AN INITIAL INVESTIGATION INTO METHODS OF COMPUTING
TRANSONIC AERODYNAMIC SENSITIVITY COEFFICIENTS

aerospace
engineering
department

Semiannual Progress Report
July 1990 -- December 1990

TEXAS A&M UNIVERSITY

TAMRF Report No. 5802-91-01

February 1991

NASA Grant No. NAG-1-793

Leland A. Carlson
Professor of Aerospace Engineering
Texas A&M University
College Station, TX 77843-3141
AN INITIAL INVESTIGATION INTO METHODS OF COMPUTING
TRANSONIC AERODYNAMIC SENSITIVITY COEFFICIENTS

Semiannual Progress Report
July 1990 -- December 1990

TAMRF Report No. 5802-91-01
February 1991
NASA Grant No. NAG 1 - 793

Leland A. Carlson
Professor of Aerospace Engineering
Texas A&M University
College Station, Texas 77843-3141
AN INITIAL INVESTIGATION INTO METHODS OF COMPUTING TRANSONIC AERODYNAMIC SENSITIVITY COEFFICIENTS

I. Introduction

This report covers approximately the period from July 1990 thru December 1990. During this reporting period, work has continued on studies necessary to develop the "quasi-analytical" sensitivity method for three dimensional transonic flow about wings. In addition, initial numerical investigations have been carried out and some very preliminary results obtained.

II. Personnel

The individuals associated with this project during the present reporting period have been Dr. Leland A. Carlson, Principal Investigator, and Hesham Elbanna, Graduate Research Assistant. Mr. Elbanna has been partially supported by the project during this period.

III. Research Progress

The efforts during the past six months and the current status of the project are summarized by a report prepared by Mr. Elbanna and contained herein as Appendix I. (Note that Appendix I contains subappendices A thru D.) As can be seen from this appendix, the primary effort has been the continued development of the three-dimensional quasi-analytical sensitivity analysis and the ancillary driver programs needed to carry out the studies and perform comparisons. Currently, the code is essentially contained in one unified package which includes the following:

(a) A three dimensional transonic wing analysis program (ZEBRA),
(b) A quasi-analytical portion which determines the matrix elements in the quasi-analytical equations,
(c) A method for computing the sensitivity coefficients from the resulting quasi-analytical equations,
(d) A package to determine for comparison purposes sensitivity coefficients via the finite difference approach, and,
(e) A graphics package.

The total program currently consists of about ten thousand FORTRAN statements, although it is hoped that this can be shortened significantly as the research progresses. Further, in the portion which determines the matrix elements, a major portion of the code from a time standpoint is for each grid only run once to determine symbolic logic that indicates where the non-zero elements are in the matrix. Once this portion is executed, a typical run requires 2-3 min for the transonic analysis, about 10 min for the quasi-analytical setup and solution (relatively independent of the number of design variables), about 2-3 minutes for a finite difference sensitivity analysis for each design variable, plus the time associated with graphical output. These times are all for the IBM 3090 at the TAMU Computer Services Center.

Thus, at this point the quasi-analytical approach and the finite difference approach each require about the same amount of computer time if only two design variables are considered. However, as the number of design variables is increased and as the quasi-analytical method is made more efficient, it is anticipated that the latter approach will be faster and more efficient.

One of the advances made during the last six months has been the investigation of various solvers for the sensitivity equations. As a result, the present scheme now uses an iterative conjugate gradient method and the generalized minimum residual algorithm (GEMRES). These
approaches appear to be very efficient and for the present test case only require a total memory for the entire code of 40 Mb. (Note that in the Appendix I, it is stated that the memory requirements are 90Mb. The larger value was initially used to ensure adequate allocation. However, it has since been determined that 40Mb at the most is actually needed.)

As indicated in Appendix I, some very preliminary results have been obtained with both the finite difference approach and the quasi-analytical method. However, as can be seen by looking at the results, the current quasi-analytical results are in error. Since this appendix was prepared, an error has been discovered in the coding for the determination of the quasi-analytical matrix elements associated with the wing boundary conditions and the wake. Consequently, the various MACSYMA codes are being re-run in order to generate the "correct" FORTRAN code. However, this is a lengthy process; and new results will probably not be available for this report.

In any event, it is believed that steady progress is being made and that useful results will be obtained soon.

IV. Project Status

During this period, additional funds were awarded to the Grant to cover the period 1 June 1990 thru 31 December 1990; and a renewal proposal to cover another twelve months was submitted. Subsequently, the facilitate interfacing with the renewal, the present period was extended thru February 28th 1991. It is anticipated that the renewal funds will be available March 1, 1991.
V. Future Efforts

During the next six months, work will continue on developing the quasi-analytical approach. In addition to debugging the program etc. and obtaining correct answers, emphasis will be placed on making the quasi-analytical method more efficient with respect to both CPU time and storage requirements. Further, work will be initiated to handle additional design variables, to extend the method to transonic and supersonic freestreams, and to generalize the geometry specification. Also, after appropriate discussions with personal at NASA Langley Research Center, consideration will be given to developing the quasi-analytical approach for a three-dimensional small perturbation potential code, which would be supplied by NASA Langley. The latter effort would allow comparison with the sensitivity results obtained using a full potential code.

VI. Technical Monitor

The technical monitor for this project is Dr. E. Carson Yates, Jr., Interdisciplinary Research Office, NASA Langley, Research Center.
APPENDIX I

Determination of Aerodynamic Sensitivity Derivatives
Based on the Full Potential Equation

H. M. Elbanna

January 1991
DETERMINATION OF AERODYNAMIC SENSITIVITY DERIVATIVES
BASED ON THE FULL POTENTIAL EQUATION

Prof. L.A. CARLSON
H.M. ELBANNA, (January, 1991)

Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANOFI</td>
<td>Boundary condition term ( ANOFI(j, k) )</td>
</tr>
<tr>
<td>AJ1, AJ2</td>
<td>Metric functions ( AJ1(j), AJ2(j) )</td>
</tr>
<tr>
<td>A1K, A2K</td>
<td>Metric functions ( A1K(k), A2K(k) )</td>
</tr>
<tr>
<td>( Cp )</td>
<td>Pressure coefficient</td>
</tr>
<tr>
<td>( c(y) )</td>
<td>Chord function</td>
</tr>
<tr>
<td>CIR</td>
<td>Circulation ( CIR(j) )</td>
</tr>
<tr>
<td>DPU</td>
<td>Wing upper surface boundary term</td>
</tr>
<tr>
<td>DPLO</td>
<td>Wing lower surface boundary term</td>
</tr>
<tr>
<td>DXII</td>
<td>Metric function ( DXII(i) )</td>
</tr>
<tr>
<td>ILE</td>
<td>I-location of leading edge</td>
</tr>
<tr>
<td>ITE</td>
<td>I-location of trailing edge</td>
</tr>
<tr>
<td>J</td>
<td>Jacobian ( X_j )</td>
</tr>
<tr>
<td>KUP</td>
<td>K-location of plane above wing</td>
</tr>
<tr>
<td>KLOW</td>
<td>K-location of plane below wing</td>
</tr>
<tr>
<td>M</td>
<td>Local Mach number ( M_{i,j,k} )</td>
</tr>
<tr>
<td>( M_c )</td>
<td>Cutoff Mach number ( 0.94 \leq M_c \leq 1.0 )</td>
</tr>
<tr>
<td>( M_\infty )</td>
<td>Freestream Mach number</td>
</tr>
<tr>
<td>( P_\infty )</td>
<td>Freestream pressure, nondimensionalized by ( \frac{2\gamma}{\gamma + 1}P_0 )</td>
</tr>
<tr>
<td>( P_0 )</td>
<td>Stagnation pressure</td>
</tr>
<tr>
<td>( q_\infty )</td>
<td>Freestream velocity, nondimensionalized by ( V^* )</td>
</tr>
<tr>
<td>RIP</td>
<td>Retarded density coefficient ( RIP(j, k) = \bar{\rho}_{i+1/2,j,k} )</td>
</tr>
<tr>
<td>RIM</td>
<td>Retarded density coefficient ( RIM(j, k) = \bar{\rho}_{i-1/2,j,k} )</td>
</tr>
<tr>
<td>RJP</td>
<td>Retarded density coefficient ( RJP(j, k) = \bar{\rho}_{i,j+1/2,k} )</td>
</tr>
<tr>
<td>RJ</td>
<td>Retarded density coefficient ( RJ(j, k) = \bar{\rho}_{i,j-1/2,k} )</td>
</tr>
<tr>
<td>RKP</td>
<td>Retarded density coefficient ( RKP(j, k) = \bar{\rho}_{i,j,k+1/2} )</td>
</tr>
<tr>
<td>RK</td>
<td>Retarded density coefficient ( RK(j, k) = \bar{\rho}_{i,j,k-1/2} )</td>
</tr>
<tr>
<td>R1KU</td>
<td>Modified retarded density coefficient for wing upper surface</td>
</tr>
<tr>
<td>R1KU</td>
<td>Modified retarded density coefficient for wing lower surface</td>
</tr>
<tr>
<td>R2KW</td>
<td>Modified retarded density coefficient for upper surface</td>
</tr>
<tr>
<td>R2KP</td>
<td>Modified retarded density coefficient for wake lower surface</td>
</tr>
<tr>
<td>U,V,W</td>
<td>Contravariant velocity components in computational plane</td>
</tr>
<tr>
<td>x,y,z</td>
<td>Physical grid system</td>
</tr>
<tr>
<td>X,Y,Z</td>
<td>Computational coordinates aligned with wing</td>
</tr>
<tr>
<td>xle(y)</td>
<td>Leading edge function</td>
</tr>
<tr>
<td>XD</td>
<td>Vector of design variables</td>
</tr>
<tr>
<td>( \rho )</td>
<td>Density, nondimensionalized by ( \rho_0 )</td>
</tr>
<tr>
<td>( \rho_\infty )</td>
<td>Freestream density, nondimensionalized by ( \rho_0 )</td>
</tr>
<tr>
<td>( \rho_0 )</td>
<td>Stagnation density</td>
</tr>
<tr>
<td>( \bar{\rho} )</td>
<td>Retarded density coefficient</td>
</tr>
<tr>
<td>( \delta() )</td>
<td>First order backward difference operator</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>Angle of attack</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>Ratio of specific heats</td>
</tr>
<tr>
<td>( \sigma )</td>
<td>Switching function ( \sigma = 1 - \nu )</td>
</tr>
<tr>
<td>( \phi )</td>
<td>Reduced potential function</td>
</tr>
<tr>
<td>( \Phi )</td>
<td>Full potential function</td>
</tr>
</tbody>
</table>
Introduction

In this progress report, work carried out during the period from July 1990 to December 1990 will be outlined. In addition, various overall steps and equations related to the three-dimensional sensitivity project will be listed herein for future reference. At this stage, it is helpful to distinguish two main phases that characterize the three-dimensional analysis/sensitivity project. Phase one of this research\(^1\) was concerned with modifying the analysis (ZEBRA) program to suit the sensitivity study, developing FORTRAN subroutines to calculate sensitivity derivatives using the finite-difference method, and, developing MACSYMA/FORTRAN algorithms to calculate the sensitivity coefficients using the quasianalytical method. These tasks were finalized by an assembly procedure that aimed at combining the above mentioned subroutines into one FORTRAN program. The main advantages of having a single FORTRAN program to carry out various analysis/sensitivity case studies are the minimization of disk read/write operations and the ability to debug/test/append any future additions to the entire project with ease, comparability, and speed. The second phase of the project will be concerned with debugging operations, addition of design variables, increasing solver efficiency, and carrying out a variety of case studies. The sections covered in this report are as follows,

- Symbolic Differentiation of the Full Potential Residual Expression.
- Structure of the Analysis/Sensitivity FORTRAN Code.
- Linear Solvers for the Sensitivity Equation.
- Primary Results and Debugging Operations.
- Future Work.
- Further Theoretical Aspects.

It is to be noted that the following sections include the effort up to the current state of work progress, this state being at the junction between the first phase and the second phase of the analysis/sensitivity project.

Symbolic Differentiation of the Full Potential Residual Expression

Following the line of formulation adopted in the two-dimensional sensitivity study, the quasianalytical method applied to the three-dimensional full potential equation yields the sensitivity equation,

$$
\left[ \frac{\partial R_{t,j,k}}{\partial \phi_{n,j,k}} \right] \left( \frac{\partial \phi_{n,j,k}}{\partial XD} \right) = - \left( \frac{\partial R_{t,j,k}}{\partial XD} \right)
$$

(1)

The residual expression of the full potential equation in conservative form (in the computational plane and using a shearing transformation) is written in terms of backward differences as,

$$
R_{t,j,k} = \delta x \left( \frac{\partial U}{\partial x} \right)_{t+1/2,j,k} + \delta y \left( \frac{\partial V}{\partial y} \right)_{t,j+1/2,k} + \delta z \left( \frac{\partial W}{\partial z} \right)_{t,j,k-1/2}
$$

(2)

The density is replaced by the retarded density coefficient in order to maintain stability in regions of supersonic flow. Therefore, Eq.(2) is written as,

$$
R_{t,j,k} = \delta x \left( \frac{\partial U}{\partial x} \right)_{t+1/2,j,k} + \delta y \left( \frac{\partial V}{\partial y} \right)_{t,j+1/2,k} + \delta z \left( \frac{\partial W}{\partial z} \right)_{t,j,k+1/2}
$$

(3)

$$
= \left[ \left( \frac{\partial U}{\partial x} \right)_{t+1/2,j,k} - \left( \frac{\partial U}{\partial x} \right)_{t-1/2,j,k} \right] + \left[ \left( \frac{\partial V}{\partial y} \right)_{t,j+1/2,k} - \left( \frac{\partial V}{\partial y} \right)_{t,j-1/2,k} \right] + \left[ \left( \frac{\partial W}{\partial z} \right)_{t,j,k+1/2} - \left( \frac{\partial W}{\partial z} \right)_{t,j,k-1/2} \right]
$$

(4)
In ZEBRA, Eq.(4) is coded as follows,

\[ R_{i,j,k} = (FIP - FIM) + (FJP - FJM) + (FKP - FKM) + \text{ANOFI} \]  

\[ = \text{RIP} U_{i-1/2,j,k - \text{RIM} U_{i+1/2,j,k - \text{RJP} V_{i-1/2,j,k - \text{RKP} W_{i-1/2,j,k}}} + \text{ANOFI} \]  

where

\[ \text{ANOFI}(i,j,k) = \begin{cases} 
-\Lambda_{33M} R1K DU, & k = KUP, \ ILE \leq i \leq ITE \\
\Lambda_{33P} R1KU DPL, & k = KLOW, \ ILE \leq i \leq ITE \\
\Lambda_{33M} R2KW CIR, & k = KUP, \ ITE < i \\
-\Lambda_{33M} R2KP CIR, & k = KLOW, \ ITE < i 
\end{cases} \]  

is the term that includes wing and wake boundary conditions. Note that the Jacobian is incorporated into the transformation coefficients of the contravariant velocity components. Next, the retarded density coefficients are given by,

\[ \text{RIP}(i,j,k) = (1 - \nu_{i-1/2,j,k}) \rho_{i+1/2,j,k} + \nu_{i+1/2,j,k} \rho_{i-1/2,j,k} \]  

\[ = \sigma_{i+1/2,j,k} (\rho_{i-1/2,j,k} - \rho_{i-1/2,j,k}) + \rho_{i-1/2,j,k} \]  

\[ \text{RJP}(i,j,k) = \frac{1}{2} (\tilde{\rho}_{i,j,k} + \tilde{\rho}_{i,j+1,k}) \]  

\[ \text{RKP}(i,j,k) = \frac{1}{2} (\tilde{\rho}_{i,j,k} + \tilde{\rho}_{i,j,k+1}) \]  

where

\[ \rho_{i,j,k} = \left[ 1 - \frac{\gamma - 1}{\gamma + 1} (U \Phi_X + V \Phi_Y + W \Phi_Z) \right]_{i,j,k}^{1/\gamma} \]  

\[ \nu_{i,j,k} = \min[1, \max(1 - \frac{M_c}{M_{i,j,k}}, 0)] \]  

Notice that the retarded density coefficient \( \text{RIP}(i,j,k) \) is evaluated only at the midsegment point \( i + 1/2, j, k \) while the values at \( i, j + 1/2, k \) and \( i, j, k + 1/2 \) \( [\text{RJP}(i,j,k) \) and \( \text{RKP}(i,j,k) \)] \) are obtained by averages of the surrounding points. The Mach number is obtained from the following relation,

\[ \frac{\rho_0}{\rho_{i,j,k}} = \left( \frac{T_0}{T} \right)^{1/\gamma} = (1 + \frac{\gamma - 1}{2} M_{i,j,k}^2) \gamma^{1/\gamma} \]  

and therefore,

\[ M_{i,j,k}^2 = \frac{2}{\gamma - 1} (\rho_{i,j,k}^{1-\gamma} - 1) \]  

where \( \rho_{i,j,k} \) is nondimensionalized by \( \rho_0 \). From Eq.(15) into Eq.(13),

\[ \nu_{i,j,k} = \begin{cases} 
0, & M_{i,j,k} < 1 \\
1 - \frac{(\gamma - 1)M_{i,j,k}^2}{\rho_{i,j,k}^{1-\gamma},} & M_{i,j,k} > 1 
\end{cases} \]  

and therefore,

\[ \nu_{i+1/2,j,k} = \begin{cases} 
0, & M_{i,j,k} < 1 \\
1 - \frac{(\gamma - 1)M_{i,j,k}^2}{(\rho_{i,j,k}^{1-\gamma}, \rho_{i+1/2,j,k}^{1-\gamma})^{1/\gamma},} & M_{i,j,k} > 1 
\end{cases} \]
The contravariant velocity components are given by,

\[ \mathbf{U} = (X_x^2 + X_y^2)\Phi_X + X_y\Phi_Y \]  

\[ V = X_y\Phi_X + \Phi_Y \]  

\[ W = \Phi_Z \]  

In order to improve the convergence of the analysis routine, the full potential is split into separate perturbation and freestream components as follows,

\[ \Phi_{i,j,k} = \phi_{i,j,k} - Xq_\infty Cos(\alpha) + Zq_\infty Sin(\alpha) \]  

Differentiating Eq.(22) with respect to \(X, Y, Z\) respectively,

\[ (\Phi_X)_i,j,k = DXIL(i)\Phi_{i-1,j,k} \]  

\[ (\Phi_Y)_i,j,k = DYIL(j)\Phi_{i,j-1,k} + Xq_\infty Cos(\alpha) \]  

\[ (\Phi_Z)_i,j,k = DZIL(k)\Phi_{i,j,k-1} + Zq_\infty Sin(\alpha) \]

where

\[ (\Phi_X)_i,j,k = A1K(k)(\Phi_{i-1,j,k} + \Phi_{i,j,k-1} - \Phi_{i,j,k} - \Phi_{i-1,j,k-1}) + A2K(k)(\Phi_{i-1,j,k} - \Phi_{i,j,k-1} + \Phi_{i-1,j,k-1}) + A2K(k)(\Phi_{i,j-1,k} - \Phi_{i,j,k})] / 2 \]  

Note that a shearing transformation is used to transform the physical grid system \((x,y,z)\) into a computational grid \((X,Y,Z)\) aligned with the wing. This transformation is given by,

\[ X(z,y) = \frac{x - xle(y)}{c(y)} \]  

\[ Y(y) = y \]  

\[ Z(z) = z \]
$$R_{i,j,k} = R_{i,j,k}(RIP, RIM, RJP, RJ, RKP, RK, U, V, W, ANOFI) \tag{32}$$

where

$$RIP = RIP(\Phi_X, \Phi_Y, \Phi_Z, U, V, W) \tag{33}$$

$$RIM = RIM(\Phi_X, \Phi_Y, \Phi_Z, U, V, W) \tag{34}$$

$$RJP = RJP(\Phi_X, \Phi_Y, \Phi_Z, U, V, W) \tag{35}$$

$$RJ = RJ(\Phi_X, \Phi_Y, \Phi_Z, U, V, W) \tag{36}$$

$$RKP = RKP(\Phi_X, \Phi_Y, \Phi_Z, U, V, W) \tag{37}$$

$$RK = RK(\Phi_X, \Phi_Y, \Phi_Z, U, V, W) \tag{38}$$

$$ANOFI = ANOFI(R1K, DPU, R1KU, DPLO, R2KW, R2KP, CIR) \tag{39}$$

and,

$$U = U(\Phi_X, \Phi_Y) \tag{40}$$

$$V = V(\Phi_X, \Phi_Y) \tag{41}$$

$$W = W(\Phi_Z) \tag{42}$$

$$\Phi_X = \Phi_X[\phi_{ii, jj, kk}, M_{\infty}, \alpha] \tag{43}$$

$$\Phi_Y = \Phi_Y[\phi_{ii, jj, kk}, M_{\infty}, \alpha] \tag{44}$$

$$\Phi_Z = \Phi_Z[\phi_{ii, jj, kk}, M_{\infty}, \alpha] \tag{45}$$

As mentioned above, once the program in Appendix A is executed, potential dependencies are used in symbolically differentiating the general residual expression and residual boundary updates (wing, wake, and right hand side vectors). This is achieved using the MACSYMA program given in Appendix B. The result of running the analytic differentiation program is a segment of FORTRAN subroutines presented in Appendix C. This segment of FORTRAN code is then transferred from the VAX machine and linked into the analysis/sensitivity program on the IBM-3090.

It is to be noted that previous work\textsuperscript{1,2} included operations similar to those mentioned above. However, residual updates were prepared separately using Eq.(5) with the last term 'ANOFI' (the term that includes wing and wake boundary conditions) replaced by the appropriate boundary terms, then each residual expression was simplified and differentiated using a different MACSYMA program. As a result, multiple MACSYMA codes (about six separate codes) had to be prepared to yield the required FORTRAN source segments. This resulted in a total size of about 12,000 lines of source code. No major problems were encountered in compiling this number of code lines since they were developed in the form of multiple subroutines. Currently, the same FORTRAN segments were reduced in size to about 7,000 lines of FORTRAN source code. This was achieved by handling both the general residual expression and the 'ANOFI' term separately thus cancelling repeated (or equivalent) portions of the FORTRAN code. Consequently, it should be noted
that the codes given in the Appendices are still being modified and optimized for size and speed and that
the enclosed versions of these codes (up to date versions) are still being debugged and refined.

For the current three-dimensional problem, design variables were previously defined as follows,
(a) Freestream design variables. These include the freestream Mach number and the angle of attack.
(b) Cross-section design variables. These include variables that define the airfoil section (such as maximum
thickness, maximum camber, and location of maximum camber) and variables that define the setting of
each spanwise section (such as geometric twist and dihedral).
(c) Planform design variables. These are variables that define the geometry of the wing planform.
These variables are used in preparing the right hand side vectors. In carrying out this step, the residual is
analytically differentiated with respect to each design variable and a corresponding segment of FORTRAN
code is generated. Refer to Appendices B and C for the details of these operations.

Finally, Appendix D includes a MACSYMA program to further process the results obtained from solving
the sensitivity equation. The result of running this program is a segment of FORTRAN code used to
calculate pressure coefficient sensitivity derivatives given the reduced potential sensitivity derivatives. A
transfer/link operation similar to the above is applied in order to merge this FORTRAN segment into the
analysis/sensitivity program.

Structure of the Analysis/Sensitivity FORTRAN Code

The analysis/sensitivity code is basically composed of the analysis program (ZEBRA), the finite-
difference sensitivity driver, and the quasianalytical sensitivity driver. Furthermore, graphics routines are
also included in the main code in order to assist in examining the results.

Execution of the main code starts thru an analysis (ZEBRA) run followed by sensitivity derivative
calculations. These calculations are carried out either using the finite-difference method or using the quasi-
analytical approach. The finite-difference portion of the code is set up to allow two consecutive ZEBRA
runs to be used to calculate a vector of sensitivity derivatives. This brute force technique while straight
forward in application has the disadvantages of being expensive to implement and exhibits accuracy prob-
lems. As for the quasianalytical sensitivity driver, it consists of two main parts. The first part is a group of
nested DO-LOOPS used to assemble the jacobian matrix and the right hand side vector(s). This is achieved
using calls to the FORTRAN segments generated via MACSYMA (see Appendix C). After the numerical
assembly step is completed, the second part of the sensitivity driver, a setup that allows execution of one
of several linear sparse solvers, is used to solve the sensitivity equation and yields the vector(s) of sensi-
tivity derivatives. Finally, the resulting sensitivity derivatives ($\partial \phi / \partial XD$) are further processed in order to
obtain pressure coefficient sensitivity derivatives ($\partial Cp / \partial XD$). This step is carried out separately using a
MACSYMA program that generates corresponding FORTRAN subroutines (see Appendix D).

Linear Solvers for the Sensitivity Equation

For the current three-dimensional problem and for the medium grid used, direct solvers that were
previously used in the two-dimensional problem (those based on tridiagonal decomposition and full Gaussian
elimination) failed to operate on the 3-D jacobian matrix basically due to memory limitations. On the other
hand, iterative routines developed earlier for the two-dimensional problem worked properly however turned
out to be somewhat slow. Later on, it was decided to try out some library routines that were available on
the IBM-3090. These turned out to be extremely efficient with regards to memory requirements and speed of
execution. Apparently, the reason for this efficiency lies in the ability of these routines to take advantage of
the IBM-3090 architecture and vectorization facility besides being written in machine code and optimized for
speed. In addition, the inclusion of these routines into the solver portion of the analysis/sensitivity program
turned out to be straightforward in the form of regular FORTRAN calls. Two scientific library solvers (based
on the iterative conjugate gradient method and the generalized minimum residual algorithm) were used with
success and a GO REGION of about 90MB was allocated in the JCL with no major problems. Notice that
the exact amount of storage needed for each of these solvers will depend on the structure of the jacobian
matrix (roughly, the structure is sparse and banded), the details of which will be determined at a later stage.
Primary Results and Debugging Operations

Currently, the MACSYMA codes are being debugged and revised to increase both the efficiency and handling of the resulting FORTRAN code segments. For example, as mentioned earlier, the last term in Eq.(5) is handled separately without revising Eq.(5) in its entirety. This has the advantage of reducing the size of both the MACSYMA program and FORTRAN generated segments. In addition, extensive debugging and review of the entire work will be performed in parallel to the above steps.

The sensitivity of the pressure coefficient \( C_p \) with respect to the design variables is obtained using \( \partial \phi / \partial X_D \). The expression for the pressure coefficient is,

\[
C_p = \frac{P - P_\infty}{\rho q_\infty^2 / 2}
\]

Substituting for the pressure using the isentropic relation, therefore

\[
C_p = \frac{(\gamma - 1)/\gamma}{\rho q_\infty^2} (\rho^* - \rho_\infty^*)
\]

where

\[
\rho = \left[ 1 - \frac{\gamma - 1}{\gamma - 1} \right] (U \Phi_X + V \Phi_Y + W \Phi_Z)
\]

and \( U, V, W, \Phi_X, \Phi_Y, \Phi_Z \) are given by equations (19)-(21) and (23)-(25) respectively. Notice also that the freestream values \( q_\infty, \rho_\infty, \text{and} P_\infty \) are obtained using the relations,

\[
q_\infty = \frac{\gamma + 1}{\gamma - 1 - 2/M_\infty^2}^{1/2}
\]

\[
\rho_\infty = \frac{\gamma - 1}{\gamma + 1} q_\infty^2 \frac{1/(\gamma - 1)}
\]

\[
P_\infty = \frac{\gamma + 1}{2\gamma} \rho_\infty
\]

Refer to Appendix D for the symbolic calculation of pressure coefficient sensitivity derivatives using reduced potential sensitivity derivatives.

Some primary results obtained by executing the analysis/sensitivity code about a fixed design point are also presented in this report following Appendix D. The planform used is that of an ONERA-M6 wing with a six percent noncambered parabolic-arc section and the flowfield (\( M_\infty = 0.8, \alpha = 0.0 \)) is computed on a 45*30*16 medium grid (i.e. symmetric subcritical flowfield). Figures (1) and (2) show the pressure coefficient for this subcritical case. Figures (3) and (4) include finite-difference pressure coefficient sensitivity derivatives with respect to Mach number and angle of attack respectively. Finally, Figures (5) and (6) contain the corresponding derivatives obtained by the quasianalytical method. Notice that the trends are different for both sets of the derivatives. It is believed that while the finite-difference results follow the trends obtained in the two-dimensional sensitivity study, the quasianalytical derivatives have different trends and therefore are in error. As mentioned earlier, debugging operations are underway with the finite-difference method being used as a reference for correct quasianalytical trends.

Future Work

As mentioned in the first section, the second phase of this project will be towards overall debugging of the analysis/sensitivity code with the objective being to match the sensitivity derivatives obtained thru the quasianalytical method with those derivatives obtained thru the finite-difference approach. Initially, focus will be on sensitivities with respect to freestream design variables (Mach number and angle of attack) followed by sensitivities with respect to both airfoil and planform design variables. It is to be noted that the inclusion of the later variables might require some sort of semi-analytical treatment to handle right hand side calculations corresponding to these variables. Next, various case studies will be conducted in order
to compare and improve on both the accuracy and efficiency of the quasianalytical and finite difference methods. This step will be followed by a physical interpretation of the results. Finally, minor modifications in the form of supersonic boundary conditions will be added to the analysis/sensitivity program in order to allow execution of supersonic test cases.

Further Theoretical Aspects

In some optimization studies, higher sensitivity derivatives might be needed. In general, it is possible to extend the quasianalytical approach in order to obtain second order sensitivity derivatives. The following ideas could be applied directly to the sensitivity equation. Consider the linear system,

\[ A \begin{bmatrix} X \\ XD_m \end{bmatrix} = B \]  

(52)

The sensitivity of \( X \) with respect to the elements of \( A \) and \( B \) \((XD_m)\) is obtained by differentiating Eq.(52) with respect to \( XD_m \),

\[ \begin{bmatrix} \frac{\partial A}{\partial XD_m} \\ \frac{\partial X}{\partial XD_m} \end{bmatrix} X - A \begin{bmatrix} \frac{\partial X}{\partial XD_m} \end{bmatrix} = \begin{bmatrix} \frac{\partial B}{\partial XD_m} \end{bmatrix} \]  

(53)

or,

\[ A^t \frac{\partial X}{\partial XD_m} = \left[ \frac{\partial B}{\partial XD_m} - \frac{\partial A}{\partial XD_m} X \right] \]  

(54)

Applying the above procedure to Eq.(1), second order sensitivity derivatives for the current three-dimensional problem could be obtained. The result is,

\[ \begin{bmatrix} \frac{\partial R_{i,j,k}}{\partial \phi_{i,j,k}} \end{bmatrix} \left( \frac{\partial^2 \phi_{i,j,k}}{\partial XD_m \partial XD} \right) = - \left( \frac{\partial^2 R_{i,j,k}}{\partial XD_m \partial XD} + \frac{\partial R_{i,j,k}}{\partial XD_m} \frac{\partial \phi_{i,j,k}}{\partial XD_m} \right) \]  

(55)

The first term in Eq.(55) is the \((n \times n)\) jacobian matrix and is obtained as explained earlier. The second term represents the unknown second order sensitivity vector \((n \times 1)\). The third term is the \((n \times 1)\) vector of derivative of the right hand side with respect to a second design variable. The fourth term is the derivative of the jacobian matrix with respect to a design variable, and is an \((n \times n)\) matrix. Finally, the last term in Eq.(55) is the first order sensitivity vector, and would be obtained typically by solving Eq.(1). Notice that the extra work required to obtain second order derivatives would be to carry out additional MACSYMA operations (basically analytical differentiation) associated with the third and fourth terms of Eq.(55). Notice that Eq.(55) is similar to Eq.(1) except for the right hand sides which are modified. Similarly, the above procedure could be applied to obtain higher derivatives for the current three dimensional problem. Examples of second order sensitivity derivatives are \(\partial^2 \phi/\partial \alpha^2\) and \(\partial^2 \phi/\partial M_\infty \partial \alpha\).

References

APPENDIX A

MACSYMA CODE TO FIND THE RESIDUAL DEPENDENCIES
PX(I,J,K) := [P(I+1,J,K),P(I,J,K)]$

PY(I,J,K) := [P(I,J,K),P(I,J+1,K),P(I,J,K+1)]$

PZ[0](I,J,K) := [P(I,J,K),P(I,J-1,K),P(I,J,K+1),P(I+1,J,K),P(I,J,K+1)]$

PHIU(J) := [P(I,J,K),P(I,J,K+1),P(I,J,K+2)]$


CIRC(J) := UNION(PHIU(J),PHIL(J))$

PZ[1](I,J,K) := [P(I,J,K),P(I,J+1,K),P(I+1,J,K)]$


PZ[3](I,J,K) := UNION(P(I,J,K),P(I,J-1,K),P(I,J,K+1),P(I+1,J,K-1),P(I+1,J,K+1),CIRC(J))$

PZ[4](I,J,K) := UNION(P(I,J,K),P(I,J-1,K),P(I,J,K+1),P(I+1,J,K-1),P(I+1,J,K+1),CIRC(J))$

FOR N:0 THRU 4 DO (  
RHN[I,J,K] := UNION(PX(I,J,K),PY(I,J,K),PZ[N](I,J,K)).  
RIN[N](I,J,K) := UNION(RHN[N,I,J,K]).  
RES(I,J,K) := [P(I,J,K),P(I,J+1,K),P(I,J,K+1),P(I+1,J,K),P(I,J,K+1)]$

R1K () := UNION(RIP[1](I,J,K),RIM[1](I,J,K),RIP[1](I,J,K+1),RIM[1](I,J,K+1))$


R2Kw() := UNION(RIP[3](I,J,K),RIM[3](I,J,K),RIP[3](I,J,K-1),RIM[3](I,J,K-1))$

R2Kp() := UNION(RIP[4](I,J,K),RIM[4](I,J,K),RIP[4](I,J,K+1),RIM[4](I,J,K+1))$

FU(I,J) := [P(I,J,K),P(I,J,K+1),P(I,J,K+2)]$

FXU(I,J) := UNION(FU(I,J),FU(I,J),FU(I+1,J))$

FYU(I,J) := UNION(FU(I,J),FU(I,J),FU(I,J-1))$

DPU() := UNION(FXU(I,J),FYU(I,J))$


FXL(I,J) := UNION(FL(I,J),FL(I,J),FL(I,J-1))$

FYL(I,J) := UNION(FL(I,J),FL(I,J-1),FL(I+1,J))$

DPLD() := UNION(FXL(I,J),FYL(I,J))$

ANDFI1() := UNION(R1K,DPU)$

ANDFI2() := UNION(R1KU,DPLD)$

ANDFI3() := UNION(R2Kw,CIRC)$

ANDFI4() := UNION(R2Kp,CIRC)$

(RJ(I,J,K) := UNION(R1K(I,J,K),RIM[0](I,J,K),RIP[I,J-1,K],RIM[0](I,J-1,K)).
  RJP(I,J,K) := RJ(I,J,K).
  RK(I,J,K) := UNION(R1K(I,J,K),RIM[0](I,J,K),RIP[I,J-1,K],RIM[0](I,J-1,K))$.

RKP(I,J,K) := RK(I,J,K).

RTOT(I,J,K) := UNION(RES,RIP,RIM,RJ,RJP,RK).

FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M] : PX (NI[N],NJ[N],NK[N]))
FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M] : PY (NI[N],NJ[N],NK[N]))
FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M] : PZ(4)(NI[N],NJ[N],NK[N]))
FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M] : PX (NI[N],NJ[N],NK[N]))
FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M] : PY (NI[N],NJ[N],NK[N]))
FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M] : PZ(4)(NI[N],NJ[N],NK[N]))

/*-----------------------------------------------*/
(RLO: [RES, RIP, RIM, RJ, RK, RJP, RKP, RTOT] , RPO : SUBST(LT,RLO) )$
M : OS$
FOR N:1 THRU 8 DO ( RPO[N]:SORT( RPO[N] ) , PRINT("DEP",N,RPO[N] ))$
M : OS$
FOR N:1 THRU 15 DO ( M:M=1, IF NT0[N]=1 THEN (PP0[M]:SORT(SUBST(LT,PP0[M])),PRINT("LPO",M,PP0[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
FOR N:1 THRU 15 DO ( M:M=1, IF NT1[N]=1 THEN (PP1[M]:SORT(SUBST(LT,PP1[M])),PRINT("LPO",M,PP1[M])) )$
$ ! This is a login command procedure template
$ IF FS$MODE () .EOS. "BATCH" THEN EXIT
$MAC

If you logged on to Venus by typing VENUS at the
ENTER RESOURCE NAME prompt of the port selector,
do NOT use the BREAK key to get out of Macsyma.

This is Macsyma 412.61 for DEC VAX 8650 Series Computers.
Copyright (c) 1982 Massachusetts Institute of Technology.
All Rights Reserved.
Enhancements (c) 1982, 1988 Symbolics, Inc. All Rights Reserved.
Type "DESCRIBE(TRADE_SECRET):" to see important legal notices.
Type "HELP();" for more information.

Checking password file: DISK$PKG1:[MACSYMA_412.SYSTEM]PASSWD-VENUS-412.TEXT
DISK$PKG1:[MACSYMA_412.SYSTEM]macsyma-init.fas:4 being loaded.
Init File Not Found: SYS$USERDISKH:[HME4905]macsyma-init.mac
BATCH("RMD.MAC");

(C1)

(C2) /************************************************************************* /
   /* RMD.MAC: POTENTIAL DEPENDENCIES */
   /**************************************************************************/
   // MACSYMA PROGRAM TO GENERATE RESIDUAL DEPENDENCIES ***************/
   /**************************************************************************/
   SHOWTIME: TRUES
   Time= 0 msecs

   (C3) PX(I,J,K) := [P(I+1,J ,K ),P(I ,J ,K )]$
   Time= 20 msecs

   (C4) PY(I,J,K) := [P(I ,J ,K ),P(I ,J-1,K ),P(I+1,J ,K ),
                    P(I+1,J-1,K ),P(I ,J+1,K ),P(I+1,J+1,K )]$
   Time= 10 msecs

   (C5) PZ[0](I,J,K) := [P(I ,J ,K ),P(I ,J ,K+1),P(I+1,J ,K-1),P(I+1,J ,K-1),
                        P(I+1,J ,K-1),P(I+1,J ,K+1)]$
   Time= 10 msecs

   (C6) PHIU(J) := [P(ITE,J,KUP),P(ITE,J,KUP+1),P(ITE,J,KUP+2)]$
   Time= 0 msecs

   (C7) PHIL(J) := [P(ITE,J,KLO),P(ITE,J,KLO-1),P(ITE,J,KLO-2)]$
   Time= 0 msecs

   (C8) CIRC(J) := UNION(PHIU(J),PHIL(J))$
   Time= 0 msecs

   (C9) PZ[1](I,J,K) := [P(I ,J ,K ),P(I ,J ,K+1),P(I ,J ,K+2),
                        P(I+1,J ,K ),P(I+1,J ,K+1),P(I+1,J ,K+2)]$
   Time= 0 msecs

                         P(I+1,J ,K ),P(I+1,J ,K-1),P(I+1,J ,K-2)]$
   Time= 10 msecs
(C11) \( PZ[3](I,J,K) := \text{UNION}\{P(I,J,K),P(I,J,K-1),P(I,J,K+1),
\quad P(I+1,J,K),P(I+1,J,K-1),P(I+1,J,K+1),CIRC(J)\}\)
Time= 0 msecs

(C12) \( PZ[4](I,J,K) := \text{UNION}\{P(I,J,K),P(I,J,K-1),P(I,J,K+1),
\quad P(I+1,J,K),P(I+1,J,K-1),P(I+1,J,K+1),CIRC(J)\}\)
Time= 10 msecs

(C13) FOR N:0 THRU 4 DO
\( RH[N](I,J,K) := \text{UNION}\{P(I,J,K),P(I,J,K-1),P(I,J,K+1),
\quad P(I+1,J,K),P(I+1,J,K-1),P(I+1,J,K+1),CIRC(J)\}\)
\( RIP[N](I,J,K) := \text{UNION}\{RH[N](I,J,K),RH[N](I-1,J,K)\}\)
\( RIM[N](I,J,K) := RIP[N](I-1,J,K)\)
Time= 80 msecs

(C14) \( RES(I,J,K) := \text{UNION}\{P(I,J,K),P(I,J,K-1),P(I,J,K+1),
\quad P(I+1,J,K),P(I+1,J,K-1),P(I+1,J,K+1),R(I+1,J+1,K),
\quad P(I,J,K-1),P(I,J,K+1),P(I+1,J,K+1),
\quad P(I,J,K-1),P(I,J,K+1)\}\)
Time= 0 msecs

(C15) \( R1K() := \text{UNION}\{R1P[1](I,J,K),R1M[1](I,J,K),R1P[1](I,J,K+1),R1M[1](I,J,K+1)\}\)
Time= 10 msecs

(C16) \( R1KU() := \text{UNION}\{R1P[2](I,J,K),R1M[2](I,J,K),R1P[2](I,J,K-1),R1M[2](I,J,K-1)\}\)
Time= 0 msecs

(C17) \( R2KW() := \text{UNION}\{R1P[3](I,J,K),R1M[3](I,J,K),R1P[3](I,J,K-1),R1M[3](I,J,K-1)\}\)
Time= 0 msecs

(C18) \( R2KP() := \text{UNION}\{R1P[4](I,J,K),R1M[4](I,J,K),R1P[4](I,J,K+1),R1M[4](I,J,K+1)\}\)
Time= 10 msecs

(C19) \( FU(I,J) := \{P(I,J,K),P(I,J,K+1),P(I,J,K+2)\}\)
Time= 0 msecs

(C20) \( FXU(I,J) := \text{UNION}\{FU(I,J),FU(I-1,J),FU(I+1,J)\}\)
Time= 0 msecs

(C21) \( FYU(I,J) := \text{UNION}\{FU(I,J),FU(I,J-1),FU(I,J+1)\}\)
Time= 0 msecs

(C22) \( DPU() := \text{UNION}\{FXU(I,J),FYU(I,J)\}\)
Time= 10 msecs

Time= 0 msecs

(C24) \( FXL(I,J) := \text{UNION}\{FL(I,J),FL(I-1,J),FL(I+1,J)\}\)
Time= 0 msecs

(C25) \( FYL(I,J) := \text{UNION}\{FL(I,J),FL(I,J-1),FL(I,J+1)\}\)
Time= 10 msecs

(C26) \( DPLQ() := \text{UNION}\{Fxl(I,J),Fyl(I,J)\}\)
Time= 10 msecs
(C27) ANOFI1() := UNION(R1K , DPU)$
Time= 10 msecs

(C28) ANOFI2() := UNION(R1KU,DPL0)$
Time= 0 msecs

(C29) ANOFI3() := UNION(R2K W,CIRC)$
Time= 0 msecs

(C30) ANOFI4() := UNION(R2KP,CIRC)$
Time= 10 msecs

RJP[I] (I,J,K):=RJ[I,J+1,K]
RKP[I] (I,J,K):=RK[I,J,K+1]
RTOT(I,J,K):=UNION(RES,RIP,RJK,RJ,RJP,RK,RKP)$
Time= 20 msecs

: Starting garbage collection due to dynamic-O space overflow.
: Finished garbage collection due to dynamic-O space overflow.
Time= 139300 msecs

(C33) /*--------------------------------------------------------------------------------*/
(RIK : RIK (), DPU : DPU () , ATT1: ANOFI1())$
: Starting garbage collection due to dynamic-O space overflow.
: Finished garbage collection due to dynamic-O space overflow.
Time= 32320 msecs

(C34) (R1KU: R1KU(), DPL0: DPL0(), ATT2: ANOFI2())$
Time= 29230 msecs

(C35) (R2KW: R2KW(), CIRC: CIRC(J), ATT3: ANOFI3())$
: Starting garbage collection due to dynamic-1 space overflow.
: Finished garbage collection due to dynamic-1 space overflow.
Time= 41350 msecs

(C36) (R2KP: R2KP(), CIRC: CIRC(J), ATT4: ANOFI4())$
: Starting garbage collection due to dynamic-1 space overflow.
: Finished garbage collection due to dynamic-1 space overflow.
Time= 40840 msecs

(C37) /*--------------------------------------------------------------------------------*/

LT :
P(I-1,J-2,K-3)=P2 ,P(I-1,J-2,K-1)=P52,P(I-1,J-2,K+1)=P102,P(I-1,J-2,K+3)=P152,
P(I+1,J-2,K-3)=P4 ,P(I+1,J-2,K-1)=P54,P(I+1,J-2,K+1)=P104,P(I+1,J-2,K+3)=P154,
P(I+2,J-2,K-3)=P5 ,P(I+2,J-2,K-1)=P55,P(I+2,J-2,K+1)=P105,P(I+2,J-2,K+3)=P155,
P(I-1,J-1,K-3)=P7 ,P(I-1,J-1,K-1)=P57,P(I-1,J-1,K+1)=P107,P(I-1,J-1,K+3)=P157,
P(I+1,J-1,K-3)=P9 ,P(I+1,J-1,K-1)=P59,P(I+1,J-1,K+1)=P109,P(I+1,J-1,K+3)=P159,
P(I+2,J-1,K-3)=P10,P(I+2,J-1,K-1)=P60,P(I+2,J-1,K+1)=P110,P(I+2,J-1,K+3)=P160,
P(I+1,J,K-3)=P14,P(I+1,J,K-1)=P64,P(I+1,J,K+1)=P114,P(I+1,J,K+3)=P164,
(C43) M : 0.
FOR N: 1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M]: PX (NI[N],NJ[N],NK[N])).
FOR N: 1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M]: PY (NI[N],NJ[N],NK[N])).
FOR N: 1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN PP4[M]: P2[4](NI[N],NJ[N],NK[N])).

Time= 6410 msecs

(C44) /*-----------------------------------------------*/
(RLO: RES, RIP, RIM, RJ, RK, RJP, RKP, RTOT), RPO : SUBST(LT,RLO)

Time= 14730 msecs

(C45) FOR N: 1 THRU 8 DO ( RPO[N]:=SORT( RPO[N] ), PRINT("DEP",N,RPO[N]))
DEP 1 [P11, P63, P82, P83, P87, P88, P89, P92, P93, P94]
DEP 2 [P11, P11, P11, P14, P62, P63, P64, P82, P83, P84, P87, P88, P89, P92, P93, P94]
DEP 3 [P11, P11, P11, P61, P62, P63, P81, P82, P83, P86, P87, P88, P89, P91, P92, P93, P94]
DEP 5 [P11, P11, P11, P14, P62, P63, P64, P76, P77, P78, P79, P81, P82, P83, P84, P86, P87, P88, P89, P91, P92, P93, P94]

Time= 1320 msecs

(C46) M : 0
Time= 10 msecs

(C47) FOR N: 1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN (PPO[M]:=SORT(SUBST(LT,PPO[M])),PRINT("LPO",M,PPO[M])))
LPO 1 [P88, P89]
LPO 2 [P87, P88]
LPO 3 [P86, P87]
LPO 4 [P83, P84]
LPO 5 [P82, P83]
; Starting garbage collection due to dynamic-1 space overflow.
; Finished garbage collection due to dynamic-1 space overflow.
LPO 6 [P81, P82]
LPO 7 [P63, P64]
LPO 8 [P62, P63]
LPO 9 [P61, P62]
LPO 10 [P93, P94]
LPO 11 [P92, P93]
LPO 12 [P91, P92]
LPO 13 [P113, P114]
LPO 14 [P112, P113]
LPO 15 [P111, P112]
Time= 7890 msecs

(C48) FOR N:1 THRU 15 DO
(M:M+1.IF NTO[N]=I THEN (PPO[M]:SORT(SUBST(LT,PPO[M])),PRINT("LPO",M,PPO[M])))$
LPO 16 [P83, P84, P88, P93, P94]
LPO 17 [P82, P83, P87, P88, P92, P93]
LPO 18 [P81, P82, P86, P87, P91, P92]
LPO 19 [P78, P79, P83, P84, P88, P89]
LPO 20 [P77, P78, P82, P83, P87, P88]
LPO 21 [P76, P77, P81, P82, P86, P87]
LPO 22 [P58, P59, P63, P64, P68, P69]
LPO 23 [P57, P58, P62, P63, P67, P68]
LPO 24 [P56, P57, P61, P62, P66, P67]
LPO 25 [P88, P89, P93, P94, P98, P99]
LPO 26 [P87, P88, P92, P93, P97, P98]
LPO 27 [P86, P87, P91, P92, P96, P97]
LPO 28 [P108, P109, P113, P114, P118, P119]
LPO 29 [P107, P108, P112, P113, P117, P118]
LPO 30 [P106, P107, P111, P112, P116, P117]
Time= 7730 msecs

(C49) FOR N:1 THRU 15 DO
(M:M+1.IF NTO[N]=I THEN (PPO[M]:SORT(SUBST(LT,PPO[M])),PRINT("LPO",M,PPO[M])))$
LPO 31 [P113, P114, P63, P64, P88, P89]
LPO 32 [P112, P113, P62, P63, P87, P88]
LPO 33 [P111, P112, P61, P62, P86, P87]
LPO 34 [P108, P109, P58, P59, P83, P84]
LPO 35 [P107, P108, P57, P58, P82, P83]
LPO 36 [P106, P107, P56, P57, P81, P82]
LPO 37 [P38, P39, P63, P64, P88, P89]
LPO 38 [P37, P38, P62, P63, P87, P88]
LPO 39 [P36, P37, P61, P62, P86, P87]
LPO 40 [P118, P119, P68, P69, P93, P94]
LPO 41 [P117, P118, P67, P68, P92, P93]
LPO 42 [P116, P117, P66, P67, P91, P92]
LPO 43 [P113, P114, P138, P139, P88, P89]
LPO 44 [P112, P113, P137, P138, P87, P88]
LPO 45 [P111, P112, P136, P137, P86, P87]

Time = 7070 msecs

(C50) (RL1: [R1K . DPU . ATT1] , RP1 : SUBST(LT,RL1) )$

Time = 5620 msecs

(C51) FOR N:I THRU 3 DO ( RP1[N]:SORT( RP1[N] ), PRINT("DEP",N,RP1[N]) )$


Time = 510 msecs

(C52) M : 0$

Time = 0 msecs

(C53) FOR N: 1 THRU 15 DO ( M:M+1, IF N1 [N]=1 THEN ( PP1[M]:SORT(SUBST(LT,PP1[M])), PRINT("LP1",M,PP1[M])) )$

LP1 1 [P88, P89]

LP1 2 [P87, P88]

LP1 3 [P86, P87]

LP1 13 [P113, P114]

LP1 14 [P112, P113]

LP1 15 [P111, P112]

Time = 1410 msecs

(C54) FOR N: 1 THRU 15 DO ( M:M+1, IF N1 [N]=1 THEN ( PP1[M]:SORT(SUBST(LT,PP1[M])), PRINT("LP1",M,PP1[M])) )$

LP1 16 [P83, P84, P88, P89, P93, P94]

LP1 17 [P82, P83, P87, P88, P92, P93]

LP1 18 [P81, P82, P86, P87, P91, P92]

LP1 28 [P108, P109, P113, P114, P118, P119]

LP1 29 [P107, P108, P112, P113, P117, P118]

LP1 30 [P106, P107, P111, P112, P116, P117]

Time = 2860 msecs

(C55) FOR N: 1 THRU 15 DO ( M:M+1, IF N1 [N]=1 THEN ( PP1[M]:SORT(SUBST(LT,PP1[M])), PRINT("LP1",M,PP1[M])) )$

LP1 31 [P113, P114, P138, P139, P88, P89]

LP1 32 [P112, P113, P137, P138, P87, P88]
LP1 33 [P111, P112, P136, P137, P86, P87]
LP1 43 [P113, P114, P138, P139, P163, P164]
LP1 44 [P112, P113, P137, P138, P162, P163]
LP1 45 [P111, P112, P136, P137, P161, P162]
Time= 2980 msecs

(C56) (RL2: [R1KU, DPLO, ATT2] . RP2 : SUBST(LT, RL2) )$  
Time= 5460 msecs

(C57) FOR N:1 THRU 3 DO ( RP2[N]:SORT(RP2[N]) , PRINT("DEP",N,RP2[N]) )$  
DEP 1 [P11, P12, P13, P14, P36, P37, P38, P39, P56, P57, P58, P59, P61, P62,  
P63, P64, P66, P67, P68, P69, P81, P82, P83, P84, P86, P87, P88, P89, P91, P92,  
P93, P94]  
DEP 2 [P33, P37, P38, P39, P43, P58, P62, P63, P64, P66, P67, P68, P69, P81, P82, P83, P84, P86, P87, P88, P89, P93]  
DEP 3 [P11, P12, P13, P14, P36, P37, P38, P39, P43, P56, P57, P58, P59,  
P61, P62, P63, P64, P66, P67, P68, P69, P81, P82, P83, P84, P86, P87, P88, P89,  
P91, P92, P93, P94]  
Time= 500 msecs

(C58) M : O$  
Time= 0 msecs

(C59) FOR N:1 THRU 15 DO  
(M:M+1, IF NT2[N]=1 THEN (PP2[M]:SORT(SUBST(LT,PP2[M])),PRINT("LP2",M,PP2[M])))$  
LP2 1 [P88, P89]  
LP2 2 [P87, P88]  
LP2 3 [P86, P87]  
LP2 7 [P63, P64]  
LP2 8 [P62, P63]  
LP2 9 [P61, P62]  
Time= 1320 msecs

(C60) FOR N:1 THRU 15 DO  
(M:M+1, IF NT2[N]=1 THEN (PP2[M]:SORT(SUBST(LT,PP2[M])),PRINT("LP2",M,PP2[M])))$  
LP2 16 [P83, P84, P88, P89, P93, P94]  
LP2 17 [P82, P83, P87, P88, P92, P93]  
LP2 18 [P81, P82, P86, P87, P91, P92]  
LP2 22 [P58, P59, P63, P64, P68, P69]  
LP2 23 [P57, P58, P62, P63, P67, P68]  
LP2 24 [P56, P57, P61, P62, P66, P67]  
Time= 3150 msecs

(C61) FOR N:1 THRU 15 DO  
(M:M+1, IF NT2[N]=1 THEN (PP2[M]:SORT(SUBST(LT,PP2[M])),PRINT("LP2",M,PP2[M])))$  
LP2 31 [P38, P39, P63, P64, P88, P89]  
LP2 32 [P37, P38, P62, P63, P87, P88]
LP2 33 [P36, P37, P61, P62, P86, P87]
LP2 37 [P13, P14, P38, P39, P63, P64]
LP2 38 [P12, P13, P37, P38, P62, P63]
LP2 39 [P11, P12, P36, P37, P61, P62]
Time= 2950 msecs

(C62) (RL3: [R2KW, CIRC, ATT3], RP3 : SUBST(LT,RL3))
; Starting garbage collection due to dynamic-O space overflow.
; Finished garbage collection due to dynamic-O space overflow.
Time= 9140 msecs

(C63) FOR N:1 THRU 3 DO (RP3[N]:SORT(RP3[N]), PRINT("DEP",N,RP3[N]))
DEP 2 [P182, P183, P184, P185, P186, P187]
Time= 530 msecs

(C64) M : 0
Time= 0 msecs

(C65) FOR N:1 THRU 15 DO (M: M + 1, IF NT3[N]=1 THEN (PP3[M]:SORT(SUBST(LT,PP3[M])), PRINT("LP3",M,PP3[M])))
LP3 1 [P88, P89]
LP3 2 [P87, P88]
LP3 3 [P86, P87]
LP3 7 [P63, P64]
LP3 8 [P62, P63]
LP3 9 [P61, P62]
Time= 1300 msecs

(C66) FOR N:1 THRU 15 DO (M: M + 1, IF NT3[N]=1 THEN (PP3[M]:SORT(SUBST(LT,PP3[M])), PRINT("LP3",M,PP3[M])))
LP3 16 [P83, P84, P88, P89, P93, P94]
LP3 17 [P82, P83, P87, P88, P92, P93]
LP3 18 [P81, P82, P86, P87, P91, P92]
LP3 22 [P58, P59, P63, P64, P68, P69]
LP3 23 [P57, P58, P62, P63, P67, P68]
LP3 24 [P56, P57, P61, P62, P66, P67]
Time= 3030 msecs

(C67) FOR N:1 THRU 15 DO (M: M + 1, IF NT3[N]=1 THEN (PP3[M]:SORT(SUBST(LT,PP3[M])), PRINT("LP3",M,PP3[M])))
LP3 31 [P113, P114, P182, P183, P184, P185, P186, P187, P63, P64, P88, P89]
LP3 32 [P112, P113, P182, P183, P184, P185, P186, P187, P62, P63, P87, P88]
| LP3 33 | P111, P112, P182, P183, P184, P185, P186, P187, P61, P62, P86, P87 |
| LP3 37 | P182, P183, P184, P185, P186, P187, P38, P39, P63, P64, P88, P89 |
| LP3 38 | P182, P183, P184, P185, P186, P187, P37, P38, P62, P63, P87, P88 |
| LP3 39 | P182, P183, P184, P185, P186, P187, P36, P37, P61, P62, P86, P87 |

Time = 5260 msecs

(C68) (RL4: [R2KP, CIRC, ATT4], RP4 : SUBST(LT,RL4) )$

Time = 5180 msecs

(C69) FOR N:1 THRU 3 DO ( RP4[N]:SORT( RP4[N] ), PRINT("DEP",N,RP4[N]) )$


DEP 2 [P182, P183, P184, P185, P186, P187]


Time = 470 msecs

(C70) M : 0$

Time = 10 msecs

(C71) FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN (PP4[M]:SORT(SUBST(LT,PP4[M])), PRINT("LP4",M,PP4[M])))$

LP4 1 [P88, P89]

LP4 2 [P87, P88]

LP4 3 [P86, P87]

LP4 13 [P113, P114]

LP4 14 [P112, P113]

LP4 15 [P111, P112]

Time = 1330 msecs

(C72) FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN (PP4[M]:SORT(SUBST(LT,PP4[M])), PRINT("LP4",M,PP4[M])))$

LP4 16 [P83, P84, P88, P89, P93, P94]

LP4 17 [P82, P83, P87, P88, P92, P93]

LP4 18 [P81, P82, P86, P87, P91, P92]

LP4 28 [P108, P109, P113, P114, P118, P119]

LP4 29 [P107, P108, P112, P113, P117, P118]

LP4 30 [P106, P107, P111, P112, P116, P117]

Time = 3030 msecs

(C73) FOR N:1 THRU 15 DO (M:M+1, IF NT4[N]=1 THEN (PP4[M]:SORT(SUBST(LT,PP4[M])), PRINT("LP4",M,PP4[M])))$

LP4 31 [P113, P114, P182, P183, P184, P185, P186, P187, P63, P64, P88, P89]

LP4 32 [P112, P113, P182, P183, P184, P185, P186, P187, P62, P63, P87, P88]

LP4 33 [P111, P112, P182, P183, P184, P185, P186, P187, P61, P62, P86, P87]
LP4 43 [P113, P114, P138, P139, P182, P183, P184, P185, P186, P187, P88, P89]


LP4 45 [P111, P112, P136, P137, P182, P183, P184, P185, P186, P187, P86, P87]

Time = 5040 msecs

(C74) /----------------------------------------------/
Accumulated Computation Time = 400140 msecs
Time = 421880 msecs

(D74)
QUIT();

(C75)
%DCL-W-SKPDAT, image data (records not beginning with "$") ignored
HME4905 job terminated at 23-JAN-1991 09:05:46.65

Duration/Completion: 374
Direct I/O count: 580
Page faults: 147965
Charged CPU time: 00:07:11.01
Peak working set size: 4096
Peak page file size: 40493
Mounted volumes: 0
Elapsed time: 00:13:18.49
APPENDIX B

MACSYMA CODE TO DIFFERENTIATE THE RESIDUAL
RESIDUAL : RIP*TA11P*(P89-P88)  
(RIP*TA12P*(TAJ1*(P89-P88)+P89-P88)+TAJ2*(P89-P88-P84))  
+ RIP*QXINF*2/DXIC(I)  
+S *(RIP*TA11M*(P89-P67)  
+ RIP*TA12M*(TAJ1*(P89-P83-P87-P82)+TAJ2*(P89-P83-P86))  
+ RIP*QXINF*2/DXIC(I))  
+ RJP*TA21P*(TAJ1*(P89-P87-P93-P92)+TAI2*(P89-P88-P94))  
+ S *(RJ*TA22M*(P88-P83)+RJ*TA21M*(P89-P87-P88))  
+ V1*(RKP*TA33P*(P113-P88)+RKP*QXINF*2*XIXI(J,I)/DZETAC(K))  
+ V2*(RK*TA33P*(P88-P63)+RK*QXINF*2*XIXI(J,I)/DZETAC(K))$  

SHOWTIME:TRUE$  
RESIDUAL : RIP'TA11P'(P88-P89)  
÷ RIP=TAi2P-(TAJI-(P88-P83+P89-P84)+TAd2-(P93-P88+P94-P89))  
÷ RIP'QXINF'2/DXlC(1) +S  
"(RIM'TAllM'(P88-P87)  
÷ RIM'TA12M'(TAjl"(P88-P83+P87-P82)÷TAJ2=(P93-P88+P92-P87))  
÷ RIM'OXINF-2/DXIC(1)) +  
RJP=TA22P-(P93-P88) + RJP,TA21P-(TAII-(P88-P8T+P93-P92)+TAI2=(P89-P88÷P84-P83))  
+ V1=(RKP,TA33P=(P113-P88)  
÷ RKP=OZINF-2=XIXXI(J.I)/DZETAC(K)) +  
V2-(RK  
"TA33M-(P88-P63) + RK -OZINF-2=XIXXI(J.I)/DZETAC(K))$  

DXII(I)'(P(U.K,I÷I)_S=P(U.K,I))  
÷ QXINF/XIXIP(d.I)$  
÷ QZINF$  
(1/2)'(AIK(K)'(P(j.K  
÷ QZINF - AiK(K)'CI(O)$  

(CCI=P(d.KUP.ITE)+S=CC2=P(J.KUP+I.ITE)+ CC3=P(J.KUP _2.ITE)) ÷S=CC4=P(d.KLOW. ITE) ÷ CC5=P(J.KLOW-I.ITE)÷S=CC6=P(J.KLOW-2.ITE)$  

(A1K(K)=(P(U.K  
÷ QZINF  

(ANOFII : S "(RIK • TA33M = DDPU + RIK "QZINF,2=XIXXI(J.I)/DZETAC(K)).  
ANOFI2  
RJKU • TA33P • DOPL + RJKU=QZINF-2-XIXXI(J.I)/DZETAC(K)  
ANOFI3 : R2KW = TA33M • CIR  
ANOFI4  
)
B-5
RIGHT HAND SIDES

XD = [XD1, XD2, XD3, XD4, XD5]

[MACH, AOA, T, C, L]

SUBROUTINE RS(J, I, K, RHSM, RHSA, RHST, RHSC, RHSL), "RMERS.FR", "INCLUDE (INTROS)"
/(TITLE1(LENGTH(RTTO), RTTO), TITLE2("PO", O))
GENTRANOPT: TRUES
(GENTRAN(RSETQ(RIP, RIP)), GENTRAN(RSETQ(RIM, RIM)), GENTRAN(RSETQ(RJ, RJP)),
GENTRAN(RSETQ(RK, RKP)), GENTRAN(RSETQ(RJP, RKP)))

FOR L:1 THRU LENGTH(SDES1) DO (PRINT(L), XDL: SDES1[L], TITLE5("PO", O, XDL),
TITLE4("RIP", XDL, DNRIP[L]), TITLE4("RIM", XDL, DNRIM[L]),
TITLE4("RJ", XDL, DNRJ[L]), TITLE4("RK", XDL, DNRK[L]),
TITLE4("RJP", XDL, DNRJP[L]), TITLE4("RKP", XDL, DNRKP[L]),
TITLE4("RES", XDL, DRS1[L]))

/GENTRAN(LITERAL("C", CR, "C", TAB, "DRESIDUAL", CR))

FOR L:1 THRU LENGTH(SDES) DO (PRINT(L), XDL: SDES[L], TITLE5("PA", I, XDL),
IF L<=LENGTH(SDESI) THEN TITLE4("RIK", XDL, DNRiK[L]),
TITLE4("DDPU", XDL, DNDPU[L]), TITLE4("ANI", XDL, DRSI[L])
GENTRAN(LITERAL("C", CR, "C", TAB, "ENDIF", CR))

FOR L:1 THRU LENGTH(SDES) DO (PRINT(L), XDL: SDES[L], TITLE5("PB", 2, XDL),
IF L<LENGTH(SDESI) THEN TITLE4("RIK", XDL, DNRiK[L]),
TITLE4("DDPL", XDL, DNDPLO[L]), TITLE4("AN2", XDL, DRSI[L])
GENTRAN(LITERAL("C", CR, "C", TAB, "ENDIF", CR))

GENTRANOPT: TRUES
(GENTRAN(RSETQ(R2KW, R2KW)), GENTRAN(LITERAL(TAB, "CIR:CIRC(J)", CR)))
/GENTRAN(LITERAL("C", CR, "C", TAB, "DANOFI3", CR))
FOR L:1 THRU LENGTH(SDES1) DO (PRINT(L), XDL: SDES1[L], TITLE5("PC", 3, XDL),
TITLE4("R2KW", XDL, DNR2KW[L]),
TITLE4("AN3", XDL, DRS3[L])
GENTRAN(LITERAL("C", CR, "C", TAB, "ENDIF", CR), TITLEB())

GENTRANOPT: TRUES
(GENTRAN(RSETQ(R2KP, R2KP)), GENTRAN(LITERAL(TAB, "CIR:CIRC(J)", CR)))
FOR L:1 THRU LENGTH(SDESI) DO (PRINT(L), XDL: SDESI[L], TITLE5("PD", 4, XDL),
TITLE4("R2KP", XDL, DNR2KP[L]),
TITLE4("AN4", XDL, DRS4[L])
GENTRAN(LITERAL("C", CR, "C", TAB, "ENDIF", CR))

( GENTRAN(TAB, "RHS = RESXD1 + AN1XD1 + AN2XD1 + AN3XD1 + AN4XD1", CR),
TAB, "RHS = RESXD2 + AN1XD2 + AN2XD2 + AN3XD2 + AN4XD2", CR,
TAB, "RHS = AN1XD3 + AN2XD3", CR,
TAB, "RHS = AN1XD4 + AN2XD4", CR,
TAB, "RHS = AN1XD5 + AN2XD5", CR), TITLEB())
WRITE SYMBOLIC PART FOR JACOBIAN

SUBROUTINE RE(J, I, K, JJ, II, KK, M)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTTO) DO (PRINT(L), RRTTO:RHS(RTTO[L]), LRTTO:LHS(RTTO[L]),
GENTRAN(LITERAL("C ", EVAL(RRTTO), CR)), EXEC1(LRTTO),
GENTRAN(LITERAL(TAB, "M = 1", CR)), EXEC3(RTTO))

SUBROUTINE R1E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT1) DO (PRINT(L), RRTT1:RHS(RTT1[L]), LRTT1:LHS(RTT1[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT1), CR)), EXEC1(LRTT1),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT1))

SUBROUTINE R2E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT2) DO (PRINT(L), RRTT2:RHS(RTT2[L]), LRTT2:LHS(RTT2[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT2), CR)), EXEC1(LRTT2),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT2))

SUBROUTINE R3E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT3) DO (PRINT(L), RRTT3:RHS(RTT3[L]), LRTT3:LHS(RTT3[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT3), CR)), EXEC1(LRTT3),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT3))

SUBROUTINE R4E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT4) DO (PRINT(L), RRTT4:RHS(RTT4[L]), LRTT4:LHS(RTT4[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT4), CR)), EXEC1(LRTT4),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT4))

SUBROUTINE RE(J, I, K, JJ, II, KK, M)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTTO) DO (PRINT(L), RRTTO:RHS(RTTO[L]), LRTTO:LHS(RTTO[L]),
GENTRAN(LITERAL("C ", EVAL(RRTTO), CR)), EXEC1(LRTTO),
GENTRAN(LITERAL(TAB, "M = 1", CR)), EXEC3(RTTO))

SUBROUTINE R1E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT1) DO (PRINT(L), RRTT1:RHS(RTT1[L]), LRTT1:LHS(RTT1[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT1), CR)), EXEC1(LRTT1),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT1))

SUBROUTINE R2E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT2) DO (PRINT(L), RRTT2:RHS(RTT2[L]), LRTT2:LHS(RTT2[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT2), CR)), EXEC1(LRTT2),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT2))

SUBROUTINE R3E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT3) DO (PRINT(L), RRTT3:RHS(RTT3[L]), LRTT3:LHS(RTT3[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT3), CR)), EXEC1(LRTT3),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT3))

SUBROUTINE R4E(J, I, K, JJ, II, KK, MM)
INCLUDE (INTROM)
FOR L:1 THRU LENGTH(RTT4) DO (PRINT(L), RRTT4:RHS(RTT4[L]), LRTT4:LHS(RTT4[L]),
GENTRAN(LITERAL("C ", EVAL(RRTT4), CR)), EXEC1(LRTT4),
GENTRAN(LITERAL(TAB, "MM = 1", CR)), EXEC3(RTT4))
APPENDIX C

FORTRAN SOURCE CODE (MACSYMA OUTPUT)
C-8
<table>
<thead>
<tr>
<th>C 2134</th>
<th>C 2137</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELSIF</td>
<td>ELSIF</td>
</tr>
<tr>
<td>(CND1</td>
<td>(CND1</td>
</tr>
<tr>
<td>[JJ, ]</td>
<td>[JJ, ]</td>
</tr>
<tr>
<td>K1, ]</td>
<td>K2, ]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C 2129</th>
<th>C 2127</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA3P</td>
<td>PA3P</td>
</tr>
<tr>
<td>PA3</td>
<td>PA3</td>
</tr>
<tr>
<td>PA3</td>
<td>PA3</td>
</tr>
<tr>
<td>PA1</td>
<td>PA1</td>
</tr>
<tr>
<td>TOXG</td>
<td>TOXG</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C 2127</th>
<th>C 2129</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORIGIN</td>
<td>ORIGIN</td>
</tr>
<tr>
<td>ENDIF</td>
<td>ENDIF</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C 2137</th>
<th>C 2134</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBROUTINE</td>
<td>SUBROUTINE</td>
</tr>
<tr>
<td>R2I,J,K</td>
<td>R2I,J,K</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C 2129</th>
<th>C 2127</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C 2134</th>
<th>C 2137</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCLUDE</td>
<td>INCLUDE</td>
</tr>
<tr>
<td>[INRO]</td>
<td>[INRO]</td>
</tr>
</tbody>
</table>
C-13
C-27
ORIGINAL PAGE IS OF POOR QUALITY
C-PE1

IF [COND[[JJ, KK, IM2, J, KMI]]] THEN

NM + 1

C-PE2 ELSEIF [COND[[JJ, KK, IM1, J, KMI]]] THEN

NM + 1

C-PE3 ELSE [COND[[JJ, KK, I, J, KMI]]] THEN

NM + 1

C-PE4 ELSEIF [COND[[JJ, KK, IP1, J, KMI]]] THEN

NM + 1

C-PE5 ELSEIF [COND[[JJ, KK, IP1, J, K]]] THEN

NM + 1

C-PE6 ELSEIF [COND[[JJ, KK, IM1, J, K]]] THEN

NM + 1

C-PE7 ELSEIF [COND[[JJ, KK, IM2, J, K]]] THEN

NM + 1

C-PE8 ELSEIF [COND[[JJ, KK, I, J, K]]] THEN

NM + 1

C-PE9 ELSEIF [COND[[JJ, KK, IM2, JP1, K]]] THEN

NM + 1

C-PE10 ELSEIF [COND[[JJ, KK, IM1, JP1, K]]] THEN

NM + 1

C-PE11 ELSEIF [COND[[JJ, KK, IM2, JP1, K]]] THEN

NM + 1

C-PE12 ELSEIF [COND[[JJ, KK, IP1, JP1, K]]] THEN

NM + 1

C-PE13 ELSEIF [COND[[JJ, KK, IM1, J, KP1]]] THEN

NM + 1

C-PE14 ELSEIF [COND[[JJ, KK, IP1, J, KP1]]] THEN

NM + 1

C-PE15 ELSEIF [COND[[JJ, KK, IM2, JP1, KP1]]] THEN

NM + 1

C-PE16 ELSEIF [COND[[JJ, KK, IP1, JP1, KP1]]] THEN

NM + 1

C-PE17 ELSEIF [COND[[JJ, KK, IM1, J, KP1]]] THEN

NM + 1

C-PE18 ELSEIF [COND[[JJ, KK, IP1, J, KP1]]] THEN

NM + 1

C-PE19 ELSEIF [COND[[JJ, KK, IM2, JP1, KP1]]] THEN

NM + 1

C-PE20 ELSEIF [COND[[JJ, KK, IP1, JP1, KP1]]] THEN

NM + 1

C-PE21 ELSEIF [COND[[JJ, KK, IM1, J, KP2]]] THEN

NM + 1

C-PE22 ELSEIF [COND[[JJ, KK, IP1, J, KP2]]] THEN

NM + 1

C-PE23 ELSEIF [COND[[JJ, KK, IP1, J, KLDW+2]]] THEN

NM + 1

C-PE24 ELSEIF [COND[[JJ, KK, I, J, KLDW+1]]] THEN

NM + 1

C-PE25 ELSEIF [COND[[JJ, KK, I, J, KLDW]]] THEN

NM + 1

C-PE26 ELSEIF [COND[[JJ, KK, I, J, KUP]]] THEN

NM + 1

C-PE27 ELSEIF [COND[[JJ, KK, I, J, KUP+1]]] THEN

NM + 1

C-PE28 ELSEIF [COND[[JJ, KK, I, J, KUP+2]]] THEN

NM + 1

ENDIF

RETURN
END
APPENDIX D

MACSYMA CODE TO FIND THE SENSITIVITY OF THE PRESSURE COEFFICIENT WITH RESPECT TO THE DESIGN VARIABLES
WING PLANFORM:

**ONERA M6**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root Chord</td>
<td>1.00</td>
</tr>
<tr>
<td>Tip Chord</td>
<td>0.56</td>
</tr>
<tr>
<td>Mean Chord</td>
<td>0.80</td>
</tr>
<tr>
<td>Area</td>
<td>1.16</td>
</tr>
<tr>
<td>Ref. Area</td>
<td>1.16</td>
</tr>
<tr>
<td>Ref. Chord</td>
<td>0.80</td>
</tr>
<tr>
<td>Ref. Moment</td>
<td>0.25</td>
</tr>
<tr>
<td>Aspect Ratio</td>
<td>3.80</td>
</tr>
<tr>
<td>Taper Ratio</td>
<td>0.56</td>
</tr>
<tr>
<td>Semi Span</td>
<td>1.48</td>
</tr>
<tr>
<td>L.E. Sweep</td>
<td>30.00</td>
</tr>
<tr>
<td>T.E. Sweep</td>
<td>15.76</td>
</tr>
<tr>
<td>Root Twist</td>
<td>0.00</td>
</tr>
<tr>
<td>Tip Twist</td>
<td>0.00</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>MACH NUMBER</td>
<td>0.80</td>
</tr>
<tr>
<td>ANGLE OF ATTACK</td>
<td>0.00</td>
</tr>
<tr>
<td>AIRFOIL MAX THICKNESS</td>
<td>0.06</td>
</tr>
<tr>
<td>AIRFOIL MAX CAMBER</td>
<td>0.00</td>
</tr>
<tr>
<td>LOCATION OF MAX CAMBER</td>
<td>0.40</td>
</tr>
</tbody>
</table>
$\eta_1 = 0.000$
$\eta_4 = 0.154$
$\eta_8 = 0.359$
$\eta_{12} = 0.564$
$\eta_{16} = 0.769$
$\eta_{20} = 0.974$

Figure (2)
\[ \text{ETA}(1) = 0.000 \quad \text{ETA}(4) = 0.154 \quad \text{ETA}(8) = 0.359 \]

\[ \text{ETA}(12) = 0.564 \quad \text{ETA}(16) = 0.769 \quad \text{ETA}(20) = 0.974 \]

Figure (3)
\[
\begin{align*}
\text{ETA}(1) &= 0.000 \\
\text{ETA}(4) &= 0.154 \\
\text{ETA}(8) &= 0.359 \\
\text{ETA}(12) &= 0.564 \\
\text{ETA}(16) &= 0.769 \\
\text{ETA}(20) &= 0.974
\end{align*}
\]

Figure (4)
\[
\eta(1) = 0.000 \\
\eta(4) = 0.154 \\
\eta(8) = 0.359 \\
\eta(12) = 0.564 \\
\eta(16) = 0.769 \\
\eta(20) = 0.974
\]

\textit{Figure (5)}
ETA(1) = 0.000
ETA(4) = 0.154
ETA(8) = 0.359

ETA(12) = 0.564
ETA(16) = 0.769
ETA(20) = 0.974

Figure (6)