AFRL-VA-WP-TR-1999-3050

DEVELOPMENT OF THE AERODYNAMIC/AEROSERVOELASTIC MODULES IN ASTROS

VOLUME 2: ZAERO PROGRAMMER'S MANUAL (F33615-96-C-3217)

P.C. CHEN
D. SARHADDI
D.D. LIU

ZONA Technology, Inc.
7430 E. Stetson Drive, Ste 205
Scottsdale, AZ 85251

FEBRUARY 1999

FINAL REPORT FOR PERIOD SEPTEMBER 1996 – SEPTEMBER 1998

Approved for public release; distribution unlimited

19991115 084
NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE UNITED STATES GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY BE RELATED TO THEM.

THIS REPORT IS RELEASEABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

VICTORIA A. TISCHLER
Aerospace Engineer
Design and Analysis Branch

VIPPERLA B. VENKAYYA
Leader, Multidisciplinary Design
Design & Analysis Branch

NELSON D. WOLF, Chief
Design and Analysis Branch
Structures Division

IF YOUR ADDRESS HAS CHANGED, IF YOU WISH TO BE REMOVED FROM OUR MAILING LIST, OR IF THE ADDRESSEE IS NO LONGER EMPLOYED BY YOUR ORGANIZATION, PLEASE NOTIFY AFRL/VASD BLDG 45, 2130 8TH STREET, SUITE 1, WRIGHT-PATTERSON AFB OH 45433-7542 TO HELP MAINTAIN A CURRENT MAILING LIST.

COPIES OF THIS REPORT SHOULD NOT BE RETURNED UNLESS RETURN IS REQUIRED BY SECURITY CONSIDERATIONS, CONTRACTUAL OBLIGATIONS, OR NOTICE ON A SPECIFIED DOCUMENT.
DEVELOPMENT OF THE AERODYNAMIC/AEROSERVOELASTIC MODULES IN ASTROS
VOLUME 2 - ZAERO PROGRAMMER’S MANUAL

AUTHOR(S)
P. C. Chen, D. D. Liu, D. Sarhaddi
ZONA Technology, Inc.

PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
ZONA Technology, Inc.
7434 E. Stetson Drive, Suite 205
Scottsdale, AZ 85251
Tel 602-945-9988 / Fax 602-945-6588

PERFORMING ORGANIZATION REPORT NUMBER
ZONA 99-11B

SPONSORING/MONITORING AGENCY(S) AND ADDRESS(ES)
Air Vehicles Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson Air Force Base, Oh 45433-7542
POC: Dr V. V. Venkaya, AFRL/VASD, 937-255-2582

DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ABSTRACT (Maximum 200 words)
This report is a part of the documentations which describe the complete development of an STTR Phase II effort entitled, “Development of the Aerodynamic/Aeroservoelastic Modules in ASTROS.” This report is one of four manuals that comprise the final report. The remaining reports consist of the ZAERO User’s Manual (Volume I), the ZAERO Applications Manual (Volume III) and the ZAERO Theoretical Manual (Volume IV).

ASTROS* is the seamless integration of the ZAERO module into ASTROS. As an aerodynamic enhancement to ASTROS, ZAERO is the ZONA aerodynamic module, unified for all Mach number ranges. This manual assumes the reader is familiar with the ASTROS system architecture, terminology and programming environment. In particular, it is geared toward system administrators and/or programmers working within the ASTROS* environment.

First, an overview of ZAERO and ASTROS* is presented. The modified system generation (SYSGEN) input for ASTROS* accommodating the ZAERO module is presented next, along with an ASTROS* system generation flow chart. Third, nine ZAERO engineering application modules within the ASTROS* environment are described. Lastly, the ZAERO specific relational and matrix database entity descriptions are presented.
Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2.0 ZAERO MODULE &amp; ASTROS*</td>
<td>2</td>
</tr>
<tr>
<td>3.0 ASTROS* SYSTEM GENERATION</td>
<td>5</td>
</tr>
<tr>
<td>3.1 Generation of the ASTROS* System</td>
<td>5</td>
</tr>
<tr>
<td>3.1.1 Compiling the ZAERO Module</td>
<td>5</td>
</tr>
<tr>
<td>3.1.2 Linking the ZAERO Module</td>
<td>5</td>
</tr>
<tr>
<td>3.2 The ZAERO/SYSGEN Input</td>
<td>8</td>
</tr>
<tr>
<td>3.2.1 Functional Module Definition</td>
<td>8</td>
</tr>
<tr>
<td>3.2.2 MAPOL Sequence Definition</td>
<td>8</td>
</tr>
<tr>
<td>3.2.3 Bulk Data Template Definition</td>
<td>8</td>
</tr>
<tr>
<td>3.2.4 Relational Schema Definition</td>
<td>8</td>
</tr>
<tr>
<td>3.2.5 Error Message Text Definition</td>
<td>8</td>
</tr>
<tr>
<td>3.3 The ZAERO Software</td>
<td>9</td>
</tr>
<tr>
<td>4.0 ENGINEERING APPLICATION MODULES</td>
<td>10</td>
</tr>
<tr>
<td>5.0 DATABASE ENTITY DESCRIPTION</td>
<td>27</td>
</tr>
<tr>
<td>6.0 REFERENCES</td>
<td>46a</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>47</td>
</tr>
<tr>
<td>ASTROS* Functional Module Definition</td>
<td></td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>50</td>
</tr>
<tr>
<td>ASTROS* MAPOL Sequence Definition</td>
<td></td>
</tr>
<tr>
<td>APPENDIX C</td>
<td>80</td>
</tr>
<tr>
<td>ASTROS* Bulk Data Template Definitions</td>
<td></td>
</tr>
<tr>
<td>APPENDIX D</td>
<td>86</td>
</tr>
<tr>
<td>ASTROS* Relational Schema Definition</td>
<td></td>
</tr>
<tr>
<td>APPENDIX E</td>
<td>89</td>
</tr>
<tr>
<td>ASTROS* Error Message Definition</td>
<td></td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ZAERO and Other Aerodynamic Modules.</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Capability of the ZAERO Module.</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>ASTROS/ZAERO (ASTROS*) Program Architecture.</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>ASTROS* System Generation Process.</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>Modified Makeqdriv file for ASTROS*.</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>Modified Makeastros file for ASTROS*.</td>
<td>7</td>
</tr>
</tbody>
</table>

List of Tables

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Computer Files Comprising ZAERO.</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>ZAERO Engineering Application Modules.</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>ZAERO Database Entities.</td>
<td>27</td>
</tr>
</tbody>
</table>
This interim report is submitted in fulfillment of CDRL CLIN 0001, Data Item A009, Title: Interface Design Document of a Small Business Technology Transfer (STTR) contract No. Contract No. F33615-96-C-3217 entitled, “Development of the Aerodynamic/Aeroservoelastic Modules in ASTROS,” covering the performance period from 24 September 1996 to 24 September 1998. This document provides the programmer’s documentation for the ZAERO module in ASTROS*.

This work was performed by ZONA Technology, Inc. and its subcontractors, the University of Oklahoma (Research Institute)/Technion (I.I.T) and Universal Analytics Inc. This work is the second phase of a continuing two-phase STTR contract supported by AFRL/Wright-Patterson. The first phase STTR contract No. F33615-95-C-3219 entitled, “Enhancement of the Aeroservoelastic Capability in ASTROS,” was completed in May 1996 and published as WL-TR-96-3119. Started in September 1996, the present second phase STTR contract was conducted by the same team members as in phase I. These contributors are: P.C. Chen (P.I.), D. Sarhaddi and D.D. Liu of ZONA Technology Inc.; Fred Striz of the University of Oklahoma; Moti Karpel of Technion/I.I.T.; and Tony Shimko and Steve Chen of Universal Analytics.

This STTR contract is sponsored by AFRL/Wright-Patterson. Capt. Gerald Andersen is the contract monitor and Dr. V.B. Venkayya is the initiator of the whole STTR effort. During the course of the present phase on the development of ASTROS*, the technical advice and assistance received from Mr. Doug Neill of The MacNeal Schwendler Corporation, Dr. V.B. Venkayya and others from AFRL are gratefully acknowledged.
1.0 INTRODUCTION

There are four major documents that describe the ZONA Aerodynamics Module (ZAERO) Module which has been seamless integrated into the Automated STRuctural Optimization System (ASTROS). These are: the ZAERO User’s, Programmer’s, Application and Theoretical Manuals for ASTROS*. While ZAERO represents the ZONA Aerodynamics Module, ASTROS* is defined as the seamless integration of ZAERO into ASTROS, i.e. ASTROS* = ZAERO + ASTROS. This Programmer’s Manual gives the detailed description of the ZAERO software and its interface with the ASTROS system. Newly created database entities in support of the ZAERO module within ASTROS* are described. Newly developed engineering application modules comprising the ZAERO module are presented in detail.

This manual assumes that the user is familiar with the ASTROS system (Version 11.0), its terminology and programming environment. A complete and comprehensive description of the ASTROS environment can be found in the ASTROS User’s and Programmer’s Manuals (Refs 1,2). In particular, this manual is geared toward system administrators and/or programmers within the ASTROS* environment.

Section 2 presents an overview of the ZAERO software, its aerodynamic capability over that of the previous modules in ASTROS, and the program architecture of ZAERO in relation to ASTROS.

Section 3 presents the computer files delivered under this contract which contain all of the subroutines of the ZAERO module, the modified System Generation (SYSGEN) input for ASTROS*, and the ASTROS* system generation process.

Section 4 presents the ZAERO engineering application modules (altogether nine modules) that make up ZAERO within the ASTROS* environment. Together with the ASTROS* object library, these ZAERO engineering applications modules constitute the entire ASTROS* executable (see ASTROS* system generation flow chart).

Section 5 presents the ZAERO specific relational and matrix database entity descriptions established upon building of the ASTROS* system that are used for communication of data among the ZAERO engineering application modules.
2.0 ZAERO MODULE AND ASTROS*

ASTROS (Automated STRuctural Optimization System) is a finite element based procedure tailored for the preliminary design of aerospace structures. As such, it includes flexibility and generality in multiple discipline integration. For aircraft, missile or spacecraft design, the unique attributes of ASTROS lie in its savings of design effort and time, improvement in flight performance and reduction in structural weight. In principle, ASTROS was aimed at the effective multidisciplinary interactions between aerodynamics, aeroelastics, structures and other modules. Although today a well-acclaimed, proven tool for Multidisciplinary Optimization (MDO) and analysis, ASTROS still requires further improvement in its capabilities in steady/unsteady aerodynamics, aeroelasticity and aeroservoelasticity (e.g. Ref 3).

The ZONA aerodynamic codes contained in the ZAERO module are the software products of ZONA Technology developed throughout the years. These include four major steady/unsteady aerodynamics codes, namely ZONA6, ZONA7, ZTAIC, and ZONA7U, that jointly cover the complete domain of all Mach number ranges. The ZONA aerodynamic system (the ZAERO System) which contains the ZAERO module and two other modules were developed under the support of AFRL/Wright-Patterson AFB for their seamless integration into the ASTROS system to improve and enhance the capability of ASTROS in aerodynamics, aeroelasticity and aeroservoelasticity (ASE). In particular, the ZAERO module improves the aerodynamics capability over the earlier aerodynamics modules in ASTROS in the following aspects (also see Figs 1 and 2):

1. Wing-Body geometry input for realistic aircraft configurations including external stores.
2. Flight regimes that include subsonic, supersonic, transonic and hypersonic Mach numbers.
3. High-order paneling scheme to assure accurate and robust solutions (without stringent paneling requirements).
4. Provides Aerodynamic Influence Coefficient (AIC) matrices for all flow regimes including the generation of transonic AIC.
5. Steady/unsteady aerodynamic options for static and dynamic aeroelastic applications.
6. Unified aerodynamic geometry bulk data input.

The development and seamless integration of the the ZAERO System into ASTROS has created a unique Multidisciplinary Design/Analysis and Optimization (MDO/MAO) tool that is currently unsurpassed in its steady/unsteady aerodynamic and aeroelastic capability. The ZAERO System consists of essentially three modules which include the ZAERO module, the AGM (aerodynamic geometry module ) and the 3D-Spline module (see Fig 3).

As can be seen in Fig 1, current capabilities of ASTROS and NASTRAN are limited to subsonic and supersonic Mach numbers and applicable to lifting surfaces only. By contrast, ZAERO is valid throughout the full range of subsonic to hypersonic Mach numbers and is applicable to complex aircraft configurations with external stores.
Fig 2 shows the capability of each code in the ZAERO Module (marked with †) along with other ZONA Codes.

<table>
<thead>
<tr>
<th>Capability</th>
<th>ZONA Unsteady/Steady Aerodynamic Codes – ZAERO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ZONAS1</td>
</tr>
<tr>
<td><strong>Geometry</strong></td>
<td></td>
</tr>
<tr>
<td>• Lifting Surface</td>
<td>⊗</td>
</tr>
<tr>
<td>(L.S.)</td>
<td></td>
</tr>
<tr>
<td>• Thickness Effect</td>
<td>⊗</td>
</tr>
<tr>
<td>• L.S. + Body =</td>
<td>⊗</td>
</tr>
<tr>
<td>Whole Aircraft</td>
<td></td>
</tr>
<tr>
<td><strong>Mach Number</strong></td>
<td></td>
</tr>
<tr>
<td>• Subsonic</td>
<td>⊗</td>
</tr>
<tr>
<td>• Transonic</td>
<td></td>
</tr>
<tr>
<td>• Supersonic</td>
<td>⊗</td>
</tr>
<tr>
<td>• Hypersonic</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. Capability of the ZAERO Module.

The seamlessly integrated ZAERO System in ASTROS is called ASTROS*. Fig 3 illustrates the role of the ZAERO System within ASTROS* and the overall ASTROS* program architecture. The ZAERO System consists of three primary modules with the following functionalities:

- **Unified Aerodynamic Geometry Module (AGM)**
  The Unified Aerodynamic Geometry Module processes the ZAERO model aerodynamic geometry input. Two newly created bulk data entries are used to define the aerodynamic geometry, namely CAERO7 for wing-like components such as wings, tails, pylons, launchers and store fins, and BODY7 for body-like components such as fuselage, stores and missile bodies.

- **3-D Spline Module**
  The 3-D Spline Module provides for the interconnection between the aerodynamic and structural models through the generation of spline matrices. Three spline methods are supported by this module. These are the infinite plate spline (IPS) method (SPLINE 1), the beam spline method (SPLINE 2) and the thin plate spline (TPS) method (SPLINE 3). The TPS
is an addition to the spline capability provided by ASTROS and unlike the IPS method does not require that a spline plane be defined.

- **The ZAERO Module**
  The ZAERO Module is made up of the four major aerodynamic codes (ZONA6, ZONA7, ZTAIC, ZONA7U) and generates the Unified Aerodynamic Influence Coefficient (UAIC) matrices, gust force vectors, control surface aerodynamic vectors and steady aerodynamic force vectors of trim parameters.

Database entities generated by AGM, 3-D Spline and ZAERO modules are computed in the ASTROS* preface phase and are not recomputed in the analysis/optimization loop.

![Figure 3. ASTROS/ZAERO (ASTROS*) Program Architecture.](image)
3.0 ASTROS* SYSTEM GENERATION

3.1 Generation of the ASTROS* System

The ASTROS System Generation Process (SYSGEN) has been modified to include the compilation of the ZAERO module source code and the linking of the ZAERO module object code into the ASTROS system. For ease of use, the system generation process has been kept the same as that of ASTROS (Version 11.0). The change made to this process to incorporate the ZAERO module are:

1. Updates to the SYSGEN input files (described in Sections 3.2.1 through 3.2.5)
2. Modified script file Makexqdriv for compiling the ZAERO module source code (described in Section 3.1.1)
3. Modified script file Makeastros for linking of the ZAERO module object code into the ASTROS* system (described in Section 3.1.2)

The entire SYSGEN process is depicted in Figure 4 and is briefly outlined as follows.

The modified SYSGEN input files (1) are processed by SYSGEN (2). SYSGEN generates the ASTROS* System Database (SYSDB) (3), SYSGEN output file (4) and the fortran source code XQDRIV (5). Both the ZAERO engineering applications modules (6) and XQDRIV source code (5) are compiled by the Makexqdriv script file (7). The object library of ASTROS (Version 11.0) (8) and object files generated by Makexqdriv (7) are linked via the Makeastros script file called by astlink (9) to generate the ASTROS* Executable Image (10). The ASTROS* System Database (3) and ASTROS* Executable (10) make up the ASTROS* system.

3.1.1 Compiling the ZAERO Module

The Makefile (Makexqdriv) used to compile the XQDRIV file generated by SYSGEN and located in the ASTROS (Version 11.0) sysgen directory has been updated to compile the ZAERO source files listed in Table 1 (see Figure 5). Should any modifications to the source code be required, the corresponding files where changes are made must be re-compiled in Makexqdriv. If no changes are made and the user wishes to re-build the ASTROS* system, it is not necessary to re-compile these files. Therefore all corresponding lines in Makexqdriv can be commented out to speed up the ASTROS* regeneration process.

3.1.2 Linking the ZAERO Module

The Makefile (Makeastros) called by the astlink script file to re-link ASTROS* and located in the ASTROS (Version 11.0) sysgen directory has been updated to link the ZAERO object files generated upon the compilation in Makexqdriv (see Figure 6).
Figure 4. ASTROS* System Generation Process.
Figure 5. Modified Makeqdriv File for ASTROS*.

Figure 6. Modified Makeastros File for ASTROS*.
3.2 ZAERO Sysgen Input

To facilitate the ASTROS* system generation described in Section 3.1, the five SYSGEN input data files, namely MODDEF.DAT, MAPOLSEQ.DAT, TEMPLATE.DAT, RELATION.DAT and SERRMSG.DAT, have been modified to include all components necessary for integration of ZAERO in ASTROS*. Modifications to each of these files are described in the following subsections. The physical changes made to each of these files are presented in Appendicies A through E, respectively.

3.2.1 Functional Module Definition (MODDEF.DAT)

The ASTROS* run-time library of MAPOL addressable modules file (MODDEF.DAT) has been updated to account for all newly developed engineering application modules presented in Section 5. These module definitions provide the additional links between the ASTROS* executive system and the ZAERO engineering application modules. The ZAERO functional module definitions are presented in Appendix A. For a detailed description of this file, please see Ref 2.

3.2.2 MAPOL Sequence (MAPOLSEQ.DAT)

For seamless integration of ZAERO into ASTROS, the ASTROS MAPOL sequence (file MAPOLSEQ.DAT) has been modified. The complete ASTROS* MAPOL sequence listing is presented in Appendix B. All changes to the original ASTROS (Version 11.0) MAPOL sequence listing are highlighted in boldface text and are demarcated by arrows on the right. For a detailed description of this file, please see Ref 2.

3.2.3 Bulk Data Template Definition (TEMPLATE.DAT)

In the development of the ZAERO module, twenty three new bulk data entries were created. Bulk data template definitions for these new bulk data entries were added to those of ASTROS (Version 11.0) and are presented in Appendix C. For a detailed description of this file, please see Ref 2.

3.2.4 Relational Schema Definition (RELATION.DAT)

Schema definitions of all relational database entities used by the ZAERO module have been defined in file RELATION.DAT. These relational entity schema definitions are presented in Appendix D. For a detailed description of this file, please see Ref 2.

3.2.5 Error Message Text Definition (SERRMSG.DAT)

Three new error message definition modules have been developed corresponding to the following engineering application modules: AEROGM, SPLINZ and ZAEROM. These ZAERO error message module definitions are presented in Appendix E. For a detailed description of this file, please see Ref 2.
3.3 The ZAERO Software

Under the current contract, six computer files containing all ZAERO engineering application and utility modules are delivered. These six files along with corresponding file descriptions are listed in Table 1. These files contain all of the ZAERO engineering application modules.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>File Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>aerogm.f</td>
<td>Code for processing of the wing/body aerodynamic geometry used by all ZAERO aerodynamic methods</td>
<td>source</td>
</tr>
<tr>
<td>fltqhz.f</td>
<td>Code for processing of matrices required for flutter analysis, including a newly developed K-method</td>
<td>source</td>
</tr>
<tr>
<td>splinz.f</td>
<td>Code for processing of spline matrices</td>
<td>source</td>
</tr>
<tr>
<td>utility.f</td>
<td>Additional math matrix in-core solvers</td>
<td>source</td>
</tr>
<tr>
<td>zaerom.f</td>
<td>Steady and unsteady aerodynamics processing for all of ZAERO's aerodynamic methods</td>
<td>source</td>
</tr>
<tr>
<td>zaerolib.o</td>
<td>ZONA's aerodynamic kernels</td>
<td>object</td>
</tr>
</tbody>
</table>

Note that all source code of ZAERO developed and integrated into ASTROS under this contract is being furnished to AFRL. The zaerolib.o code was developed prior to the current STTR Phases I & II and is ZONA Technology proprietary. This file is delivered in object code format only for specified computer platforms. To acquire updated object code for different computer platforms, please contact ZONA Technology at (602) 945-9988, POC: Darius Sarhaddi.
4.0 ZAERO ENGINEERING APPLICATION MODULES

Nine new engineering application modules have been developed as the ZAERO interface to ASTROS. The modules along with a brief functional descriptions are presented in Table 2.

Table 2. ZAERO Engineering Application Modules.

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>AEROGM</td>
<td>Aerodynamic Geometry Module</td>
</tr>
<tr>
<td>CONMOD</td>
<td>Control Surface Modes Generation</td>
</tr>
<tr>
<td>FLUTQHHHZ</td>
<td>Process matrix [AJK] with normal modes for flutter</td>
</tr>
<tr>
<td>FLUTSENZ</td>
<td>To compute the sensitivities of active flutter constraints in the current boundary condition</td>
</tr>
<tr>
<td>FLUTTRAZ</td>
<td>Perform flutter analysis in the current boundary condition and to evaluate any flutter constraints if it is an optimization boundary condition with applied flutter constraints</td>
</tr>
<tr>
<td>QHHLGENZ</td>
<td>Compute the unsteady aerodynamic matrices in the modal dynamic degrees of freedom for gust analysis</td>
</tr>
<tr>
<td>SPLINZ</td>
<td>Generate the spline matrix that relates displacements and forces between the structural model and aerodynamic models</td>
</tr>
<tr>
<td>SZAERO</td>
<td>Generate steady aerodynamic AIC matrices and aerodynamic forces of unit configurations</td>
</tr>
<tr>
<td>UZAERO</td>
<td>Unsteady aeroelastic analysis preface</td>
</tr>
</tbody>
</table>

For ease of understanding, these new engineering modules are documented in the same format as those presented in the ASTROS Programmer’s Manual (Ref 2). The modules presented provide the programmer a general description of the algorithm and clearly defines the module’s arguments. In addition, the purpose, MAPOL calling sequence, FORTRAN subroutine name and method (i.e. function) of the module is presented. In cases of similar methods employed by modules to those of ASTROS (Version 11.0), the user is referred to the ASTROS Programmer’s Manual (Ref 2).
Engineering Application Module: AEROGM

Entry Point: AEROGM

Purpose:
ZAERO geometry preface module.

MAPOL Calling Sequence:

CALL AEROGM ( AECOMPZ, GEOMZA, AGRIDZ );

AECOMPZ A relation describing aerodynamic components (Output)
GEOMZA A relation describing the aerodynamic boxes (Output)
AGRIDZ A relation describing the corner points of aerodynamic boxes (Output)

Application Calling Sequence:
None

Method:
The AEROGM module processes all BODY7 and CAERO7 bulk data entries and computes the geometric data stored in the relational entities AECOMPZ, GEOMZA, and AGRIDZ. These relational entites are to be used by the CONMOD, SPLINZ, UZAERO, and SZAERO modules.

Design Requirements:
The AEROGM module is executed in the preface phase. It is the aerodynamic geometry module for the ZAERO module.

Error Conditions:
None
Engineering Application Module: CONMOD

Entry Point: CONMOD

Purpose:
Control surface modes generation.

MAPOL Calling Sequence:
CALL CONMOD ( AECOMPZ, GEOMZA, [SCNTLG], [SCNTLK], [ACNTLG], [ACNTLK], 
[LMODEG], [LMODEK] );

AECOMPZ A relation created by the AEROGM module describing aerodynamic components 
(Character, Input)

GEOMZA A relation created by the AEROGM module describing the aerodynamic boxes 
(Character, Input)

[SCNTLG] Matrix whose rows contain the symmetric control surface modes defined at the G-set 
D.O.F. and columns are associated with the AESURFZ bulk data entries. [SCNTLG] 
is used to compute the inertia loads by unit deflection angle of control surfaces. (Output)

[SCNTLK] Matrix whose rows contain the symmetric control surface modes defined at the K-set 
D.O.F. and columns are associated with the AESURFZ bulk data entries. [SCNTLK] 
is used to compute the unsteady aerodynamic forces [AJC] and steady aerodynamic 
forces [AIRFRFC] by unit deflection angle of the control surfaces. (Output)

[ACNTLG] Same as [SCNTLG] but for antisymmetric control surface modes (Output)

[ACNTLK] Same as [SCNTLK] but for antisymmetric control surface modes (Output)

[LMODEG] Matrix whose rows contain the load modes at the G-set D.O.F. and columns are 
associated with the LOADMOD bulk data entries (Output)

[LMODEK] Matrix whose rows contain the load modes at the K-set D.O.F. and columns are 
associated with the LOADMOD bulk data entries (Output)

Application Calling Sequence:
None

Method:
First, the CONMOD module processes all AESURFZ bulk data entries (if there are any) and generates the 
control surface modes due to unit deflection angle of the control surfaces about the hinge lines in both G-set and 
K-set D.O.F. If TYPE = 'SYM' or 'ASYM', the control surface modes are stored in [SCNTLG] and 
[SCNTLK]. If TYPE = 'ANTISYM', the control surface modes are stored in [ACNTLG] and [ACNTLK].

Next, the CONMOD module processes all LOADMOD bulk data entries (if there are any) and generates the 
load modes of each LOADMOD. The load modes are defined in the G-set and K-set D.O.F. and stored in each 
row of the matrix [LMODEG] and [LMODEK], respectively.
Design Requirements:
None

Error Conditions:
None
Engineering Application Module: FLUTQHHZ

Entry Point: FLTQHZ

Purpose:
Processes matrix \([AJK]\) with normal modes for flutter.

MAPOL Calling Sequence:

```fortran
CALL FLUTQHHZ ( NITER, BCID, SUB, ESIZE(BC), PSIZE(BC), [AJK], [SKJ],
                [UGTKA], [PHIA], USET(BC), [TMN(BC)], [GSUBO(BC)], NGDR,
                AECOMPZ, GEOMZA, [PHIKH], [QHHLFL(BC, SUB)], OAGRDDSP );
```

- **NITER**
  Design iteration number (Integer, Input)

- **BCID**
  Boundary condition number (Integer, Input)

- **SUB**
  Flutter subcase number (Integer, Input)

- **ESIZE(BC)**
  Number of extra points for the current boundary condition (Integer, Input)

- **PSIZE(BC)**
  Number of physical degrees of freedom in the current boundary conditions (Integer, Input)

- **[AJK]**
  Unsteady AIC matrices generated by the UZAERO module (Input)

- **[SKJ]**
  Integration matrix generated by the UZAERO module (Input)

- **[UGTKA]**
  The matrix of splining coefficients relating the aerodynamic pressures and forces at the structural grids and relating the structural displacements to the streamwise slopes of the aerodynamic boxes. \([UGTKA]\) is reduced to the a-set DOF from \([UGTKG]\). (Input)

- **[PHIA]**
  Matrix of normal modes eigenvectors in the a-set (Input)

- **USET(BC)**
  Current boundary condition's unstructured entity of set definition masks (expanded to include extra points and any GDR scalar points) (Input)

- **[TMN(BC)]**
  Multipoint constraint transformation matrix for the current boundary condition (Input)

- **[GSUBO(BC)]**
  Static condensation or GDR reduction matrix for the current boundary condition (Input)

- **NGDR**
  Denotes dynamic reduction in the boundary condition
  
  \[= 0 \text{ No GDR}\]
  
  \[= -1 \text{ GDR is used}\]
  
  (Input, Integer)

- **AECOMPZ**
  A relation describing aerodynamic components created by the AEROGM module (Character, Input)
GEOMZA A relation describing the aerodynamic boxes created by the AEROGM module
(Character, Input)

[PHIKH] A modal transformation matrix that relates the box-on-box aerodynamic motions to
unit displacements of the generalized structural coordinates (modes) (Output)

[QHHLFL (BC, SUB)] A matrix containing the list of h x h unsteady aerodynamics matrices for the current
flutter subcase related to the generalized (modal) coordinates and including control
effectiveness (CONEFFS), extra points and CONTROL matrix inputs, where BC
represents the MAPOL boundary condition loop index number (Output)

OAGRDDSP A relation containing the structural eigenvectors (generalized DOF) mapped to the
aerodynamic boxes for those AIRDISP requests in the Solution Control. These
terms are the columns of PHIKH put in relational form to satisfy the output
requests. (Output)

Application Calling Sequence:

None

Method:

FLUTQHHZ is very similar to the FLUTQHHL module (see FLUTQHHL Engineering Application Module
of ASTROS Programmer’s Manual for description of Method). There are only two differences between these
two modules.

1. FLUTQHHZ reads in [AJK] and [SJK] matrices and computes the QKK matrices as

   \[ QKK = [\text{SJK}]^T [\text{AJK}]^T \]

   then computes the generalized aerodynamic forces as

   \[ \text{QHHLFL} = \text{PHIKH}^T \text{QKK} \text{PHIKH} \]

   therefore, the [QKK] matrix is a intermediate matrix created in FLUTQHHZ. However, the actual
   procedure to compute [QHHLFL] in the FLUTQHHZ is described in ENTITY DESCRIPTIONS of AJK

2. FLUTQHHZ uses the relational entity REUNMK to retrieve the AIC matrices of the Mach number and
   associated reduced frequencies as defined in the IDMK of the FLUTTER bulk data entry.
Engineering Application Module: FLUTSENZ

Entry Point: FLTSTZ

Purpose:
To compute the sensitivities of active flutter constraints in the current boundary condition.

MAPOL Calling Sequence:

```fortran
CALL FLUTSENZ ( NITER, BC, SUB, LOOP, GSIZEB, NDV, GLBDES, CONST, GMKCT,
    DKVI, GMMCT, DMVI, CLAMBD, LAMBDA, [QHHLFL(BC, SUB)],
    [BHHFL(BC, SUB)], [KHHFL(BC, SUB)], [PHIG(BC)], [AMAT],
    AEROZ );
```

**NITER**
Design iteration number (Integer, Input)

**BC**
Boundary condition identification number (Integer, Input)

**SUB**
Flutter subcase number (Integer, Input)

**LOOP**
Logical flag indicating whether more flutter subcases exist in the current boundary condition (Logical, Input)

**GSIZEB**
The size of the structural set (Integer, Input)

**NDV**
The number of global design variables (Integer, Input)

**GLBDES**
Relation of global design variables (Character, Input)

**CONST**
Relation of constraint values (Character, Input)

**GMKCT**
Relation containing the connectivity data for the DKVI sensitivity matrix (Character, Input)

**DKVI**
Unstructured entity containing the stiffness design sensitivity matrix in a highly compressed format (Character, Input)

**GMMCT**
Relation containing connectivity data for DMVI sensitivity matrix (Character, Input)

**DMVI**
Unstructured entity containing the mass design sensitivity matrix in a highly compressed format (Character, Input)

**CLAMBD**
Relation containing results of flutter analysis (Character, Input)

**LAMBDA**
Relation containing the output from the real eigenanalysis (Character, Input)

**[QHHLFL(BC, SUB)]**
A matrix containing the list of h x h unsteady aerodynamics matrices for the current flutter subcase related to the generalized (modal) coordinates and including control effectiveness (CONEFFS), extra points and CONTROL matrix inputs, where BC represents the MAPOL boundary condition loop index number (Input)

16
Modal mass matrix (Input)
[BHHFL (BC, SUB) ]  Modal flutter damping matrix (Input)
[KHHFL (BC, SUB) ]  Modal flutter stiffness matrix (Input)
[PHIG (BC) ]        Matrix of real eigenvectors in the structural set (Input)
[AMAT]              Matrix of constraint sensitivities (Output)
AEROZ               Relation containing the definition of the aerodynamic coordinate system (Input)

Application Calling Sequence:
None

Method:

FLUTSENZ is very similar to the FLUTSENS module (see FLUTSENZ Engineering Application Module for description of Method). There is only one difference between these two modules. FLUTSENZ uses the relational entity REUNMK to retrieve the AIC matrices of the Mach number and associated reduced frequencies as defined in the IDMK of the FLUTTER bulk data entry.

Design Requirements:

The module assumes that at least one flutter subcase exists in the current boundary condition.

Error Conditions:

None.
Engineering Application Module: FLUTTRAZ

Entry Point: FLTTAZ

Purpose:
To perform flutter analyses in the current boundary condition and to evaluate any flutter constraints if the current boundary condition is an optimization boundary condition with applied flutter constraints.

MAPOL Calling Sequence:
CALL FLUTTRAZ ( NITER, BCID, SUB, [QHHLFL(BC, SUB)], LAMBDA, HSIZE(BC),
ESIZE(BC), GMKCT, [MHHFL(BC, SUB)], [BHHFL(BC, SUB)],
KHHFL(BC, SUB)], CLAMBDA, AEROZ );

NITER
Design iteration number (Integer, Input)

BCID
User defined boundary condition identification number (Integer, Input)

SUB
Flutter subcase number (ranging from 1 to the total number of FLUTTER subcases) of the subcase to be processed in this pass (Integer, Input)

[QHHLFL(BC, SUB)]
Matrix list of modal unsteady aerodynamic coefficients (Input)

LAMBDA
Relational entity containing the output from the real eigenanalysis (Character, Input)

HSIZE(BC)
Number of modal dynamic degrees of freedom in the current boundary condition (Input)

ESIZE(BC)
The number of extra point degrees of freedom in the current boundary condition (Integer, Input)

[MHHFL(BC, SUB)]
Modal mass matrix (Input)

[BHHFL(BC, SUB)]
Modal flutter damping matrix (Input)

[KHHFL(BC, SUB)]
Modal flutter stiffness matrix (Input)

CLAMBDA
Relation containing results of flutter analyses (Character, Input)

AEROZ
Relational entity of the configuration parameters defined by the AEROZ bulk data entry (Character, Input)

Application Calling Sequence:
None

Method:
FLUTTRAZ is very similar to the FLUTTRAN module (see FLUTTRAN Engineering Application Module of the ASTROS Programmer's Manual for a description of the Method). The difference is that rather than processing the UNMK unstructured entity, FLUTTRAZ reads the relational entity REUNMK for retrieving the Mach number and reduced frequency pairs.
Design Requirements:

The module assumes that at least one flutter subcase exists in the current boundary condition.

Error Conditions:

Referenced data on FLUTTER entries that do not exist on the database are flagged and the execution is terminated.
Engineering Application Module: **QHHLGENZ**

Entry Point: **QHJGEN**

**Purpose:**

To compute the unsteady aerodynamic matrices in the modal dynamic degrees of freedom for gust analysis.

**MAPOL Calling Sequence:**

```fortran
CALL QHHLGENZ ( BC, ESIZE(BC), [AJK], [SKJ], [QGK], [UGTKA], [PHIA], [PHIKH], [QHHL], [QHJL], AEROZ );
```

- **BC**
  Boundary condition identification number (Integer, Input)

- **ESIZE(BC)**
  The number of extra point degrees of freedom in the boundary condition (Integer, Input)

- **[AJK]**
  Unsteady AIC matrices generated by the UZAERO module (Input)

- **[SKJ]**
  Integration matrix generated by the UZAERO module (Input)

- **[QGK]**
  A matrix containing the intermediated gust vectors generated by the UZAERO module (Input)

- **[UGTKA]**
  The matrix of splining coefficients relating the aerodynamic pressures and forces at the structural grids and relating the structural displacements to the streamwise slopes of the aerodynamic boxes reduced to the a-set DOF. Generated by the SPLINZ module. (Input)

- **[PHIA]**
  Matrix of normal modes eigenvectors in the a-set (Input)

- **[PHIKH]**
  A modal tranformation matrix that relates the box-on-box aerodynamic motions to unit displacements of the generalized structural coordinates (modes) (Output)

- **[QHHL]**
  A matrix containing the list of h x h unsteady aerodynamics matrices of each reduced frequency for the current gust subcase related to the generalized (modal) coordinates (Output)

- **[QHJL]**
  A matrix containing the list of h x 1 unsteady hormonic gust vector of each reduced frequency (Output)

- **AEROZ**
  A relation containing the definition of the aerodynamic coordinate system (Input)

**Application Calling Sequence:**

None

**Method:**
QHHLGENZ is very similar to the QHHLGEN module (see QHHLGEN Engineering Application Module of the ASTROS Programmer's Manual for a description of the Method). There are only two differences between these two modules.

1. QHHLGENZ reads in \([AJK]\) and \([SJK]\) matrices and computes the \([QKK]\) matrices as

\[
[QKK] = [SJK]^T [AJK]^T
\]

then computes the generalized aerodynamic forces as

\[
[QHHL] = [PHIKH]^T [QKK] [PHIKH]
\]

therefore, the \([QKK]\) matrix is an intermediate matrix created in QHHGENZ.

2. The gust vector is computed as:

\[
[QHJL] = [PHIKH]^T [QGK] \exp(i*k/(REFC/2)\cdot x_0)
\]

where

- \(k\) is the reduced frequency.
- \(REFC\) is the reference chord.

and

- \(x_0\) is the location of the reference plane defined in the GUST bulk entry.

3. QHHLGENZ uses the relational entity REUNMK to retrieve the AIC matrices of the Mach number and associated reduced frequencies as defined in the IDMK of the GUST bulk data entry.
Engineering Application Module: SPLINZ

Entry Point: SPLINZ

Purpose: Generates the spline matrix that relates displacements and forces between the structural model and the ZAERO aerodynamic model.

MAPOL Calling Sequence:

CALL SPLINZ' ( GSIZEB, GEOMZA, AECOMPZ, AEROZ, [UGTKG] );

GSIZEB The number of degrees of freedom in the set of all structural GRID and SCALAR points (Integer, Input)

GEOMZA A relation describing the aerodynamic boxes for the ZAERO model. The location of the box centroid, normal and pitch moment axis are given. It is used in splining the aerodynamics to the structure and to map responses back to the aerodynamic boxes. (Character, Input)

AECOMPZ A relation describing aerodynamic components for the ZAERO model. It is used in splining the aerodynamics to the structural model. (Character, Input)

AEROZ A relation created by the AEROZ bulk entry (Character, Input)

[UGTKG] Spline matrix relating the structural displacements at G-set d.o.f to the displacements ans slopes at the K-set d.o.f of the aerodynamic boxes. (Output)

Application Calling Sequence:

None

Method:
The SPLINZ module is very similar to the SPLINES and SPLINEU modules (see ASTROS Programmer's Manual), except:

1. It only relates the aerodynamic boxes associated with BODY7 and CAERO7 to the structural model.
2. In addition to the SPLINE1, SPLINE2 and ATTACH bulk data entries, it also reads the SPLINE3 bulk data entry for 3D spline.
3. The spline matrix is used for both the steady and unsteady aeroelastic modules.

The spline matrix [UGTKG] is used for both steady aeroelastic analysis and dynamic aeroelastic analysis. For the definition of K-set d.o.f., please see entity descriptions of entity UGTKG.

Design Requirements:

None

Error Conditions:

1. Each aerodynamic box may appear on only one SPLINE1, SPLINE2, SPLINE3 or ATTACH entry, although not all boxes need appear. Missing boxes will not influence the aeroelastic response.
2. Missing structural grids or aerodynamic elements appearing on the spline definitions will be flagged.
Engineering Application Module: SZAERO

Entry Point: SZAERO

Purpose:
Generates steady aerodynamic AIC matrices and aerodynamic forces of unit configuration parameters by the ZAERO module.

MAPOL Calling Sequence:

```
CALL SZAERO' ( [AJK] , MINDEX, LOOP, AECOMPZ, GEOMZA, AGRIDZ, STABCF,
[AICMAT(MINDEX) , [AAICMAT(MINDEX) , [AIRFRC(MINDEX) ,
[SCNTLK] , [ACNTLK] ]);
```

AJK Unsteady AIC matrices generated by the UZAERO module (Input)

MINDEX Mach number index for the current pass. Controls which Mach number/symmetry conditions will be processed in this pass by SZAERO. One pass for each unique Mach number will be performed with MINDEX incrementing by one until SZAERO returns LOOP = .FALSE. (Input)

LOOP A logical flag set by SZAERO to indicate whether additional MINDEX subscripts are needed to complete the processing of all Mach number/symmetry conditions on all the TRIM bulk data entries. One pass for each unique Mach number will be performed with MINDEX incrementing by one until SZAERO returns LOOP = .FALSE. (Output)

AECOMPZ A relation created by the AEROGM module describing aerodynamic components (Character, Input)

GEOMZA A relation created by the AEROGM module describing the aerodynamic boxes (Character, Input)

AGRIDZ A relation created by the AEROGM module describing the corner points of aerodynamic boxes (Character, Input)

STABCF A relation of rigid aerodynamic stability coefficients for unit configuration parameters. The coefficients are stored in STABCF and the corresponding distributed forces are stored in [AIRFRC(MINDEX)]. The STABCF relation is used to pick the appropriate rigid loads from [AIRFRC(MINDEX)] when performing the aeroelastic trim as well as for retrieving the RIGID/FLEXIBLE stability coefficients for each configuration parameters. (Output)

[AICMAT(MINDEX)] Matrix containing the steady aerodynamic influence coefficients for symmetric flight condition (Output)

[AAICMAT(MINDEX)] Same as [AICMAT(MINDEX)] but for antisymmetric flight condition (Output)

[AIRFRC(MINDEX)] Matrix containing the steady aerodynamic distributed forces for unit configuration parameters for the current Mach number index. If both symmetric and antisymmetric conditions exist for the Mach number, both sets of configuration parameters will coexist in [AIRFRC]. (Output)
Matrix (created by the CONMOD module) whose rows contain the symmetric control surface modes defined at the K-set D.O.F. and columns are associated with the AESURFZ bulk data entries. [SCNTLK] is used to compute the aerodynamic stability coefficients and distributed forces contained in STABCF and [AIRFRC] by unit deflection of control surfaces. (Input)

Same as [SCNTLK] but for antisymmetric control surface modes.

Application Calling Sequence:
None

Method:
The SZAERO module is very similar to the STEADY module (see ASTROS Programmer’s Manual) except that SZAERO processes the aerodynamic geometry generated by the AEROGM module and computes the AIC matrices from ZONA6, ZONA7, ZTAIC, and ZONA7U methods for wing-body configurations. The output data format of SZAERO is identical to that of the STEADY module so that the output data can be directly used by the downstream steady aeroelastic trim modules.

The steady AIC matrices are obtained by taking the real part of the lowest reduced frequency of the matrix [AJK], where [AJK] is generated by UZAERO module.

Design Requirements:
See STEADY module.

Error Conditions:
See STEADY module.
Engineering Application Module: UZAERO

Entry Point: UZAERO

Purpose:

Unsteady aeroelastic analysis preface by ZAERO module.

MAPOL Calling Sequence:

CALL UZAERO ( AECOMPZ, GEOMZA, AGRIDZ, [AJK], [AJC], [AJL] [QGK], [SKJ], [SCNTLK], [ACNTLK], [LMODEK] );

AECOMPZ
A relation created by the AEROGM module describing aerodynamic components (Character, Input)

GEOMZA
A relation created by the AEROGM module describing the aerodynamic boxes (Character, Input)

AGRIDZ
A relation created by the AEROGM module describing the corner points of aerodynamic boxes (Character, Input)

[AJK]
Matrix containing the transposed unsteady aerodynamic influence coefficient (AIC) matrices for all Mach, and reduced frequency pairs defined in all MKAEROZ bulk data entries (Output)

[AJC]
Matrix containing the unsteady pressure in J-set D.O.F. on aerodynamic boxes due to the control surface modes for all Mach number and reduced frequency pairs defined in all MKAEROZ bulk data entries (Output)

[AJL]
Matrix containing the unsteady pressure in J-set D.O.F. on aerodynamic boxes due to the load modes for all Mach number and reduced frequency pairs defined in all MKAEROZ bulk data entries (Output)

[QGK]
Gust matrix containing the intermediated gust force vectors at the K-set D.O.F. for all Mach number and reduced frequency pairs defined in all MKAEROZ bulk data entries (Output)

[SKJ]
Integration matrix to take pressures in J-set D.O.F. to forces in K-set D.O.F (Output)

[SCNTLK]
Matrix (created by the CONMOD module) whose rows contain the symmetric control surface modes defined at the K-set D.O.F. and columns are associated with the AESURFZ bulk data entries. [SCNTLK] is used to compute the unsteady aerodynamic forces [AJC] by unit deflection of control surfaces. (Input)

[ACNTLK]
Same as [SCNTLK] but for antisymmetric control surface modes (Input)

[LMODEK]
Matrix (created by CONMOD module) whose rows contain load modes defined at the K-set D.O.F. and columns are associated with the LOADMOD bulk data entries. [LMODEK] is used to compute the unsteady aerodynamic forces [AJL] of the load modes. (Input)
Application Calling Sequence:

None

Method:

The UZAERO module first reads in the relational entity AEROZ to check the symmetric condition of the aerodynamic geometry. If XZSYM = 'YES', the symmetric AIC and antisymmetric AIC matrices will be generated regardless of whether they are required for the downstream unsteady aeroelastic modules. The AIC matrices are generated according to the input sequence of MKAEROZ bulk data entries. Each MKAEROZ will produce a set of AIC matrices at the given Mach number and its associated list of reduced frequencies. The geometric data of the aerodynamic model is based on the relations AECOMPZ, GEOMZA, and AGRIDZ.

The AIC matrices of Mach, reduced frequency, symmetry pairs are stored in $[AJK]$. $[AJC]$ is computed by:

$$[AJC] = [AJK]^T [SCNTLK], [ACNTLK]$$

pre-multiplied $[AJC]$ by $[SKJ]^T$ will yield the control surface aerodynamic forces at K-set D.O.F.

The intermediated gust force vector $[QGK]$ is computed by:

$$[QGK] = [SKJ]^T [AJK]^T \{\exp(-i*K*X/(REFC/2))\}$$

where $K$ is the reduced frequency.
$X$ is the aerodynamic box control point locations.
$REFC$ is the reference chord.

$[AJL]$ is computed by:

$$[AJL] = [AJK]^T [LMODEK]$$

pre-multiplied $[AJL]$ by $[SKJ]^T$ will yield the load mode aerodynamic forces at K-set D.O.F.

The method to retrieve the $[AJK]$ and $[AJC]$, and $[AJL]$ matrices of a given Mach number, reduced frequency, and symmetry pair is described in relational entity REUNMK.

Design Requirements:

Unlike the AMP module, the UZAERO module does not generate the $[QKK]$ matrix. The $[QKK]$ matrix is computed by the FLUTQHHZ module from:

$$[QKK] = [SKJ]^T [AJK]^T$$

The unsteady forces due to control surface modes (defined as $[QKC]$) can be computed by:

$$[QKC] = [SKJ]^T [AJC]$$

Error Conditions:

None
To facilitate the communication of data among the ZAERO engineering application modules, fifteen new database entities (11 Matrix and 4 Relational) are created and are presented in Table 3.

Table 3. ZAERO Database Entities.

<table>
<thead>
<tr>
<th>Entity Name</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AJC</td>
<td>Basic name of the unsteady aerodynamic matrix containing unsteady pressure coefficients at J-set d.o.f due to unit control surface deflections.</td>
<td>Matrix</td>
</tr>
<tr>
<td>QGK</td>
<td>Basic name of the unsteady aerodynamic gust force vector containing the intermediated unsteady forces at K-set d.o.f.</td>
<td>Matrix</td>
</tr>
<tr>
<td>SKJ</td>
<td>Integration matrix relating the unsteady aerodynamic pressure coefficients at the J-set d.o.f to the unsteady aerodynamic forces at the K-set d.o.f.</td>
<td>Matrix</td>
</tr>
<tr>
<td>AJK</td>
<td>Basic name of the unsteady aerodynamic AIC matrix relating the displacements at the K-set d.o.f to the pressure coefficients at the J-set d.o.f.</td>
<td>Matrix</td>
</tr>
<tr>
<td>ACNTLK</td>
<td>Displacements and slopes defined at K-set d.o.f due to unit anti-symmetric control surface deflection.</td>
<td>Matrix</td>
</tr>
<tr>
<td>SCNTLK</td>
<td>Translational and rotational displacements defined at G-set d.o.f due to unit symmetric control surface deflection.</td>
<td>Matrix</td>
</tr>
<tr>
<td>SCNTLG</td>
<td>Displacements and slopes defined at K-set d.o.f due to unit symmetric control surface deflection.</td>
<td>Matrix</td>
</tr>
<tr>
<td>ACNTLG</td>
<td>Translational and rotational displacements defined at G-set d.o.f due to unit anti-symmetric control surface deflection.</td>
<td>Matrix</td>
</tr>
<tr>
<td>LMODEG</td>
<td>Translational and rotational displacements defined at G-set d.o.f due to the load modes specified in bulk entries LOADMOD.</td>
<td>Matrix</td>
</tr>
<tr>
<td>LODEK</td>
<td>Displacements and slopes defined at K-set d.o.f due to the load modes specified in bulk entries LOADMOD.</td>
<td>Matrix</td>
</tr>
<tr>
<td>UGTKG</td>
<td>Spline matrix relating the structural displacements at G-set d.o.f to the displacements and slopes at the K-set d.o.f of the aerodynamic boxes, but stored in the transposed form.</td>
<td>Matrix</td>
</tr>
<tr>
<td>AECOMPZ</td>
<td>Contains data on the aerodynamic components in the CAERO7 and BODY7 bulk entries.</td>
<td>Relation</td>
</tr>
<tr>
<td>GEOMZA</td>
<td>Contains data on the aerodynamic boxes of the CAERO7 and BODY7 bulk entries.</td>
<td>Relation</td>
</tr>
<tr>
<td>AGRIDZ</td>
<td>Contains data of the corner grid points on the CAERO7 and BODY7 boxes.</td>
<td>Relation</td>
</tr>
<tr>
<td>REUNMK</td>
<td>Contains the relations between the unsteady aerodynamic matrices generated by the UZAERO module to the bulk entries MKAEROZ.</td>
<td>Relation</td>
</tr>
</tbody>
</table>

The ZAERO database entities are documented similar to those in the ASTROS Programmer’s Manual (Ref 2). A Usage section has been added to aide and clearly define to the programmer data stored on each database entity.
AJC contains a three characters string 'AJC' defined by MAPOL. To retrieve the AJC of a given Mach number, reduced frequency pair and symmetry condition, please see entity REUNMK.

The actual matrix name stored on the data base is $\text{AJC}_{siij}$, where $s$='S' for symmetric or asymmetric case, ='A' for antisymmetric case. $ii$=index of Mach number. $jj$=index of reduced frequency.

The matrix $\text{QKC}$ defined as the unsteady aerodynamic forces due to unit control surface deflections at K-set is computed by:

$$[\text{QKC}] = [\text{SKJ}]^T[\text{AJC}_{siij}]$$

The unsteady generalized aerodynamic control forces $[\text{QHCLFL}]$ is computed by:

$$[\text{QHCLFL}] = [\text{PHIKH}]^T[\text{QKC}]$$

where $[\text{PHIKH}]$ is the modal matrix at K-set d.o.f.

Therefore the number of rows of $[\text{QHCLFL}]$ is the number of modes. Each column of $[\text{QHCLFL}]$ corresponds to the generalized aerodynamic control forces due to each of the bulk entry AESURFZ with TYPE=SYM for $\text{AJC}_{siij}$ and TYPE=ANTISYM for $\text{AIC}_{aijj}$. 
QGK

Matrix

Basic name of the unsteady aerodynamic gust force vector containing the intermediated unsteady forces at K-set d.o.f. QGK is used by the aeroservoelastic gust analysis.

Complex matrix with one column and K-set number of rows.

UZAERO

QGK contains a three character string 'QGK' defined by MAPOL. To retrieve the QGK of a given Mach number, reduced frequency pair and symmetry condition, please see entity REUNMK.

The actual matrix name stored on the data base is QGK_siijjj,
where s='S' for symmetric or asymmetric case, ='A' for antisymmetric case.
   ii=index of Mach number.
   jj=index of reduced frequency.

The actual gust generalized forces in modal space is computed by:

\[ [QGK_{siijj}] = [QGK_{siijj}] \times \exp(i*k*x_0/(REFC/2.)) \]

where \( x_0 \) is the location of the reference plane defined in the bulk entry GUST.
   \( k \) is the corresponding reduced frequency.
   \( \text{and REFC is the reference chord defined in bulk entry AEROZ.} \)
Entity: SKJ
Entity Type: MATRIX
Description: Integration matrix relating the unsteady aerodynamic pressure coefficients at the J-set d.o.f. to the unsteady aerodynamic forces at the K-set d.o.f.
Matrix Form: Real matrix with J-set number of column and K-set number of rows but stored in the transposed form.
Created by: UZAERO
Usage: SKJ depends on the geometry of the aerodynamic model only and is independent of Mach number and reduced frequency.
Entity: AJK
Entity Type: MATRIX
Description: Basic name of the unsteady aerodynamic AIC matrix relating the displacements at the K-set d.o.f to the pressure coefficients at the J-set d.o.f.
Matrix Form: Complex matrix with K-set number of columns and J-set number of rows but stored in the transposed form.
Created by: UZAERO

Usage:

AJK contains a three characters string 'AJK' defined by MAPOL. To retrieve the AJK of a given Mach number, reduced frequency pair and symmetry condition, please see entity REUNMK.

The actual matrix name stored on the data base is AJK_{sijj}.
where s='S' for symmetric or asymmetric case, ='A' for antisymmetric case.
ii=index of Mach number.
jj=index of reduced frequency.

The matrix QKK relating displacements at K-set to unsteady aerodynamic forces at K-set is computed by:

\[
[QKK] = [SKJ]^T [AJK_{sijj}]^T
\]

The unsteady generalized aerodynamic forces \([QHHLFL]\) is computed by:

\[
[QHHLFL] = [PHIKH]^T [QKK][PHIKH]
\]
where \([PHIKH]\) is the modal matrix at K-set d.o.f.

However, in the FLUTQHHZ module and QHHLGENZ module, \([QHHLFL]\) is computed by the following procedure:

The unsteady aerodynamic pressure coefficients \([CP]\) at J-set d.o.f. is first obtained

\[
[CP] = [AJK_{sijj}]^T [PHIKH]
\]
Then, the aerodynamic forces at K-set d.o.f are computed:

\[
[FORCE] = [SKJ]^T [CP]
\]
Finally, the generalized aerodynamic forces are computed:

\[
[QHHLFL] = [PHIKH]^T [FORCE]
\]
Matrices \([CP]\) and \([FORCE]\) are deleted after \([QHHLFL]\) is obtained.
Entity: ACNTLK
Entity Type: MATRIX
Description: Displacements and slopes defined at K-set d.o.f. due to unit anti-symmetric control surface deflection. Each column is corresponding to each AESURFZ bulk entry with TYPE=ANTISYM.
Matrix Form: Real matrix with K-set number of rows and number of columns being equal to the number of AESURFZ bulk entries with TYPE=ANTISYM.
Created by: CONMOD
Usage:
1. ACNTLK is used by both UZAERO and SZAERO modules.
   For the UZAERO module, it generates the \([AJC]\) matrix for all MKAEROZ bulk entries by:
   \[ [AJC] = [AKJ]^T[ACNTLK] \]
   For the SZAERO module, it generates the matrix \([AIRFRC]\) and the aerodynamic stability coefficients of control surfaces (stored in relation STABCF) for each TRIM bulk entry by:
   \[ [AIRFRC] = [AAICMAT]^T[ACNTLK] \]
2. ACNTLK does not exist if there are no AESURFZ with TYPE=ANTISYM.
Entity:        SCNTLK
Entity Type:   MATRIX
Description:   Displacements and slopes defined at K-set d.o.f. due to unit symmetric control surface deflection. Each column corresponds to each AESURFZ bulk entry with TYPE=SYM or ASYM.
Matrix Form:   Real matrix with K-set number of rows and number of columns being equal to the number of AESURFZ bulk entries with TYPE=SYM or ASYM.
Created by:   CONMOD
Usage:

1. SCNTLK is used by both the UZAERO and SZAERO modules.
   For UZAERO module, it generates the [AJC] matrix for all MKAEROZ bulk entries by:
   \[[AJC]=[AJK]^T[SCNTLK]\]
   For the SZAERO module, it generates the matrix [AIRFRC] and the aerodynamic stability coefficients of control surfaces (stored in relation STABCF) for each TRIM bulk entry by:
   \[[AIRFRC]=[AICMAT]^T[SCNTLK]\]

2. SCNTLK does not exist if there are no AESURFZ with TYPE=SYM or ASYM.
Entity: SCNTLG
Entity Type: MATRIX
Description: Translational and rotational displacements defined at G-set d.o.f. due to unit symmetric control surface deflection. Each column corresponds to an AESURFZ bulk entry with TYPE=SYM or ASYM.
Matrix Form: Real matrix with G-set number of rows and number of columns being equal to the number of AESURFZ bulk entries with TYPE=SYM or ASYM.
Created by: CONMOD

Usage:

1. SCNTLG is used to compute the inertial matrix of the control surfaces in modal space by:

\([\text{PHIG}]^T[\text{MGG}][\text{SCNTLG}]\) in G-set d.o.f.

or

\([\text{PHIA}]^T[\text{MAA}][\text{SCNTLA}]\) in A-set d.o.f. Where [SCNTLA] can be computed by the reduction of [SCNTLG] from G-set to A-set.

2. SCNTLG does not exist if there are no AESURFZ with TYPE=SYM or ASYM.
Entity:          ACNTLG
Entity Type:     MATRIX
Description:    Translational and rotational displacements defined at G-set d.o.f. due to unit anti-
symmetric control surface deflection. Each column corresponds to an AESURFZ
bulk entry with TYPE=ANTISYM.
Matrix Form:     Real matrix with G-set number of rows and number of columns being equal to the
number of AESURFZ bulk entries with TYPE=ANTISYM.
Created by:      CONMOD
Usage:
1. ACNTLG is used to compute the inertial matrix of the control surfaces in modal space by:
   \[ \text{[PHIG]}^T \text{[MGG]} \text{[ACNTLG]} \]  in G-set d.o.f.
   or
   \[ \text{[PHIA]}^T \text{[MAA]} \text{[ACNTLA]} \]  in A-set d.o.f. Where \([\text{ACNTLA}]\) can be computed by the
   reduction of \([\text{ACNTLG}]\) from G-set to A-set.
2. ACNTLG does not exist if there are no AESURFZ with TYPE=ANTISYM.
Entity: LMODEG
Entity Type: MATRIX
Description: Translational and rotational displacements defined at G-set d.o.f due to the load modes specified in bulk entries LOADMOD.
Matrix Form: Real matrix with G-set number of rows and number of columns being equal to the number of LOADMOD bulk entries.
Created by: CONMOD
Usage:

1. LMODEG is used to compute the sectional forces or moments at the structural grid points defined by the LOADMOD bulk entries. LMODEG can be reduced from G-set to A-set d.o.f. by the A-set reduction procedures.

2. LMODEG does not exist if there are no LOADMOD bulk data entries.
Entity: **LMODEK**

Entity Type: **MATRIX**

Description: Displacements and slopes defined at K-set d.o.f due to the load modes specified in bulk entries **LOADMOD**.

Matrix Form: Real matrix with K-set number of rows and number of columns being equal to the number of **LOADMOD** bulk entries.

Created by: **CONMOD**

Usage:

1. **LMODEK** is used to compute the sectional forces or moments at the aerodynamic boxes defined by the **LOADMOD** bulk entries.

2. **LMODEK** does not exist if there are no **LOADMOD** bulk data entries.
1. The definition of K-set d.o.f. is:

For each aerodynamic box, six d.o.f.'s are assigned and defined as:

\{T1, T2, T3, d(T1)/dx, d(T2)/dx, d(T3)/dx\}, where T1, T2, and T3 are the displacements at the centroid of the aerodynamic box along x, y, and z directions, respectively. d( )/dx denotes as the slope of ( ) with respect to the free stream direction (the x-axis of the aerodynamic coordinates).

Therefore, for N number of aerodynamic boxes (number of J-set d.o.f.'s = N), number of K-set d.o.f.'s = 6 * N.

2. [UGTKG] can be reduced to [UGTKA] by the A-set reduction procedures, where [UGTKA] is used to transform the displacements at A-set to K-set and transform the aerodynamic forces from K-set to A-set by the transposed of [UGTKA].

3. [UGTKG] is computed according to the SPLINE1, SPLINE2, SPLINE3, and ATTACH bulk entries.
AECOMPZ

Entity: AECOMPZ
Entity Type: Relation
Description: Contains data on the aerodynamic components in the CAER07 and BODY7 bulk data entries.

Relation Attributes:

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE/KEY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACID</td>
<td>Integer&gt;0</td>
<td>Identification number of CAERO7 or BODY7 bulk entries.</td>
</tr>
<tr>
<td>MACROTYP</td>
<td>Text(8)</td>
<td>Either 'CAER07' or 'BODY7'.</td>
</tr>
<tr>
<td>GROUP</td>
<td>Integer</td>
<td>Identification number of the ACOORD bulk entry.</td>
</tr>
<tr>
<td>ACMPNT</td>
<td>Text(8)</td>
<td>Component type. One of WING or BODY.</td>
</tr>
<tr>
<td>TYPE</td>
<td>Integer&gt;0</td>
<td>TYPE=2 for CAER07, 3 for BODY7.</td>
</tr>
<tr>
<td>FIINTID</td>
<td>Integer&gt;0</td>
<td>First internal aerodynamic box identification number.</td>
</tr>
<tr>
<td>NCBOX</td>
<td>Integer&gt;0</td>
<td>Number of chordwise boxes for CAER07.</td>
</tr>
<tr>
<td>NSBOX</td>
<td>Integer&gt;0</td>
<td>Number of spanwise boxes for CAER07.</td>
</tr>
<tr>
<td>BNDRY</td>
<td>R Vector(12)</td>
<td>For CAER07: BNDRY(i), i=1,3: x, y, z of leading edge at root. BNDRY(i), i=4,6: x, y, z of trailing edge at root. BNDRY(i), i=7,9: x, y, z of leading edge at tip. BNDRY(i), i=10,12: x, y, z of trailing edge at tip. For BODY7: BNDRY(i), i=1,3: x, y, z of the nose. BNDRY(4): base pressure of the body wake. BNDRY(5): X location of the steady point singularity of the body wake. BNDRY(6): X location of the unsteady point singularity of the body wake. BNDRY(i), i=7,8: Y and Z offset for the point singularity of the body wake. BNDRY(9): Body length. BNDRY(10): Flag for body wake. (Integer) BNDRY(11): Number of inlet boxes. (Integer) BNDRY(12): Number of wake boxes on the body.</td>
</tr>
<tr>
<td>WCOS</td>
<td>For CAER07: Cos(theta), where theta = dihedral angle. For BODY7: Number of segments. (Integer)</td>
<td></td>
</tr>
<tr>
<td>WSIN</td>
<td>For CAER07: Sin(theta), where theta = dihedral angle. For BODY7: Not used.</td>
<td></td>
</tr>
<tr>
<td>IWING</td>
<td>Integer</td>
<td>Flag for vertical fin on the X-Z plane. =0: yes. =1, no.</td>
</tr>
<tr>
<td>ATTR</td>
<td>Integer</td>
<td>=0: CAERO7 root is not attached to BODY7. &gt;0: CAERO7 root is attached to BODY7 with ID=ATTR. Not used for BODY7.</td>
</tr>
<tr>
<td>YRB</td>
<td>Real</td>
<td>Y location of the center line of BODY7 to which the CAERO7 root is attached.</td>
</tr>
<tr>
<td>ZRB</td>
<td>Real</td>
<td>Z location of the center line of BODY7 to which the CAERO7 root is attached.</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FLCOSR</td>
<td>Real</td>
<td>$\cos(\theta)$, where $\theta$ is the dihedral angle of the vortex-carry-through boxes at root.</td>
</tr>
<tr>
<td>FLSINR</td>
<td>Real</td>
<td>$\sin(\theta)$, where $\theta$ is the dihedral angle of the vortex-carry-through boxes at root.</td>
</tr>
</tbody>
</table>
| ATTT     | Integer| $=0$: CAERO7 Tip is not attached to BODY7.  
$>0$: CAERO7 Tip is attached to BODY7 with ID=ATTT.  
Not used for BODY7. |
| YTB      | Real   | Y location of the center line of BODY7 if CAERO7 tip is attached to it. |
| ZTB      | Real   | Z location of the center line of BODY7 if CAERO7 root is attached to it. |
| FLCOST   | Real   | $\cos(\theta)$, where $\theta$ is the dihedral angle of the vortex-carry-through boxes at tip. |
| FLSINT   | Real   | $\sin(\theta)$, where $\theta$ is the dihedral angle of the vortex-carry-through boxes at tip. |
| LABEL    | Text(8)| Label of CAERO7 or BODY7 bulk entries. |

**Created by:** AEROGM

**Usage:**

AECOMPZ is used by SPLINZ, UZAERO and SZAERO modules.
Entity: GEOMZA
Entity Type: Relation
Description: Contains data on the aerodynamic boxes of the CAERO7 and BODY7 bulk data entries.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE/KEY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACROID</td>
<td>Integer</td>
<td>Component identification number of the associated CAERO7 or BODY7.</td>
</tr>
<tr>
<td>ACMPNT</td>
<td>Text(8)</td>
<td>=&quot;FUSEL&quot; for BODY7 box, =&quot;WING&quot; for CAERO7 box.</td>
</tr>
<tr>
<td>NDOF</td>
<td>Integer</td>
<td>=3 for BODY7 box, =2 for CAERO7 box.</td>
</tr>
<tr>
<td>EXTID</td>
<td>Integer</td>
<td>External identification number of the box.</td>
</tr>
<tr>
<td>INTID</td>
<td>Integer</td>
<td>Internal identification number of the box.</td>
</tr>
<tr>
<td>AREA</td>
<td>Real</td>
<td>Area of the box.</td>
</tr>
<tr>
<td>X</td>
<td>Real</td>
<td>X location of centroid of the box.</td>
</tr>
<tr>
<td>Y</td>
<td>Real</td>
<td>Y location of centroid of the box.</td>
</tr>
<tr>
<td>Z</td>
<td>Real</td>
<td>Z location of centroid of the box.</td>
</tr>
<tr>
<td>N1</td>
<td>Real</td>
<td>X component of the box normal in basic coordinates.</td>
</tr>
<tr>
<td>N2</td>
<td>Real</td>
<td>Y component of the box normal in basic coordinates.</td>
</tr>
<tr>
<td>N3</td>
<td>Real</td>
<td>Z component of the box normal in basic coordinates.</td>
</tr>
<tr>
<td>R1</td>
<td>Real</td>
<td>X component of the box local pitch axis in basic coordinates.</td>
</tr>
<tr>
<td>R2</td>
<td>Real</td>
<td>Y component of the box local pitch axis in basic coordinates.</td>
</tr>
<tr>
<td>R3</td>
<td>Real</td>
<td>Z component of the box local pitch axis in basic coordinates.</td>
</tr>
<tr>
<td>RTHETA</td>
<td>Real</td>
<td>For BODY7 box: dihedral angel of the box.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For CAERO7 box: Thickness slope at 50% chord.</td>
</tr>
<tr>
<td>RDELTA</td>
<td>Real</td>
<td>For BODY7 box: Inclination angel of the box.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For CAERO7 box: Camber slope at 50% chord.</td>
</tr>
<tr>
<td>CHORD</td>
<td>Real</td>
<td>Chord length.</td>
</tr>
<tr>
<td>ID1</td>
<td>Integer</td>
<td>Aerodynamic grid identification number at left hand side corner of the box leading edge.</td>
</tr>
<tr>
<td>ID2</td>
<td>Integer</td>
<td>Aerodynamic grid identification number at left hand side corner of the box trailing edge.</td>
</tr>
<tr>
<td>ID3</td>
<td>Integer</td>
<td>Aerodynamic grid identification number at right hand side corner of the box leading edge.</td>
</tr>
<tr>
<td>ID4</td>
<td>Integer</td>
<td>Aerodynamic grid identification number at right hand side corner of the box trailing edge.</td>
</tr>
<tr>
<td>CAM85</td>
<td>Real</td>
<td>Camber slope at 85% chord for CAERO7 box. Not used for BODY7 box.</td>
</tr>
<tr>
<td>CAM95</td>
<td>Real</td>
<td>Camber slope at 95% chord for CAERO7 box. Not used for BODY7 box.</td>
</tr>
<tr>
<td>DZX85</td>
<td>Real</td>
<td>Thickness slope at 85% chord for CAERO7 box. Not used for BODY7 box.</td>
</tr>
<tr>
<td>DZX95</td>
<td>Real</td>
<td>Thickness slope at 95% chord for CAERO7 box. Not used for BODY7 box.</td>
</tr>
<tr>
<td>Variable</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>DZXLE</td>
<td>Real</td>
<td>Thickness slope at leading edge of the mid-chord for CAERO7 box. Not used for BODY7 box.</td>
</tr>
<tr>
<td>DZXTE</td>
<td>Real</td>
<td>Thickness slope at trailing edge of the mid-chord for CAERO7 box. Inlet panel flow ratio in percentage for BODY7 box.</td>
</tr>
<tr>
<td>IWAKE</td>
<td>Integer</td>
<td>For BODY7 box=1, box is inlet panel. =0, box is not inlet panel. Not used for CAERO7 box.</td>
</tr>
</tbody>
</table>

Created by: **AEROGM**

**Usage:**

GEOMZA is used by SPLINZ, UZAERO and SZAERO modules.
Entity: AGRIDZ

Entity Type: Relation

Description: Contains data of the corner grid points on the CAERO7 and BODY7 boxes.

Relation Attributes:

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE/KEY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXTID</td>
<td>Integer&gt;0</td>
<td>External identification of the grid point.</td>
</tr>
<tr>
<td>INTID</td>
<td>Integer&gt;0</td>
<td>Internal identification of the grid point.</td>
</tr>
<tr>
<td>CORD</td>
<td>Integer</td>
<td>Identification number of ACOORD bulk entry.</td>
</tr>
<tr>
<td>X</td>
<td>Real</td>
<td>X location of the grid point.</td>
</tr>
<tr>
<td>Y</td>
<td>Real</td>
<td>Y location of the grid point.</td>
</tr>
<tr>
<td>Z</td>
<td>Real</td>
<td>Z location of the grid point.</td>
</tr>
</tbody>
</table>

Created by: AEROGM

Usage: AGRIDZ is used by UZAERO and SZAERO modules.
Contents the relations between the unsteady aerodynamic matrices generated by the UZAERO module to the bulk entries MKAEROZ.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE/KEY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDMK</td>
<td>Integer&gt;0</td>
<td>Identification number specified in the bulk entries MKAEROZ.</td>
</tr>
<tr>
<td>MACH</td>
<td>Real≥0</td>
<td>Mach number specified in bulk entries MKAEROZ.</td>
</tr>
<tr>
<td>METHOD</td>
<td>Integer</td>
<td>Method flag specified in bulk entries MKAEROZ.</td>
</tr>
<tr>
<td>SYMXZ</td>
<td>Integer</td>
<td>Symmetry flag. SYMXZ=1 for symmetric case, =-1 for antisymmetric case, =0 for asymmetric case.</td>
</tr>
<tr>
<td>ALPHA</td>
<td>Real</td>
<td>Angle of attack specified in the TRIMFLT bulk entry of the current MKAEROZ.</td>
</tr>
<tr>
<td>BETA</td>
<td>Real</td>
<td>Side slip angle specified in the TRIMFLT bulk entry of the current MKAEROZ.</td>
</tr>
<tr>
<td>PRATE</td>
<td>Real</td>
<td>Non-dimensional roll rate specified in the TRIMFLT bulk entry of the current MKAEROZ.</td>
</tr>
<tr>
<td>QRATE</td>
<td>Real</td>
<td>Non-dimensional pitch rate specified in the TRIMFLT bulk entry of the current MKAEROZ.</td>
</tr>
<tr>
<td>RRATE</td>
<td>Real</td>
<td>A non-dimensional yaw rate specified in the TRIMFLT bulk entry of the current MKAEROZ.</td>
</tr>
<tr>
<td>MINDEX</td>
<td>integer&gt;0</td>
<td>Index of the MKAEROZ bulk entry ranging from 1 to the number of the MKAEROZ bulk entries.</td>
</tr>
<tr>
<td>KINDEX</td>
<td>Integer&gt;0</td>
<td>Index of the reduced frequency ranging from 1 to the number of reduced frequencies specified in the current MKAEROZ.</td>
</tr>
<tr>
<td>RFREQ</td>
<td>Real&gt;0.0</td>
<td>The KINDEX'th reduced frequency specified in the current MKAEROZ.</td>
</tr>
</tbody>
</table>

Created by: UZAERO

Usage:

The UZAERO module generates the unsteady aerodynamic matrices [AJK], [AJC], and [QGK] of all MKAEROZ bulk entries in the input file regardless of whether or not they are required for the downstream unsteady aeroelastic modules. To retrieve these matrices, please see the example on the following page.
For a given pair of IDMK and SYMXZ found in either the FLUTTER or GUST bulk entry, to retrieve the corresponding matrix $[AJK]$:

```
CHARACTER*8 UNLIST(12), NAME
DATA UNLIST/ 'IDMK', 'MACH', 'METHOD', 'SYMXZ', 'ALPHA', 'BETA', 'PRATE', 'QRATE',
'RRATE', 'MINDEX', 'KINDEX', 'RFREQ' /
INTEGER INFO(20), IGET(12), MINDEX(100), KINDEX(100), SYMXZ
REAL ROET(12), K(100), MACH
EQUIVALENCE (ROET(1), IGET(1))
CHARACTER*1 S
CALL DBOPEN(REUNMK, INFO, 'RO', 'NOFLUSH', ISTAT)
CALL REPROJ(REUNMK, 12, UNLIST)
NMK = INFO(1)
INDEX = 0
DO I = 1, NMK
   CALL REGET(REUNMK, IGET, ISTAT)
   IF(IDMK .EQ. IGET(1)) THEN
      INDEX = INDEX + 1
      MACH = REGET(2)
      METHOD = IGET(3)
      ISYM = IGET(4)
      MINDEX(INDEX) = IGET(10)
      KINDEX(INDEX) = IGET(11)
      K(INDEX) = REGET(12)
   ENDIF
ENDDO
CALL DBCLOS(REUNMK)
KTOTAL = INDEX
C KTOTAL is the total number of reduced frequencies specified in the MKAERO2 bulk entry.
C With IDMK as the identification number.
C If one wishes to retrieve the $[AJK]$ matrix of the second reduced frequency, do the following:
KTH = 2
IF(SYMXZ .EQ. 1 .OR. SYMXZ .EQ. 0) THEN
   S = 'S'
ELSE
   S = 'A'
ENDIF
C Subroutine MYNAME is an utility routine to assemble the matrix name.
C INPUT: AJK A three characters string contains the basic name of the matrix.
C $S$ = 'S' for symmetric or asymmetric case, = 'A' for antisymmetric case.
C MINDEX(KTH) KTH'th Mach number index found in the REUNMK relation.
C KINDEX(KTH) KTH'th reduced frequency index found in the REUNMK relation.
C OUTPUT: A character*8 string of the matrix created by UZAERO module with the form:
C AJK$s_{ii}j{j}$, where $s = S$, $i = MINDEX(KTH)$, and $j = KINDEX(KTH)$
C Now, NAME is the matrix name of the AIC matrix of the corresponding Mach number and
C reduced frequency.
C Now, NAME is the matrix name of the control surface forces matrix of the corresponding
C Mach number and reduced frequency.
C Now, NAME is the matrix name of the gust force matrix of the corresponding
C Mach number and reduced frequency.
C ....................
```

46
6.0 REFERENCES

APPENDIX A

ZAERO FUNCTIONAL MODULE DEFINITION

(MODDEF.DAT)
The following is a list of all ZAERO module definitions added to ASTROS and found in file MODDEF.DAT.

AEROGM 3
102 7 7 7
C AERO GEOMETRY FOR ZAERO MODULE
C NOTE: ALPHABETICAL ORDER IN FILE MODDEF.DAT IS NOT REQUIRED
C CALL AEROGM ( EP(1), EP(2), EP(3) )
END

CONNOD 8
102 7 7 8 8 8 8 8
C ZAERO CONTROL MODE GENERATOR
1 EP(8) )
END

FLUTQHZ 18
102 -1 1 1 8 8 8 8 8 8 1 7 1 7 1 7 8 8 8 8 7
C PROCESS THE 'FLUTQHZ' MODULE - FLUTTER AEROMATRIX PROCESSOR
C CALL FLUTQHZ { IP(1), IP(2), IP(3), IP(4), IP(5), EP(6), EP(7),
END

FLUTSENZ 21
102 1 1 4 1 1 7 7 7 9 7 9 7 7 8 8 8 8 8 8
C PROCESS THE 'FLUTSENZ' MODULE TO OBTAIN FLUTTER CONST. SENSITIV.
* EP(20), EP(21) }
END

FLUTTANZ 13
102 -1 1 1 8 7 1 1 8 8 8 7 7 7 7
C PROCESS THE 'FLUTTANZ' MODULE TO PERFORM FLUTTER ANALYSIS
C CALL FLUTTANZ { IP(1), IP(2), IP(3), EP(4), EP(5), IP(6), IP(7),
END

QHHLGENZ 11
102 1 1 8 8 8 8 8 8 8 7
C 'QHHLGENZ' - GENERATE THE QHH MATRIX LIST FOR FLUTTER ANALYSIS
END

SPLINZ 5
102 1 7 7 7 8
C PROCESS THE UNSTEADY AERODYNAMIC SPLINE
END
SZAERO 12
102 8 1 4 7 7 7 0 8 0 8 0
C
C PROCESS SZAERO STEADY AERODYNAMICS
C (PREFACE TO STATIC AEROELASTICITY DISCIPLINE)
C
END

UZAERO 11
102 7 7 7 8 8 8 8 8 8 8 8
C
C AIC GENERATION BY ZAERO MODULE
C
END

INPUT4 6
102 -1 1 7 8 8 8
C
C READ MODAL RESULTS FROM NASTRAN OUTPUT4 SOLUTION
C AND REPLACE THE ASTROS DATABASE MATRICES KAA, MAA, PHIA
C AND RELATION LAMBDA
C
END
APPENDIX B

ASTROS* MAPOL SEQUENCE LISTING
The following ASTROS* MAPOL sequence listing documents all changes made to the original ASTROS MAPOL sequence. All newly added lines and commented lines for integration of ZAERO into ASTROS are highlighted in boldfaced text. Arrows are also used at the ends of the lines to demarcate the beginning and ending of changes.

ASTROS* MAPOL Sequence Listing:

```
***** MAPOL SOURCE CODE LISTING *****
STAT.LEVL
1 1$i***$ 1
2 1$i CSCIID <8($) MCO083-MAPOLSEQ 11.1 4/29/94 17:00:35$ 1
3 1$i***$ 1
4 1$i EXECUTIVE SEQUENCE FOR ASTROS$ 1
5 1$i* 1
6 1$i* CONSTANTS FOR SDCOMP SET SINGULARITY MESSAGES$ 1
7 1$i* 1
8 1$i* INTEGER SINGSET, SINGSET, SINGLESET; 1
9 1$i* 1
10 1$i* VARIABLE DECLARATION SEGMENT$ 1
11 1$i* 1
12 1$i* 1
13 1$i* 1
14 1$i* 1
15 1$i* INTEGER GSIZE, NVT, NITER, BC; 1
16 1$i* ESIZE(IOOO), PRIZE(IOOO), GSIZE; 1
17 1$i* 1
18 1$i* REAL CTLE, CTIMIN; 1
19 1$i* 1
20 1$i* UNSTRUCT grid, GRIDTEMP, SMPLID; 1
21 1$i* 1
22 1$i* RELATION MF, PRINT, GRID, SPOINT, EPOINT, 1
23 1$i* 1
24 1$i* TEMP, TEMP, OCP, TEMP, OCP, OCP2; 1
25 1$i* 1
26 1$i* RELATION CORDC, CORDR, CORDS, CORD2C, CORD2R; 1
27 1$i* 1
28 1$i* RELATION CORD2S, GPWGRID, OCP, GRADIENT; 1
29 1$i* 1
30 1$i* 1
31 1$i* 1
32 1$i* DECLARATIONS FOR MODULE MKUSET 1
33 1$i* 1
34 1$i* 1
35 1$i* 1
36 1$i* USTRUCT USET(IOOO), GSPT(IOOO); 1
37 1$i* 1
38 1$i* RELATION SPEC, SPEC, SPEC, MPC, MPC; 1
39 1$i* 1
40 1$i* ASET, ASET, OMT, OMT, SUCR; 1
41 1$i* 1
42 1$i* RELATION JSET, JSET, JSET; 1
43 1$i* 1
44 1$i* MATRIX FOM(IOOO), FTOF(IOOO), FTOF(IOOO), FTOF(IOOO), FTOF(IOOO); 1
45 1$i* 1
46 1$i* YS(IOOO); 1
47 1$i* 1
48 1$i* RELATION FDMFS(IOOO), FDMFS(IOOO), FDMFS(IOOO), FDMFS(IOOO), FDMFS(IOOO); 1
49 1$i* 1
50 1$i* 1
51 1$i* RELATION TREF, TREF, TREF; 1
52 1$i* 1
53 1$i* RELATION KSEM, KSEM, KSEM, KSEM, KSEM; 1
54 1$i* 1
55 1$i* RELATION CHER, CHER, CHER, CHER, CHER; 1
56 1$i* 1
57 1$i* RELATION CMES2, CMES2, CMES2, CMES2, CMES2; 1
58 1$i* 1
59 1$i* RELATION CMES2, CMES2, CMES2, CMES2, CMES2; 1
60 1$i* 1
61 1$i* RELATION DCONVM, DCONVM, DCONVM, DCONVM, DCONVM; 1
```

51
<table>
<thead>
<tr>
<th>Line</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>1: DCONTM, DCONENM, DCONTM, DCONVM, DCONTM, DCONMT,</td>
</tr>
<tr>
<td>66</td>
<td>1: DCONEP, DCONET, DCONAE, DCONCA, DCONET,</td>
</tr>
<tr>
<td>67</td>
<td>1: DCONEN, DCONEF, DCONAF, DCONCEF, DCONAF,</td>
</tr>
<tr>
<td>68</td>
<td>1: RELATION DCONDFS, DCONFQ, DCONTHK, DCONHT2;</td>
</tr>
<tr>
<td>69</td>
<td>1: RELATION DCONPMN, DCONNMN, DCONNL;</td>
</tr>
<tr>
<td>70</td>
<td>1: RELATION DCONK, DCONKE;</td>
</tr>
<tr>
<td>71</td>
<td>1: RELATION GLODES, GLOE, TFIXED, LOCVAR, DVC;</td>
</tr>
<tr>
<td>72</td>
<td>1: MATRIX [TRANS];</td>
</tr>
<tr>
<td>73</td>
<td>1: MATRIX [PMINT], [PMAXT], [SMAT];</td>
</tr>
<tr>
<td>74</td>
<td>1: $</td>
</tr>
<tr>
<td>75</td>
<td>1: DECLARATIONS FOR OUTPUT FILE PROCESSING (EDR/OFF)</td>
</tr>
<tr>
<td>76</td>
<td>1: $</td>
</tr>
<tr>
<td>77</td>
<td>1: $</td>
</tr>
<tr>
<td>78</td>
<td>1: $</td>
</tr>
<tr>
<td>79</td>
<td>1: RELATION GRIDLIST, MODELST, ELENST, FREQLIST, TIMELIST,</td>
</tr>
<tr>
<td>80</td>
<td>1: ITERLIST, GDVLST, LDVLST, DCONLIST, PPLYLIST;</td>
</tr>
<tr>
<td>81</td>
<td>1: $</td>
</tr>
<tr>
<td>82</td>
<td>1: RELATION GFELEM, EOBMLN, EOBAR, EOBAS, EBMA, EOBEX1,</td>
</tr>
<tr>
<td>83</td>
<td>1: EOBEX2, EOBEX3, EOBEX4, EOBEX5, EOBEX6, EOBEX7,</td>
</tr>
<tr>
<td>84</td>
<td>1: EOBHEAR, EOBTR, GBDATA, EOBTRI;</td>
</tr>
<tr>
<td>85</td>
<td>1: UNSTRUCT EODISC;</td>
</tr>
<tr>
<td>86</td>
<td>1: $</td>
</tr>
<tr>
<td>87</td>
<td>1: RELATION OGRODLN, OGRODSF, OLOCALDV, OAGRDSF, OAGRDLN;</td>
</tr>
<tr>
<td>88</td>
<td>1: MATRIX [FLUTMODE], [FLGLOAD], [PFLOAD], [FHPHLOAD];</td>
</tr>
<tr>
<td>89</td>
<td>1: $</td>
</tr>
<tr>
<td>90</td>
<td>1: DECLARATIONS FOR MODULES EMAI, EMAS AND GLOBAL</td>
</tr>
<tr>
<td>91</td>
<td>1: $</td>
</tr>
<tr>
<td>92</td>
<td>1: MATRIX PARTITION/REDUCTION</td>
</tr>
<tr>
<td>93</td>
<td>1: $</td>
</tr>
<tr>
<td>94</td>
<td>1: $</td>
</tr>
<tr>
<td>95</td>
<td>1: UNSTRUCT DVI, DMVI;</td>
</tr>
<tr>
<td>96</td>
<td>1: RELATION GMLCT, GMLCT;</td>
</tr>
<tr>
<td>97</td>
<td>1: MATRIX [KGG], [KNN], [KFF], [KAA], [KLL],</td>
</tr>
<tr>
<td>98</td>
<td>1: [MGG], [MMN], [MFF], [MAA], [MLL],</td>
</tr>
<tr>
<td>99</td>
<td>1: [MRBR], [MLR], [KRS], [KGG], [KODINV(1000)],</td>
</tr>
<tr>
<td>100</td>
<td>1: [GSUBO(1000)], [KLINN(1000)], [MRU(1000)],</td>
</tr>
<tr>
<td>101</td>
<td>1: [IFM(1000)], [MGG], [IFR(1000)], [KRR], [D(1000)],</td>
</tr>
<tr>
<td>102</td>
<td>1: [KRR], [KGG], [INT(1000)], [MEG], [NOO],</td>
</tr>
<tr>
<td>103</td>
<td>1: [KDD], [KDD], [MAABAR];</td>
</tr>
<tr>
<td>104</td>
<td>1: MATRIX [TM1], [TM2];</td>
</tr>
<tr>
<td>105</td>
<td>1: MATRIX [FG], [FN], [PP], [PA],</td>
</tr>
<tr>
<td>106</td>
<td>1: [FG], [FLBAR], [FR], [RHS(1000)], [UG(1000)],</td>
</tr>
<tr>
<td>107</td>
<td>1: [UN], [UF], [UL], [UM],</td>
</tr>
<tr>
<td>108</td>
<td>1: [AG(1000)], [AN], [AF], [AA], [AK],</td>
</tr>
<tr>
<td>109</td>
<td>1: [AL], [UU], [UO], [FS];</td>
</tr>
<tr>
<td>110</td>
<td>1: LOGICAL MGGFLAG, K2GFLAG;</td>
</tr>
<tr>
<td>111</td>
<td>1: $</td>
</tr>
<tr>
<td>112</td>
<td>1: $</td>
</tr>
<tr>
<td>113</td>
<td>1: $</td>
</tr>
<tr>
<td>114</td>
<td>1: DECLARATIONS FOR SOLUTION CONTROL</td>
</tr>
<tr>
<td>115</td>
<td>1: $</td>
</tr>
<tr>
<td>116</td>
<td>1: INTEGER NUMOPTC, NCOND, MAXITER,</td>
</tr>
<tr>
<td>117</td>
<td>1: MFS, MFE,</td>
</tr>
<tr>
<td>118</td>
<td>1: GCS, GCC,</td>
</tr>
<tr>
<td>119</td>
<td>1: GDS, FSD;</td>
</tr>
<tr>
<td>120</td>
<td>1: INTEGER BLOAD, BNAMES, BMODES, BSAERO, BFLUTR,</td>
</tr>
<tr>
<td>121</td>
<td>1: BOXN, BDRSP, BDR, BTR, BDR, BDPR,</td>
</tr>
<tr>
<td>122</td>
<td>1: BMFL, BMOU, BBFL, BBLAST, MNPC, NMPD,</td>
</tr>
<tr>
<td>123</td>
<td>1: NOMIC, NRSET, DMODES;</td>
</tr>
<tr>
<td>124</td>
<td>1: REAL MOVIL, MOVIL, COMOVIL, CHI, CNVRGILIM,</td>
</tr>
<tr>
<td>125</td>
<td>1: EPS, OPTIMIZE, CASE;</td>
</tr>
<tr>
<td>126</td>
<td>1: RELATION JOB, OPTIMIZE, CASE;</td>
</tr>
<tr>
<td>127</td>
<td>1: $</td>
</tr>
<tr>
<td>128</td>
<td>1: $</td>
</tr>
<tr>
<td>129</td>
<td>1: $</td>
</tr>
<tr>
<td>130</td>
<td>1: DECLARATIONS FOR SENSITIVITY EVALUATION</td>
</tr>
<tr>
<td>131</td>
<td>1: $</td>
</tr>
<tr>
<td>132</td>
<td>1: INTEGER DDFL, NACS, NAUS, NAU;</td>
</tr>
<tr>
<td>133</td>
<td>1: LOGICAL ACTBOUND, ACTBOUND, ACTP, ACTP, ACTP, ACTP,</td>
</tr>
<tr>
<td>134</td>
<td>1: ACTU, ACTUAG, ACTUAG, ACTP, ACTP;</td>
</tr>
<tr>
<td>135</td>
<td>1: UNSTRUCT PCAA, PCAA, PCAA;</td>
</tr>
<tr>
<td>136</td>
<td>1: RELATION POLIST;</td>
</tr>
<tr>
<td>137</td>
<td>1: MATRIX (DPFP), [PGAS], [UGA], [DUG], [DMUG],</td>
</tr>
<tr>
<td>138</td>
<td>1: (DPFP), [DPV], [DPV], [DPV], [DPV], [DUP],</td>
</tr>
<tr>
<td>139</td>
<td>1: (DUAD), [DUAD], [AGA], [AMAT], [DGG],</td>
</tr>
<tr>
<td>140</td>
<td>1: (DPFP), [DPFP], [DPFP], [DPFP], [DUP], [DULL];</td>
</tr>
<tr>
<td>141</td>
<td>1: (DAP), [DAP], [DAP], [DAP], [DAP], [PAA],</td>
</tr>
<tr>
<td>142</td>
<td>1: (DAP), [DAP], [DAP], [DAP], [DAP], [DAP],</td>
</tr>
<tr>
<td>143</td>
<td>1: (DAP), [DAP], [DAP], [DAP], [DAP], [DAP],</td>
</tr>
<tr>
<td>144</td>
<td>1: (DAP), [DAP], [DAP], [DAP], [DAP], [DAP],</td>
</tr>
<tr>
<td>145</td>
<td>1: (DAP), [DAP], [DAP], [DAP], [DAP], [DAP],</td>
</tr>
</tbody>
</table>
AERODYNAMIC ENTITIES

**INTEGER**
- SYM, MINDEX, SUB, S;
- REAL
- QDF, MACH;
- LOGICAL
- LOOP, AERFLG(1000), NONPOLY;
- UNSTRUCT
- ACPT, UNK;
- RELATION
- AESURF, AIRFOIL, AEROS, AEFCT, AXSTA,

**INTEGER**
- BODY, SPINEL1, SET1, SET2, ATTACH,
- TRIM, AERO, BLAST, CAEROS, PAEROS,
- GEOMA, AECOMP, STBCF, CAERO1, PAERO1,
- CAERO2, PAERO2, MCAERO1, MCAERO2, FLUTTER,
- FLFACT, CLAMBOA, CONEFTS, CONLINK, GEOMUA,
- AECOMP, SPINEL2, CONEFT, AEROS, CAROS,
- AERGEOM, CAROUGE, ACORO, AGRID, AGRIDE,

**REAL**
- QMODY, SROHSEK, CHORDCF, MACHCF, STACIC,
- AECOMP, GEOMIA, MAAERO, AERO, RESMIN,
- PANEL1, PANEL2, SPINEL2, AEROS, TRIMCF,

**INTEGER**
- LOADMOD;

**MATRIX**
- [AIRFOIL(1000)], [AICAT(1000)], [AICHAT(1000)],
- [AIRFRC(1000)], [AIRFORCE(1000)],
- [AIRFLG(1000)], [AAPOLY(1000)], [AAPOLY(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],
- [AIRGEOM(1000)], [AIRGEOM(1000)],

**INTEGER**
- HEIZE(1000);
BEGIN MAPOL SOLUTION SEQUENCE

PREFACE MODULES

INITIALIZE SUBSCRIPT VALUES TO "1" TO AVOID RUN TIME PROBLEMS

TRY USING A UTILITY TO PRINT OUT THE GRID RELATIONAL ENTITY

GENERATE THE ELEMENT MATRICES

GENERATE THE SIMPLE LOAD VECTORS AND LOAD SENSITIVITIES

GENERATE THE STEADY AIC MATRIX AND THE STEADY SPLINE TRANSFORMATION MATRICES

IF NONPOWLY CALL EXIT;

ASSEMBLE THE ELEMENT MATRICES

TO THE SENSITIVITY MATRICES

PRINT("LOG-('BEGIN PREFACE MODULES')")

CALL SOLUTION ( NUMOPTC, NUMCOND, MPE, OCS, OCL, FSDS, FSDE, MAXITER, MOVLIM, WINDOW, OCMOVLIM, NRFAC, EPS );

CALL IFP ( GSIZEB );

TRY USING A UTILITY TO PRINT OUT THE GRID RELATIONAL ENTITY

PRINT ( "LOG-('ELEMENT MATRIX GENERATION')")

CALL EMG ( NDV, GSIZEB, GLBDES, DESLINK, SMAT, DVCT, DVSIZE, KELM, MELM, TELM, TREF );

CALL PFBULK { GSIZEB, EOSUMMRY, EODISC, GPFELEM );

HANDLE THE NON-PLANAR STEADY AERODYNAMICS ANALYSES

TERMINATE THE EXECUTION IF THE ONLY DISCIPLINE IS NPSAERO

PRINT ( "LOG-('NON-PLANAR STEADY AERODYNAMICS')")

CALL STEADYTPH ( NONPOWLY, AECOMPS, GEOMSA, STABCF, AIRFORCE, AEROGEOM, CAROGEOM, GASHELOD );

IF NONPOWLY CALL EXIT;

ASSUME THE ELEMENT MATRICES

TO THE SENSITIVITY MATRICES

PRINT("LOG-('PHASE 1 ELEM. MATRIX ASSEMBLY')")

CALL EMG1 ( NDV, GLBDES, KELM, MELM, GMKCT, DKVI, GMMCT, DMVI );

CALL LODGEN ( GSIZEB, GLBDES, DVCT, DVSIZE, GMMCT, DMVI, TELM, TREF, SMPLOD, DPTHVI, DPGRVI ));

GENERATE THE SIMPLE LOAD VECTORS

AND LOAD SENSITIVITIES

IF NONPOWLY CALL EXIT;

ASSEMBLE THE ELEMENT MATRICES

TO THE SENSITIVITY MATRICES

PRINT("LOG-('PHASE 1 ELEM. MATRIX ASSEMBLY')")

CALL EMG1 ( NDV, GLBDES, KELM, MELM, GMKCT, DKVI, GMMCT, DMVI );

CALL LODGEN ( GSIZEB, GLBDES, DVCT, DVSIZE, GMKCT, DMVI, TELM, TREF, SMPLOD, DPTHVI, DPGRVI ));

GENERATE THE STEADY AIC MATRIX AND THE STEADY SPLINE TRANSFORMATION MATRICES

IF NONPOWLY CALL EXIT;

ASSEMBLE THE ELEMENT MATRICES

TO THE SENSITIVITY MATRICES

PRINT("LOG-('PHASE 1 ELEM. MATRIX ASSEMBLY')")

CALL EMG1 ( NDV, GLBDES, KELM, MELM, GMKCT, DKVI, GMMCT, DMVI );

CALL LODGEN ( GSIZEB, GLBDES, DVCT, DVSIZE, GMKCT, DMVI, TELM, TREF, SMPLOD, DPTHVI, DPGRVI ));

GENERATE THE STEADY AIC MATRIX AND THE STEADY SPLINE TRANSFORMATION MATRICES

IF NONPOWLY CALL EXIT;
1$PRINT("LOG- ('STEADY AERODYNAMICS ') ");
2$LOOP := TRUE;
3$MINDEX := 0;
4$WHILE LOOP DO
5$MINDEX := MINDEX + 1;
6$CALL STEADY ( MINDEX, LOOP, AECOMPS, GEOMSA, STABCF, [AIICMAT (MINDEX)],
7[AICMAT (MINDEX)], [AIRFRC (MINDEX)], AEROGEOM, CAROGEOM );
8$ENDDO;
9$CALL SPLINES ( GSIZEB, GEOMSA, AECOMPS, AEROS, [GSTG], [GSTG] );
10$GENERATE THE UNSTEADY AIC MATRIX AND THE
11UNSTEADY SPLINE TRANSFORMATION MATRIX
12$PRINT("LOG- ('UNSTEADY AERODYNAMICS') ");
13CALL UNSTEADY ( GEOMUA, AECOMPU, [AJJTL], [D1JK], [D2JK], [SKJ],
14AEROGM, CAROGEO );
15$CALL AMP ( [AJJTL], [D1JK], [D2JK], [SKJ], [QQL], [QUJL], [QQL], [QJJL] );
16$CALL SPLINEU ( GSIZEB, GEOMUA, AECOMPU, AERO, [USTG] );
17$*********************************************************
18$**********
19$BEGIN OPTIMIZATION LOOP
20$*********************************************************
21IF NUMOPTBC > 0 THEN
22$PRINT("LOG- ('************************') ");
23$PRINT("LOG- ('BEGIN OPTIMIZATION') ");
24$BEGIN CONVERGENCE LOOP
25$WHILE NOT GLBCNVRG AND NITER <- MAXITER DO
26$END;
27$END;
28$END;
29$END;
30$END;
31$END;
32$END;
33$END;
34$END;
35$END;
36$END;
37$END;
38$END;
39$END;
40$END;
41$END;
42$END;
43$END;
44$END;
45$END;
46$END;
47$END;
48$END;
49$END;
50$END;
51$END;
52$END;
53$END;
54$END;
55$END;
56$END;
57$END;
58$END;
59$END;
60$END;
61$END;
62$END;
63$END;
64$END;
65$END;
66$END;
67$END;
68$END;
69$END;
70$END;
71$END;
72$END;
73$END;
74$END;
75$END;
76$IF NUMOPTBC > 0 THEN
77$PRINT("LOG- ('************************') ");
78$PRINT("LOG- ('BEGIN OPTIMIZATION') ");
79$BEGIN CONVERGENCE LOOP
80$WHILE NOT GLBCNVRG AND NITER <- MAXITER DO
81$END;
82$END;
83$END;
84$END;
85$END;
86$END;
87$END;
88$END;

55
ASSEMBLE THE GLOBAL MATRICES

NITER := NITER + 1;
PRINT("LOG="("DESIGN ITERATION "",13)",NITER);  
CALL INITINIT ( NITER, CONST );
CALL UMPRG ( [GLB Sig] );
CALL TCEVAL ( NITER, NDV, MOVIM, WINDOW, GLBDES, LOCLVAR, [PMINT],  
[PHACT], TFIXED, CONST );
CALL LAMINON ( NITER, NDV, DCONLAM, DCONLMN, DCONPMN, TFIXED, GLBDES,  
LOCLVAR, [TRANS], CONST );
CALL EM2 ( NITER, NDV, GSIZEB, GLBDES, GMMCT, DKVI, [KIGG],  
GMMCT, DMVI, [M1GG] );

BEGIN BOUNDARY CONDITION LOOP FOR OPTIMIZATION

FOR BC = 1 TO NUMOPTBC DO
PRINT("LOG="("BOUNDARY CONDITION "",13)",BC);  

ESTABLISH THE BASE USET AND PARTITIONING DATA FOR THE BC  
THIS DATA MUST BE RECREATED EACH ITERATION SINCE GDR CAN CHANGE IT  
CALL MKUSET ( BC, GSIEB, [Y(BC)], [TMN(BC)], [POM(BC)], [RNSF(BC)],  
[PSO(BC)], [PARL(BC)], USET(BC) );

MAKE B.C.-DEPENDENT BGPDT FROM BASE, ADDING THE EXTRA POINTS FOR  
THIS B.C.
CALL BCBGPDT( BC , GSIZEB , BGPDT(BC) , ESIZE(BC) );
GSIZE := GSIZEB;
PSIZE := ESIZE(BC) + GSIZE;
PROCESS MATRICES, TRANSFER FUNCTIONS, AND INITIAL CONDITIONS FOR  
THIS B.C.
CALL BCBULK( BC , PSIZE(BC) , BGPDT(BC) , USET(BC) );
CALL BOUND ( BC, GSIZE, ESIZE(BC), USET(BC), BLOAD, BMASS, DMODES,  
BMODES, BSAERO, BFLUTR, BDYN, BDSP, BDTR, BDTR,  
BMFR, BOUND, BBLAST, NMPC, NSPC, NOMIT, NRSET, NGDR );

DETERMINE IF ANY M2GG/K2GG INPUT DATA ARE TO BE ADDED
CALL NULLMAT ( [KGG], [MGG] );
CALL MK2GG( BC, GSIZEB, [M2GG], M2GGFLAG, [K2GG], K2GGFLAG );
MGG := M1GG + M2GG;
ELSE
MGG := [M1GG];
ENDIF;
IF K2GGFLAG THEN
KGG := K1GG + [K2GG];
ELSE
KGG := [K1GG];
ENDIF;

CALL THE GRID POINT WEIGHT GENERATOR FOR THIS BOUNDARY CONDITION
CALL GPWG ( NITER, BC, GPWGGRID, [MGG], OGPWG );
IF BLOAD <> 0 CALL GTLOAD (NITER, BC, GSIZE, BGPDT(BC), GLBDES,  
SMPLOD, [DPTHVI], [DPGRVI], [PG], OGRIDLOD);  

PARTITION-REDUCTION OF GLOBAL MATRICES

IF NUMOPTBC > 1 CALL NULLMAT ( [KNN], [PN], [MNN],  
[GTKN], [GSTKN], [UGTKN] );
IF NMPC <> 0 THEN
PERFORM MPC REDUCTION
PRINT("LOG="("MPC REDUCTION")");
CALL GREDUCE ( [KGG], [PG], [PGMN(BC)], [TMN(BC)], [KNN], [PN] );
CALL GREDUCE ( [KGG], [PMO(BC)], [TMN(BC)], [KNN], [PN] );
CALL GREDUCE ( [KGG], [PMG(BC)], [TMN(BC)], [KNN], [PN] );

IF BSAERO <> 0 THEN
CALL GREDUCE ( [KGG], [PMO(BC)], [TMN(BC)], [KNN], [PN] );
CALL GREDUCE ( [KGG], [PMG(BC)], [TMN(BC)], [KNN], [PN] );
CALL GREDUCE ( [KGG], [PMO(BC)], [TMN(BC)], [KNN], [PN] );

************************************************************** TAKEN OUT FOR EAERO **************************************************************
IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 OR BSAERO <> 0
CALL GREDUCE (, [UGTKG], [PGMN(BC)], [TMN(BC)], , [UGTKN] );
ELSE
NO MPC REDUCTION
[KNN] := [KGG];
IF BLOAD <> 0
[PN] := [PG];
IF BMASS <> 0
[MNN] := [MGG];
NO SPC REDUCTION
[KFF] := [KNN];
IF BLOAD <> 0
[PF] := [P];
IF BMASS <> 0
[PS] := [PS];
ELSE
CALL NREDUCE ( [UGTKN], [PNSF(BC)], , [UGTKF] );
ENDIF;
IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 OR BSAERO <> 0
CALL NREDUCE (, [UGTKG], [PNSF(BC)], , [UGTKF] );
ENDIF;
PERFORM AUTOSPC CALCULATIONS ON THE KNN MATRIX
PRINT("LOG* AUTOSPC COMPUTATIONS");
CALL GPSP ( NITER/ BC, NGDR, [KNN], BGPDT(BC), [YS(BC)],
USET(BC), GPST(BC) );
CALL MKPVECT ( USET(BC), [PGMN(BC)], [PNSF(BC)],
[PSF(BC)], [PARL(BC)];
CALL BOUNDUPD ( BC, GSI2E, ESIZE{BC), USET{BC), NSPC, NMR{BC);
PRINT("LOG\(*\)(' DYNAMIC REDUCTION')");

OBTAIN THE OMITTED DOF PARTITION OF KFF AND MFF

CALL PARTN ( [KFF], [KOO], [KOA], [PFOA(BC)] );

CALL PARTN ( [MFF], [MOO], [MOA], [PFOA(BC)] );

LSIZE := GSIZE - GMPH - NSPC - NOMIT;

CALL GDR1 ( [KOO], [MOO], [KSOO], [GGO], LKSET, LJSET, NEIV,
            FMAX, BC, BGPDT(BC), USET(BC), NOMIT, LSIZE );

IF LKSET <> 0 THEN
    CALL SDCOMP ( [KSOO], [LSOO], USET(BC), SINGOSET );
    CALL GDR2 ( [LSOO], [MOO], [PHIOK], LKSET, LJSET,
                NEIV, FMAX, BC );
ENDIF;

CALL GDR3 ( [KOO], [KOA], [MGG], [PHIOK], [TMN(BC)],
            [GGO], [PGMN(BC)], [PNSF(BC)], [PFOA(BC)],
            [GSUBO(BC)], BGPDT(BC), USET(BC),
            LKSET, LJSET, ASIZE, GNORM, BC );

CALL GDR4 ( BC, GSIZE, PSIZE(BC), LKSET, LJSET, NUMOPTBC,
            NBNDCOND, [PGMN(BC)], [TMN(BC)], [PNSF(BC)],
            [PFOA(BC)], [PARL(BC)], [PGDRG(BC)], [PAJK],
            [PFJK], BGPDT(BC), USET(BC) );

ENDIF;

IF BLOAD <> 0 OR BMODES <> 0 OR BFLUTR <> 0 OR BDYN <> 0 THEN
    REDUCE THE MATRICES WITHOUT AEROELASTIC CORRECTIONS
    PRINT("LOG\(* SYMMETRIC DYNAMIC REDUCTION')")

    [MAA] := TRANS ( [GSUBO(BC)] ) * [MFF] * [GSUBO(BC)];
    [KAA] := TRANS ( [GSUBO(BC)] ) * [KFF] * [GSUBO(BC)];
    [PA] := TRANS ( [GSUBO(BC)] ) * [PF];

    IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 THEN
        [TMPl] := TRANS ( [UGTKF] ) * [GSUBO(BC)];
        CALL TRNSPOSE ( [TMPl], [UGTKA] );
    ELSE
        NO F-SET REDUCTION
    ENDIF;
ELSE
    IF NOMIT <> 0 THEN
        PERFORM THE STATIC REDUCTION
        PRINT("LOG\(* STATIC CONDENSATION')")

        CALL FREDUCE ( [KFF], [PF], [PFOA(BC)], [KOOINV(BC)],
                      [GSUBO(BC)], [KAA], [PA], [PO], USET(BC) );

        IF BMSS <> 0 THEN
            PERFORM GUYAN REDUCTION OF THE MASS MATRIX

            [MAA] := [MAABAR] + TRANS([MOA]) * [GSUBO(BC)] +
                    TRANS([GSUBO(BC)]) * [MOA] +
                    TRANS([GSUBO(BC)]) * [MOO] * [GSUBO(BC)];

            [KAA] := [KFF] + TRANS([MMA]) * [GSUBO(BC)] +
                    TRANS([GSUBO(BC)]) * [MOA] +
                    TRANS([GSUBO(BC)]) * [MOO] * [GSUBO(BC)];

            IF NRSET <> 0 THEN
                [IFM(BC)] := [MOO] * [GSUBO(BC)] + [MOA];
            ELSE
                NO F-SET REDUCTION
            ENDIF;
ENDIF;
ELSE
    ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ELSE
    ENDIF;
ENDIF;
IF BMFSM <> 0 [MAA] := [MFF];
ENDIF;

IF NRSET <> 0 THEN
PERFORM THE SUPPORT SET REDUCTION
PRINT("LOG-1'
  SUPPORT REDUCTION");
ENDIF;

IF NITER = 1 THEN
CALL PARTN ([KAA], [KKR], [KLR], [KLL], [PARL(BC)]);
CALL SDCOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLESET);
CALL RTSCHAR (BC, USET(BC), BGPU(BC), [D(BC)], [KLL],
  [KRR], [KLR]);
ELSE
IF BLOAD <> 0 THEN
  CALL PARTN ([KAA], [KLR], [KLL], [PARL(BC)]);
  CALL SDCOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLESET);
ENDIF;
ENDIF;

IF NITER = 1 THEN
  CALL PARTN ([MAA], [MRBAR], [MLR], [KLL], [PARL(BC)]);
  [FRR(BC)] := [FLL] * [D(BC)] + [MLL] * [D(BC)];
  [MRR(BC)] := [MRBAR] + TRANS ([KLL]) * [D(BC)];
ENDIF;

IF BMODES <> 0 THEN
  PRINT("LOG-1'
  PROCESS STATICS WITH INERTIA RELIEF");
ENDIF;

CALL PARTN ([MAA], [MRBAR], [MLR], [KLL], [PARL(BC)]);

LHS(BC) := TRANS ([D(BC)]);
RHS(BC) := TRANS ([D(BC)]) * [MLR];

IF BMODES <> 0 THEN
  PRINT("LOG-1'
  NO SUPPORT SET REDUCTION");
ENDIF;

IF BLOAD <> 0 THEN
  PRINT("LOG-1'
  ►DISCIPLINE: STATICS.INERTIA RELIEF');
  CALL PARTN ([PA], [PR], [FRBAR], [PARL(BC)]);
  [FRR(BC)] := TRANS ([D(BC)]) * [FRBAR];
  CALL INERTIA ([LHS(BC)], [RHS(BC)], [AR]);
  [AR] := TRANS ([D(BC)]) * [AR];
  CALL ROWMERGE ([AA], [AR], [UA], [PARL(BC)]);
ENDIF;

IF BMODES <> 0 THEN
  PRINT("LOG-1'
  ►DISCIPLINE: NORMAL MODES');
  CALL REIG (NITER, BC, USET(BC), [KAA], [MAA], [MRR(BC)],
  [D(BC)], LAMBDA, [PHIA], [MII], HSIZE(BC));
  CALL OFPROOT (NITER, BC, NUMOPTBC, LAMBDA);
  PRINT ("LOG- (• DISCIPLINE: STATICS (INERTIA RELIEF)')");
ENDIF;

IF BMASS <> 0 [MAA] := [MFF];
ENDIF;

IF BLOAD <> 0 THEN
  PRINT("LOG-1'
  ►DISCIPLINE: STATICS');
  CALL PARTN ([KAA], [KKR], [KLR], [KLL], [PARL(BC)]);
  CALL SDCOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLESET);
ENDIF;

IF BMODES <> 0 THEN
  PRINT("LOG-1'
  ►DISCIPLINE: NORMAL MODES');
  CALL REIG (NITER, BC, USET(BC), [KAA], [MAA], [MRR(BC)],
  [D(BC)], LAMBDA, [PHIA], [MII], HSIZE(BC));
  CALL OFPROOT (NITER, BC, NUMOPTBC, LAMBDA);
  CALL FCEVAL (NITER, BC, LAMBDA, CONST);
ENDIF;

ENDIF;

IF NITER = 1 THEN
  CALL PARTN ([KAA], [KKR], [KLR], [KLL], [PARL(BC)]);
  CALL SDCOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLESET);
ENDIF;

IF BSCAERO <> 0 THEN
  PRINT("LOG-1'
  SAERO INITIALIZATION');
ELSE
  PRINT("LOG-1'
  STARTED FROM COORD SYSTEM');
ENDIF;

CALL TRNSPOSE ([GSKF], [GSKF]);
CALL TRNSPOSE ([UGSKF], [GSKF]);
LOOP := TRUE;
IF NRSET <> 0 THEN

IF SIM = 1 [AICS] := [GKFP] * [TRANS ([ALCMAT(MINDEX)])] * [GSKFP]]; 
ELSE [KATT] := [KFP] - [GMP] * [AIAGC(MINDEX)];
ENDIF;

REDUCE THE MATRICES WITH AEROELASTIC CORRECTIONS;
SAVE THE SUBCASE/BC DEPENDENT DATA FOR SENSITIVITY ANALYSIS;

IF NGRID <> 0 THEN

PERFORM THE GENERAL DYNAMIC REDUCTION

PRINT("LOG= ' SAERO DYNAMIC REDUCTION' ");

[MAAA] := TRANS ([GSSUBO(BC)]) * [MF] * [GSSUBO(BC)]]; 
[KAAD] := TRANS ([GSSUBO(BC)]) * [KAD] * [GSSUBO(BC)]];
[PAA] := TRANS ([GSSUBO(BC)]) * [PAF];

ELSE IF NMIT <> 0 THEN

IF NITER = 1 AND SUB = 1 AND NRSET <> 0 AND BLOAD = 0 AND 
BNODES = 0 AND BFLTR = 0 AND BDIN = 0 THEN

FORM [KAA] ON FIRST PASS SO [D] CAN BE FORMED

CALL REDUCE ([KFP], [FPOA(BC)], [ROGINV(BC)];

ENDIF;

ELSE IF NRSET <> 0 THEN

PERFORM THE SUPPORT SET REDUCTION

IF NITER = 1 AND SUB = 1 AND NRSET <> 0 AND BLOAD = 0 AND 
BNODES = 0 AND BFLTR = 0 AND BDIN = 0 THEN

FORM [KAA] ON FIRST PASS SO [D] CAN BE FORMED

ELSE

ENDIF;

IF NRSET <> 0 THEN

PERFORM THE SUPPORT SET REDUCTION

ENDIF;
PRINT("LOG-";' SAERO SUPPORT REDUCTION";)

IF NITER = 1 AND SUB = 1 AND BLOAD = 0 AND BMODES = 0 AND
BFLUTR = 0 AND BDRM = 0 THEN

[D] WAS NOT COMPUTED FOR NON-SAERO DISCIPLINES SO
NEED TO COMPUTE IT NOW

CALL PARTN [ [KAA], [KKR], [KLR], [KLL], [PARL(BC)]]
CALL SDCOMP [ [KLL], [KLLINV(BC)], USET(BC), SINGLESET ]
CALL FBS [ [KLLINV(BC)], [KLR], [D(BC)]]
CALL RBCHECK [ BC, USET(BC), BGFOT(BC), [D(BC)], [KLL],
[KKR], [KLR]]

ENDIF;

CALCULATE THE REDUCED MASS MATRIX

PROCESS STEADY AEROELASTIC DISCIPLINE

PRINT("LOG-";' DISCIPLINE: STEADY AERO";)

CALL PARTN [ [KAAA], [MRRBAR], [MLR], [KLL], [PARL(BC)]]

[R13(BC, SUB)] := [ML] * [D(BC)] + [MLR];
[R33] := [MRRBAR] + TRANS([MLR]) * [D(BC)] + [MLR];
[R22] := TRANS([D(BC)]) * [MLR] + [MRRBAR];
CALL TRNPOSE [ [R13(BC, SUB)], [R21(BC, SUB)]]

PRINT("LOG-";' >>>DISCIPLINE: STEADY AERO'";)

CALL ROWPART [ [PAA], [PARR], [PAL], [PARL(BC)]]
CALL DECOMP [ [KLL1(BC, SUB)], [R1112(BC, SUB)], [R1113(BC, SUB)],
[R31(BC, SUB)] := TRANS([D(BC)]) * [R12(BC, SUB)] + [KARR];
[R31(BC, SUB)] := TRANS([D(BC)]) * [R11] + [KARR];
[R32(BC, SUB)] := TRANS([D(BC)]) * [R12(BC, SUB)] + [KARR];
[R32(BC, SUB)] := TRANS([D(BC)]) * [R11] + [KARR];

CALL DECOMP [ [R11], [R1112(BC, SUB)], [R1113(BC, SUB)];

CALL ROWMERGE [ [AAA(SUB)], [AARC], [AAPL], [DELC];
CONST, [AAPL], [DELC];

CALL SAERO [ [NITER, BC, MINDEX, SUB, SYM, QDP, STABCF,
BGFOT(BC), [LHSA(BC, SUB)], [RHSA(BC, SUB)], [AAPL],
[DELC];]

CALL ROWMERGE [ [AAA(SUB)], [AAPL], [DELC];]
CALL ROWMERGE (\[UAAC(SUB)\], \[UAR\], \[UAL\], \[PARL(BC)\]);
ENDIF;
END;
ELSE

NO SUPPORT SET REDUCTION

PRINT("LOG=\' '' '' '' '' '' '' '' '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''}

ENDIF;
BEGIN THE DATA RECOVERY OPERATIONS

PRINT("LOG*(' DATA RECOVERY")

IF NGDR <> 0 THEN
  BEGIN THE DATA RECOVERY OPERATIONS
  ENDIF;

BEGIN THE DATA RECOVERY OPERATIONS

PRINT("LOG*(' DATA RECOVERY")

IF NUMOPTBC > 1 CALL NULLMAT ([UF], [AF], [PHIF], [UTRANF], [UFREQF]);

IF NGDR <> 0 THEN
  BEGIN THE DATA RECOVERY OPERATIONS
  ENDIF;

MERGE THE CURRENT SUBCASE DEPENDENT RESULTS INTO A SINGLE
MATRIX OF RESPONSE QUANTITIES FOR FURTHER RECOVERY

ENDIF;

ENDIF;

IF BSAERO <> 0 THEN

IF BLOAD <> 0 THEN
  PRINT("LOG*(' DYNAMIC REDUCTION RECOVERY")

IF NRSET <> 0 THEN

ENDIF;

ENDDO;

FOR S = 1 TO SUB DO

[UFGRD] := [GSUBO(BC)] * [UAA(S)];

CALL ROWPART ([UAA(S)], [UJK], , [PAJK]);

CALL ROWMERGE ([UAA(S)], [UJK], [UFGDR], [PFJK]);

CALL SAEROMRG (BC, S, [UAF], [UAFTMP]);

CALL ROWPART ([UAAC(S)], [UJK], , [PAJK]);

CALL ROWMERGE ([UAAC(S)], [UJK], [AFGDR], [PFJK]);

CALL SAEROMRG (BC, S, [AAF], [AAFTMP]);

ENDIF;

ENDIF;

Else

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:

ENDIF:
DATA RECOVERY WITH STATIC CONDENSATION

PRINT("LOG('"");
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDER
PRINT("LOG=' SPC RECOVERY'");

IF BLOAD <> 0 THEN
  CALL YSMERGE ([UN], [YS(BC)], [UF], [PNSF(BC)]);
  CALL OFPSpcf ( NITER, BC, 1, 1, GSIZE, ESIZE(BC), NGDR, [KFS], [KSS], [UF], [YS(BC)], [PS], [PNSF(BC)], [PGMN(BC)], [PFJK], , , BGFDT(BC), OGRIDLOD);
ENDIF;

IF NRSET <> 0 CALL YSMERGE ([AN], [AF], [PNSF(BC)]);
ENDDO;
ENDIF;

IF BSAER0 <> 0 THEN
  CALL YSMERGE ([UN], [YS(BC)], [UF], [PNSF(BC)]);
ENDIF;

IF MRSET <> 0 CALL YSMERGE ([AN], [AF], [PNSF(BC)]);
ENDIF;

IF BDTR <> 0 OR BMTR <> 0
  CALL YSMERGE ([UTRANN], [YS(BC)], [UTRANF], [PNSF(BC)], BDTR);
ENDIF;

IF BDTR <> 0 OR BMFR <> 0
  CALL YSMERGE ([UFRQMN], [YS(BC)], [UFRQMF], [PNSF(BC)], BMFR);
ENDIF;

IF FLAST <> 0 THEN
  CALL OFPSpcf ( NITER, BC, 8, 1, GSIZE, ESIZE(BC), NGDR, [KFS], , [PFJK], , , BGFDT(BC), OGRIDLOD);
ENDIF;

IF BDTR <> 0 OR BMTR <> 0
  CALL YSMERGE ([UTRANN], [YS(BC)], [UTRANF], [PNSF(BC)], BDTR);
ENDIF;

IF BDTR <> 0 OR BMFR <> 0
  CALL YSMERGE ([UFREQMN], [YS(BC)], [UFREQMF], [PNSF(BC)], BMFR);
ENDIF;

IF NUMOPTBC > 1 CALL NULLMAT ([UG(BC)], [AG(BC)], [UAG(BC)], [AAG(BC)], [PHIG(BC)], , , , BGPDT(BC), OGRIDLOD);
ENDIF;

ELSE
  DATA RECOVERY WITHOUT SPC-REDUCTION
  IF BLOAD <> 0 THEN
    [UN] := [UF];
    IF NRSET <> 0 [AN] := [AF];
  ENDDO;
  ENDIF;

  IF BSAERO <> 0 THEN
    [UN] := [UF];
    IF NRSET <> 0 [AN] := [AF];
    FOR S = 1 TO SUB DO
      IF AEPLO(S) THEN
        [UANC(S)] := [UAFC(S)];
        [AANC(S)] := [AAFC(S)];
        ENDDO;
      ENDIF;
    ENDIF;
  ENDIF;

  IF BMODES <> 0 THEN
    IF BDTR <> 0 OR BMTR <> 0
      [UTRANN] := [UTRANF];
      IF BDTR <> 0 OR BMTR <> 0
        [UTRANF] := [UTRANN];
      ENDIF;
    ENDIF;
  ENDIF;

  IF BMODES <> 0 THEN
    IF BDTR <> 0 OR BMTR <> 0
      [UFREQMN] := [UFREQMF];
      IF BDTR <> 0 OR BMFR <> 0
        [UFREQMF] := [UFREQMN];
      ENDIF;
    ENDIF;
  ENDIF;

  IF NUMOPTBC > 1 CALL NULLMAT ([UG(BC)], [AG(BC)], [UAG(BC)], [AAG(BC)], [PHIG(BC)], , , , BGPDT(BC), OGRIDLOD);
ENDIF;

ELSE
  DATA RECOVERY WITH MPC-REDUCTION
  PRINT("LOG=' MPC RECOVERY'");

  IF BLOAD <> 0 THEN
    [UN] := [TMN(BC)] * [UN];
    CALL RORMEGR ([UG(BC)], [UN], [UN], [PGMN(BC)]);
  ENDIF;

  IF NRSET <> 0 THEN
    [UN] := [TMN(BC)] * [AN];
    CALL RORMEGR ([AG(BC)], [UN], [AN], [PGMN(BC)]);
  ENDIF;

  ENDIF;

  IF BSAERO <> 0 THEN
    [UN] := [TMN(BC)] * [UN];
  ENDIF;
CALL ROWMERGE ([UAG(BC)], [UN], [UM], [PGMN(BC)])

IF NRSET <> 0 THEN
    [UM] := [TMN(BC)] * [AAN];
    CALL ROWMERGE ([AAG(BC)], [UM], [AAN], [PGMN(BC)]);
ENDIF;

ENDIF;

FOR S = 1 TO SUB DO
    IF AEFLG(S) THEN
        [UM] := [TMN(BC)] * [UANC(S)];
        CALL ROWMERGE ([UAGC(BC,S)], [UM], [UANC(S)], [PGMN(BC)])
    ENDIF;
    ENDIF;
    ENDIF;
    ENDIF;

IF BMODES <> 0 THEN
    [UM] := [TMN(BC)] * [PHIN];
    CALL ROWMERGE ([PHIG(BC)], [UM], [PHIN], [PGMN(BC)]);
ENDIF;

ENDIF;

ENDDO;

ENDIF;

ENDIF;

IF BMODES <> 0 THEN
    [UM] := [TMN(BC)] * [UTRANN];
    CALL ROWMERGE ([UTRANG], [UM], [UTRANN], [PGMN(BC)]);
ENDIF;

ENDIF;

ELSE

DATA RECOVERY WITHOUT MPC-REDUCTION

IF BLOAD <> 0 THEN
    [UG(BC)] := [UN];
    IF NRSET <> 0 [AG(BC)] := [AN];
ENDIF;

IF BSAERO <> 0 THEN
    [UM] := [TMN(BC)] * [PHIN];
    CALL ROWMERGE ([PHIG(BC)], [UM], [PHIN], [PGMN(BC)]);
ENDIF;

IF BDFR <> 0 OR BMFR <> 0 THEN
    CALL DCEVAL (NITER, BC, [UG(BC)], CONST, BSAERO);
ENDIF;

ENDDO;

ENDIF;

IF NRSET <> 0 [AAG(BC)] := [AAN];
FOR S = 1 TO SUB DO
    IF AEFLG(S) THEN
        [AAGC(BC,S)] := [AANC(S)];
        CALL ROWMERGE ([AAGC(BC,S)], [UM], [AANC(S)], [PGMN(BC)]);
    ENDIF;
    [UAGC(BC,S)] := [UANC(S)];
    [AAGC(BC,S)] := [AANC(S)];
    [AAGC(BC,S)] := [AANC(S)];
ENDIF;

ENDDO;

RECOVER PHYSICAL BLAST DISCIPLINE DISPLACEMENTS

IF BBLAST <> 0 [UBLASTG] := [PHIG(BC)] * [UBLASTI];

PERFORM CONSTRAINT EVALUATION FOR STATIC DISCIPLINES

PRINT("LOG="
      CONSTRAINT EVALUATION")

IF BLOAD <> 0 THEN
    CALL DCEVAL (NITER, BC, [UG(BC)], CONST);
    CALL SCEVAL (NITER, BC, [UG(BC)], [SNAT], TREF, [GLBSIG], CONST);
ENDIF;

IF BSAERO <> 0 THEN
    CALL DCEVAL (NITER, BC, [UG(BC)], CONST, BSAERO);
    CALL SCEVAL (NITER, BC, [UG(BC)], [SNAT], TREF, [GLBSIG], CONST, BSAERO);
ENDIF;

HANDLE OUTPUT REQUESTS

PRINT("LOG=" OUTPUT PROCESSING")

IF BSAERO <> 0 THEN
    RECOVER STATIC AEROELASTIC LOADS DATA
    LOOP := TRUE;
    SUB := 0;
    WHILE LOOP DO
        SUB := SUB + 1;
        CALL SAERODRV (BC, SUB, MINDEX, SYM, MACH, QDP);
    ENDLOOP;
CALL THE TRIMMED LOADS COMPUTATION WITH PROPER MATRICES
CALL OFPALOAD (NITER, BC, MINDEX, SUB, GSIZE, BGPDT(BC), 1
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)], $!
ENDIF;
1
IF SYM - -1 THEN
CALL OFFAEROM (NITER, BC, MINDEX, SUB, GSIZIE, GEMOEA, 1
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)], $!
ENDIF;
1
CALL TO COMPUTE THE TRIMMED LOADS/DISPLACEMENTS ON THE AERODYNAMIC MODEL $!
CALL OFPSCF (NITER, BC, 5, 1, GSIZE, ESIZE(BC), 1
[PHIG(BC)], [PTGLOAD], [PTHLLOAD], [PPFLLOAD], 1
ENDIF;
1
ENDDO; J
ENDDO; G
CALL OFPALOAD (NITER, BC, MINDEX, SUB, GSIZIE, GEMOEA, 1
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)], $!
IF BDRSF <= 0 THEN
CALL OFFLOAD (NITER, BC, BGPDT(BC), 1
ENDIF;
1
IF SYM - -1 THEN
CALL OFFAEROM (NITER, BC, MINDEX, SUB, GSIZIE, GEMOEA, 1
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)], $!
ENDIF;
1
ENDDO; G
CALL OFPSCF (NITER, BC, 5, 1, GSIZIE, ESIZE(BC), 1
ENDIF;
1
CALL OFPDISP ( NUMOPTBC, BC, NITER, GSIZE, BGPDT(BC), ESIZE(BC), [PTRANS], [PMAXT], [PNSF(BC)], [PFOA(BC)], [PARL(BC)], USET(BC));

CALL OFPLOAD ( NUMOPTBC, BC, NITER, GSIZE, BGPDT(BC), ESIZE(BC), [PTRANS], [PMAXT], [PNSF(BC)], [PFOA(BC)], [PARL(BC)], USET(BC));

FOR BC = 1 TO NUMOPTBC DO
  CALL ABOUND ( NITER, BC, CONST, ACTBOUND, NAUS, NACSD, [PGAS], [PMINT], [PMAXT], [PARL(BC)], USET(BC));
  IF ACTBOUND THEN
    REESTABLISH THE BASE USET AND PARTITIONING DATA FOR THE BC
    NOTE, THIS LEAVES AN INCOMPATIBILITY BETWEEN USET(BC) AND
    BGPDT(BC) SINCE THE LATTER IS NOT REGENERATED.
    THESE INCOMPATIBILITIES WILL NOT AFFECT THE SENSITIVITY ANALYSES:
    AND WILL BE CORRECTED IN THE SUBSEQUENT ANALYSIS
  ENDIF;
  CALL ABOUND ( NITER, BC, CONST, ACTBOUND, NAUS, NACSD, [PGAS], [PMINT], [PMAXT], [PARL(BC)], USET(BC));
ENDFOR;

CALL LAMINSNS ( NITER, NDV, GLBDES, LOCLVAR, [PTRANS], CONST, [AMAT] );
CALL MAKDFV ( NITER, NDV, [PMINT], [PMAXT], CONST, [AMAT] );
CALL LAMINSNS ( NITER, NDV, GLBDES, LOCLVAR, [PTRANS], CONST, [AMAT] );

***********************************************************************
SENSITIVITY EVALUATION FOR BOUNDARY CONDITION DEPENDENT CONSTRAINTS:
***********************************************************************

FOR BC = 1 TO NUMOPTBC DO
  CALL ABOUND ( NITER, BC, CONST, ACTBOUND, NAUS, NACSD, [PGAS], [PMINT], [PMAXT], [PARL(BC)], USET(BC));
  IF ACTBOUND THEN
    REESTABLISH THE BASE USET AND PARTITIONING DATA FOR THE BC
    IF GDR CHANGED IT
      NOTE, THIS LEAVES AN INCOMPATIBILITY BETWEEN USET(BC) AND
      BGPDT(BC) SINCE THE LATTER IS NOT REGENERATED.
      THESE INCOMPATIBILITIES WILL NOT AFFECT THE SENSITIVITY ANALYSES:
      AND WILL BE CORRECTED IN THE SUBSEQUENT ANALYSIS
    ENDIF;
    CALL ABOUND ( NITER, BC, CONST, ACTBOUND, NAUS, NACSD, [PGAS], [PMINT], [PMAXT], [PARL(BC)], USET(BC));
  ENDIF;
ENDFOR;

***********************************************************************
SENSITIVITY EVALUATION FOR BOUNDARY CONDITION DEPENDENT CONSTRAINTS:
***********************************************************************
ENDIF;

EVALUATE FREQUENCY CONSTRAINT SENSITIVITIES

IF ACTDYN THEN
  IF NGDR <> 0 THEN
    CALL ROWPART ([PHIG(BC)], [GMP]; [PGDRG(BC)])
    CALL FREQSNS ([NITER, BC, NDV, GLEDES, CONST, LAMBDA, GMCT, D oper, GMCT, D oper], [GMP], [AMAT]);
  ELSE
    CALL FREQSNS ([NITER, BC, NDV, GLEDES, CONST, LAMBDA, GMCT, D oper, GMCT, D oper], [GMP], [AMAT]);
  ENDIF;
ENDIF;

EVALUATE FLUTTER CONSTRAINT SENSITIVITIES

IF ACTFLUT THEN
  SUB := 0;
  WHILE LOOP DO
    IF NGDR <> 0 CALL ROWPART ([PHIG(BC)], [GMP]; [PGDRG(BC)]);
    WHILE LOOP DO
      SUB := SUB + 1;
      IF NGDR <> 0 THEN
        CALL FLUTSNS ([NITER, BC, SUB, LOOP, GSIZEB, NDV, GLEDES, CONST, GMCT, D oper, GMCT, D oper, GMCT, D oper], [GMP], [AMAT], AEROS);
      ELSE
        CALL FLUTSNS ([NITER, BC, SUB, LOOP, GSIZEB, NDV, GLEDES, CONST, GMCT, D oper, GMCT, D oper, GMCT, D oper], [GMP], [AMAT], AEROS);
      ENDIF;
    ENDDO:
  ENDIF;
ENDIF;

EVALUATE ACTIVE DISPLACEMENT DEPENDENT CONSTRAINTS FROM THE STATICS DISCIPLINE

IF NAUS > 0 THEN
  SENSITIVITIES OF CONSTRAINTS WRT DISPLACEMENTS FOR STATICS
  CALL NULLMAT ([DFDU], [DPGV]);
  IF NACSD > NAUS * NDV THEN
    USE GRADIENT METHOD
    CALL MAKDFU ([NITER, BC, GSIZEB, [SMAT], [GLBSIG]], [CONST, [DFDU]]);
  ELSE
    USE VIRTUAL LOAD METHOD
    CALL MAKDFU ([NITER, BC, GSIZEB, [SMAT], [GLBSIG]], [CONST, [DFSV]]);
  ENDIF;
  ENDIF;

SOME RELATIVELY SIMPLE CALCULATIONS THAT PREcede THE LOOP ON THE DESIGN VARIABLES

IF NGDR <> 0 THEN
  CALL PARTN ([UG(BC)]; [UGA], [PGAS], [PGDRG(BC)])
ELSE
  CALL COLPART ([UG(BC)], [UGA], [PGAS]);
ENDIF;

OBTAIN THE SENSITIVITIES OF THE DESIGN DEPENDENT LOADS

CALL DDLOAD(ndv, gsizeb, bc, smload, ddfgl, [PGAS], [DPV])
CALL MAKDVU ( NITER, NDV, GLBDES, [UGA], [DKUG], GMKCT, DKVI );
CALL NULLMAT ( [DUG] );
IF NRSET <> 0 THEN
  IF NGDR <> 0 THEN
    CALL PARTN ( [AG(BC)], [AGA], [PGAS], [PDRG(BC)] );
  ELSE
    CALL COLPART ( [AG(BC)], [AGA], [PGAS] );
  ENDIF;
ELSE
  [DUG] := [DKUG] + [DMAG];
ELSE
  [DUG] := [DKUG];
ENDIF;
ACCOUNT FOR VIRTUAL LOAD METHOD
IF NACSD > NAUS * NDV THEN
  USE GRADIENT METHOD
  IF DDFLG > 0 THEN
    [DPGV] := [DPVJ] + [DUG];
  ELSE
    [DPGV] := [DUG];
  ENDIF;
ELSE
  USE VIRTUAL LOAD METHOD
  IF DDFLG > 0 THEN
    [DFDU] := [DPVJ] + [DUG];
  ELSE
    [DFDU] := [DUG];
  ENDIF;
ENDIF;
ELSE
  REDUCE THE RIGHT HAND SIDES TO THE L SET
  CALL NULLMAT ( [DPNV], [DMUN] );
  IF NSPC <> 0 THEN
    CALL NREDUCE ( [DPNV], [PNSFS(BC)], [KNS(BC)], [DPFV]);
  ELSE
    [DPFV] := [DPGV];
  ENDIF;
ENDIF;
IF NRSET <> 0 THEN
  CALL ROWPART ( [DPAV], [DPRV], [DPLV], [PARLS(BC)]) ;
  [DRHS] := TRANS ( [GSUBO(BC)] ) * [DURD];
  CALL INERTIA ( [MRR(BC)], [DRHS], [DURD] );
  [DULD] := [D(BC)] * [DURD];
  CALL FBS ( [KLLINV(BC)], [DPLV], [DULV] );
  CALL YSMERGE ( [DUAV], [DULV], [PARLS(BC)] );
ELSE
  CALL FBS ( [KLLINV(BC)], [DPAV], [DUAV] );
ENDIF;
ENDIF;
RECOVER TO THE F SET
CALL NULLMAT ( [DUFV] );
IF NGDR <> 0 THEN
[DUFV] := [GSUBO(BC)] * [DUAV];
ELSE
IF NOMIT <> 0 THEN
IF NRSET <> 0 THEN
[TMP1] := [DPOV] - [DFM(BC)] * [DUAD];
ELSE
[TMP1] := [DPOV];
ENDIF;
CALL FBS ( [KOOINV(BC)], [TMP1], [UOO] );
[UOJ := [GSUBO(BC)] * [DUAV] + [UOO];
CALL ROWMERGE ([DUFV], [UO], [DUAV], [PFOAS(BC)]);
ELSE
[DUFV] := [DUAV];
ENDIF;
ENDIF;
ENDIF;
REDUCE THE LEFT HAND SIDE MATRIX
IF NMPC <> 0 THEN
CALL GREDUCE ([DFDU], [PGMNS(BC)], [TMN(BC)], [DFDUN]);
ELSE
[DFDUN] := [DFDU];
ENDIF;
IF NSPC <> 0 THEN
CALL ROWPART ([DFDUN], [DFDUF], [PNSFS(BC)]);
ELSE
[DFDUF] := [DFDUN];
ENDIF;
ACCOUNT FOR VIRTUAL LOAD METHOD
IF NACSD > NAUS * NDV THEN
USE GRADIENT METHOD
CALL MKAMAT ([AMAT], [DFDUF], [DUFV], PCAS, [PGAS]);
ELSE
USE VIRTUAL LOAD METHOD
CALL MKAMAT ([AMAT], [DUFV], [DFDUF], PCAS, [PGAS]);
ENDIF;
ENDIF; $ END IF ON ACTIVE APPLIED STATIC LOADS $!
EVALUATE ACTIVE CONSTRAINTS FROM THE STATIC AEROELASTICITY DISCIPLINE
IF ACTAERO THEN
LOOP TRUE;
ACTUAG := FALSE;
SUB := 0;
CALL NULLMAT ( [DUFV] );
WHILE LOOP DO
SUB := SUB + 1;
CALL AROSNSDR (NITER, BC, SUB, LOOP, MINDEX, CONST, STM, NGDR,
[PGDRC(BC)], [UAG(BC)], [AGA(BC)],
ACTUAG, [UGA], [AGA], [GAAU],
FCAS, [UAGC(BC,SUB)], [AGGC(BC,SUB)],
ACTAEFF, [UAGC], [AAGC], PCAE ];
IF ACTAEFF THEN
PROCESS PSEUDO DISPLACEMENTS FOR EFFECTIVENESS
CALL MAKDVU ( NITER, NDV, GLBDES, [AUAGC], [DKUG],
GMKCT, DKVI );
IF NRSET <> 0 THEN
CALL MAKDVU ( NITER, NDV, GLBDES, [AAAGC], [DMAG],
GMKCT, DMVI );
[DPGV] := [DKUG] + [DMAG];
CALL MAKDVU ( NITER, NDV, GLBDES, [UAGC], [DKUG],
GMKCT, DMVI );
ENDIF;
ELSE
[DPGV] := [DKUG];
ENDIF;

REDUCE THE RIGHT HAND SIDES TO THE L SET

CALL NULLMAT ([DPNV], [DMUN]) ;
IF NSFC <> 0 THEN
CALL GREduce (,[DPGV],[POMNS(BC)], [TWN(BC)],
[DRNV]);
IF NRSET <> 0 CALL GREduce ( , [DMUG],
[POMNS(BC)], [TWN(BC)], [DMUN]);
ELSE
[DPGV] := [DPGV];
IF NRSET <> 0 [DMUN] := [DMUG];
ENDIF;

CALL NULLMAT ( [DPFV], [DMUF] ) ;
IF NSFC <> 0 THEN
CALL NRREDUCE ([DPNV], [PNSFS(BC)],.....,[DPFV]);
IF NRSET <> 0 CALL NRREDUCE (,[DMUN],[PNSFS(BC)],.....,[DMUF]);
ELSE
[DPFV] := [DPGV];
IF NRSET <> 0 [DMUF] := [DMUG];
ENDIF;

CALL NULLMAT ( [DPAV], [DMUA] ) ;
IF NGDF <> 0 THEN
[DPAV] := TRANS( [GSUBO(BC)] ) * [DPFV];
IF NRSET <> 0 [DMUA] := TRANS( [GSUBO(BC)] )* [DMUF];
ELSE
IF NOMIT <> 0 THEN
CALL FREDUCE ( , [DMUF], [PFOAS(BC)], 1,
[KOOG(BC, SUB)], [KOGG(BC, SUB)],
[KAO(BC, SUB)], [GASUBO(BC, SUB)],
[DPAV], [DKUG]);
IF NRSET <> 0 CALL FREDUCE ( , [DPAV], [PFOAS(BC)], 1,
[KOOG(BC, SUB)], [KOGG(BC, SUB)],
[KAO(BC, SUB)], [GASUBO(BC, SUB)],
[DMUA], [DKUG]);
ELSE
[DPAV] := [DPFV];
IF NRSET <> 0 [DMUA] := [DMUF];
ENDIF;
ENDIF;

IF NRSET <> 0 THEN
CALL ROWPART ( [DPAV], [DPPV], [DPVL], [PARLS(BC)] );
CALL ROWPART ([DMUA], [DMPV], [DMLV], [PARLS(BC)] );
CALL GFS ( [RL11(BC, SUB)], [RL11(BC, SUB)],
[RL11(BC, SUB)], [RL11DPL]);
[DP1] := TRANS([D(BC)]) * [DML] + [DMP] -
[R21(BC, SUB)] * [R11DPL];
[DRHS] := TRANS( [D(BC)] ) * [DPV] + [DPFV] -
[R31(BC, SUB)] * [R11DPL];

PROCESS ACTIVE CONSTRAINTS FOR SAERO DISCIPLINE

CALL GFS ( [KL11(BC, SUB)], [KL11(BC, SUB)],
[DP1], [DKIV]);
[DRHS] := [DRHS] - ([K21(BC, SUB)] * [DKIV]);

CALL DECOMP ( [LHSA(BC, SUB)], [LHSU], [LHSL] );
CALL GFS ( [LHSL], [LHSU], [DRHS], [DU2] );
[DUIR] := [DKIV] + (K1112(BC, SUB)] * [DU2];
[DUIL] := [R11DPL] + (K1112(BC, SUB)] * [DUIR];
[EFFSENS] := - (R31(BC, SUB)] * [DUIL] -
[R32(BC, SUB)] * [DUIR];

CALL AEROFEFF ( NITER, BC, SUB, SYM, NOV, CONST,
PCAL, [EFFSENS], [AMAT]);
ELSE
NOTE THAT SAERO W/O SUPPORT IS NOT SUPPORTED
ENDIF;
ENDIF; $ END IF ON ACTAEFF
IF ACTUAG THEN

SENSITIVITIES OF CONSTRAINTS WRT DISPLACEMENTS
FALSE IF ONLY TRIM PARAMETER CONSTRAINTS ARE ACTIVE

CALL NULLMAT ( [DFDU] );
CALL MAKDFU ( NITER, BC, GSIZEB, [SMAT], [GLBSIG],
CONST, [DFDU], ACTUAGG, SUB );

SOME RELATIVELY SIMPLE CALCULATIONS THAT PRECEDE
THE LOOP ON THE DESIGN VARIABLES

CALL MAKDVU ( NITER, NDV, GLBDES, [UGA], [DRUG],
GMKCT, DKVI );
CALL NULLMAT ( [DPGV] ) ;
IF NR5ET <> 0 THEN
CALL MAKDVU ( NITER, NDV, GLBDES, [AGA], [DMAG],
GMKCT, DMVI );
ELSE
[DPGV] := [DKUG];
ENDIF;

REDUCE THE RIGHT HAND SIDES TO THE L SET

CALL NULLMAT ( [DPNV], [DMUN] );
CALL GREDUCE ( , [DPGV], [PMNS(BC)], [TMN(BC)],
[DPNV]);
IF NRSET <> 0 CALL GREDUCE ( , [DMUG],
[PMNS(BC)], [TMN(BC)], , [DMUN] );
ELSE
[DPGV] := [DPGV];
ENDIF;

CALL NULLMAT ( [DPFV], [DMUF] );
CALL NREDUCE ( , [DPGV], [PNSFS(BC)],,, ,, [DPFV]);
IF NRSET <> 0 CALL NREDUCE ( , [DMUF], [PNSFS(BC)],,, , [DMUF] );
Else
[DPFV] := [DPFV];
ELSE IF NRSET <> 0 [DMUN] := [DMUN];
ENDIF;

CALL NULLMAT ( [DPAV], [DMUA] );
IF NGDR <> 0 THEN
[DPAV] := TRANS ( (GSUBO(BC)) * [DPFV];
IF NRSET <> 0 [DMUA] := TRANS ( [GSUBO(BC)]*DMUF);[DMUA];
ELSE IF NOMIT <> 0 THEN
CALL FREDUCE ( , [DPFV], [PFOAS(BC)], 1,
[KOOL(BC,SUB)], [KOOU(BC,SUB)],
[GAO(BC,SUB)], [GASUBO(BC,SUB)] , ,
[DPAV], [DPFV], );
IF NRSET <> 0 CALL FREDUCE ( , [DMUA], [PFOAS(BC)],,, , [DMUF] );
ELSE
[DPFV] := [DPFV];
ENDIF;

CALL NULLMAT ( [RL11(BC,SUB)], [RU11(BC,SUB)],
[RL11(BC,SUB)], [RU11(BC,SUB)],
[DP1] := TRANS ( [D(BC)] ) * [DMUL] + [DMUR] -
[MP1(BC,SUB)] * [R11DPL];
[DRHS] := TRANS ( [D(BC)] ) * [DPGV] + [DPAV] -
[MP1(BC,SUB)] * [R11DPL];
ENDIF;
PROCESS ACTIVE CONSTRAINTS FOR SAERO DISCIPLINE

CALL GFRS ( [KL11(BC,SUB)], [KU11(BC,SUB)], [DF], [DK1V]);

[DRHS] := [DRHS] - [K21(BC,SUB)] * [DK1V];

CALL AEROSENS ( NITER, BC, MINDEX, SUB, CONST, SIM, NDV,

BOP2T(BC), STABCF, [PGAA], [LHSA(BC,SUB)], [RSA(BC,SUB)],

[DRHS], [AAR], [DDEDLV], [AMAT]);

[DURV] := [K1112(BC,SUB)] * [AAR] +

[PAR(BC,SUB)] * [DDELDV] + [DK1V];

[DULV] := [R1112(BC,SUB)] * [DURV] +

[R1113(BC,SUB)] * [AAR] -

[R11P1(BC,SUB)] * [DDELDV] + [R11DPL];

CALL ROWMERGE ([DUAV],[DURV],[DULV],[PARLS(BC)]);

ELSE

NOTE THAT SAERO W/O SUPPORT IS NOT SUPPORTED

ENDIF;

ENDDO; $ END IF ON ACTIVE AEROELASTIC CONSTRAINTS

CALL NULLMAT { [UAFTMP] ) ;

IF NGDR <> 0 THEN

[UAFTMP] := [GASUBO(BC,SUB)] * [DUAV];

ELSE

IF NOMIT <> 0 THEN

IF NRSET <> 0 THEN

[TMP1] := [DPOV] + [POARO(BC,SUB)] * [DDELDV] ;

ELSE

[TMP1] := [DPOV];

ENDIF;

CALL GFRS ( [KOO1(BC,SUB)], [KOU(BC,SUB)],

[TMP1], [UO]);

[DUAV] := [GASUBO(BC,SUB)] * [DUAV] + [UO];

CALL ROWMERGE ([UAFTMP],[DUAV],[UO],[PARLS(BC)]);

ELSE

[UAFTMP] := [DUAV];

ENDIF;

ENDIF; $ END IF ON ACTUAG

ENDDO; $ END DO ON SUBSCRIPT LOOP

IF ACTUAGG THEN

REDUCE THE LEFT HAND SIDE MATRIX

CALL NULLMAT { [DFDUN] );

IF NMPC <> 0 THEN

CALL GREDUCE ( [DFDUN], [PGMNS(BC)], [TMN(BC)],[DFDUN]);

ELSE

[DFDUN] := [DFDUN];

ENDIF;

CALL NULLMAT ( [DFDU] );

IF NSPC <> 0 THEN

CALL ROWPART ( [DFDUN], [DFDU],[PNSFS(BC) ]);

ELSE

[DFDU] := [DFDUN];

ENDIF;

TAKE MERGED SENSITIVITIES OF DISPLACEMENTS AND

COMPUTE THE AMAT MATRIX TERMS FOR THE SAERO

CONSTRAINTS

CALL MKAMAT ([AMAT], [DFDUF], [DUFV], PCAA, [PGAU], [DUAV] ) ;

ENDIF; $ END IF ON ANY ACTIVE DISPLACEMENTS

ENDIF; $ END IF ON ANY ACTIVE AEROELASTIC CONSTRAINTS

EVALUATE PANEL BUCKLING CONSTRAINT SENSITIVITIES

74
1928 71 IF ACTPNL THEN
1929 81 CALL PBKLSENS ( BC, NITER, NDV, GLBDES, LOCVAR, [TRANS], 
1930 81 PDLIST );
1931 81 ENDIF;
1932 71 IF ACTBAR THEN
1933 81 CALL PBKLSENS ( BC, NITER, NDV, GLBDES, LOCVAR, [TRANS]);
1934 81 ENDIF;
1935 71 ENDDO;
1936 61 $ END IF ON ACTIVE BOUNDARY CONDITION $ 1937 51$ END DO ON ACTIVE BOUNDARY CONDITIONS $ 1938 51$ END IF ON FSD METHOD $ 1939 41$ END IF ON FSD METHOD $ 1940 31$ END WHILE LOOP FOR GLOBAL CONVERGENCE $ 1941 21$ END IF ON OPTIMIZATION $ 1942 11$ END IF ON OPTIMIZATION $ 1943 11$ BEGIN FINAL ANALYSIS LOOP $ 1944 11$ BEGIN FINAL ANALYSIS LOOP $ 1945 11$ IF NBNCOND > NUMOPTBC THEN
1949 21$ PRINT("BOUNDARY CONDITION ',I3)",BC)
1957 21$ FOR BC = NUMOPTBC + 1 TO NBNCOND DO
1958 21$ CALL ANALINIT;
1959 21$ CALL ANALINIT;
1960 21$ CALL ANALINIT;
1961 21$ CALL ANALINIT;
1962 21$ CALL ANALINIT;
1963 21$ CALL ANALINIT;
1964 21$ ASSEMBLY THE GLOBAL MATRICES $ 1965 21$ ASSEMBLY THE GLOBAL MATRICES $ 1966 21$ PRINT("LOG=('BOUNDARY CONDITION ',I3)"; BC);
1967 21$ PRINT("BOUNDARY CONDITION ',I3)"; BC);
1973 21$ PRINT("BEGIN FINAL ANALYSIS")
1974 21$ CALL EMU2 ( BC, GSIZE, GLBDES, GMKCT, DMV1, [KGG],
1975 21$ PRINT("BEGIN FINAL ANALYSIS")
1976 21$ CALL EMU2 ( BC, GSIZE, GLBDES, GMKCT, DMV1, [MGG] );
1977 31$ FOR BC = NUMOPTBC + 1 TO NBNCOND DO
1978 31$ FOR BC = NUMOPTBC + 1 TO NBNCOND DO
1979 31$ ESTABLISH THE BASE USET AND PARTITIONING DATA FOR THE BC
1980 31$ ESTABLISH THE BASE USET AND PARTITIONING DATA FOR THE BC
1981 31$ CALL MKUSET ( BC, GSIZE, [YS(BC)], [TMN(BC)], [PAMS(BC)],
1982 31$ CALL MKUSET ( BC, GSIZE, [YS(BC)], [TMN(BC)], [PAMS(BC)],
1983 31$ [PFA(BC)], [PARL(BC)], USET(BC) );
1984 31$ [PFA(BC)], [PARL(BC)], USET(BC) );
1985 31$ MAKE B.C.-DEPENDENT BGPDT FROM BASE, ADDING THE EXTRA POINTS FOR
1986 31$ MAKE B.C.-DEPENDENT BGPDT FROM BASE, ADDING THE EXTRA POINTS FOR
1987 31$ THIS B.C.
1988 31$ CALL BCGBPDT ( BC, GSIZE, BGPDT(BC) , ESIZE(BC) );
1989 31$ CALL BCGBPDT ( BC, GSIZE, BGPDT(BC) , ESIZE(BC) );
1990 31$ GSIZE := GSIZE;
1991 31$ GSIZE := GSIZE;
1992 31$ PSIZE(BC) := ESIZE(BC) + GSIZE;
1993 31$ PSIZE(BC) := ESIZE(BC) + GSIZE;
1994 31$ PROCESS MATRICES, TRANSFER FUNCTIONS, AND INITIAL CONDITIONS FOR
1995 31$ PROCESS MATRICES, TRANSFER FUNCTIONS, AND INITIAL CONDITIONS FOR
1996 31$ THIS B.C.
1997 31$ CALL BBOUND ( BC, GSIZE, ESIZE(BC), USET(BC), BLOAD, BMASS, DMODES, 
1998 31$ CALL BBOUND ( BC, GSIZE, ESIZE(BC), USET(BC), BLOAD, BMASS, DMODES, 
1999 31$ BMFR, BGST, BBLAST, NMPC, NSKFC, NOMIT, NRSET, NGDR );
2000 31$ BMFR, BGST, BBLAST, NMPC, NSKFC, NOMIT, NRSET, NGDR );
2001 31$ DETERMINE IF ANY M2GG/K2GG INPUT DATA ARE TO BE ADDED
2002 31$ DETERMINE IF ANY M2GG/K2GG INPUT DATA ARE TO BE ADDED
2003 31$ CALL NULLMAT ( [KGG], [MGG] );
2004 31$ CALL NULLMAT ( [KGG], [MGG] );
2005 31$ CALL MK2GG ( BC, GSIZE, [MGG], [M2GG], [K2GG], K2GGFLAG );
2006 31$ CALL MK2GG ( BC, GSIZE, [MGG], [M2GG], [K2GG], K2GGFLAG );
2007 31$ IF M2GGFLAG THEN
2008 31$ IF M2GGFLAG THEN
2009 41$ [MGG] := [MGG] + [M2GG];
2010 41$ [MGG] := [MGG] + [M2GG];
2011 41$ ELSE
2012 41$ ELSE
2013 41$ [MGG] := [MGG];
2014 41$ [MGG] := [MGG];
2015 41$ ENDIF;
2016 41$ ENDIF;
IF K20GFLAG THEN
  [KGG] := [K1GG] + [K2GG];
ELSE
  [KGG] := [K1GG];
ENDIF;

CALL THE GRID POINT WEIGHT GENERATOR FOR THIS BOUNDARY CONDITION

CALL GPWG ( , BC, GPGMGRID, [MGG], OFWG );

IF BLOAD <> 0 CALL GTLOAD ( , BC, GSIZE, BGPOD(BC), GLOBES, SMLOD, [DPTHVT], [DPSVT], [PG], OGRLDL);
IF BMASS <> 0 THEN

PERFORM GUYAN REDUCTION OF THE MASS MATRIX

CALL PARTN ( [MPF], [MOO], [MDA], [MAABAR], [PFOA(BC)] );
[MAA] := [MAABAR] + TRANS([MDA]) * [GSUBO(BC)] +
TRANS([GSUBO(BC)]) * [MDA] +
TRANS([GSUBO(BC)]) * [MOO] * [GSUBO(BC)];
IF NRSET <> 0 THEN [IPI(BC)] := [MOO] * [GSUBO(BC)] + [MDA];
ENDIF;
IF BFULTHR <> 0 OR BGUST <> 0 OR BBLAST <> 0 THEN
CALL ROWPART ( [UGTF], [UGTKO], [UGTKAB], [PFOA(BC)] );
[THP1] := TRANS([UGTKO]) * [GSUBO(BC)];
CALL TRANSPOSE ([TMP1], [TMP2]);
[UGTKA] := [UGTKAB] + [TMP2];
ENDIF;
ELSE
NO F-SET REDUCTION
ENDIF;

IF BMASS <> 0 THEN

CALL PARTN ([KAA], [KRR], [KLR], [KLL], [PARL(BC)] );
CALL SDCOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLSET);
CALL PARTN ([MAA], [MRRBAR], [MLR], [KLL], [PARL(BC)];
[IFR(BC)] := [MLR] * [D(BC)] + [MLR];
[MRBAR(BC)] := [MRBARBAR] + TRANS([MLR]) * [D(BC)] +
TRANS ([D(BC)]) * [IFR(BC)];
[R22] := TRANS ([D(BC)]) * [MLR] + [MRBARBAR];
ENDIF;

IF NRSET <> 0 THEN

PERFORM THE SUPPORT SET REDUCTION

PRINT('LOG=' SUPPORT REDUCTION');
CALL PARTN ([KAA], [KRR], [KLR], [KLL], [PARL(BC)] );
CALL SDCOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLSET);
CALL PARTN ([MAA], [MRRBAR], [MLR], [KLL], [PARL(BC)];
CALL ROWPART ( [PA], [PR], [PLBAR], [PARL(BC)] );
CALL ROWMERGE ( [AA], [AR], [AL], [PARL(BC)] );
CALL YSMERGE ([UA], , [UL], [PARL(BC)] );
CALL REIG ( , BC; DBC, USET(BC), [KAA], [MAA], [MRR(BC)] ,
CALL OFPMROOT ( , BC; NUMOPTBC, LAMBDA );
[LHS(BC)] := [MRR(BC)];
[LHS(BC)] := [MRR(BC)] + TRANS([D(BC)]) * [PR];
[CALON(BC)] := [D(BC)] + [PR];
[CALON(BC)] := [D(BC)] * [AR];
CALL ROWMERGE ( [AA], [AR], [AL], [PARL(BC)] );
[CALON(BC)] := [PLBAR] - [IFR(BC)] * [AR];
CALL FBS ( [KLLINV(BC)], [RHS(BC)], [UG]);
CALL YSMERGE ( [UA], , [UL], [PARL(BC)];
ENDIF;
IF BMODES <> 0 THEN

NO SUPPORT SET REDUCTION
ENDIF;

IF BLOAD <> 0 THEN

PRINT('LOG=' NORMAL MODES');
CALL REIG ( , BC, USET(BC), [KAA], [MAA], [MRR(BC)] ,
[D(BC)], LAMBDA, [PHIA], [Mil], HSIZE(BC) );
CALL OFPMROOT ( , BC, NUMOPTBC, LAMBDA );
ENDIF;
ELSE
NO SUPPORT SET REDUCTION
ENDIF;

IF BLOAD <> 0 THEN

PRINT('LOG=' NORMAL MODES');
CALL REIG ( , BC, USET(BC), [KAA], [MAA], [MRR(BC)] ,
[D(BC)], LAMBDA, [PHIA], [Mil], HSIZE(BC) );

78
CALL OFFMROOT ( , BC, NUMOPTBC, LAMBDA );

ENDIF;

ENDIF;

IF BSAERO <> 0 THEN

PERFORM STATIC AEROELASTIC ANALYSES

ENDIF;

ENDIF;

IF BSAERO <> 0 THEN

PERFORM STATIC AEROELASTIC ANALYSES

ENDIF;

PRINT("LOG<- ' SAERO INITIALIZATION' ");

******************************************************************************* TAKEN OUT FOR ZAERO *******************************************************

CALL TRANSPOSE ( [GSTKF], [GSKF] );

******************************************************************************* TAKEN OUT FOR ZAERO *******************************************************

CALL TRANSPOSE ( [GSTKF], [GSKF] );

******************************************************************************* TAKEN OUT FOR ZAERO *******************************************************

CALL TRANSPOSE ( [UTKF], [GSKF] );

LOOP TRUE;

SUB : - 0 ;

WHILE LOOP DO

SUB SUB + 1;

CALL SAERODRV (BC, SUB, LOOP, MINDEX, SYM, MACH, QDP, 1 )

ADJUST THE KFF MATRIX AND DETERMINE THE RIGID AIR LOADS

******************************************************************************* TAKEN OUT FOR ZAERO *******************************************************

CALL TRANSPOSE ( [GSKF], [MSUBO(BC)] );

******************************************************************************* TAKEN OUT FOR ZAERO *******************************************************


[PAF] := [QDP ] * [GSKF ] * [AIRFRC (MINDEX) ] ;

******************************************************************************* TAKEN OUT FOR ZAERO *******************************************************


[PAF] := [QDP ] * [UGTKF] * [AIRFRC (MINDEX) ] ;

******************************************************************************* TAKEN OUT FOR ZAERO *******************************************************

[KAFF] := [KFF] * [QDP ] [AICS] ;

REDUCE THE MATRICES WITH AEROELASTIC CORRECTIONS

SAVE THE SUBCASE/BC DEPENDENT DATA FOR SENSITIVITY ANALYSIS

IF NGDR <> 0 THEN

PERFORM THE GENERAL DYNAMIC REDUCTION

ELSE IF NOMIT <> 0 THEN

PERFORM THE STATIC REDUCTION

ELSE

ENDIF;

CALL FREDUCE ( [KAFF], [PAF], [PFOA(BC)], [GASUBO(BC,SUB)], [KAAA], [PAA], [POARO(BC,SUB)], USET(BC)) ;

ELSE IF BMASS <> 0 THEN

PERFORM GUYAN REDUCTION OF THE MASS MATRIX

ENDIF;

CALL FREDUCE ( [KAFF], [PAF], [PFOA(BC)], [MAABAR],

[MAAA] := [MFF] * [MOO] * [GASUBO(BC, SUB)];

[MAOO] := [GASUBO (BC, SUB)] ;

[MAABAR] + TRANS ([MOA] ) + TRANS ([GASUBO (BC, SUB) ]+

[MAAO] + TRANS ([GASUBO (BC, SUB) ] * [MOA] +

[GASUBO (BC, SUB) ] ) ;

IF NRSET <> 0

ENDIF;

ELSE

NO F-SET REDUCTION

ENDIF;
IF NRSET <> 0 AND SUB = 1 AND BLOAD = 0 AND BMODES = 0 AND BFLTR = 0 THEN

FORM [KAA] ON FIRST PASS SO (D) CAN BE FORMED

ENDIF;

[KAA] := [KFF];

ENDIF;

[KAA] := [KFF];

[MAA] := [MFF];

[PAF] := [PAF];

ENDIF;

ENDIF;

IF NRSET <> 0 THEN

PERFORM THE SUPPORT SET REDUCTION

PRINT('"LOG*(' GAERO SUPPORT REDUCTION"');

ENDIF;

IF SUB = 1 AND BLOAD = 0 AND BMODES = 0 AND BFLTR = 0 AND BDYN = 0 THEN

[D] WAS NOT COMPUTED FOR NON-SAERO DISCIPLINES SO NEED TO COMPUTE IT NOW

ENDIF;

CALL PARTN ([KAA], [KKR], [KLR], [KLL], [PARL(BC)]);

CALL ECOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLSET);

CALL FBS ([KLLINV(BC)], [KKR], [D(BC)], -1);

CALL RBCHECK ([BC], USET(BC), BGPDT(BC), [D(BC)], [KLL]);

ENDIF;

CALL DECOMP ([MAAA], [MRRBAR], [MLR], [MLL], [PARL(BC)]);

NEED TO COMPUTE (D) NOW

CALL PARTN ([MAAA], [MRRBAR], [MRL], [PLL], [PARL(BC)]);

CALL DECOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLSET);

CALL SDCOMP ([KLL], [KLLINV(BC)], USET(BC), SINGLSET);

CALL FBS ([KLLINV(BC)], [KLR], (D(BC)), -1);

CALL RBCHECK ([BC], USET(BC), BGPDT(BC), [D(BC)], [KLL]);

ENDIF;

CALL DECOMP ([R11], [R1112(BC, SUB)], [R1113(BC, SUB)]);

CALL ROWPART ([PAF], [PARBAR], [PAL], [PARL(BC)]);

CALL GFBS ([R1112(BC, SUB)], [R1113(BC, SUB)], [PRIGID], [PARL(BC)], -1);

CALL GFBS ([R1113(BC, SUB)], [R1113(BC, SUB)], [R1113(BC, SUB)], -1);

CALL DECOMP ([K11], [K1112(BC, SUB)], [K1113(BC, SUB)]);

CALL GFBS ([K1112(BC, SUB)], [K1113(BC, SUB)], [K1113(BC, SUB)], -1);

CALL GFBS ([K1113(BC, SUB)], [K1113(BC, SUB)], [K1113(BC, SUB)], -1);

CALL DECOMP ([K12], [K1213(BC, SUB)], [K1214(BC, SUB)]);

CALL GFBS ([K1213(BC, SUB)], [K1214(BC, SUB)], [K1214(BC, SUB)], -1);

CALL DECOMP ([K13], [K1312(BC, SUB)], [K1313(BC, SUB)]);

CALL GFBS ([K1312(BC, SUB)], [K1313(BC, SUB)], [K1313(BC, SUB)], -1);

CALL DECOMP ([K14], [K1413(BC, SUB)], [K1414(BC, SUB)]);

CALL GFBS ([K1413(BC, SUB)], [K1414(BC, SUB)], [K1414(BC, SUB)], -1);

CALL DECOMP ([K11], [K1112(BC, SUB)], [K1113(BC, SUB)]);

CALL GFBS ([K1112(BC, SUB)], [K1113(BC, SUB)], [K1113(BC, SUB)], -1);

CALL DECOMP ([K12], [K1213(BC, SUB)], [K1214(BC, SUB)]);

CALL GFBS ([K1213(BC, SUB)], [K1214(BC, SUB)], [K1214(BC, SUB)], -1);

CALL DECOMP ([K13], [K1312(BC, SUB)], [K1313(BC, SUB)]);

CALL GFBS ([K1312(BC, SUB)], [K1313(BC, SUB)], [K1313(BC, SUB)], -1);

CALL DECOMP ([K14], [K1413(BC, SUB)], [K1414(BC, SUB)]);

CALL GFBS ([K1413(BC, SUB)], [K1414(BC, SUB)], [K1414(BC, SUB)], -1);

CALL DECOMP ([K11], [K1112(BC, SUB)], [K1113(BC, SUB)]);

CALL GFBS ([K1112(BC, SUB)], [K1113(BC, SUB)], [K1113(BC, SUB)], -1);

CALL DECOMP ([K12], [K1213(BC, SUB)], [K1214(BC, SUB)]);

CALL GFBS ([K1213(BC, SUB)], [K1214(BC, SUB)], [K1214(BC, SUB)], -1);

CALL DECOMP ([K13], [K1312(BC, SUB)], [K1313(BC, SUB)]);

CALL GFBS ([K1312(BC, SUB)], [K1313(BC, SUB)], [K1313(BC, SUB)], -1);

CALL DECOMP ([K14], [K1413(BC, SUB)], [K1414(BC, SUB)]);

CALL GFBS ([K1413(BC, SUB)], [K1414(BC, SUB)], [K1414(BC, SUB)], -1);
PROCESS STEADY AEROELASTIC DISCIPLINE

PRINT("LOG=" ' >>>DISCIPLINE: STEADY AERO')

ENDIF;

ENDDO:

3:

IF BDYN <> 0 THEN

PRINT("LOG=" ' >>>DISCIPLINE: FLUTTER')

ENDIF;

ENDDO;

3:

IF BFLTR <> 0 THEN

PRINT("LOG=" ' >>>DISCIPLINE: TRANSIENT RESPONSE')

ENDIF;

ENDDO;

3:

IF BMTR <> 0 THEN

PRINT("LOG=" ' >>>DISCIPLINE: FREQUENCY RESPONSE')

ENDIF;

3:

IF BMFR <> 0 THEN

PRINT("LOG=" ' >>>DISCIPLINE: TRANSIENT RESPONSE')

ENDD0;

3:

IF BDYN <> 0 THEN

PRINT("LOG=" ' >>>DISCIPLINE: FLUTTER')

ENDIF;

ENDDO;

3:

IF BFLTR <> 0 THEN

PRINT("LOG=" ' >>>DISCIPLINE: TRANSIENT RESPONSE')

ENDIF;

ENDDO;

3:

END DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT OF THE SUPPORT SET

ENDIF;

3:

END DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT OF THE SUPPORT SET

ENDIF;

3:

END DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT OF THE SUPPORT SET

ENDIF;

3:

END DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT OF THE SUPPORT SET

ENDIF;

3:

END DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT OF THE SUPPORT SET

ENDIF;

3:

END DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT OF THE SUPPORT SET

ENDIF;

3:

SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;

3:

END SUPPORT SET REDUCTION

ENDIF;
BEGIN THE DATA RECOVERY OPERATIONS
IF NGDR <> 0 THEN
      IF NBNDCOND > 1 CALL NULLMAT ( [UF], [AF], [PHIP] );
      IF NODR <> 0 THEN
      DATA RECOVERY WITH GDR
      APPEND THE GDR-GENERATED DOPs TO THE F-SET
      PRINT("LOG="
      IF BLOAD <> 0 THEN
      IF BLOD <> 0 OR BMTR <> 0 THEN
      IF BDTR <> 0 OR BMDR <> 0 THEN
      IF NOMIT <> 0 THEN
      DATA RECOVERY WITH STATIC CONDENSATION

NSPC <> 0 THEN
  DATA RECOVERY WITH SPC-REDUCTION
  PRINT("LOG('')
  IF BSAERO <> 0 THEN
    FOR S = 1 TO SUB DO
      CALL RECOVA ([UAF], [UAF], [UAF]);
      CALL SAEROMRG ( BC, S, [UAF], [UAF]);
    ENDDO;
    IF BMDV <> 0 THEN
      CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
      CALL XSMERGE ( [UAF], [UAF]);
    ENDDO;
  ENDIF;
ENDIF;
IF BMDV <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
  CALL XSMERGE ( [UAF], [UAF]);
ENDIF;
IF BSAERO <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
  CALL XSMERGE ( [UAF], [UAF]);
ENDIF;
IF BMDV <> 0 OR BMTR <> 0 THEN
  CALL RECOVA ([UAF], [UAF], [UAF]);
  ENDIF;
ENDIF;
IF BSAERO <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
  CALL XSMERGE ( [UAF], [UAF]);
ENDIF;
IF BMDV <> 0 OR BMTR <> 0 THEN
  CALL RECOVA ([UAF], [UAF], [UAF]);
ENDIF;
ELSE
  DATA RECOVERY WITHOUT F-SET REDUCTION
ENDIF;
IF BMDV <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
  CALL XSMERGE ( [UAF], [UAF]);
ENDIF;
IF BSAERO <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
  CALL XSMERGE ( [UAF], [UAF]);
ENDIF;
IF BMDV <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
  CALL XSMERGE ( [UAF], [UAF]);
ENDIF;
IF BSAERO <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)];
  CALL XSMERGE ( [UAF], [UAF]);
ENDIF;
IF BMDV <> 0 OR BMTR <> 0 THEN
  CALL RECOVA ([UAF], [UAF], [UAF]);
  ENDIF;
ENDIF;
ENDIF;

2658 41
IF BDFR <> 0 OR BMFR <> 0

2659 51
CALL YSMERGE ( 

2660 [UTRANN], [YS(BC)], [UTRANF], [PNSF(BC)], BDFR );

2661 41
IF BFLUTR <> 0

2662 51
CALL OFPSPCF { 0, BC, 4, 2, GSIZE, ESIZE(BC), NGDR, [KFS], [PHIF], [PNSF(BC)], [PGMN(BC)], [PFJK], [BGPDT(BC)], OGRIDLOD };

2663 41
IF BBLAST <> 0 THEN

2664 51
UBLASTF := [PHIF]*[UBLASTI];

2665 41
CALL OFPSPCF ( 0, BC, 8, 1, GSIZE, ESIZE(BC), NGDR, [KFS], [UBLASTF], [PNSF(BC)], [PGMN(BC)], [PFJK], [BGPDT(BC)], OGRIDLOD );

2666 41
ELSE

2667 41$1
DATA RECOVERY WITHOUT SPC-REDUCTION

2668 41$1
IF BLOAD <> 0 THEN

2669 51
[UN] := [UF];

2670 41$1
IF NRSET <> 0 THEN

2671 61
[UN] := [AF];

2672 41$1
ENDIF;

2673 51
ENDIF;

2674 41
IF BSAERO <> 0 THEN

2675 51
[UN] := [AF];

2676 41$1
IF NRSET <> 0 THEN

2677 61
[UN] := [AF];

2678 41$1
ENDIF;

2679 51
IF BMODES <> 0 THEN

2680 51
PHIN := [PHIF];

2681 41$1
IF BDTR <> 0 OR BMTR <> 0 THEN

2682 51
UTRANN := [UTRANN];

2683 41$1
ENDIF;

2684 51
ENDIF;

2685 41
IF BDTR <> 0 OR BMTR <> 0 THEN

2686 51
UTRANN := [UTRANN];

2687 41$1
ENDIF;

2688 51
IF BDTR <> 0 OR BMTR <> 0 THEN

2689 51
UTRANN := [UTRANN];

2690 41$1
ENDIF;

2691 31
IF NBNDCOND > 1 CALL NULLMAT ( [UG(BC)], [AG(BC)], [UAG(BC)], [AAG(BC)], [PHIG(BC)] );

2692 41
IF NMPC <> 0 THEN

2693 31
PRINT("LOG=('MPC RECOVERY')");

2694 41$1
IF BLOAD <> 0 THEN

2695 51
[UN] := [TMN(BC)];

2696 41$1
IF NRSET <> 0 THEN

2697 61
[UN] := [TMN(BC)];

2698 41$1
ENDIF;

2699 51
IF BSAERO <> 0 THEN

2700 51
[UN] := [TMN(BC)];

2701 41$1
IF NRSET <> 0 THEN

2702 61
[UN] := [TMN(BC)];

2703 41$1
ENDIF;

2704 51
ENDIF;

2705 41
IF BMODES <> 0 THEN

2706 51
[PHIN] := [PHIF];

2707 41
IF BDTR <> 0 OR BMTR <> 0 THEN

2708 51
[UTRANN] := [UTRANN];

2709 41
IF NRSET <> 0 THEN

2710 61
[UTRANN] := [UTRANN];

2711 41
ENDIF;

2712 51
ENDIF;

2713 41
ELSE

2714 41$1
DATA RECOVERY WITHOUT MPC-REDUCTION

2715 41$1
IF BLOAD <> 0 THEN

2716 51
[UG(BC)] := [UN];

2717 41$1
IF NRSET <> 0 THEN

2718 51
[UG(BC)] := [UN];

2719 41$1
ENDIF;

2720 51
IF BSAERO <> 0 THEN

2721 51
[UG(BC)] := [UN];

2722 41$1
IF NRSET <> 0 THEN

2723 51
[UG(BC)] := [UN];

2724 41$1
ENDIF;

2725 51
ENDIF;

2726 41$1
ELSE
IF BMODES < 0 [PHIG] := [PHIN];
IF BDTR < 0 OR BMTR < 0 [UTRANG] := [UTRANN];
IF BDFR < 0 OR BMFR < 0 [UFREQG] := [UFREQN];
ENDIF;
RECOVER PHYSICAL BLAST DISCIPLINE DISPLACEMENTS
IF BLAST < 0 [UBLASTG] := [PHIG] * [UBLASTI];
HANDLE OUTPUT REQUESTS
PRINT(["LOG\" = ' OUTPUT PROCESSING']);
IF BSAERO <> 0 THEN
RECOVER STATIC AEROElastic LOADS DATA
LOOP := TRUE;
SUB := 0;
WHILE LOOP DO
SUB := SUB + 1;
CALL SAERODRV (BC, SUB, LOOP, MINDEX, SYM, MACH, QDP);
CALL THE TRIMMED LOADS COMPUTATION WITH PROPER MATRICES
ENDIF;
IF SYM = 1 THEN
616************************************************************************** TAKEN OUT FORENAERO **************************************************************************
CALL OFFLOAD ( , BC, MINDEX, SUB, GSIZE, BGPDT(BC),
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)],
[DELTA(SUB)], [AICMAT(MINDEX)],
[UAG(BC)], [MGG], [AAG(BC)], [KFS],
[KSS], [UAF], [YS(BC)], [PSF(BC)],
[PGMN(BC)], [PFJK], UDDR, USET(BC),
OGRIDLOD );
ENDIF;
ELSE
616************************************************************************** TAKEN OUT FORENAERO **************************************************************************
CALL OFFLOAD ( , BC, MINDEX, SUB, GSIZE, BGPDT(BC),
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)],
[DELTA(SUB)], [AICMAT(MINDEX)],
[UAG(BC)], [MGG], [AAG(BC)], [KFS],
[KSS], [UAF], [YS(BC)], [PSF(BC)],
[PGMN(BC)], [PFJK], UDDR, USET(BC),
OGRIDLOD );
ENDIF;
IF SYM = -1 THEN
616************************************************************************** TAKEN OUT FORENAERO **************************************************************************
CALL OFFLOAD ( , BC, MINDEX, SUB, GSIZE, BGPDT(BC),
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)],
[DELTA(SUB)], [AICMAT(MINDEX)],
[UAG(BC)], [MGG], [AAG(BC)], [KFS],
[KSS], [UAF], [YS(BC)], [PSF(BC)],
[PGMN(BC)], [PFJK], UDDR, USET(BC),
OGRIDLOD );
ENDIF;
CALL TO COMPUTE THE TRIMMED LOADS/DISPLACEMENTS ON THE
AERODYNAMIC MODEL
616************************************************************************** TAKEN OUT FORENAERO **************************************************************************
CALL OFFAEROM ( NITER, BC, MINDEX, SUB, GSIZE, GEOMA,
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)],
[DELTA(SUB)], [AICMAT(MINDEX)],
[UAG(BC)], OGRIDLOD, OGRDQSF );
CALL OFFAEROM ( NITER, BC, MINDEX, SUB, GSIZE, GEOMA,
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)],
[DELTA(SUB)], [AICMAT(MINDEX)],
[UAG(BC)], OGRIDLOD, OGRDQSF );
ELSE
616************************************************************************** TAKEN OUT FORENAERO **************************************************************************
CALL OFFAEROM ( NITER, BC, MINDEX, SUB, GSIZE, GEOMA,
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)],
[DELTA(SUB)], [AICMAT(MINDEX)],
[UAG(BC)], OGRIDLOD, OGRDQSF );
ENDIF;
ELSE
616************************************************************************** TAKEN OUT FORENAERO **************************************************************************
CALL OFFAEROM ( NITER, BC, MINDEX, SUB, GSIZE, GEOMA,
[GTKG], [GSTKG], QDP, [AIRFRC(MINDEX)],
[DELTA(SUB)], [AICMAT(MINDEX)],
[UAG(BC)], OGRIDLOD, OGRDQSF );
ENDIF;
CALL OFPROM ( NITER, BC, MINDEX, SUB, GSIZE, GEOMSA, 
[UGTKGJ, UTRANF], [PNSF(BC)], [PGMN(BC)], [PFJK], [PHIG(BC)], 
[PGLOAD], [PFLOAD], [PFOGLOAD], [PGLOAD], [PGRIDLOD] );

CALL OFPSPCF ( 0, BC, 5, 1, GSIZE, ESIZE(BC), 
[UGTKGJ, UTRANF], [PNSF(BC)], [PGMN(BC)], [PFJK], [PHIG(BC)], 
[PGLOAD], [PFLOAD], [PFOGLOAD], [PGGRIDLOD] );

CALL OFPSPCF ( 0, BC, 6, 2, GSIZE, ESIZE(BC), 
[UGTKGJ, UTRANF], [PNSF(BC)], [PGMN(BC)], [PFJK], [PHIG(BC)], 
[PGLOAD], [PFLOAD], [PFOGLOAD], [PGGRIDLOD] );

CALL OFPLOAD ( NUMOPTBC, BC, GSIZE, BGPDT(BC), PSIZE(BC), 
[UG(BC)], [AG(BC)], [UG(BC)], [AG(BC)], [UG(BC)], [AG(BC)], 
[UBLASTG], [UWRANG], [UWRANE], [UFREQG], [UFREQE], 
[PHIG(BC)];

CALL EDR ( NUMOPTBC, BC, NDV, GSIZE, EOSUMMAR, EODISC, 
GLBDES, LOCLVAR, [PTRANS], 
[UG(BC)], [UAG(BC)], [UWRANG], [UFREQG], [PHIG(BC)];

CALL OFPEDR ( BC, HSIZE(BC) );

ENDIF;

2857 2:ENDIF;

2858 1:END;

2859 31:ENDIF;

2860 31:ENDIF;

2861 4:ENDIF;

2862 4:ENDIF;

2863 4:IF BDRSP <> 0 THEN

2864 31:CALL OFFLOAD ( , BC, BGPDT(BC), PSIZE(BC), ESIZE(BC), 
[PHIG(BC)], 
[PGLOAD], [PFLOAD], [PFOGLOAD], [PGGRIDLOD] );

2865 31:IF BDFR <> 0 OR BMFR <> 0

2866 31:CALL OFFLOAD ( NUMOPTBC, BC, GSIZE, BGPDT(BC), PSIZE(BC), 
[UG(BC)];

2867 31:CALL OFFDISP ( NUMOPTBC, BC, GSIZE, BGPDT(BC), ESIZE(BC), PSIZE(BC), 
[UG(BC)], [AG(BC)], [UG(BC)], [AG(BC)], 
[UBLASTG], [UWRANG], [UWRANE], [UFREQG], [UFREQE], 
[PHIG(BC)];

2868 31:CALL EDR ( NUMOPTBC, BC, NDV, GSIZE, EOSUMMAR, EODISC, 
GLBDES, LOCLVAR, [PTRANS], 
[UG(BC)], [UAG(BC)], [UWRANG], [UFREQG], [PHIG(BC)];

2869 31:CALL OFPEDR ( BC, HSIZE(BC) );

2870 31:ENDIF;

2871 2:ENDIF;

2872 1:END;
APPENDIX C

ZAERO BULK DATA TEMPLATE DEFINITIONS

(TEMPLATE.DAT)
The following lists the twenty three (23) new bulk data templates in file (TEMPLATE.DAT) used to define the ZAERO bulk data cards:

<table>
<thead>
<tr>
<th>Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACOORD</td>
<td>ID</td>
</tr>
<tr>
<td>+COORD</td>
<td>ZMCNT</td>
</tr>
<tr>
<td>AEROZ</td>
<td>ACSID</td>
</tr>
<tr>
<td>AESURFZ1</td>
<td>LABEL</td>
</tr>
<tr>
<td>ATTACH</td>
<td>EID, MODEL</td>
</tr>
<tr>
<td>BODY7</td>
<td>IDBODY, LABEL, IPBODY, ACOORD, NSEG, ID(1), ID(2), ID(3), CONT</td>
</tr>
</tbody>
</table>

**ACOORD**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Default</th>
<th>Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XORDG</td>
<td>REAL</td>
<td>0.0</td>
<td>GT 0</td>
</tr>
<tr>
<td>YORDG</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>ZORDG</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>DELTA</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>THETA</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>XMNT</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>YMNT</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>CONT</td>
<td>CHAR</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

**+COORD**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Default</th>
<th>Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XORDG</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>YORDG</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>ZORDG</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>DELTA</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>THETA</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>XMNT</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>YMNT</td>
<td>REAL</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>CONT</td>
<td>CHAR</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

**AEROZ**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Default</th>
<th>Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACSID</td>
<td>CHAR</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>XZSYM</td>
<td>CHAR</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>RHOREF</td>
<td>CHAR</td>
<td>0.0</td>
<td>GE 0</td>
</tr>
<tr>
<td>RFC</td>
<td>REAL</td>
<td>1.0</td>
<td>GE 0</td>
</tr>
<tr>
<td>RFEB</td>
<td>REAL</td>
<td>0.0</td>
<td>GE 0</td>
</tr>
<tr>
<td>IREFS</td>
<td>REAL</td>
<td>0.0</td>
<td>GE 0</td>
</tr>
<tr>
<td>GREF</td>
<td>INT</td>
<td>0</td>
<td>GE 0</td>
</tr>
</tbody>
</table>

**AESURFZ1**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Default</th>
<th>Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>CHAR</td>
<td>NULL</td>
<td>GE 0</td>
</tr>
<tr>
<td>CID</td>
<td>CHAR</td>
<td>NULL</td>
<td>GE 0</td>
</tr>
<tr>
<td>SETK</td>
<td>INT</td>
<td>NULL</td>
<td>GT 0</td>
</tr>
<tr>
<td>SETG</td>
<td>INT</td>
<td>NULL</td>
<td>GT 0</td>
</tr>
</tbody>
</table>

**ATTACH**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Default</th>
<th>Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>EID</td>
<td>INT</td>
<td>1</td>
<td>GT 0</td>
</tr>
<tr>
<td>MODEL</td>
<td>CHAR</td>
<td>NULL</td>
<td>FLEX</td>
</tr>
<tr>
<td>SETK</td>
<td>INT</td>
<td>NULL</td>
<td>GT 0</td>
</tr>
<tr>
<td>FEEDBK</td>
<td>INT</td>
<td>NULL</td>
<td>GT 0</td>
</tr>
</tbody>
</table>

**BODY7**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Default</th>
<th>Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>LABEL</td>
<td>CHAR</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>IPBODY</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ACOORD</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>NSEG</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ID(1)</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ID(2)</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ID(3)</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>CONT</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

**+BODY7**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Default</th>
<th>Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ACOORD</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>NSEG</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ID(1)</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ID(2)</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ID(3)</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>CONT</td>
<td>INT</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

88
### CAER07 EID LABEL ACOORD NSPAN NCHORD LS PAN ZTAIC PAFOIL ICONT

<table>
<thead>
<tr>
<th>CHAR</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>GT 0</th>
<th>GE 0</th>
<th>GE 2</th>
<th>GE 2</th>
<th>GE 0</th>
<th>GE 0</th>
<th>GE 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

### CAER07 EID LABEL ACOORD NSPAN NCHORD LS PAN ZTAIC PAFOIL ICONT

<table>
<thead>
<tr>
<th>CHAR</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>GE 0.</th>
<th>GE 0.</th>
<th>GE 0.</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
</tbody>
</table>

### +CAER07 XTL YTL ZTL TCH LTCHD IATT |

<table>
<thead>
<tr>
<th>CHAR</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>GE 0.</th>
<th>GE 0.</th>
<th>GE 0.</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>17</td>
<td>18</td>
</tr>
</tbody>
</table>

### CHORDCP ID IX CPU CPL IX CPU CPL ICONT

<table>
<thead>
<tr>
<th>CHAR</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>REAL</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>GE 0.</th>
<th>GE 0.</th>
<th>GE 0.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

### FLUTTER SETID METHOD DENS IDMK VEL MLIST KLIST EFFID ICONT

<table>
<thead>
<tr>
<th>CHAR</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>IB</th>
<th>GT 0.</th>
<th>FLTFIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

### SYMXZ SYMXY EPS CURVFIT PRINT

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>

### GUST SID GLoad WG XO IV QDP IDMK ICONT

<table>
<thead>
<tr>
<th>CHAR</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>IB</th>
<th>GT 0.</th>
<th>SYMXY</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1</td>
<td>-9</td>
</tr>
</tbody>
</table>

### GUST SID GLoad WG XO IV QDP IDMK ICONT

<table>
<thead>
<tr>
<th>CHAR</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>IB</th>
<th>GT 0.</th>
<th>SYMXY</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1</td>
<td>-9</td>
</tr>
</tbody>
</table>

### LOADMOD LID LABEL CP SETK SETG ICONT

<table>
<thead>
<tr>
<th>CHAR</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CHECKS

<table>
<thead>
<tr>
<th>IB</th>
<th>GT 0.</th>
<th>SYMXY</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1</td>
<td>-9</td>
</tr>
<tr>
<td>MACHCP</td>
<td>ID</td>
<td>MACH</td>
</tr>
<tr>
<td>---------</td>
<td>----</td>
<td>------</td>
</tr>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>REAL</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>0.9</td>
<td>0</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GE 0.</td>
</tr>
<tr>
<td>MACHCP</td>
<td>ISPIND</td>
<td>CHIDCP</td>
</tr>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>5</td>
<td>-6</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GE 0.</td>
</tr>
</tbody>
</table>

$\text{MACHCP ID MACH IGRID INDICIA ISPIND CHIDCP ISPIND CHIDCP| CONT |}$

<table>
<thead>
<tr>
<th>MKAEROZIDMK</th>
<th>MACH</th>
<th>METHOD</th>
<th>IDFLT</th>
<th>SAVE</th>
<th>FILE1</th>
<th>FILE2</th>
<th>PRINT</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>REAL</td>
<td>INT</td>
<td>INT</td>
<td>REAL</td>
<td>CHAR</td>
<td>CHAR</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GE 0.0</td>
<td>GE 0</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MKAEROZ</td>
<td>IDMK</td>
<td>MACH</td>
<td>METHOD</td>
<td>IDFLT</td>
<td>SAVE</td>
<td>FILE1</td>
<td>FILE2</td>
<td>PRINT</td>
</tr>
<tr>
<td>+MKAEROZIDMK</td>
<td>FREQ(1)</td>
<td>FREQ(2)</td>
<td>FREQ(3)</td>
<td>FREQ(4)</td>
<td>FREQ(5)</td>
<td>FREQ(6)</td>
<td>FREQ(7)</td>
<td>FREQ(9)</td>
</tr>
<tr>
<td>CHAR</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>CHAR</td>
</tr>
<tr>
<td>DEFAULT 0.</td>
<td>GE 0</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td></td>
</tr>
<tr>
<td>RFREQ</td>
<td>-12</td>
<td>-12</td>
<td>-12</td>
<td>-12</td>
<td>-12</td>
<td>-12</td>
<td>-12</td>
<td>-12</td>
</tr>
</tbody>
</table>

$\text{MKAEROZ IDMK MACH METHOD IDFLT SAVE FILE1 FILE2 PRINT| CONT |}$

<table>
<thead>
<tr>
<th>PAFOIL7</th>
<th>ID</th>
<th>IAFX</th>
<th>ITHR</th>
<th>ICAMR</th>
<th>IRADR</th>
<th>ITHT</th>
<th>ICAMT</th>
<th>RADT</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>REAL</td>
<td>INT</td>
<td>INT</td>
<td>REAL</td>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GE 0.</td>
<td>GE 0.</td>
<td>GE 0.0</td>
<td>GE 0.0</td>
<td>GE 0.0</td>
<td>GE 0.0</td>
<td>GE 0.0</td>
<td>GE 0.0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>-8</td>
<td></td>
</tr>
</tbody>
</table>

$\text{PAFOIL7 ID IAFX ITHR ICAMR IRADR ITHT ICAMT RADT| CONT |}$

<table>
<thead>
<tr>
<th>PANLST1</th>
<th>SETID</th>
<th>MACROID</th>
<th>BOX1</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GT 0</td>
<td>GT 0</td>
<td>GE</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>-4</td>
<td></td>
</tr>
</tbody>
</table>

$\text{PANLST1 SETID MACROID BOX1 BOX2| BOX2 |}$

<table>
<thead>
<tr>
<th>PANLST2</th>
<th>SETID</th>
<th>MACROID</th>
<th>B(1)</th>
<th>B(2)</th>
<th>B(3)</th>
<th>B(4)</th>
<th>B(5)</th>
<th>B(6)</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT/CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GT 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
</tr>
</tbody>
</table>

$\text{PANLST2 SETID MACROID BOX1| BOX2 |}$

<table>
<thead>
<tr>
<th>PANLST3</th>
<th>SETID</th>
<th>MACROID</th>
<th>BOX1</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
<th>BOX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GT 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
</tr>
<tr>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
</tr>
</tbody>
</table>

$\text{PANLST3 SETID MACROID BOX1 BOX2| BOX2 |}$

90
The image contains a page with a table and some code elements. The table appears to be a part of a document, possibly related to programming or data configuration. The code elements seem to be part of a larger context that is not fully visible in the image. The table includes rows and columns with various entries, some of which include numerical values and text descriptions. The elements are not clearly discernible due to the resolution or quality of the image.
<table>
<thead>
<tr>
<th>TRIM</th>
<th>SETID</th>
<th>IDMK</th>
<th>QDP</th>
<th>TRMTYP</th>
<th>EFFID</th>
<th>VO</th>
<th>PRINT</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>INT</td>
<td>INT</td>
<td>REAL</td>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GT 0</td>
<td>GT 0</td>
<td>TRIM</td>
<td>GE 0</td>
<td>GE 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TRIMFLT</th>
<th>IDFLT</th>
<th>TILTA</th>
<th>ALPHA</th>
<th>BETA</th>
<th>PRATE</th>
<th>QRATE</th>
<th>RRATE</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CHAR</td>
<td>INT</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td></td>
<td>CHAR</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ZTAIC</th>
<th>ID</th>
<th>INFLAP</th>
<th>MACHCP1</th>
<th>MACHCP2</th>
<th>MACHCP3</th>
<th>MACHCP4</th>
<th>MACHCP5</th>
<th>MACHCP6</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>CHAR</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GE 0</td>
<td>GT 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ZTAIC</th>
<th>LABEL</th>
<th>HINGE</th>
<th>INBDY</th>
<th>OUTBDY</th>
<th>LABEL</th>
<th>HINGE</th>
<th>INBDY</th>
<th>OUTBDY</th>
<th>ETC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>CHAR</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>CHAR</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GE 1</td>
<td>GE 1</td>
<td>GE 2</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>-12</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>LABEL</td>
<td>HINGE</td>
<td>INBDY</td>
<td>OUTBDY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$</td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX D

ZAERO RELATIONAL SCHEMA DEFINITION

(RELATION.DAT)
The following are the relational SCHEMA definitions (from file RELATION.DAT) for all database relational entities used by the ZAERO module:

```
RELATION ACOORD
  ID INT
  XINTG RSP
  YINTG RSP
  ZINTG RSP
  DLTG RSP
  THETAG RSP
  XMCT RSP
  YMCNT RSP
  ZMCNT RSP
  XEND RSP
  YEND RSP
  ZEND RSP
  XTORQ RSP
  YTORQ RSP
  ZTORQ RSP
  END

RELATION BCOORD
  IDBODI INT
  LABELB STR 8
  IPBODI INT
  ACCORD INT
  NEK INT
  IDMESHA INT
  IDMESHB INT
  IDMESHC INT
  IDMESHD INT
  IDMESHE INT
  IDMESHF INT
  IDMESHG INT
  IDMESHK INT
  END

RELATION BODIES
  MND INT
  THETAB INT
  DELTAB INT
  ID INT
  YBEND INT
  XBEND INT
  YBEND INT
  XBEND INT
  ZBEND INT
  ZMCNT INT
  XCORD INT
  LABELB INT
  TYPE INT
  END

RELATION GEOMEA
  MACROID INT
  AMOUNT STR 8
  NEOF INT
  ENDINT INT
  AREA RSP
  X RSP
  Y RSP
  Z RSP
  N1 RSP
  N2 RSP
  N3 RSP
  R1 RSP
  R2 RSP
  RI RSP
  END

RELATION PANLST1
  SETID INT
  MACROID INT
  BOX1 INT
  BOX2 INT
  END

RELATION PANLST2
  SETID INT
  MACROID INT
  BOX1 INT
  END

RELATION PROD7
  IPBOD7 INT
  WAKE INT
  CPBASE RSP
  XWAKE RSP
  XDRAKE RSP
  YWAKE RSP
  ZWAKE RSP
  INLET INT
  IDP INT
  FLOWRT RSP
  END

RELATION AGRID2
  EXID INT
  INTID INT
  CORD INT
  X RSP
  Y RSP
  Z RSP
  END

RELATION AERO
  ACSID INT
  AXSYM INT 4
  NCHORD INT
  REFSC RSP
  RRF RSP
  RFS RSP
  REFSP RSP
  GREF INT
  END

RELATION AEROUT
  LABEL STR 8
  TYPE STR 8
  CID INT
  SEK INT
  SETG INT
  END

RELATION AQUAD
  MACROID INT
  AMOUNT INT
  NOOF INT
  ENDINT INT
  END

RELATION CHORDCP
  MACROID INT
  LABEL INT
  TYPE STR 8
  X INT
  CPU RSP
  END

RELATION FLOWRT
  MACROID INT
  MACH INT
  IGRID INT
  INDICIA INT
  SPIND INT
  CHORDCP INT
  END

RELATION FLUTTER
  SETID INT
  METHOD STR 4
  DMES INT
  IDMKE INT
  VELE INT
  MELIST INT
  KEIST INT
  EFFID INT
  SYNX INT
  SYNY INT
  EPS RSP
  CURVFINT STR 8
  MACHVAL RSP
  PRINT INT
  END

RELATION AEROUTP
  MACROID INT
  MACH INT
  METHOD INT
  DFTLT INT
  FILE1 STR 8
  FILE2 STR 8
  PRINT INT
  RFPREQ RSP
  END

RELATION PRBOARD
  ID INT
  IAF INT
  ITHR INT
  ICAMB INT
  RDR INT
  IHT INT
  ICAMT INT
  BINT INT
  END

RELATION AEROUTP
  MACROID INT
  MACH INT
  METHOD INT
  DFTLT INT
  FILE1 STR 8
  FILE2 STR 8
  PRINT INT
  RFPREQ RSP
  END

RELATION PANLST3
  SETID INT
  MACROID INT
  BOX1 INT
  END

RELATION SEGME
  MACROID INT
  NAXIS INT
  NRAD INT
  ITYPE INT
  END

RELATION SEGMESH
  MACROID INT
  NAXIS INT
  NRY INT
  IDY INT
  END

RELATION SPLINE3
  EID INT
  XINT INT
  MODEL STR 8
  CP INT
  BOXID INT
  GRDF INT
  FLEX RSP
  EPS RSP
  END

RELATION TRIMFLT
  ID INT
  IAF INT
  ITHR INT
  ICAMB INT
  RDR INT
  IHT INT
  ICAMT INT
  BINT INT
  END
```

94
<table>
<thead>
<tr>
<th>RELATION</th>
<th>ZTAIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>INT</td>
</tr>
<tr>
<td>NFLAP</td>
<td>INT</td>
</tr>
<tr>
<td>MACHCP1</td>
<td>INT</td>
</tr>
<tr>
<td>MACHCP2</td>
<td>INT</td>
</tr>
<tr>
<td>MACHCP3</td>
<td>INT</td>
</tr>
<tr>
<td>MACHCP4</td>
<td>INT</td>
</tr>
<tr>
<td>MACHCP5</td>
<td>INT</td>
</tr>
<tr>
<td>MACHCP6</td>
<td>INT</td>
</tr>
<tr>
<td>LABEL</td>
<td>STR 4</td>
</tr>
<tr>
<td>HINGE</td>
<td>INT</td>
</tr>
<tr>
<td>INBDY</td>
<td>INT</td>
</tr>
<tr>
<td>OUTBDY</td>
<td>INT</td>
</tr>
<tr>
<td>END</td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX E

ZAERØ ERROR MESSAGE DEFINITION

(SERRMSG.DAT)
In following the ASTROS format for error message definitions, three new error message modules (numbers 35 through 37) have been generated for the ZAERO software and added to the SERRMSG.DAT file. These ZAERO error message modules are listed as follows:

**MODULE 35 ZONAS AEROGM MODULE MESSAGES**

'NO $ BULK DATA ENTRIES ARE DEFINED, BUT BODY? BULK DATA EXISTS IN THE INPUT.'

'$ BULK DATA ENTRY WITH BID: $ HAS $ NUMBER OF SEGMENTS, BUT THERE ARE ONLY $ NUMBER OF SEGMENTS. '

'BULK DATA ENTRIES DEFINED.'

'BULK DATA ENTRY $ IS REFERED BY A ID: $ BUT NO $ EXISTS IN THE INPUT.'

'ID NUMBER: $ OF BULK DATA CARD $ IS NOT DEFINED.'

'BULK DATA ENTRY $ WITH ID: $, REFERS TO BULK DATA ENTRY $ WITH ID: $ WHICH DOES NOT EXIST.'

'BULK DATA CARD WITH ID: $ SPECIFIES $ NUMBER OF AXIAL STATIONS, BUT ONLY $ ARE DEFINED.'

'THE X-LOCATIONS OF A $ BULK DATA ENTRY WITH IMESH: $ ARE NOT IN ASCENDING ORDER AT AXIAL STATIONS $ AND $.'

'$ WITH WID: $ HAS $ NUMBER OF SPANWISE DIVISIONS DEFINED, BUT THERE ARE $ NUMBER OF VALUES.'

'LISTED IN THE CORRESPONDING $ BULK DATA ENTRY WITH ID: $.'

'$ WITH ID: $ REFERENCED BY $ WITH WID: $ DOES NOT BEGIN WITH 0.0 OR END AT 100.0.'

'THE SPANWISE DIVISIONS OF A $ BULK DATA CARD, ID: $ REFERENCED BY A $ CARD WITH WID: $, ARE NOT.'

'IN ASCENDING ORDER.'

'THE TOTAL NUMBER OF MACH NUMBERS LISTED IN ALL MACHCP BULK DATA ENTRIES EXCEEDS 6.'

'CAER07 ENTRY WITH WID: $, HAS NO STEADY PRESSURE INPUT ON SPANWISE STRIP INDEX $ AND MACH NUMBER $.'

'THEREFORE LINEAR UNSTEADY PRESSURE WILL BE COMPUTED FOR THIS STRIP.'

'CAER07 ENTRY WITH WID: $, HAS MORE THAN ONE SPANWISE STRIP INDEX DEFINED FOR A MACHCP BULK DATA ENTRY'

'FOR SPANWISE STRIP INDEX $ AND MACH NUMBER $.'

'A SEGMESH BULK DATA CARD WITH IDMESH: $ HAS $ NUMBER OF 5-VALUE CIRCUMFERENTIAL POINTS (NRAD) DEFINED, '

'BUT THERE ARE ONLY $ NUMBER OF VALUES LISTED IN AEFACT WITH ID: $.'

'INCOMPLETE LIST OF LABEL-HINGE-INBDY-OUTBDY PAIRS FOR NFLAP -'

'ERROR IN $ WITH ID: $. ENTRY LABEL - $ IS NOT $ OR $.'

'ERROR IN $ WITH ID: $. ENTRY HINGE = $ IS NOT GREATER THAN 1 AND LESS THAN $ (NCHORD) .'

'ERROR IN $ WITH ID: $. ENTRY INBDY = $ IS NOT GREATER OR EQUAL TO 1 AND LESS THAN $ (NSPAN) .'

'ERROR IN $ WITH ID: $. ENTRY OUTBDY = $ IS GREATER THAN OR EQUAL TO $ (NSPAN) .'

'A $ BULK DATA CARD WITH ID: $ HAS A BOX OF ZERO AREA WITH ID: $.'

'A $ BULK DATA CARD WITH ID: $ SPECIFIES $ (ITAX) NUMBER OF CHORDWISE HALF THICKNESS VALUES ($), '}

'BUT ONLY $ ARE LISTED IN THE CORRESPONDING $ BULK DATA CARD WITH ID: $.'

'A $ BULK DATA CARD WITH ID: $ SPECIFIES $ (ITAX) NUMBER OF CHORDWISE CAMBER VALUES ($), '}

'BUT ONLY $ ARE LISTED IN THE CORRESPONDING $ BULK DATA CARD WITH ID: $.'

'A $ BULK DATA CARD WITH ID: $ HAS $ NUMBER OF CHORDWISE DIVISIONS (NCHORD) SPECIFIED, '}

'BY ONLY $ ARE LISTED IN THE CORRESPONDING $ BULK DATA CARD WITH ID: $.'

'A $ BULK DATA CARD WITH ID: $ REFERENCED BY A $ BULK DATA CARD WITH ID: $'}

'IS NOT DEFINED AS THE CENTERLINE OF THE BODY.'

'A $ WING MACROELEMENT WITH WID: $ HAS ZERO AREA.'

'ERROR IN $ BULK DATA CARD WITH ID: $.'

'BUT THERE ARE $ NUMBER OF BOX ID SPECIFIED.'

**MODULE 36 ZONAS SPLINZ MODULE MESSAGES**

'$ ENTRY $ REFERENCES AN AERODYNAMIC BODY COMPONENT. ONLY KING-LIKE COMPONENTS ALLOWED.'

'COORDINATE SYSTEM $, REFERENCED ON $ ENTRY $, CANNOT BE FOUND.'

'GRID POINT $, REFERENCED ON $ ENTRY $, CANNOT BE FOUND.'

'GRID POINT $, REFERENCED ON $ ENTRY $, CANNOT BE FOUND.'

'THE STRUCTURAL SET DEFINED BY SET1 ENTRY $, REFERENCED ON $ ENTRY $, IS EMPTY.'

'THE STRUCTURAL POINT DEFINITION PRISM DEFINED BY SET2 ENTRY $ ON $ ENTRY $ HAS ILLEGAL GEOMETRY.'

'THE STRUCTURAL SET DEFINED BY SET2 ENTRY $ ON $ ENTRY $ HAS ILLEGAL GEOMETRY.'

'NO COORDINATE SYSTEM FOR THE SPLINE Y-AXIS IS DEFINED ON $ ENTRY $.'

'DUPLICATE ID IN BULK DATA CARD $ WITH ID: $.'

'IS NOT DEFINED AS THE CENTERLINE OF THE BODY.'

'A $ WING MACROELEMENT WITH WID: $ HAS ZERO AREA.'

'DUPLICATE ID IN BULK DATA CARD $ WITH ID: $.'

'ERROR IN $ BULK DATA CARD $ WITH ID: $. NUMBER OF INLET PANELS EQUALS $ (INLET).'

'BUT THERE ARE $ NUMBER OF BOX ID SPECIFIED.'

'ERROR IN $ BULK DATA CARD $ WITH ID: $. NUMBER OF BOX ID SPECIFIED.'

'ERROR IN $ BULK DATA CARD $ WITH ID: $. NUMBER OF BOX ID SPECIFIED.'
"$ SETID $ SPECIFIES NON-EXISTENT AERODYNAMIC BOX MACROID $, EXTID $.
"COORD SYS $, REFERENCED ON $ ENTRY, CANNOT BE FOUND.
"$ SETID $ SPECIFIES A SPLINE PLANE WHICH IS NEARLY PERPENDICULAR TO THE FREE STREAM VELOCITY.
"$ SETID $ SPECIFIES AERODYNAMIC BOXES BELONGING TO MORE THAN ONE MACRO-ELEMENT.
"$ SETID $ FAILS WHEN USING DEFAULT SPLINE PLANE (CP-BLANK) BECAUSE THE BOUNDARY FOR
" MACRO-ELEMENT $ DOES NOT DEFINE A PLANE. USE CP OPTION TO SPECIFY A REFERENCE PLANE.
"SPLINE2 WITH ID: $ CAN ONLY BE USED WITH CAERO7.
"AERODYNAMIC GRID WITH INTERNAL ID: $ CANNOT BE FOUND IN ATTACH BULK DATA ENTRY.
"STRUCTURAL GRID WITH EXTERNAL ID: $ CANNOT BE FOUND IN ATTACH BULK DATA ENTRY.
"SPLINE2 WITH ID: $ HAS LESS THAN TWO STRUCTURAL GRIDS.
"$ WITH ID: $ ERROR. STRUCTURAL GRID WITHINTERNAL ID: $ CANNOT BE FOUND.
"SPLINE2 WITH ID: $ HAS TWO STRUCTURAL GRIDS WITH ID: $ AND $ THAT SHARE THE SAME
" LOCATION ALONG THE LINE OF THE SPLINE.
"THE $ $ AERODYNAMIC BOX IS NOT ATTACHED TO THE STRUCTURE, THEREFORE, NO DISPLACEMENT
" IS ASSUMED FOR THIS BOX.
"AERODYNAMIC GRID WITH INTERNAL ID: $ CANNOT BE FOUND.
"SPLINE1 WITH ID: $ SPECIFIES A SPLINE PLANE WHICH IS NEARLY PERPENDICULAR TO THE FREE
" STREAM VELOCITY.
"$ WITH ID: $ REFERS TO A SETI THAT HAS LESS THAN $ GRIDS.
"$ WITH ID: $ REFERS TO A SETI THAT HAS ALL GRIDS ALIGNED ALONG A LINE.
"$ WITH ID: $ REFERS TO A SETI THAT HAS TWO GRIDS AT THE SAME LOCATION.
"$ WITH ID: $ GIVES A SINGULAR MATRIX.
"A REFERENCED LOCAL COORDINATE SYSTEM WITH ID: $ CANNOT BE FOUND.
"SPLINE3 WITH ID: $ REFERS TO A SETI THAT HAS ALL GRIDS LOCATED ON THE SAME PLANE.
" THE NORMAL VECTOR OF THE PLANE IS $XN = $, $YN = $, $ZN = $ .

\*MODULE 37 ZOME'S ZAEROM MODULE MESSAGES
"$ WITH ID: $ HAS DUPLICATED REDUCED FREQUENCIES.
"THERE IS NO CAERO7 OR BODY7 INPUT FOR THE ZAERO MODULE.
"THERE IS NO OR MORE THAN ONE $ INPUT FOR THE ZAERO MODULE STEADY/UNSTEADY AERODYNAMIC ANALYSIS.
"THE CONTROL POINT OF AN AERODYNAMIC BOX WITH ID: $ LOCATED ON A CAERO7 MACROELEMENT
" WITH WID: $ ALIGNS WITH THE EDGE OF ANOTHER AERODYNAMIC BOX WITH ID: $ LOCATED ON A
" CAERO7 WITH WID: $.
"THE CONTROL POINT OF AN AERODYNAMIC BOX WITH INTERNAL ID: $ LOCATED ON A CAERO7 WITH INTERNAL
" ID: $ ALIGNS WITH THE EDGE OF ANOTHER AERODYNAMIC BOX WITH INTERNAL ID: $ LOCATED ON A
" CAERO7 WITH INTERNAL ID: $.