLK, JMAX(NTB), KMAX(NTB), LMAX(NTB)
C  READ RESTART TAPE
      NBLK=0
      NTP=8
      READ(NTP)NC1,G
   50 READ(NTP,END=99)JMAX(NBLK+1),KMAX(NBLK+1),LMAX(NBLK+1)
      NBLK=NBLK+1
      WRITE(*,*)' READING BLOCK ',NBLK
      READ(NTP) ((( X(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
     *        L=1,LMAX(NBLK)),
     *         ((( Y(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
     *        L=1,LMAX(NBLK)),
     *         ((( Z(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
     *        L=1,LMAX(NBLK))
C
C  TRANSLATE AND SCALE...
C       DO 10 J=1,JMAX(NBLK)
C       DO 10 K=1,KMAX(NBLK)
C       DO 10 L=1,LMAX(NBLK)
C       X(J,K,L)=X(J,K,L)*184.
C       Y(J,K,L)=Y(J,K,L)*184.
C       Z(J,K,L)=Z(J,K,L)*184.
C       X(J,K,L)=X(J,K,L)+1595.3
C       Z(J,K,L)=Z(J,K,L)+454.44
C    10 CONTINUE

       WRITE(9+NBLK) ((( X(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         ((( Y(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         ((( Z(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK))

       READ(NTP)((( R(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         (((RU(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         (((RV(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         (((RW(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         ((( E(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK))
       WRITE(29+NBLK)((( R(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         (((RU(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         (((RV(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         (((RW(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
      *        L=1,LMAX(NBLK)),
      *         ((( E(J,K,L),J= 1,JMAX(NBLK)),K=1,KMAX(NBLK)),
```

```
     *       L=1,LMAX(NBLK))
        GO TO 50
     99 CONTINUE
        DO 5 I=1,NBLK
        REWIND 9+I
        REWIND 29+I
      5 CONTINUE
        RETURN
        END

        SUBROUTINE SCNGEN
        PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
c    adjustable :  max print, max B.C.'s, max patches  per block
        PARAMETER (MPS=10,    MBC=25,     MP=50)
c    block specific info
        COMMON /CPOIN/
       * GAMMA,GAMI,GM1R,GGM1,GSGM,RPR,RPRT,COFMIX,DTBLK,DIS2,DIS4,
       * RTPTS,
       * PRESSJ(MBC),TEMPJ(MBC),
       * PRESSK(MBC),TEMPK(MBC),
       * PRESSL(MBC),TEMPL(MBC),
       * IPWX,IPWY,IPWZ,IPWR,IPWRU,IPWRV,IPWRW,IPWE,IPWQ,
       * IPWS1,IPWVDT,IPWXX,IPWXY,IPWXZ,IPWYX,
       * IPWYY,IPWYZ,IPWZX,IPWZY,IPWZZ,IPWTMU,IPWSPT,IPWCOF,
       * IPW28,IPW29,IPW30,IPWRK1,IPWRK2,IPWEPS,IPWAK,
       * NXYZ,LMODE,LTURB,
       * JMAX,KMAX,LMAX,JM,KM,LM,INVISC(3),LAMIN(3),
       * NPSEG,JKLPI(3,3,MPS),IPORD(2,MPS),
       * NJSEG,JLINE(MBC),JKLOW(MBC),JKHIGH(MBC),JLLOW(MBC),JLHIGH(MBC),
       * JTYPE(MBC),JSIGN(MBC),INTERJ(MBC),
       * NKSEG,KLINE(MBC),KJLOW(MBC),KJHIGH(MBC),KLLOW(MBC),KLHIGH(MBC),
       * KTYPE(MBC),KSIGN(MBC),INTERK(MBC),
       * NLSEG,LLINE(MBC),LJLOW(MBC),LJHIGH(MBC),LKLOW(MBC),LKHIGH(MBC),
       * LTYPE(MBC),LSIGN(MBC),INTERL(MBC),
       * JEDGE(MBC),KEDGE(MBC),LEDGE(MBC),JDIR(MBC),KDIR(MBC),LDIR(MBC),
       * NORMJ(MBC),NORMK(MBC),NORML(MBC),
       * NJPAT,JPJ2(MP),JPJM(MP),JPK2(MP),JPKM(MP),JPL2(MP),JPLM(MP),
       * NKPAT,KPJ2(MP),KPJM(MP),KPK2(MP),KPKM(MP),KPL2(MP),KPLM(MP),
       * NLPAT,LPJ2(MP),LPJM(MP),LPK2(MP),LPKM(MP),LPL2(MP),LPLM(MP)
c
        COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
        NAMELIST/INPUTS/ PREF,TREFR,VRAT,TSUTH,RE,PR,PRT,DDUMP,
       * DTCAP,PCQMAX,SPLEND,SMOO,STOPL2,STOPTR,ALPHA,BETA,XMACH,
       * NBLOCK,NMAX,NC,NSPRT,NP,IFXPRT,IFXPLT,L2PLOT,
       * IPLOT,NUMDT,IVARDT,ISOLVE,IRHS,IFILTR,IMUTUR,
       * MBORD,IBORD,NSECTR,LPRES,LREST,GAMMA,DIS2,DIS4,
       * NOBORT,IFMAX,IMASS,P0sd,T0sd,Lrec45,
       * ORDER,NTURB,IFILTR, NSKIP, STOPTM, CBAR, SSPAN, SREF,
       * REFMX, REFMY, REFMZ
        NAMELIST/BOUNDS/ NJSEG,JLINE,JKLOW,JKHIGH,JLLOW,JLHIGH,
       *                 NKSEG,KLINE,KJLOW,KJHIGH,KLLOW,KLHIGH,
       *                 NLSEG,LLINE,LJLOW,LJHIGH,LKLOW,LKHIGH,
       *                 JTYPE,JSIGN,PRESSJ,TEMPJ,INTERJ,
       *                 KTYPE,KSIGN,PRESSK,TEMPK,INTERK,
       *                 LTYPE,LSIGN,PRESSL,TEMPL,INTERL,
       *                 JEDGE,KEDGE,LEDGE,JDIR,KDIR,LDIR,
       *                 NORMJ,NORMK,NORML
c
        DIMENSION R(JMX,KMX,LMX),RU(JMX,KMX,LMX),RV(JMX,KMX,LMX),
       1  RW(JMX,KMX,LMX),E(JMX,KMX,LMX),X(JMX,KMX,LMX),Y(JMX,KMX,LMX),
```

```
     2  Z(JMX,KMX,LMX)
      COMMON/DATA/ NBLK, JMM(NTB), KMM(NTB), LMM(NTB)
      LOGICAL FORM
      DATA FORM/.TRUE./
      DATA MINUS1/-1/,IZERO/0/,IONE/1/
      BODSYM=0
      INCENT=0
      NC=0
      NDUM=0
      NTI = 7
      NTP = 9
      REWIND NTP
      NPNL=0
      ITTR=1


C*********************************************************
C  BASIC DATA
C*********************************************************

      READ(NTI,INPUTS)        ! Cray version of NAMELIST
c     READ(NTI,NML=INPUTS)    ! IRIS version of NAMELIST

C*********************************************************
C  PATCH GEOMETRY
C*********************************************************

C  BODY SURFACE: L=1; J=1,JMX; K=1,KMX
C  SIDE WALL UPPER: K=2; J=1,JMX; L=1,LMX
C  SIDE WALL LOWER: K=KMX-1; J=1,JMX; L=1,LMX
C  END WALL: J=JMX; K=2,KMX-1; L=1,LMX
C     CALL BGOM(NP,NPNL,IORDR,N1,N2,N3,N4,N5)
C      WHERE:
C            NP= PATCH NO.
C            NPNL= RUNNING PANEL COUNT
C            IORDR= J,K OR L CONSTANT (1,2 OR 3)
C            N1 THRU N5= LOOP COUNTERS
C               N1,N2 RUN FIRST
C               N3,N4 RUN SECOND
C               N5 IS CONSTANT COUNTER
C

C ASSUMES C-C GRID...

      WRITE(*,*)'   BODY GEOMETRY...'
      NPCH=0
      DO 25 NN=1,NBLK
      READ(NTI,BOUNDS)        ! Cray version of NAMELIST
c     READ(NTI,NML=BOUNDS)    ! IRIS version of NAMELIST
      DO 26 J=1,NJSEG
      IF(JTYPE(J).EQ.7.OR.JTYPE(J).EQ.8)THEN
        NPCH=NPCH+1
        IORDR=1
        CALL SGOM(NN,NPCH,IORDR,JLLOW(J),JLHIGH(J),JKLOW(J),JKHIGH(J),
     1                          JLINE(J),NPNL)
      ENDIF
   26 CONTINUE

      DO 27 K=1,NKSEG
      IF(KTYPE(K).EQ.7.OR.KTYPE(K).EQ.8)THEN
```

```
      NPCH=NPCH+1
      IORDR=2
      CALL SGOM(NN,NPCH,IORDR,KLLOW(K),KLHIGH(K),KJLOW(K),KJHIGH(K),
     1                        KLINE(K),NPNL)
    ENDIF
 27 CONTINUE

    DO 28 L=1,NLSEG
    IF(LTYPE(L).EQ.7.OR.LTYPE(L).EQ.8)THEN
      NPCH=NPCH+1
      IORDR=3
      CALL SGOM(NN,NPCH,IORDR,LJLOW(L),LJHIGH(L),LKLOW(L),LKHIGH(L),
     1                        LLINE(L),NPNL)
    ENDIF
 28 CONTINUE
 25 CONTINUE

    WRITE(*,*)' SCAN DATA INPUT IN FILE SCAN.DAT'
    RETURN
701 FORMAT(2A4)
702 FORMAT(5I5)
703 FORMAT(6E12.5)
704 FORMAT(3I5/(6E12.5))
705 FORMAT(6E12.5/2E12.5,I5)
706 FORMAT(6E12.5)
707 FORMAT(20A4)
    END

    SUBROUTINE SGOM(NB,NPTCH,IORDR,N1,N2,N3,N4,N5,NPNL)
    PARAMETER(NTB=3, JMX=230, KMX=32, LMX=50)
    DIMENSION R(JMX,KMX,LMX),RU(JMX,KMX,LMX),RV(JMX,KMX,LMX),
   1  RW(JMX,KMX.LMX),E(JMX,KMX,LMX),X(JMX,KMX,LMX),Y(JMX,KMX,LMX),
   2  Z(JMX,KMX,LMX)
    COMMON/DATA/ NBLK, JMAX(NTB), KMAX(NTB), LMAX(NTB)
    LOGICAL FORM
    DATA FORM/.TRUE./
    NTP = 9
    IZERO=0
    ID=2
    NR=N2-N1
    NC=N4-N3
    NTOT = (NR+1)*(NC+1)

    READ(9+NB) ((( X(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
   *       L=1,LMAX(NB)),
   *        ((( Y(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
   *       L=1,LMAX(NB)),
   *        ((( Z(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
   *       L=1,LMAX(NB))
    REWIND 9+NB

    IF(FORM)THEN
     WRITE(NTP,702)          -3,-4,0,0
     WRITE(NTP,702)          NTOT,1,1,0
      IF(IORDR.EQ.1)THEN
       WRITE(NTP,703)((X(N5,K,L),Y(N5,K,L),Z(N5,K,L),NB,N5,K,L,
   1         L=N1,N2), K=N3,N4)
      ELSEIF(IORDR.EQ.2)THEN
       WRITE(NTP,703)((X(J,N5,L),Y(J,N5,L),Z(J,N5,L),NB,J,N5,L,
   1         L=N1,N2), J=N3,N4)
```

```
         ELSEIF(IORDR.EQ.3)THEN
          WRITE(NTP,703)((X(J,K,N5),Y(J,K,N5),Z(J,K,N5),NB,J,K,N5,
     1          J=N1,N2), K=N3,N4)
         ENDIF
        ELSE
         WRITE(NTP)                -3,-4,0,0
         WRITE(NTP)                NTOT,1,1,0
         DO 100 I=N3,N4
          IF(IORDR.EQ.1)THEN
           WRITE(NTP)((X(N5,K,L),Y(N5,K,L),Z(N5,K,L),NB,N5,K,L,
     1          L=N1,N2), K=N3,N4)
         ELSEIF(IORDR.EQ.2)THEN
          WRITE(NTP)((X(J,N5,L),Y(J,N5,L),Z(J,N5,L),NB,J,N5,L,
     1          L=N1,N2), J=N3,N4)
         ELSEIF(IORDR.EQ.3)THEN
          WRITE(NTP)((X(J,K,N5),Y(J,K,N5),Z(J,K,N5),NB,J,K,N5,
     1          J=N1,N2), K=N3,N4)
         ENDIF
  100   CONTINUE
        ENDIF
        NPNL=NPNL+NR*NC
  701 FORMAT(2A4)
  702 FORMAT(5I5)
C 703 FORMAT(6E12.5)
  703 FORMAT(3F10.5,4I5)
        RETURN
        END

        SUBROUTINE PLTGEN
        PARAMETER(NTB=3,JMX=230,KMX=32,LMX=50)
C  adjustable :  max print, max B.C.'s, max patches  per block
        PARAMETER (MPS=10,    MBC=25,     MP=50)
C  block specific info
        COMMON /CPOIN/
       * GAMMA,GAMI,GM1R,GGM1,GSGM,RPR,RPRT,COFMIX,DTBLK,DIS2,DIS4,
       * RTPTS,
       * PRESSJ(MBC),TEMPJ(MBC),
       * PRESSK(MBC),TEMPK(MBC),
       * PRESSL(MBC),TEMPL(MBC),
       * IPWX,IPWY,IPWZ,IPWR,IPWRU,IPWRV,IPWRW,IPWE,IPWQ,
       * IPWS1,IPWVDT,IPWXX,IPWXY,IPWXZ,IPWYX,
       * IPWYY,IPWYZ,IPWZX,IPWZY,IPWZZ,IPWTMU,IPWSPT,IPWCOF,
       * IPW28,IPW29,IPW30,IPWRK1,IPWRK2,IPWEPS,IPWAK,
       * NXYZ,LMODE,LTURB,
       * JMAX,KMAX,LMAX,JM,KM,LM,INVISC(3),LAMIN(3),
       * NPSEG,JKLPI(3,3,MPS),IPORD(2,MPS),
       * NJSEG,JLINE(MBC),JKLOW(MBC),JKHIGH(MBC),JLLOW(MBC),JLHIGH(MBC),
       * JTYPE(MBC),JSIGN(MBC),INTERJ(MBC),
       * NKSEG,KLINE(MBC),KJLOW(MBC),KJHIGH(MBC),KLLOW(MBC),KLHIGH(MBC),
       * KTYPE(MBC),KSIGN(MBC),INTERK(MBC),
       * NLSEG,LLINE(MBC),LJLOW(MBC),LJHIGH(MBC),LKLOW(MBC),LKHIGH(MBC),
       * LTYPE(MBC),LSIGN(MBC),INTERL(MBC),
       * JEDGE(MBC),KEDGE(MBC),LEDGE(MBC),JDIR(MBC),KDIR(MBC),LDIR(MBC),
       * NORMJ(MBC),NORMK(MBC),NORML(MBC),
       * NJPAT,JPJ2(MP),JPJM(MP),JPK2(MP),JPKM(MP),JPL2(MP),JPLM(MP),
       * NKPAT,KPJ2(MP),KPJM(MP),KPK2(MP),KPKM(MP),KPL2(MP),KPLM(MP),
       * NLPAT,LPJ2(MP),LPJM(MP),LPK2(MP),LPKM(MP),LPL2(MP),LPLM(MP)
C
        COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
        NAMELIST/INPUTS/ PREF,TREFR,VRAT,TSUTH,RE,PR,PRT,DDUMP,
```

```
     * DTCAP,PCQMAX,SPLEND,SMOO,STOPL2,STOPTR,ALPHA,BETA,XMACH,
     * NBLOCK,NMAX,NC,NSPRT,NP,IFXPRT,IFXPLT,L2PLOT,
     * IPLOT,NUMDT,IVARDT,ISOLVE,IRHS,IFILTR,IMUTUR,
     * MBORD,IBORD,NSECTR,LPRES,LREST,GAMMA,DIS2,DIS4,
     * NOBORT,IFMAX,IMASS,P0sd,T0sd,Lrec45,
     * ORDER,NTURB,IFILTR, NSKIP, STOPTM, CBAR, SSPAN, SREF,
     * REFMX, REFMY, REFMZ
      NAMELIST/BOUNDS/ NJSEG,JLINE,JKLOW,JKHIGH,JLLOW,JLHIGH,
     *                 NKSEG,KLINE,KJLOW,KJHIGH,KLLOW,KLHIGH,
     *                 NLSEG,LLINE,LJLOW,LJHIGH,LKLOW,LKHIGH,
     *                 JTYPE,JSIGN,PRESSJ,TEMPJ,INTERJ,
     *                 KTYPE,KSIGN,PRESSK,TEMPK,INTERK,
     *                 LTYPE,LSIGN,PRESSL,TEMPL,INTERL,
     *                 JEDGE,KEDGE,LEDGE,JDIR,KDIR,LDIR,
     *                 NORMJ,NORMK,NORML
C
      DIMENSION R(JMX,KMX,LMX),RU(JMX,KMX,LMX),RV(JMX,KMX,LMX),
     1  RW(JMX,KMX,LMX),E(JMX,KMX,LMX),X(JMX,KMX,LMX),Y(JMX,KMX,LMX),
     2  Z(JMX,KMX,LMX)
      COMMON/DATA/ NBLK, JMM(NTB), KMM(NTB), LMM(NTB)
      LOGICAL FORM
      DATA FORM/.TRUE./
      DATA MINUS1/-1/,IZERO/0/,IONE/1/
      BODSYM=0
      INCENT=0
      NC=0
      NDUM=0
      NTI = 7
      NTP = 9
      REWIND NTP
      NPNL=0
      ITTR=1


C*************************************************************
C  BASIC DATA
C*************************************************************

      READ(NTI,INPUTS)        ! Cray version of NAMELIST
c     READ(NTI,NML=INPUTS)    ! IRIS version of NAMELIST
      CALL BSIC

C*************************************************************
C  PATCH GEOMETRY
C*************************************************************

C  BODY SURFACE: L=1; J=1,JMX; K=1,KMX
C  SIDE WALL UPPER: K=2; J=1,JMX; L=1,LMX
C  SIDE WALL LOWER: K=KMX-1; J=1,JMX; L=1,LMX
C  END WALL: J=JMX; K=2,KMX-1; L=1,LMX
C     CALL BGOM(NP,NPNL,IORDR,N1,N2,N3,N4,N5)
C      WHERE:
C            NP= PATCH NO.
C            NPNL= RUNNING PANEL COUNT
C            IORDR= J,K OR L CONSTANT (1,2 OR 3)
C            N1 THRU N5= LOOP COUNTERS
C               N1,N2 RUN FIRST
C               N3,N4 RUN SECOND
C               N5 IS CONSTANT COUNTER
C
```

```
      IF(FORM)THEN
       WRITE(NTP,701)          'BGIN','BODG'
      ELSE
       WRITE(NTP)           'BGIN','BODG'
      ENDIF

C ASSUMES C-C GRID...

      WRITE(*,*)'   BODY GEOMETRY...'
      NPCH=0
      DO 25 NN=1,NBLK
      READ(NTI,BOUNDS)        ! Cray version of NAMELIST
c     READ(NTI,NML=BOUNDS)    ! IRIS version of NAMELIST
      DO 26 J=1,NJSEG
      IF(JTYPE(J).EQ.60.OR.JTYPE(J).EQ.61)THEN
        NPCH=NPCH+1
        IORDR=1
        CALL BGOM(NN,NPCH,IORDR,JLLOW(J),JLHIGH(J),JKLOW(J),JKHIGH(J),
     1                          JLINE(J),NPNL)
       ENDIF
   26 CONTINUE

      DO 27 K=1,NKSEG
      IF(KTYPE(K).EQ.60.OR.KTYPE(K).EQ.61)THEN
        NPCH=NPCH+1
        IORDR=2
        CALL BGOM(NN,NPCH,IORDR,KLLOW(K),KLHIGH(K),KJLOW(K),KJHIGH(K),
     1                          KLINE(K),NPNL)
       ENDIF
   27 CONTINUE

      DO 28 L=1,NLSEG
      IF(LTYPE(L).EQ.60.OR.LTYPE(L).EQ.61)THEN
        NPCH=NPCH+1
        IORDR=3
        CALL BGOM(NN,NPCH,IORDR,LJLOW(L),LJHIGH(L),LKLOW(L),LKHIGH(L),
     1                          LLINE(L),NPNL)
       ENDIF
   28 CONTINUE
   25 CONTINUE

      IF(FORM)THEN
       WRITE(NTP,701)          'END ','GRUP'
      ELSE
       WRITE(NTP)           'END ','GRUP'
      ENDIF
C  IF ITTR.GT.0 THEN WRITE AERO AWAY TO TAPE...

C*********************************************************
C AERODYNAMICS
C*********************************************************


       IF(FORM)THEN
        WRITE(NTP,701)          'BGIN','AERO'
        WRITE(NTP,701)          'STRT','ITER'
        WRITE(NTP,702)          NPNL,ITTR,IZERO,IZERO,IZERO
       ELSE
        WRITE(NTP)           'BGIN','AERO'
        WRITE(NTP)           'STRT','ITER'
```

```
        WRITE(NTP)          NPNL,ITTR,IZERO,IZERO,IZERO
       ENDIF

C      CALL AERO(IORDR,N1,N2,N3,N4,N5)
       WHERE:
C            IORDR= J,K OR L CONSTANT (1,2 OR 3)
C            N1 THRU N5= LOOP COUNTERS

       REWIND NTI
       WRITE(*,*)'   AERODYNAMICS...'
       DO 35 NN=1,NBLK
       READ(NTI,BOUNDS)        ! Cray version of NAMELIST
C      READ(NTI,NML=BOUNDS)    ! IRIS version of NAMELIST
       DO 36 J=1,NJSEG
       IF(JTYPE(J).EQ.60.OR.JTYPE(J).EQ.61)THEN
         IORDR=1
         CALL AERO(NN,IORDR,JLLOW(J),JLHIGH(J),JKLOW(J),JKHIGH(J),
     1                          JLINE(J))
       ENDIF
    36 CONTINUE

       DO 37 K=1,NKSEG
       IF(KTYPE(K).EQ.60.OR.KTYPE(K).EQ.61)THEN
         IORDR=2
         CALL AERO(NN,IORDR,KLLOW(K),KLHIGH(K),KJLOW(K),KJHIGH(K),
     1                          KLINE(K))
       ENDIF
    37 CONTINUE

       DO 38 L=1,NLSEG
       IF(LTYPE(L).EQ.60.OR.LTYPE(L).EQ.61)THEN
         IORDR=3
         CALL AERO(NN,IORDR,LJLOW(L),LJHIGH(L),LKLOW(L),LKHIGH(L),
     1                          LLINE(L))
       ENDIF
    38 CONTINUE
    35 CONTINUE

       IF(FORM)THEN
        WRITE(NTP,701)          'END ','GRUP'
       ELSE
        WRITE(NTP)          'END ','GRUP'
       ENDIF

C*************************************************************
C  OFF-BODY DATA SCAN
C*************************************************************
C      CALL OFFB(IORDR,N1,N2,N3,N4,N5)
C       WHERE:
C            IORDR= J,K OR L CONSTANT (1,2 OR 3)
C            N1 THRU N5= LOOP COUNTERS
C

       IF(FORM)THEN
        WRITE(NTP,701)          'BGIN','OFFB'
       ELSE
        WRITE(NTP)          'BGIN','OFFB'
       ENDIF

       REWIND NTI
```

```
      WRITE(*,*)'   OFF BODY DATA...'
      DO 45 NN=1,NBLK
      READ(NTI,BOUNDS)        ! Cray version of NAMELIST
C     READ(NTI,NML=BOUNDS)    ! IRIS version of NAMELIST
      DO 46 J=1,NJSEG
      IF(JTYPE(J).EQ.50)THEN
        IORDR=1
        CALL OFFB(NN,IORDR,JLLOW(J),JLHIGH(J),JKLOW(J),JKHIGH(J),
     1                          JLINE(J))
      ENDIF
   46 CONTINUE

      DO 47 K=1,NKSEG
      IF(KTYPE(K).EQ.50)THEN
        IORDR=2
        CALL OFFB(NN,IORDR,KLLOW(K),KLHIGH(K),KJLOW(K),KJHIGH(K),
     1                          KLINE(K))
      ENDIF
   47 CONTINUE

      DO 48 L=1,NLSEG
      IF(LTYPE(L).EQ.50)THEN
        IORDR=3
        CALL OFFB(NN,IORDR,LJLOW(L),LJHIGH(L),LKLOW(L),LKHIGH(L),
     1                          LLINE(L))
      ENDIF
   48 CONTINUE
   45 CONTINUE

C  'ID-SPAN PLANE...
      KMID=KMM(1)/2
      IF(2*KMID.NE.KMM(1))KMID=KMID+1
      CALL OFFB(1,2,2,LMM(1),1,JMM(1),KMID)
      KMID=KMM(NBLK)/2
      IF(2*KMID.NE.KMM(NBLK))KMID=KMID+1
      CALL OFFB(NBLK,2,1,LMM(NBLK),1,JMM(NBLK),KMID)

      IF(FORM)THEN
       WRITE(NTP,701)           'END ','GRUP'
       WRITE(NTP,701)           'END ','FILE'
      ELSE
       WRITE(NTP)          'END ','GRUP'
       WRITE(NTP)          'END ','FILE'
      ENDIF


C     DO 95 J=1,NBLK
C     CLOSE 9+J
C     CLOSE 29+J
C  95 CONTINUE

      WRITE(*,*)' PLOT FILE DATA IN FILE PARC.FMT'
      RETURN
  701 FORMAT(2A4)
  702 FORMAT(5I5)
   03 FORMAT(6E12.5)
  704 FORMAT(3I5/(6E12.5))
  705 FORMAT(6E12.5/2E12.5,I5)
  706 FORMAT(6E12.5)
  707 FORMAT(20A4)
```

```
      END

      SUBROUTINE BSIC
      COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
      LOGICAL FORM
      CHARACTER *80 TITL
      DATA FORM/.TRUE./
      NTP = 9
      WRITE(*,*)
      WRITE(*,*)' PLEASE INPUT TITLE FOR PLOT FILE.'
      READ(*,80)TITL
      RSYM = 0.0
      RGPR = 0.0
      ALDEG = ALPHA
      YAWDEG = BETA
      RMACH = 0.0
      RMINF = XMACH
      CBAR = 1.0
      SREF = 1.0
      SSPAN = 1.0
      REFL = 1.0
      SCALX = 1.0
      REFMX = 0.0
      REFMY = 0.0
      REFMZ = 0.0

      IF(FORM)THEN
        WRITE(NTP,701) 'BGIN','BSIC'
        WRITE(NTP,707) TITL
        WRITE(NTP,703) RSYM,RGPR,ALDEG,YAWDEG,RMACH,RMINF
        WRITE(NTP,703) CBAR*REFL,SREF*REFL*REFL,SSPAN*REFL,REFMX*SCALX,
     1             REFMY*REFL,REFMZ*REFL
        WRITE(NTP,703) CBAR,CBAR,CBAR,CBAR,CBAR,CBAR
        WRITE(NTP,701) 'END ','GRUP'
      ELSE
        WRITE(NTP) 'BGIN','BSIC'
        WRITE(NTP) TITL
        WRITE(NTP) RSYM,RGPR,ALDEG,YAWDEG,RMACH,RMINF
        WRITE(NTP) CBAR*REFL,SREF*REFL*REFL,SSPAN*REFL,REFMX*SCALX,
     1             REFMY*REFL,REFMZ*REFL
        WRITE(NTP) CBAR,CBAR,CBAR,CBAR,CBAR,CBAR
        WRITE(NTP) 'END ','GRUP'
      ENDIF
      RETURN
   80 FORMAT(A80)
  701 FORMAT(2A4)
  703 FORMAT(6E12.5)
  707 FORMAT(A80)
      END

      SUBROUTINE BGOM(NB,NPTCH,IORDR,N1,N2,N3,N4,N5,NPNL)
      PARAMETER(NTB=3,JMX=230,KMX=32,LMX=50)
      DIMENSION R(JMX,KMX,LMX),RU(JMX,KMX,LMX),RV(JMX,KMX,LMX),
     1  RW(JMX,KMX,LMX),E(JMX,KMX,LMX),X(JMX,KMX,LMX),Y(JMX,KMX,LMX),
     2  Z(JMX,KMX,LMX)
      COMMON/DATA/ NBLK, JMAX(NTB), KMAX(NTB), LMAX(NTB)
      LOGICAL FORM
      DATA FORM/.TRUE./
      NTP = 9
      IZERO=0
```

```
      ID=2
      NR=N2-N1
      NC=N4-N3

      READ(9+NB) ((( X(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *        L=1,LMAX(NB)),
     *         ((( Y(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *        L=1,LMAX(NB)),
     *         ((( Z(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *        L=1,LMAX(NB))
      REWIND 9+NB

      IF(FORM)THEN
       WRITE(NTP,701)           'STRT','PTCH'
       WRITE(NTP,702)           NPTCH,ID,NR,NC,IZERO
        IF(IORDR.EQ.1)THEN
         WRITE(NTP,703)((X(N5,K,L),Y(N5,K,L),Z(N5,K,L),L=N1,N2)
     1          ,K=N3,N4)
        ELSEIF(IORDR.EQ.2)THEN
         WRITE(NTP,703)((X(J,N5,L),Y(J,N5,L),Z(J,N5,L),L=N1,N2)
     1          ,J=N3,N4)
        ELSEIF(IORDR.EQ.3)THEN
         WRITE(NTP,703)((X(J,K,N5),Y(J,K,N5),Z(J,K,N5),J=N1,N2)
     1          ,K=N3,N4)
        ENDIF
      ELSE
       WRITE(NTP)           'STRT','PTCH'
       WRITE(NTP)           NPTCH,ID,NR,NC,IZERO
       DO 100 I=N3,N4
        IF(IORDR.EQ.1)THEN
         WRITE(NTP)((X(N5,K,L),Y(N5,K,L),Z(N5,K,L),L=N1,N2),
     1          K=N3,N4)
        ELSEIF(IORDR.EQ.2)THEN
         WRITE(NTP)((X(J,N5,L),Y(J,N5,L),Z(J,N5,L),L=N1,N2),
     1          J=N3,N4)
        ELSEIF(IORDR.EQ.3)THEN
         WRITE(NTP)((X(J,K,N5),Y(J,K,N5),Z(J,K,N5),J=N1,N2),
     1          K=N3,N4)
        ENDIF
  100  CONTINUE
      ENDIF
      NPNL=NPNL+NR*NC
  701 FORMAT(2A4)
  702 FORMAT(5I5)
c 703 FORMAT(6E12.5)
  703 FORMAT(6E20.9)
      RETURN
      END

      SUBROUTINE AERO(NB,IORDR,N1,N2,N3,N4,N5)
      PARAMETER(NTB=3,JMX=230,KMX=32,LMX=50)
      COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
      DIMENSION R(JMX,KMX,LMX),RU(JMX,KMX,LMX),RV(JMX,KMX,LMX),
     1  RW(JMX,KMX,LMX),E(JMX,KMX,LMX),X(JMX,KMX,LMX),Y(JMX,KMX,LMX),
     2  Z(JMX,KMX,LMX)
      COMMON/DATA/ NBLK, JMAX(NTB), KMAX(NTB), LMAX(NTB)
      LOGICAL FORM
      DATA FORM/.TRUE./, GAMMA/1.4/, GAMI/0.4/

      NTP = 9
```

```
      GD = GAMMA*GAMI
      FSMACH = XMACH

      CPS        = 1./(.5*GAMMA*FSMACH**2)

      READ(29+NB)((( R(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *       L=1,LMAX(NB)),
     *        (((RU(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *       L=1,LMAX(NB)),
     *        (((RV(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *       L=1,LMAX(NB)),
     *        (((RW(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *       L=1,LMAX(NB)),
     *        ((( E(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
     *       L=1,LMAX(NB))
      REWIND 29+NB

      ZER=0.
      IF(IORDR.EQ.1)THEN
      DO 5 K=N3,N4-1
      DO 10 L=N1,N2-1
      RR         = 1./R(N5,K,L)
      Q1         = R(N5,K,L)
      Q2         = RU(N5,K,L)*RR
      Q3         = RV(N5,K,L)*RR
      Q4         = RW(N5,K,L)*RR
      Q5         = E(N5,K,L)/GAMMA
      PP         =          GD*(Q5-.5*(Q2**2+Q3**2+Q4**2)*Q1)
      CPTMP          = (PP-1.)*CPS
      QQ=SQRT(Q2*Q2+Q3*Q3+Q4*Q4)
        Q2=Q2/FSMACH
CC      Q3=Q3/FSMACH
CC      Q4=Q4/FSMACH

      IF(FORM)THEN
       WRITE(NTP,703)Q2,Q3,Q4,ZER,CPTMP,QQ,R(N5,K,L),Q5
      ELSE
       WRITE(NTP)Q2,Q3,Q4,ZER,CPTMP,QQ,R(N5,K,L),Q5
      ENDIF

   10 CONTINUE
    5 CONTINUE

      ELSEIF(IORDR.EQ.2)THEN
      DO 15 J=N3,N4-1
      DO 20 L=N1,N2-1
      RR         = 1./R(J,N5,L)
      Q1         = R(J,N5,L)
      Q2         = RU(J,N5,L)*RR
      Q3         = RV(J,N5,L)*RR
      Q4         = RW(J,N5,L)*RR
      Q5         = E(J,N5,L)/GAMMA
      PP         =          GD*(Q5-.5*(Q2**2+Q3**2+Q4**2)*Q1)
      CPTMP          = (PP-1.)*CPS
      QQ=SQRT(Q2*Q2+Q3*Q3+Q4*Q4)
        Q2=Q2/FSMACH
CC      Q3=Q3/FSMACH
CC      Q4=Q4/FSMACH

      IF(FORM)THEN
```

```
      WRITE(NTP,703)Q2,Q3,Q4,ZER,CPTMP,QQ,R(J,N5,L),Q5
     ELSE
      WRITE(NTP)Q2,Q3,Q4,ZER,CPTMP,QQ,R(J,N5,L),Q5
     ENDIF

  20 CONTINUE
  15 CONTINUE

     ELSEIF(IORDR.EQ.3)THEN
     DO 25 K=N3,N4-1
     DO 30 J=N1,N2-1
     RR          = 1./R(J,K,N5)
     Q1          = R(J,K,N5)
     Q2          = RU(J,K,N5)*RR
     Q3          = RV(J,K,N5)*RR
     Q4          = RW(J,K,N5)*RR
     Q5          = E(J,K,N5)/GAMMA
     PP          =          GD*(Q5-.5*(Q2**2+Q3**2+Q4**2)*Q1)
     CPTMP          = (PP-1.)*CPS
     QQ=SQRT(Q2*Q2+Q3*Q3+Q4*Q4)
CC     Q2=Q2/FSMACH
CC     Q3=Q3/FSMACH
CC     Q4=Q4/FSMACH

     IF(FORM)THEN
      WRITE(NTP,703)Q2,Q3,Q4,ZER,CPTMP,QQ,R(J,K,N5),Q5
     ELSE
      WRITE(NTP)Q2,Q3,Q4,ZER,CPTMP,QQ,R(J,K,N5),Q5
     ENDIF

  30 CONTINUE
  25 CONTINUE
     ENDIF

 703 FORMAT(6E12.5)
     RETURN
     END

     SUBROUTINE OFFB(NB,IORDR,N1,N2,N3,N4,N5)
     PARAMETER(NTB=3,JMX=230,KMX=32,LMX=50)
     COMMON/VISFAC/ ALPHA,BETA,XMACH,C2B,C2BP,VRAT,RE
     DIMENSION R(JMX,KMX,LMX),RU(JMX,KMX,LMX),RV(JMX,KMX,LMX),
    1  RW(JMX,KMX,LMX),E(JMX,KMX,LMX),X(JMX,KMX,LMX),Y(JMX,KMX,LMX),
    2  Z(JMX,KMX,LMX)
     COMMON/DATA/ NBLK, JMAX(NTB), KMAX(NTB), LMAX(NTB)
     LOGICAL FORM
     DATA FORM/.TRUE./, GAMMA/1.4/, GAMI/0.4/

     FSMACH = XMACH
     GD=GAMMA*GAMI
     CPS = 1./(0.5*GAMMA*FSMACH*FSMACH)
     NTP = 9
     ZER=0.
     IONE=1
     IZERO=0
     JN=N2-N1+1
     KN=N4-N3+1

     READ(9+NB) ((( X(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
    *       L=1,LMAX(NB)),
```

```
*         ((( Y(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
*        L=1,LMAX(NB)),
*         ((( Z(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
*        L=1,LMAX(NB))
 REWIND 9+NB

 READ(29+NB)((( R(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
*        L=1,LMAX(NB)),
*         (((RU(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
*        L=1,LMAX(NB)),
*         (((RV(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
*        L=1,LMAX(NB)),
*         (((RW(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
*        L=1,LMAX(NB)),
*         ((( E(J,K,L),J= 1,JMAX(NB)),K=1,KMAX(NB)),
*        L=1,LMAX(NB))
 REWIND 29+NB

 IF(FORM)THEN
  WRITE(NTP,701)          'STRT','BLCK'
  WRITE(NTP,702)          IONE,IZERO,IZERO,IZERO,IZERO
  WRITE(NTP,702)          KN,JN,IZERO,IZERO,IZERO
 ELSE
  WRITE(NTP)          'STRT','BLCK'
  WRITE(NTP)          IONE,IZERO,IZERO,IZERO,IZERO
  WRITE(NTP)          KN,JN,IZERO,IZERO,IZERO
 ENDIF

 IF(IORDR.EQ.1)THEN
 DO 5 K=N3,N4
 DO 10 L=N1,N2
 IF(ABS(R(N5,K,L)).GT.1.E-10)THEN
   RR          = 1./R(N5,K,L)
 ELSE
   RR          = 1.
 ENDIF
 Q1          = R(N5,K,L)
 Q2          = RU(N5,K,L)*RR
 Q3          = RV(N5,K,L)*RR
 Q4          = RW(N5,K,L)*RR
 Q5          = E(N5,K,L)/GAMMA
 PP          =          GD*(Q5-.5*(Q2**2+Q3**2+Q4**2)*Q1)
 CPTMP          = (PP-1.)*CPS
 QQ=SQRT(Q2*Q2+Q3*Q3+Q4*Q4)

 IF(FORM)THEN
  WRITE(NTP,703)X(N5,K,L),Y(N5,K,L),Z(N5,K,L),Q2,Q3,Q4,
1  CPTMP,Q5
 ELSE
  WRITE(NTP)X(N5,K,L),Y(N5,K,L),Z(N5,K,L),Q2,Q3,Q4,
1  CPTMP,Q5
 ENDIF

10 CONTINUE
 5 CONTINUE

 ELSEIF(IORDR.EQ.2)THEN
 DO 15 J=N3,N4
 DO 20 L=N1,N2
 IF(ABS(R(J,N5,L)).GT.1.E-10)THEN
```

```
      RR          = 1./R(J,N5,L)
    ELSE
      RR          = 1.
    ENDIF
    Q1          = R(J,N5,L)
    Q2          = RU(J,N5,L)*RR
    Q3          = RV(J,N5,L)*RR
    Q4          = RW(J,N5,L)*RR
    Q5          = E(J,N5,L)/GAMMA
    PP          =          GD*(Q5-.5*(Q2**2+Q3**2+Q4**2)*Q1)
    CPTMP          = (PP-1.)*CPS
    QQ=SQRT(Q2*Q2+Q3*Q3+Q4*Q4)

    IF(FORM)THEN
     WRITE(NTP,703)X(J,N5,L),Y(J,N5,L),Z(J,N5,L),Q2,Q3,Q4,
   1  CPTMP,Q5
     ELSE
     WRITE(NTP)X(J,N5,L),Y(J,N5,L),Z(J,N5,L),Q2,Q3,Q4,
   1  CPTMP,Q5
     ENDIF

  20 CONTINUE
  15 CONTINUE

    ELSEIF(IORDR.EQ.3)THEN
    DO 25 K=N3,N4
    DO 30 J=N1,N2
    IF(ABS(R(J,K,N5)).GT.1.E-10)THEN
      RR          = 1./R(J,K,N5)
    ELSE
      RR          = 1.
    ENDIF
    Q1          = R(J,K,N5)
    Q2          = RU(J,K,N5)*RR
    Q3          = RV(J,K,N5)*RR
    Q4          = RW(J,K,N5)*RR
    Q5          = E(J,K,N5)/GAMMA
    PP          =          GD*(Q5-.5*(Q2**2+Q3**2+Q4**2)*Q1)
    CPTMP          = (PP-1.)*CPS
    QQ=SQRT(Q2*Q2+Q3*Q3+Q4*Q4)

    IF(FORM)THEN
     WRITE(NTP,703)X(J,K,N5),Y(J,K,N5),Z(J,K,N5),Q2,Q3,Q4,
   1  CPTMP,Q5
     ELSE
     WRITE(NTP)X(J,K,N5),Y(J,K,N5),Z(J,K,N5),Q2,Q3,Q4,
   1  CPTMP,Q5
     ENDIF

  30 CONTINUE
  25 CONTINUE
     ENDIF

 701 FORMAT(2A4)
 702 FORMAT(5I5)
  03 FORMAT(6E12.5)
 /03 FORMAT(6E20.9)
 704 FORMAT(3I5/(6E12.5))
 705 FORMAT(6E12.5/2E12.5,I5)
 706 FORMAT(6E12.5)
```

```
707 FORMAT(20A4)
    RETURN
    END
```

APPENDIX C

Included here is an example NPARC input file utilized in the zonal aerodynamic analysis of the Boeing 727-200 with horizontal and vertical tail leading-edge ice. This example is for the un-iced t-tail at zero degrees angle of attack and sideslip, Mach 0.12 and a Reynolds number 1.2 million. Note the use of the type 8 boundary condition and the LDIR variable in conjunction with the type 70 boundary conditions. The grid structure for this particular problem is summarized below:

Block 1	JMAX = 194	(Freestream direction)
	KMAX = 25	(Spanwise direction)
	LMAX = 50	(Normal direction)
Block 2	JMAX = 194	(Freestream direction)
	KMAX = 25	(Spanwise direction)
	LMAX = 49	(Normal direction)
Block 3	JMAX = 194	(Freestream direction)
	KMAX = 25	(Spanwise direction)
	LMAX = 49	(Normal direction)

For Block 1, L=1 coincides with L=2 of Block 3 for so-called wake-cut treatment. For Block 2, K=1 and K=KMAX coincides K=KMAX-1 of Block 1 and K=2 of Block 3 respectively. These overlap regions were added automatically with the H-O topology option in PRPAMI. The horizontal and vertical tail surfaces are defined by the following indicial ranges:

SURFACE	BLOCK	J Range	K Range	L Range
Upper vertical	1	20 - 170	1 - 1	2 - 35
Upper horizontal	1	64 - 163	1 - 25	2 - 2
Horizontal tip	2	64 - 163	2 - 24	1 - 1
Lower horizontal	3	64 - 163	3 - 25	1 - 1
Lower vertical	3	20 - 170	25 - 25	1 - 49

```
$INPUTS
  NMAX=5000,      NP=3000,
  PREF=15.0,      TREFR=500.,
  IFXPRT=1,       NBLOCK=3,
  DIS2=0.16,      DIS4=0.32,        IMUTUR = 3,  IFILTR = 2,
  DTCAP=2.0,      PCOMAX=10.0,       IVARDT=2,
  NSPRT=50,       STOPL2=1.E-20,     STOPTR=50,  STOPTM ='00002500',
  RE = 1.2E+06,XMACH = 0.12,    NOBORT = 1,
  ALPHA = 0.0,    BETA = 0.0,
$END
$BLOCK :#1
  NPSEG=0,
  INVISC(1)=1,    INVISC(2)=1,    INVISC(3)=1,
  LAMIN(1) =0,    LAMIN(2) =0,    LAMIN(3) =1,
$END
$BOUNDS
  NJSEG=2,
    JLINE(1)=1,     JTYPE(1)=8,          JSIGN(1)=1,
      JKLOW(1)=2,   JKHIGH(1)=25,
      JLLOW(1)=1,   JLHIGH(1)=50,
      PRESSJ(1)=1.0,    TEMPJ(1)=1.0,
    JLINE(2)=194,    JTYPE(2)=8,          JSIGN(2)=-1,
      JKLOW(2)=2,   JKHIGH(2)=25,
      JLLOW(2)=1,   JLHIGH(2)=50,
      PRESSJ(2)=1.0,    TEMPJ(2)=1.0,
  NKSEG=5,
    KLINE(1)=1,     KTYPE(1)=60,         KSIGN(1)=1,
      KJLOW(1)=20,  KJHIGH(1)=170,
      KLLOW(1)=2,   KLHIGH(1)=35,        KEDGE(1)=25,
    KLINE(2)=25,    KTYPE(2)=70,         KSIGN(2)=-1,
      KJLOW(2)=1,   KJHIGH(2)=193,
      KLLOW(2)=2,   KLHIGH(2)=49,        INTERK(2)=1,
    KLINE(3)=1,     KTYPE(3)=50,         KSIGN(3)=1,
      KJLOW(3)=171,KJHIGH(3)=194,
      KLLOW(3)=1,   KLHIGH(3)=49,
    KLINE(4)=1,     KTYPE(4)=50,         KSIGN(4)=1,
      KJLOW(4)=1,   KJHIGH(4)=19,
      KLLOW(4)=1,   KLHIGH(4)=49,
    KLINE(5)=1,     KTYPE(5)=50,         KSIGN(5)=1,
      KJLOW(5)=20,  KJHIGH(5)=170,
      KLLOW(5)=36,  KLHIGH(5)=49,
  NLSEG=4,
    LLINE(1)=2,     LTYPE(1)=60,         LSIGN(1)=1,
      LJLOW(1)=64,  LJHIGH(1)=163,
      LKLOW(1)=1,   LKHIGH(1)=25,        LEDGE(1)=45,
    LLINE(2)=1,     LTYPE(2)=70,         LSIGN(2)=1,
      LJLOW(2)=164,  LJHIGH(2)=193,
      LKLOW(2)=2,   LKHIGH(2)=25,        INTERL(2)=2,     LDIR(2)=2,
    LLINE(3)=50,     LTYPE(3)=8,         LSIGN(3)=-1,
      LJLOW(3)=1,   LJHIGH(3)=194,
      LKLOW(3)=1,   LKHIGH(3)=25,
      PRESSL(3)=1.0, TEMPL(3)=1.0,
    LLINE(4)=1,     LTYPE(4)=70,         LSIGN(4)=1,
      LJLOW(4)=2,   LJHIGH(4)=64,
      LKLOW(4)=2,   LKHIGH(4)=25,        INTERL(4)=3,     LDIR(4)=2,
$
$BLOCK :#2
  NPSEG=0,
  INVISC(1)=1,    INVISC(2)=1,    INVISC(3)=1,
  LAMIN(1) =0,    LAMIN(2) =0,    LAMIN(3) =1,
$END
$BOUNDS
```

```
  NJSEG=2,
    JLINE(1)=1,     JTYPE(1)=8,          JSIGN(1)=1,
      JKLOW(1)=2,   JKHIGH(1)=24,
      JLLOW(1)=2,   JLHIGH(1)=48,
      PRESSJ(1)=1.0,  TEMPJ(1)=1.0,
    JLINE(2)=194,    JTYPE(2)=8,          JSIGN(2)=-1,
      JKLOW(2)=1,   JKHIGH(2)=25,
      JLLOW(2)=1,   JLHIGH(2)=49,
      PRESSJ(2)=1.0,    TEMPJ(2)=1.0,
  NKSEG=2,
    KLINE(1)=1,     KTYPE(1)=70,         KSIGN(1)=1,
      KJLOW(1)=1,   KJHIGH(1)=193,
      KLLOW(1)=1,   KLHIGH(1)=48,        INTERK(1)=1,
    KLINE(2)=25,    KTYPE(2)=70,         KSIGN(2)=-1,
      KJLOW(2)=1,   KJHIGH(2)=193,
      KLLOW(2)=1,   KLHIGH(2)=48,        INTERK(2)=4,
  NLSEG=4,
    LLINE(1)=1,     LTYPE(1)=60,         LSIGN(1)=1,
      LJLOW(1)=64,  LJHIGH(1)=163,
      LKLOW(1)=1,   LKHIGH(1)=25,        LEDGE(1)=45,
    LLINE(2)=1,     LTYPE(2)=82,         LSIGN(2)=1,
      LJLOW(2)=164,  LJHIGH(2)=194,
      LKLOW(2)=2,   LKHIGH(2)=24,
    LLINE(3)=49,     LTYPE(3)=8,         LSIGN(3)=-1,
      LJLOW(3)=1,   LJHIGH(3)=194,
      LKLOW(3)=1,   LKHIGH(3)=25,
      PRESSL(3)=1.0, TEMPL(3)=1.0,
    LLINE(4)=1,     LTYPE(4)=82,         LSIGN(4)=1,
      LJLOW(4)=1,   LJHIGH(4)=64,
      LKLOW(4)=2,   LKHIGH(4)=24,
$
$BLOCK :#3
  NPSEG=0,
  INVISC(1)=1,   INVISC(2)=1,    INVISC(3)=1,
  LAMIN(1) =0,   LAMIN(2) =0,    LAMIN(3) =1,
$END
$BOUNDS
  NJSEG=2,
    JLINE(1)=1,     JTYPE(1)=8,          JSIGN(1)=1,
      JKLOW(1)=1,   JKHIGH(1)=24,
      JLLOW(1)=1,   JLHIGH(1)=48,
      PRESSJ(1)=1.0,  TEMPJ(1)=1.0,
    JLINE(2)=194,    JTYPE(2)=8,          JSIGN(2)=-1,
      JKLOW(2)=1,   JKHIGH(2)=24,
      JLLOW(2)=1,   JLHIGH(2)=49,
      PRESSJ(2)=1.0,    TEMPJ(2)=1.0,
  NKSEG=4,
    KLINE(1)=25,     KTYPE(1)=60,        KSIGN(1)=-1,
      KJLOW(1)=20,  KJHIGH(1)=170,
      KLLOW(1)=1,   KLHIGH(1)=49,        KEDGE(1)=25,
    KLINE(2)=1,    KTYPE(2)=70,         KSIGN(2)=1,
      KJLOW(2)=1,   KJHIGH(2)=193,
      KLLOW(2)=1,   KLHIGH(2)=48,        INTERK(2)=4,
    KLINE(3)=25,    KTYPE(3)=50,         KSIGN(3)=-1,
      KJLOW(3)=171,KJHIGH(3)=194,
      KLLOW(3)=1,   KLHIGH(3)=49,
    KLINE(4)=25,     KTYPE(4)=50,        KSIGN(4)=-1,
      KJLOW(4)=1,   KJHIGH(4)=19,
      KLLOW(4)=1,   KLHIGH(4)=49,
  NLSEG=4,
    LLINE(1)=1,     LTYPE(1)=60,         LSIGN(1)=1,
      LJLOW(1)=64,  LJHIGH(1)=163,
      LKLOW(1)=1,   LKHIGH(1)=25,        LEDGE(1)=45,
```

```
    LLINE(2)=1,     LTYPE(2)=70,          LSIGN(2)=1,
      LJLOW(2)=164,  LJHIGH(2)=193,
      LKLOW(2)=1,   LKHIGH(2)=24,         INTERL(2)=2,      LDIR(2)=2,
    LLINE(3)=49,     LTYPE(3)=8,          LSIGN(3)=-1,
      LJLOW(3)=1,   LJHIGH(3)=194,
      LKLOW(3)=1,   LKHIGH(3)=25,
      PRESSL(3)=1.0, TEMPL(3)=1.0,
    LLINE(4)=1,     LTYPE(4)=70,          LSIGN(4)=1,
      LJLOW(4)=2,   LJHIGH(4)=64,
      LKLOW(4)=1,   LKHIGH(4)=24,         INTERL(4)=3,      LDIR(4)=2,
$END
```

APPENDIX D

Included here is a discussion of the specific additions made to the NPARC input stream. The user should consult the NPARC users manual[2] as a reference for this discussion. Additional pages to be inserted into the VSAERO Revision E.5 User's Manual are also included.

SUMMARY OF NPARC INPUT MODIFICATIONS

The parameter STOPTM was added to the namelist INPUTS section, (manual, page 69). This variable acts much like STOPTR (manual, page 83), but it represents the specific time the job execution should terminate normally. The format is in month, day, hour and minute format (MMDDHHMM). This option is useful for dedicated time execution on a CRAY computer.

The boundary condition type 8 was added to the namelist BOUNDS section which allows the specification of external boundary flow-conditions via the data file PARC.SCN (manual, page 53). This option acts much the same as the freestream type 7 boundary condition except the flow takes on local characteristics provided by the PARC.SCN data file. Further information may be found in section 2.3 of this report.

The input variables JDIR, KDIR and LDIR were added to the namelist BOUNDS section to remove a built-in directionality in the type 70 boundary conditions (manual, page 42). These parameters represent arrays similar to the other input within the BOUNDS group (manual, page 57). The user is given control of the directionality of a boundary condition region. The values for JDIR, KDIR and LDIR are summarized below:

Segment type	JDIR	KDIR	LDIR	Value	Action
JLINE	x			2	Reverses K index
	x			3	Reverses L index
KLINE		x		1	Reverses J index
		x		3	Reverses L index
LLINE			x	1	Reverses J index
			x	2	Reverses K index

Any other value for these variables will not modify the directionality of the boundary condition region.

4.3.5 OFF-BODY VELOCITY SCAN INPUT

CARD 27: SCAN BOX.

Cols	Variable	Value	Description	Frmt
1-5	***MOLD***		Shape of velocity scan volume, see Sections 3.9.1—3.9.7	6I5
		0	Stops the scan	
		1	Skewed box option: allows a single point, points along a straight line or straight lines within a parallelogram or within a parallelepiped. Requires CARD 28, etc.	
		2	Allows points along radial lines in a cylindrical volume. Requires CARD 29, etc.	
		3	x,y,z array. Requires CARD 30	
		4	Single panel-to-panel (center-to-center) scan. Requires CARD 31	
		5	Multiple panel-to-panel scan. Requires CARD 32	
		6	Single panel-to-point scan. Requires CARD 33	
		7	Multiple panel-to-point scan. Requires CARD 32 and CARD 34	
		9	X,Y,Z array for zonal NS applications. Requires CARD 34.1.	

CARD 27: SCAN BOX. (Cont'd)

Cols	Variable	Value	Description			
6-10	*MEET*		Controls the intersection line routine (see 3.9.1)			
			Interpolat'n[1]	Moved[2]	Flagged[3]	B.L.[4]
		0	Yes	Yes	Yes	No
		1	No	No	No	No
		2	Yes	No	Yes	No
		-1	No	Yes	Yes	No
		-2	No	No	Yes	No
		-3	No	Yes	Yes	Yes

Note:[1] Interpolation is used along the scan line to set velocity vectors on points inside panel near-field regions using velocities from outside the region and surface intersection point velocities.

Note:[2] Inside points just adjacent to a surface intersection are moved to the intersection point and take the surface velocity value.

Note:[3] Points inside bodies are flagged as such.

Note:[4] Include boundary layer profile if NVPI > 1.

11-15	*NEAR*		Controls the near-field routine in the velocity calculation (see 3.9.1)
		0	Active
		1	Active only for surface panels (wake near field off)
		-1	Inactive

Note: Meet of -3 recommended for MOLD 9 option.

CARD 34.1: **GLOBAL COORDINATE ARRAY.** (Include if MOLD=9 on CARD 27)

Cols	Variable	Value	Description	Frmt
1-5 6-10 11-15	NP1 NP2 NP3		Dimensions of array of points entered on CARD 34.2. If NP1 <0 array of points read from separate Unit 31 data file, NAME.SCI.	4I5
16-20	IPVPRT	0	Suppress print of results	
		1	Print results	
75A	NAME		Prefix of scan output data file path name and optional scan input data file path name left adjusted, (e.g., NAME.SCO, NAME.SCI).	

CARD 34.2: (Include NP1 • NP2 • NP3 points)

Cols	Variable	Value	Description	Frmt
1-10 11-20 21-30	X(J,K,L) Y(J,K,L) Z(J,K,L)		Global coordinates of scan point. Order of input is (((J=1,NP1),K=1, NP2),L=1,NP3)	3F10.
31-35	NB		Block index number transferred to NAME.SCO data file.	
36-40	JP		Grid point indicial location (J,K,L), transferred to NAME.SCO data file.	
41-45	KP			
46-50	LP			

Note: Format of optional NAME.SCI input data file is a repeat of CARD 34.1 and the CARD 34.2 set. This option is useful for very large data input arrays.

APPENDIX E

ENHANCEMENT OF THE DHC-6-200 TWIN OTTER PANEL MODEL

The existing VSAERO panel model of the DHC-6-200 Twin Otter was refined and enhanced under Task 5.3 of the contract. Initially, the existing VSAERO model was reviewed and compared to a geometry description of the aircraft supplied by DeHavilland Canada. Several areas of the existing panel model were reworked to comply with the available aircraft documentation. These areas included the nose of the fuselage, the wing and the vertical and horizontal tail definition.

The fuselage was first rebuilt, with a continuous maximum width line established through the nose. Based upon the DeHavilland documentation, the wing was revised and the intersection with the fuselage and the engine nacelle was determined. Similarly, the vertical and horizontal tail was rebuilt and a new intersection was defined. The basic refined model is presented in Fig. E.1.

Further, the wing lift struts, main landing gear strut and wheel and main gear/strut attach-fitting-faring were defined. Also, the nose landing gear strut and wheel were added to the model. All struts were modeled in their in-flight, extended positions. This enhanced model is shown in Fig. E.2.

Finally, an actuator disk propeller model was included for simulation of power effects. The wake system attached to this disk provides a model of the propeller swirl. As the swirl must have a specified rotation, it was decided to represent the propeller on the starboard side of the aircraft. Since a symmetric panel model was previously assumed, the port propeller is represented as the mirror image of the starboard propeller, which results in a reverse rotation to that on the actual aircraft. This simplification may be removed by explicitly defining both port and starboard sides. The resulting model of the DHC-6-200 Twin Otter is presented in Fig. E.3.

At each stage described here, fine-tuning of the panel model was accomplished by examination of the VSAERO flow solution and subsequent repair as necessary.

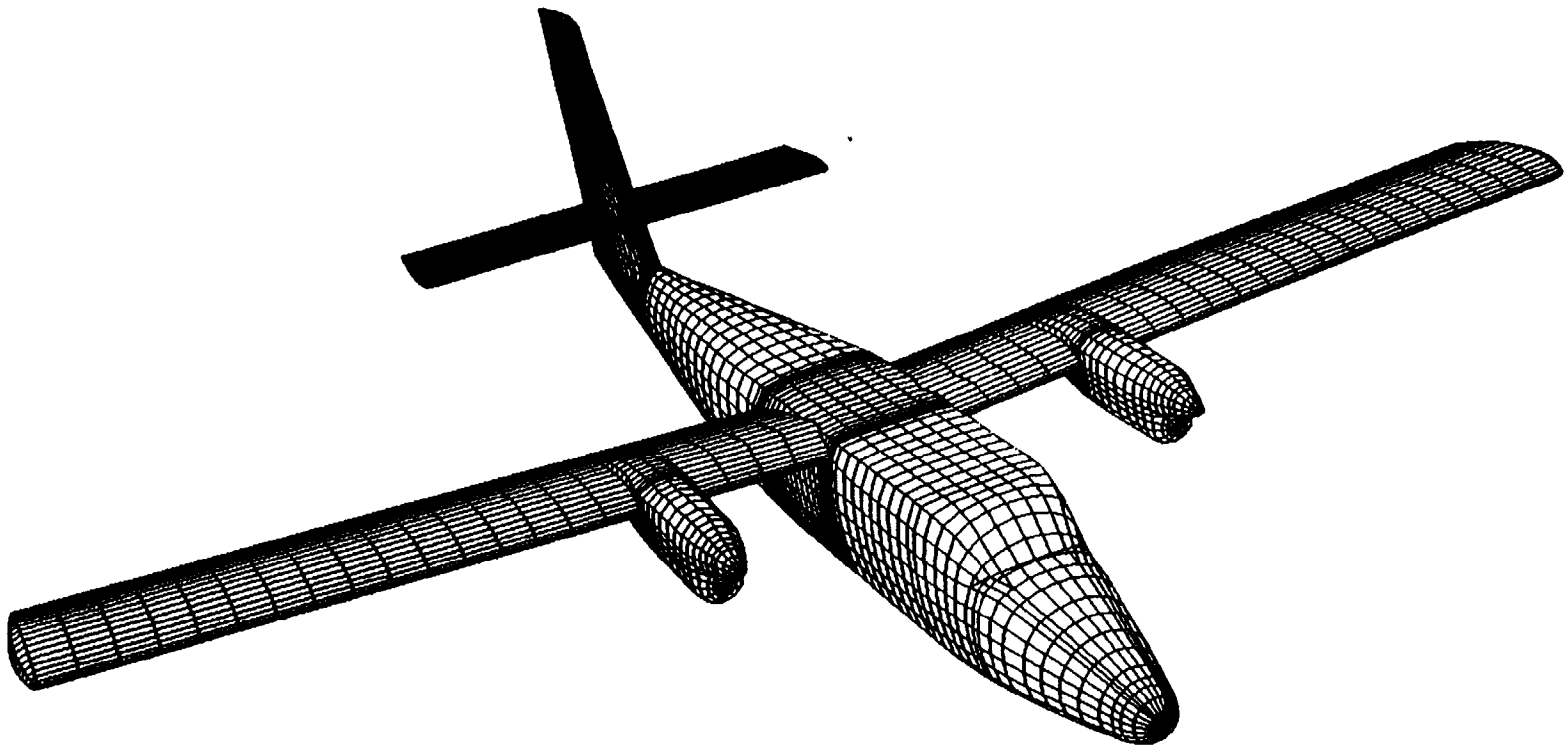


Fig. E.1. Baseline Panel Model of the DHC-6-200 Twin Otter.

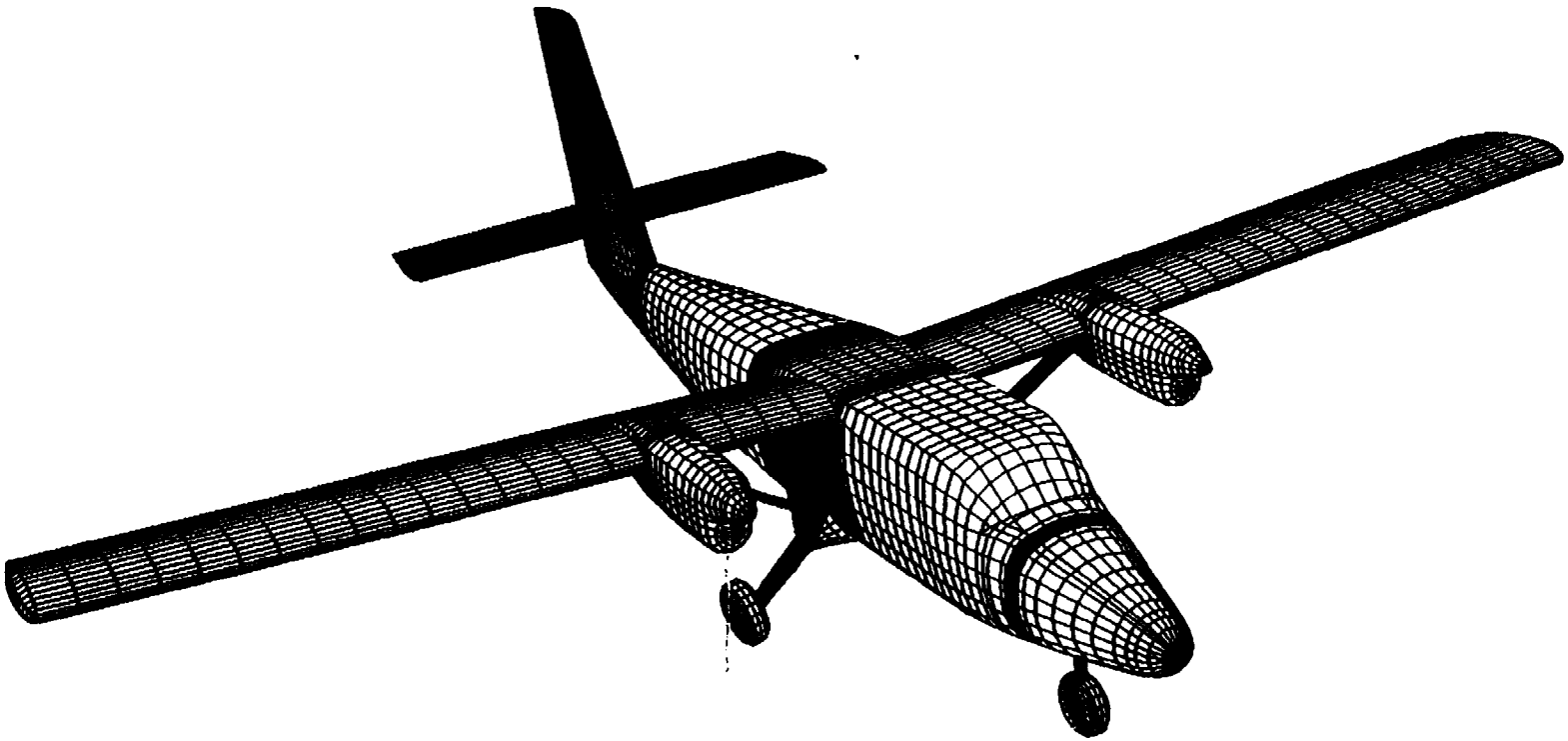


Fig. E.2. Front View of the Enhanced Panel Model of the DHC-6-200 Twin Otter.

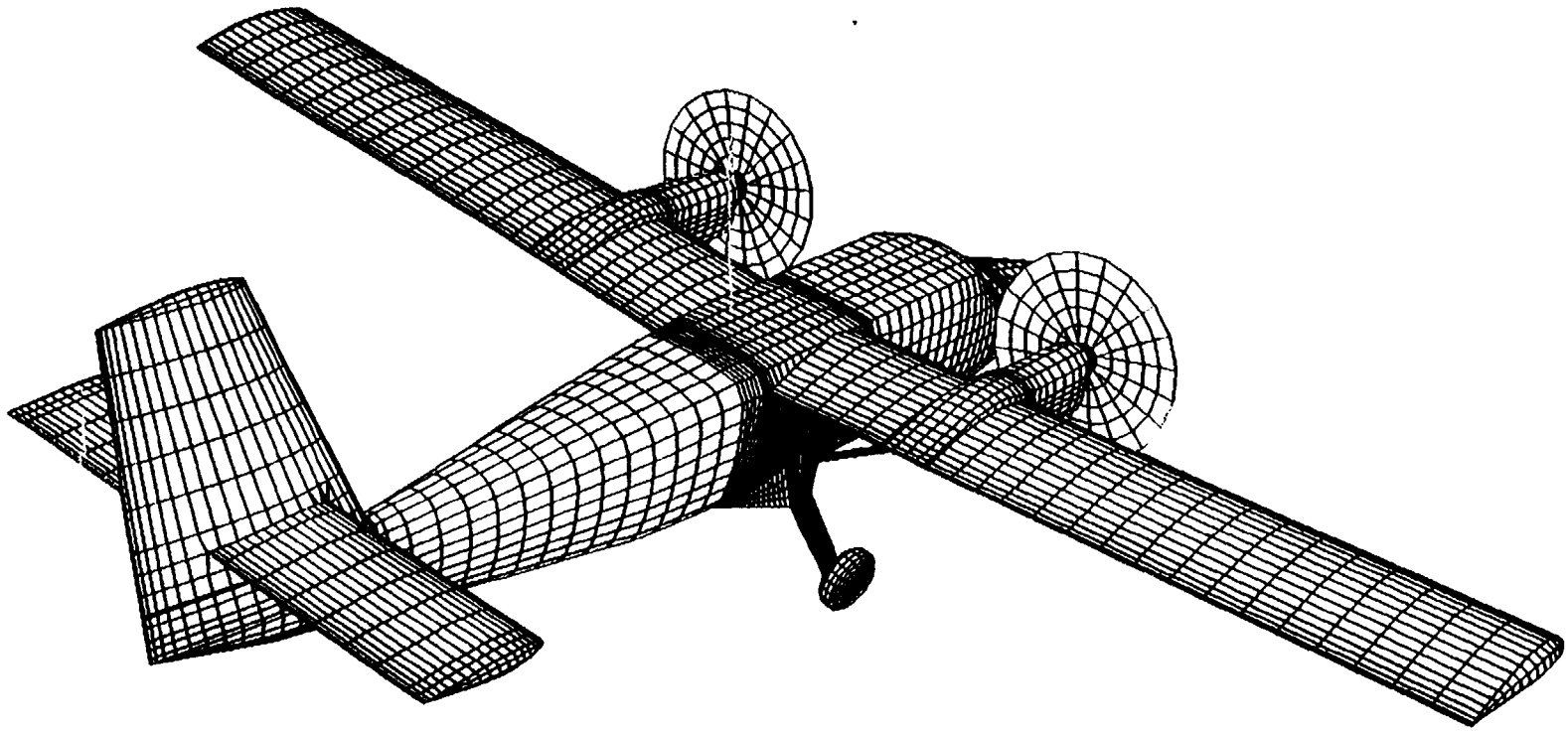


Fig. E.3. Rear View of the Enhanced Panel Model of the DHC-6-200 Twin Otter With Propeller Simulation.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1996	Final Contractor Report

4. TITLE AND SUBTITLE

Development of Three-Dimensional Flow Code Package to Predict Performance and Stability of Aircraft With Leading Edge Ice Contamination

5. FUNDING NUMBERS

WU-505-68-10
C-NAS3-26310

6. AUTHOR(S)

D.J. Strash and J.M. Summa

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Analytical Methods, Inc.
2133 152nd Avenue N.E.
Redmond, Washington 98052

8. PERFORMING ORGANIZATION REPORT NUMBER

E-10399

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA CR-198519
AMI Report No. 9408

11. SUPPLEMENTARY NOTES

Project Manager, Andrew L. Reehorst, Propulsion Systems Division, NASA Lewis Research Center, organization code 2720, (216) 433-3938.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 02

This publication is available from the NASA Center for AeroSpace Information, (301) 621-0390.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

In the work reported herein, a simplified, uncoupled, zonal procedure is utilized to assess the capability of numerically simulating icing effects on a Boeing 727-200 aircraft. The computational approach combines potential flow plus boundary layer simulations by VSAERO for the un-iced aircraft forces and moments with Navier-Stokes simulations by NPARC for the incremental forces and moments due to iced components. These are compared with wind tunnel force and moment data, supplied by the Boeing Company, examining longitudinal flight characteristics. Grid refinement improved the local flow features over previously reported work with no appreciable difference in the incremental ice effect. The computed lift curve slope with and without empennage ice matches the experimental value to within 1%, and the zero lift angle agrees to within 0.2 of a degree. The computed slope of the un-iced and iced aircraft longitudinal stability curve is within about 2% of the test data. This work demonstrates the feasibility of a zonal method for the icing analysis of complete aircraft or isolated components within the linear angle of attack range. In fact, this zonal technique has allowed for the viscous analysis of a complete aircraft with ice which is currently not otherwise considered tractable.

14. SUBJECT TERMS

Aircraft icing; Computational fluid dynamics; Grid generation; Navier-Stokes equation; Panel method; Multigrid methods

15. NUMBER OF PAGES

98

16. PRICE CODE

A05

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	