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

AIR VEHICLES DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATIERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542

DTIC QUALITY INSPECTED 4
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.
This report is a part of the documentation 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>
</tbody>
</table>

APPENDIX A ......................................................... 47
ASTROS* Functional Module Definition (MODDEF.DAT)

APPENDIX B ......................................................... 50
ASTROS* MAPOL Sequence Definition (MAPOLSEQ.DAT)

APPENDIX C ......................................................... 80
ASTROS* Bulk Data Template Definitions (TEMPLATE.DAT)

APPENDIX D ......................................................... 86
ASTROS* Relational Schema Definition (RELATION.DAT)

APPENDIX E ......................................................... 89
ASTROS* Error Message Definition (SERRMSG.DAT)
### 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>
FOREWORD

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* enviroment.

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.
Figure 1. ZAERO and Other Aerodynamic Modules.

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>ZONA51</td>
</tr>
<tr>
<td>Geometry</td>
<td></td>
</tr>
<tr>
<td>· Lifting Surface (L.S.)</td>
<td>•</td>
</tr>
<tr>
<td>· Thickness Effect</td>
<td></td>
</tr>
<tr>
<td>· L.S. + Body = Whole Aircraft</td>
<td>•</td>
</tr>
<tr>
<td>Mach Number</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-url)
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 Makeqdriv 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 XQDRV (5). Both the ZAERO engineering applications modules (6) and XQDRV source code (5) are compiled by the Makeqdriv script file (7). The object library of ASTROS (Version 11.0) (8) and object files generated by Makeqdriv (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 (Makeqdriv) used to compile the XQDRV 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 Makeqdriv. 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 Makeqdriv 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 Makeqdriv (see Figure 6).
Figure 4. ASTROS* System Generation Process.
Figure 5. Modified Makeqdriv File for ASTROS*. 

Figure 6. Modified Makeastros File for ASTROS*. 

7
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 Appendixes 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 1. Computer Files Comprising ZAERO.

<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 description 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>FLUTQHHZ</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>QHLGGENZ</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 entities 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, GEMZ, [SCNTLG], [SCNTLK], [ACNTLG], [ACNTLK],
               [LMODEG], [LMODEK] );

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

GEOMZA
A relation created by the AEROGM module describing the aerodynamic boxes
(Characteristic, 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 [AIRFRK] 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 = 'ANTISY', 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:


MAPOL Calling Sequence:

CALL FLUTQHZ ( NITER, BCID, SUB, ESIZE(BC), PSIZE(BC), [AJK], [SKJ],
                [UGTKA], [PHIA], USET(BC), [TNM(BC)], [GSUBO(BC)], NGDR,
                AECOMPZ, GEOMZA, [PHIKH], [QHHLPL(BC, SUB)], OAGRDDS
                );

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
(GSIZE+ESIZE) (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)

[TNM(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 No GDR
= -1 GDR is used
(Input, Integer)

AECOMPZ

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

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

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)

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 PHIHK 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

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

   then computes the generalized aerodynamic forces as

   \[ [QHHLFL] = [PHIHK]^T [QKK] [PHIHK] \]

   therefore, the [QKK] matrix is an 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:

CALL FLUTSENZ (NITER, BC, SUB, LOOP, GSIZEB, NDV, GLBDES, CONST, G MkTC, DKVI, G MMCT, DMVI, CLAMBDA, LAMBDA, [QHHLFL(BC, SUB)], [BHFL(BC, SUB)], [KHFL(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)
G MkTC 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)
G MMCT 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)
CLAMBDA 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)
[MHHPL (BC, SUB)] Modal mass matrix (Input)
[BHHPL (BC, SUB)] Modal flutter damping matrix (Input)
[KHHPL (BC, SUB)] Modal flutter stiffness matrix (Input)
[PHIG (BC)] Matrix of real eigenvectors in the structural set (Input)
[AMAT] Matrix of constraint sensitivities (Output)
AEROS 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: FLUTTAZ

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)], CLAMBDAX, 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)

CLAMBDAX  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:

    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 transformation 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 harmonic 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 a intermediate matrix created in QHHGENZ.

2. The gust vector is computed as:

\[ [QHJL] = [PHIKH]^T [QGK] \exp(i*k/(RFC/2.)*x_o) \]

where \( k \) is the reduced frequency.
\( RFC \) is the reference chord.

and \( x_o \) 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 aerelastic 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 STABC and [AIRFC] 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 ECOMPZ, 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
5.0 ZAERO DATABASE ENTITY DESCRIPTIONS

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 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>LMODEK</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 MAEROZ.</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.
Entity: AJC
Entity Type: MATRIX
Description: Basic name of the unsteady aerodynamic matrix containing unsteady pressure coefficients at J-set d.o.f. due to unit control surface deflections. AJC is used during the aeroelastic analysis.
Matrix Form: Complex matrix with number of columns being equal to the number of control surfaces and J-set number of rows being equal to the number of J-set d.o.f.
Created by: UZAERO

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 database 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 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 \( \text{AI}_{siij} \).
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 $QGK_{siijj}$.
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:

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

where $x_0$ is the location of the reference plane defined in the bulk entry GUST.
   $k$ is the corresponding reduced frequency.
   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.
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 $AJK_{siij}$,
where $s='S'$ for symmetric or asymmetric case, $= 'A'$ for antisymmetric case.  
$ii=index$ of Mach number.  
$jj=index $ of reduced frequency.

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

$$[QKK] = [SKJ]^T [AJK_{siij}]^T$$

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

$$[CP] = [AJK_{siij}]^T [PHIKH]$$

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:

   
   \[ [AJC] = [AJK]^T [ACNTLK] \]

   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 \text{[SCNTLA]} can be computed by the reduction of \text{[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 antisymmetric 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:


   or


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.
<table>
<thead>
<tr>
<th>Entity</th>
<th>LMODEK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity Type</td>
<td>MATRIX</td>
</tr>
<tr>
<td>Description</td>
<td>Displacements and slopes defined at K-set d.o.f due to the load modes specified in bulk entries LOADMOD.</td>
</tr>
<tr>
<td>Matrix Form</td>
<td>Real matrix with K-set number of rows and number of columns being equal to the number of LOADMOD bulk entries.</td>
</tr>
<tr>
<td>Created by</td>
<td>CONMOD</td>
</tr>
</tbody>
</table>

**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:

\( \{T1, T2, T3, \frac{dT1}{dx}, \frac{dT2}{dx}, \frac{dT3}{dx} \} \), where T1, T2, and T3 are the displacements at the centroid of the aerodynamic box along x, y, and z directions, respectively. \( \frac{d(\cdot)}{dx} \) 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.
**Entity:** AECOMPZ  
**Entity Type:** Relation  
**Description:** Contains data on the aerodynamic components in the CAERO7 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 'CAERO7' 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 CAERO7, 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 CAERO7. =1 for BODY7.</td>
</tr>
<tr>
<td>NSBOX</td>
<td>Integer&gt;0</td>
<td>Number of spanwise boxes for CAERO7. Number of boxes for BODY7.</td>
</tr>
</tbody>
</table>
| BNDRY    | R Vector(12) | For CAERO7: 
|          |          | 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. |
| WCOS     |          | For CAERO7: Cos(theta), where theta = dihedral angle. 
|          |          | For BODY7: Number of segments. (Integer) |
| WSIN     |          | For CAERO7: Sin(theta), where theta = dihedral angle. 
<p>|          |          | For BODY7: Not used. |
| IWING    | Integer  | Flag for vertical fin on the X-Z plane. =0: yes, =1, no. |
| ATTR     | Integer  | =0: CAERO7 root is not attached to BODY7. &gt;0: CAERO7 root is attached to BODY7 with ID=ATTR. Not used for BODY7. |
| YRB      | Real     | Y location of the center line of BODY7 to which the CAERO7 root is attached. |
| ZRB      | Real     | Z location of the center line of BODY7 to which the CAERO7 root is attached. |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>
| ATIT    | Integer | =0: CAERO7 Tip is not attached to BODY7.  
                   >0: CAERO7 Tip is attached to BODY7 with ID=ATIT  
                   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. |
| FLSINTI | 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.
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>DZXLE</td>
<td>Real</td>
<td>Thickness slope at leading edge of the mid-chord for <strong>CAERO7</strong> box. Not used for <strong>BODY7</strong> box.</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DZXTE</td>
<td>Real</td>
<td>Thickness slope at trailing edge of the mid-chord for <strong>CAERO7</strong> box. Inlet panel flow ratio in percentage for <strong>BODY7</strong> box.</td>
</tr>
<tr>
<td>IWAKE</td>
<td>Integer</td>
<td>For <strong>BODY7</strong> box=$1$, box is inlet panel. $=0$, box is not inlet panel. Not used for <strong>CAERO7</strong> 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.
Contains 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>Nondimensional 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>

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]:

```fortran
CHARACTER*8 UNLIST(12), NAME
DATA UNLIST//"IDMK", "MACH", "METHOD", "SYMXZ", "ALPHA", "BETA", "FRATE", "QRATE"
*, "RRATE", "MINDEX", "KINDEX", "RFREQ"
INTEGER IINFO(20), IGET(12), MINDEX(100), KINDEX(100), SYMXZ
REAL RGET(12), K(100), MACH
EQUIVALENCE (RGET(1), IGET(1))
CHARACTER*4 $ CALL DBOPEN(REUNMK, INFO, ',RO', ',NOFLUSH', ISTAT)
CALL REPROJ(REUNMK, 12, UNLIST)
NMK = INFO(4)
C INDEX = total number of MKAERO2 bulk entries.
DO I=1, NMK
  CALL REGST(REUNMK, IGET, ISTAT)
  IF(IDMK.EQ.IGET(11)) THEN
    INDEX = INDEX + 1
    MACH = REGST(2)
    METHOD = IGET(3)
    ISYM = IGET(4)
    MINDEX(INDEX) = IGET(10)
    KINDEX(INDEX) = IGET(11)
    K(INDEX) = RGET(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
C following:
C KTH = 2
C 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's Mach number index found in the REUNMK realtion.
C      KINDEX(KTH) KTH's reduced frequency index found in the REUNMK relation.
C OUTPUT: A character*8 string of the matrix created by UAERO module with the form:
C AJK3ijjj, where s=S, i=MINDEX(KTH), and j=KINDEX(KTH)
CALL MYNAME(AJK, S, MINDEX(KTH), KINDEX(KTH), NAME)
C Now, NAME is the matrix name of the AIC matrix of the corresponding Mach number and
C reduced frequency.
C CALL MYNAME(AJC, S, MINDEX(KTH), KINDEX(KTH), NAME)
C Now, NAME is the matrix name of the control surface forces matrix of the corresponding
C Mach number and reduced frequency.
C CALL MYNAME(QGK, S, MINDEX(KTH), KINDEX(KTH), NAME)
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
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

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

FLUTQHNL 18
102 -1  1  1  1  0  0  0  0  0  0  0  0  0  1  7  7  8  8  7
C
C PROCESS THE 'FLUTQHNL' MODULE - FLUTTER AEROMATRIX PROCESSOR
C
CALL FLUTQHNL ( IP(1), IP(2), IP(3), IP(4), IP(5), IP(6), IP(7),
1     IP(8), IP(9), IP(10), IP(11), IP(12), IP(13),
2     IP(14), IP(15), IP(16), IP(17), IP(18) )
END

FLUTSENZ 21
102  1  1  4  1  1  7  7  7  9  7  9  7  7  8  8  8  8  8  8
8  7
C
C PROCESS THE 'FLUTSENZ' MODULE TO OBTAIN FLUTTER CONST. SENSITIV.
C
CALL FLUTSENZ ( IP(1), IP(2), IP(3), IP(4), IP(5), IP(6), IP(7),
*     IP(8), IP(9), IP(10), IP(11), IP(12), IP(13),
*     IP(14), IP(15), IP(16), IP(17), IP(18), IP(19),
*     IP(20), IP(21) )
END

FLUTTRAZ 13
102 -1  1  1  8  7  1  1  8  8  8  7  7
C
C PROCESS THE 'FLUTTRAZ' MODULE TO PERFORM FLUTTER ANALYSIS
C
CALL FLUTTRAZ ( IP(1), IP(2), IP(3), IP(4), IP(5), IP(6), IP(7),
1     IP(8), IP(9), IP(10), IP(11), IP(12), IP(13) )
END

QHILGENZ 11
102  1  1  8  8  8  8  8  8  8  8  8  7
C
C 'QHILGENZ' - GENERATE THE QHIL MATRIX LIST FOR FLUTTER ANALYSIS
C
CALL QHILGENZ ( IP(1), IP(2), IP(3), IP(4), IP(5), IP(6),
1     IP(7), IP(8), IP(9), IP(10), IP(11) )
END

SPLINZ 5
102  1  7  7  7  8
C
C PROCESS THE UNSTEADY AEROdynamic SPLINE
C
CALL SPLINZ ( IP(1), IP(2), IP(3), IP(4), IP(5) )
END

48
PROCESS ZAERO STEADY AERODYNAMICS
(FRAME TO STATIC AEROELASTICITY DISCIPLINE)

END

AIC GENERATION BY ZAERO MODULE

END

READ MODAL RESULTS FROM NASTRAN OUTPUT4 SOLUTION
AND REPLACE THE ASTROS DATABASE MATRICES KAA, MAA, PHIA
AND RELATION LAMBDA

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 *****

<table>
<thead>
<tr>
<th>STAT. LEVEL</th>
<th>LINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>116***$</td>
</tr>
<tr>
<td>2</td>
<td>116 CSCID &lt;8($) MC0083-MAPOLSEQ 11.1 4/29/94 17:00:35&gt; $</td>
</tr>
<tr>
<td>3</td>
<td>116*</td>
</tr>
<tr>
<td>4</td>
<td>116**</td>
</tr>
<tr>
<td>5</td>
<td>116 EXECUTIVE SEQUENCE FOR ASTROS $</td>
</tr>
<tr>
<td>6</td>
<td>116*</td>
</tr>
<tr>
<td>7</td>
<td>116*</td>
</tr>
<tr>
<td>8</td>
<td>116*</td>
</tr>
<tr>
<td>9</td>
<td>116*</td>
</tr>
<tr>
<td>10</td>
<td>116 INTEGER SINGSET, SINGSET, SINGSET;</td>
</tr>
<tr>
<td>11</td>
<td>116*</td>
</tr>
<tr>
<td>12</td>
<td>116*</td>
</tr>
<tr>
<td>13</td>
<td>116*</td>
</tr>
<tr>
<td>14</td>
<td>116*</td>
</tr>
<tr>
<td>15</td>
<td>116 INTEGER GSIZE, NSET, NITER, BC,</td>
</tr>
<tr>
<td>16</td>
<td>116 EFSIZE(1000), NSIZE(1000), GSIZES;</td>
</tr>
<tr>
<td>17</td>
<td>116 REAL CIT, CTIMIN;</td>
</tr>
<tr>
<td>18</td>
<td>116 LOGICAL GBSCVGR, APFCVGR, PFLAG;</td>
</tr>
<tr>
<td>19</td>
<td>116 UNSTRUCT DEVENT, GRIDTEMP, SMILID;</td>
</tr>
<tr>
<td>20</td>
<td>116 RELATION DESHIFT, CONST, MFRAM, CONVERT, OCPARM,</td>
</tr>
<tr>
<td>21</td>
<td>116 MOMENT1, GRID, SPOINT, EPOINT, SEQFP,</td>
</tr>
<tr>
<td>22</td>
<td>116 BENVET(1000), CSTM, FORCE, FORCE1, MOMENT1,</td>
</tr>
<tr>
<td>23</td>
<td>116 TEMP, TEMPL, OMILBUCK, OELBUCK,</td>
</tr>
<tr>
<td>24</td>
<td>116 CORDC, CORDC, CORS, CORDC2, CORS2;</td>
</tr>
<tr>
<td>25</td>
<td>116 CORD2S, GPNGGRID, OPGAP, GRADIENT;</td>
</tr>
<tr>
<td>26</td>
<td>116*</td>
</tr>
<tr>
<td>27</td>
<td>116*</td>
</tr>
<tr>
<td>28</td>
<td>116*</td>
</tr>
<tr>
<td>29</td>
<td>116*</td>
</tr>
<tr>
<td>30</td>
<td>116*</td>
</tr>
<tr>
<td>31</td>
<td>116*</td>
</tr>
<tr>
<td>32</td>
<td>116 UNSTRUCT USEI(1000), GFS11(1000);</td>
</tr>
<tr>
<td>33</td>
<td>116*</td>
</tr>
<tr>
<td>34</td>
<td>116*</td>
</tr>
<tr>
<td>35</td>
<td>116*</td>
</tr>
<tr>
<td>36</td>
<td>116 MATRIX [PFOA(1000)], [PFOA(1000)], [PFOA(1000)];</td>
</tr>
<tr>
<td>37</td>
<td>116*</td>
</tr>
<tr>
<td>38</td>
<td>116*</td>
</tr>
<tr>
<td>39</td>
<td>116*</td>
</tr>
<tr>
<td>40</td>
<td>116*</td>
</tr>
<tr>
<td>41</td>
<td>116*</td>
</tr>
<tr>
<td>42</td>
<td>116*</td>
</tr>
<tr>
<td>43</td>
<td>116*</td>
</tr>
<tr>
<td>44</td>
<td>116*</td>
</tr>
<tr>
<td>45</td>
<td>116 UNSTRUCT TREE, DVSIZE, PCOMPS;</td>
</tr>
<tr>
<td>46</td>
<td>116 UNSTRUCT KELM, HELM, TELM;</td>
</tr>
<tr>
<td>47</td>
<td>116 RELATION CPQMEM1, CPQMEM2, CROD, CONROD, RODEST,</td>
</tr>
<tr>
<td>48</td>
<td>116 SHEAREST, CTREM, TRMEST, CNASS1;</td>
</tr>
<tr>
<td>49</td>
<td>116 CHASS, CMASS2, MASSEST, CONN1, COMM1, COMM2;</td>
</tr>
<tr>
<td>50</td>
<td>116 CONKEM2, CBAR, BEAMEST, CQUAD, QUADEST;</td>
</tr>
<tr>
<td>51</td>
<td>116 CINEX1, CINEX2, CINEX3, CINEX4, CINEX5;</td>
</tr>
<tr>
<td>52</td>
<td>116 INEXE2, ELAS1, ELAS2, ELASEST;</td>
</tr>
<tr>
<td>53</td>
<td>116 PCOMP, PCQMEM1, PCQMEM2, PROD, PSPHEAR;</td>
</tr>
<tr>
<td>54</td>
<td>116 RTMREM, PHASS, PFLAS, PBAR, PSHELL;</td>
</tr>
<tr>
<td>55</td>
<td>116 PCOMP1, PCOMP2, PHICE, PHICE1, MAT1, MAT2;</td>
</tr>
<tr>
<td>56</td>
<td>116 MAT3, MAT4, CTRIA3, TRIASEST;</td>
</tr>
<tr>
<td>57</td>
<td>116*</td>
</tr>
<tr>
<td>58</td>
<td>116*</td>
</tr>
<tr>
<td>59</td>
<td>116*</td>
</tr>
<tr>
<td>60</td>
<td>116*</td>
</tr>
<tr>
<td>61</td>
<td>116*</td>
</tr>
<tr>
<td>62</td>
<td>116 RELATION DESEL, DESVARS, PLIST, ELIST,</td>
</tr>
<tr>
<td>63</td>
<td>116 SHAPE, SPGEN;</td>
</tr>
<tr>
<td>64</td>
<td>116 RELATION DCONVM, DCONTW, DCONFD, DCONF, DCONVM;</td>
</tr>
</tbody>
</table>

51
BEGIN MAPOL SOLUTION SEQUENCE

SUBST := 1;
PRINT("LOG('BEGIN PREFACE MODULES')");
CALL SOLUTION ( NUMOPSEC, NUMCOND, MPS, MPE, OCS, OCL, FSS, FSSL, MAXITER, MOVLIN, WINDOW, OCMOVLIN, ALPHA, CNVRGLIM, NRPAP, EPS );
CALL IFP( GSIZEB );
TRY USING A UTILITY TO PRINT OUT THE GRID RELATIONAL ENTITY
GENERATE THE ELEMENT MATRICES
PRINT("LOG('ELEMENT MATRIX GENERATION')");
CALL MakeST( NV, GLOBES, [TRANS], [MIN], [MAXST], LOCVAR,
TIXED, DESLINK );
CALL EMG( NV, GSIZEB, GLOBES, DESLINK, [MAT], DUCT, DVSIZE,
KMLM, KMLM, KELM, TELM, TREF );
CALL IFBULK( GSIZEB, EGSUMINARY, EODISC, OPELELEM );
HANDLE THE NON-PLANAR STEADY AERODYNAMICS ANALYSES
TERMINATE THE EXECUTION IF THE ONLY DISCIPLINE IS NPSAERO
PRINT("LOG('NON-PLANAR STEADY AERODYNAMICS')");
CALL STADYPF( NONFORM, RECOMPS, GEOMSA, STABCF, [AIRFORCE], AEROGRAH,
CARSCEM, GASHLED );
IF NONFORMLY CALL EXIT;
ASSEMBLE THE ELEMENT MATRICES
TO THE SENSITIVITY MATRICES
PRINT("LOG('PHASE 1 ELEM. MATRIX ASSEMBLY')");
CALL EMG2( NV, GLOBES, DUCT, KMLM, KMLM, GMCT, DVI, GMCT, DMVI );
GENERATE THE SIMPLE LOAD VECTORS
AND LOAD SENSITIVITIES
PRINT("LOG('PHASE 1 STATIC LOADS GENER.')");
CALL LODGEN( GSIZEB, GLOBES, DUCT, DVSIZE, GMCT, DVI, TELM, TREF,
SMLD, [DPTVI], [DPTVI] );
GENERATE THE STEADY AIC MATRIX AND THE
STEADY SPLINE TRANSFORMATION MATRICES
11$PRINT("LOG="'STeady AERodynamics'";)
11$LOOP := TRUE;
11$MINDEX := 0;
11$WHILE LOOP DO
11$ MINDEX := MINDEX + 1;
11$ CALL STEADY ( MINDEX, LOOP, AECOMPS, GEOMSA, STABCF, [AICMAT(MINDEX)],
11$ [AIRFX(MINDEX)], AEROgeom, CAROGEOM ;
11$ENDDO;
11$CALL SPLINES ( GSIZE, GEOMSA, AECOMPS, AERO, [GTS], [GTS]);
11$ GENERATE THE UNSTADY AIC MATRIX AND THE
11$ UNSTADY SPLINE TRANSFORMATION MATRIX
11$
11$PRINT("LOG="'UNSTADY AERodynamicS'";)
11$CALL UNSTEADY ( GEOMSA, AECOMPS, [AJT], [D1JK], [D2JK], [SKJ],
11$ AEROgeom, CAROGEOM );
11$CALL AMP [ [AJT], [D1JK], [D2JK], [SKJ], [QOC], [QTL], [QUL] ];
11$CALL SPLINEU ( GSIZE, GEOMSA, AECOMPS, AERO, [GTS]);
11$***********************************************************************
11$
11$***********************************************************************
11$
11$EAERO MODULE P. C. CHEM 3-28-1997
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$PRINT("LOG="'EAERO AERODYNAMIC GEOMETRY'";)
11$PRINT("LOG="'EAERO AERODYNAMIC GEOMETRY'";)
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$CALL AEROM Module
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$FOR BOTH STEADY AND UNSTEADY GEOMETRY GENERATIONS
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$CALL AEROM ( AECOMPS, GEOMSA, AGRIDS );
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$PRINT("LOG="'EAERO CONTROL MODE MODULE'";)
11$PRINT("LOG="'EAERO CONTROL MODE MODULE'";)
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$CALL COMMOD ( GEOMSA, AECOMPS, [SCNTLG], [SCNTLK], [ACHTLG], [ACHTLK], [IANGDEG],
11$ [IANGDEG] );
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$PRINT("LOG="'EAERO SPLINE MODULE '";)
11$PRINT("LOG="'EAERO SPLINE MODULE '";)
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$CALL SPLINES ( GSIZE, GEOMSA, AECOMPS, AERO, [GTS]);
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$FOR BOTH STEADY AND UNSTEADY AIC GENERATIONS
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$PRINT("LOG="'EAERO UNSTADY AERODYNAMICS '";)
11$PRINT("LOG="'EAERO UNSTADY AERODYNAMICS '";)
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$CALL UAVERO [ AECOMPS, GEOMSA, AGRIDS, [AJK], [AJC], [AJL], [QMK],
11$ [SKJ], [SCNTLG], [SCNTLK], [IANGDEG] ];
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$PRINT("LOG="'EAERO STEADY AERODYNAMICS '";)
11$PRINT("LOG="'EAERO STEADY AERODYNAMICS '";)
11$***********************************************************************
11$
11$***********************************************************************
11$
11$***********************************************************************
11$
11$LOOP := TRUE;
11$MINDEX := 0;
11$WHILE LOOP DO
21$ MINDEX := MINDEX + 1;
21$ CALL UAVERO ( [AJK], MINDEX, LOOP, AECOMPS, GEOMSA, AGRIDS, STABCF,
21$ [AICMAT(MINDEX)], [AIRFX(MINDEX)], [IANGDEG] );
21$ [SCNTLG], [SCNTLK] );
21$ENDDO;
21$***********************************************************************
21$***********************************************************************
21$***********************************************************************
21$IF NUMOFTRC > 0 THEN
21$ PRINT("LOG="'BEGIN OPTIMIZATION'";)
21$ PRINT("LOG="'BEGIN OPTIMIZATION'";)
21$***********************************************************************
21$***********************************************************************
21$***********************************************************************
21$INITIALIZE MAPOF PARAMETERS
21$***********************************************************************
21$***********************************************************************
21$***********************************************************************
21$GLOBALVREG := FALSE;
21$GLOBALVREG := FALSE;
21$BEGIN CONVERGENCE LOOP
21$WHILE NOT GLOBALVREG AND NITER <= MAXITER DO
31$
ASSEMBLE THE GLOBAL MATRICES

BEGIN BOUNDARY CONDITION LOOP FOR OPTIMIZATION

FOR BC = 1 TO NUMDFBC DO

Establish the base Uset and partitioning data for the BC
This data must be recreated each iteration since GDR can change it

MAKE B.C.-DEPENDENT BGPDT FROM BASE, ADDING THE EXTRA POINTS FOR
THIS B.C.

CALL BCBPDT ( BC , GSIZE , BGPDT( BC ), ESIZE( BC ) );

GSIZE := GSIZE;

FSIZE := ESIZE( BC ) + GSIZE;

PROCESS MATRICES, TRANSFER FUNCTIONS, AND INITIAL CONDITIONS FOR
THIS B.C.

CALL BCBULK ( BC , FSIZE( BC ) , BGPDT( BC ), USET( BC ) );

CALL BOUND ( BC , GSIZE , ESIZE( BC ) , USET( BC ) , BLOAD, BMASS, DMODES, BMODES, BSAREA, BELTUR, Bdyn, BDSP, BDR, BMFR, BUST, BELAST, NMPC, NSPC, NOMIT, NSET, NGDR );

DETERMINE IF ANY M2GG/K2GG INPUT DATA ARE TO BE ADDED

CALL NULLMAT ( [KGG], [MGG] );

CALL MK2GG ( BC , GSIZE , M2GG( BC ), M2GFLAG, [K2GG], [K2GGFLAG] );

IF M2GFLAG THEN

[MGG] := [MIGG] + [M2GG];

ELSE

[MGG] := [MIGG];

ENDIF;

IF K2GFLAG THEN

[KGG] := [K2G] + [K2GG];

ELSE

[KGG] := [K2G];

ENDIF;

CALL THE GRID POINT WEIGHT GENERATOR FOR THIS BOUNDARY CONDITION

CALL GWPG ( NITER, BC, GPRGRID, [MGG], [OFPWG] );

IF BLOAD <= 0 CALL GTLOAD (NITER, BC, GSIZE, BGPDT( BC ), GLBDES, SMPLD, [PDMTHVI], [RDMV2], [PG], [OGRIDL0])

PARTITION-REDUCTION OF GLOBAL MATRICES

**********************************************************************

IF NUMDFBC > 1 CALL NULLMAT ( [BNH], [PH], [BNH] );

**********************************************************************

IF NUMDFBC > 1 CALL NULLMAT ( [BNH], [PH], [BNH], [USPMN] );

IF NFMP <= 0 THEN

PERFORM MFC REDUCTION

PRINT("LOG\n" MPC REDUCTION");

CALL MGREDUCE ( [KGG], [PG], [PMG(BC)], [TMW(BC)], [BNH], [PH] );

IF BMASS <= 0 CALL MGREDUCE ( [KGG], [FPMG(BC)], [TMW(BC)], [BNH] );

**********************************************************************

IF BSAREA <= 0 THEN

CALL MGREDUCE ( [GKGG], [FPMG(BC)], [TMW(BC)], [USTHM] );

CALL MGREDUCE ( [GKGG], [FPMG(BC)], [TMW(BC)], [GSTM] );

**********************************************************************
ENDIF;

IF BLUTLR < 0 OR BGUST < 0 OR BLEAST < 0 OR BSAERO < 0
CALL GREduce ([UGTKG], [FGMN(BC)], [TMN(BC)], [UGTKN]);
ELSE
    NO_MPC_REDUCTION
ENDIF;

[KNN] := [KSG];
IF BLOAD < 0 [PN] := [PG];
IF BMASS < 0 [MNN] := [MGG];

************** TAKEN OUT FOR EAERO **************

IF BSAERO < 0 THEN
    [GTFN] := [GTFK];
    [GSTFN] := [GTFK];
ENDIF;

************** TAKEN OUT FOR EAERO **************

IF BLUTLR < 0 OR BGUST < 0 OR BLEAST < 0 OR BSAERO < 0
    [UGTKN] := [UGTKG];
ENDIF;

PERFORM AUTOSPC CALCULATIONS ON THE KNM MATRIX

PRINT("LOG="
    AUTOSPC COMPUTATIONS");
CALL GPSS ([HIER, BC, NGDR, [KNN], BQPT(BC), [YS(BC)]);
CALL MKPVCT (USBC[BC], GPST[BC]);
CALL MKPVCT (USBC[BC], [FGMN(BC)], [PNSF(BC)]);
    [FPOA(BC)], [PARL(BC)]
    BOUNDARY (BC, GSIZE, ESIZE(BC), USBC[BC], NSFC, NOMIT, NRES); FOR SENSITIVITY ANALYSIS, SAVE A COPY OF THE PRE-GDR PART. VECTORS.
CALL MKPVCT (USBC[BC], [FGMN(BC)], [PNSF(BC)]);
CALL MKPVCT (USBC[BC], [FGMN(BC)], [PNSF(BC)]);
ENDIF;

************** TAKEN OUT FOR EAERO **************

IF NUMOPTBC > 1 CALL NULLMAT ([KFF], [PF], [NFF], [GTFN], [GSTFN], [UGTKF]);

************** TAKEN OUT FOR EAERO **************

IF NUMOPTBC > 1 CALL NULLMAT ([KFF], [PF], [NFF], [UGTKF]);
IF NSFC < 0 THEN
    PERFORM SPC REDUCTION
    PRINT("LOG="
    SPC REDUCTION");
CALL NRREDUCE ([KNN], [PN], [PNSF(BC)], [YS(BC)], [KFF], [KFS], [KF], [PF], [PS]);
    IF BMASS < 0 CALL NRREDUCE ([MNN], [PNSF(BC)], [MFF]);

************** TAKEN OUT FOR EAERO **************

IF BSAERO < 0 THEN
    CALL NRREDUCE ([GTFN], [PNSF(BC)], [GTFK], [GSTFN]);
CALL NRREDUCE ([GSTFN], [PNSF(BC)], [GTFK], [GSTFN]);
ENDIF;

************** TAKEN OUT FOR EAERO **************

IF BLUTLR < 0 OR BGUST < 0 OR BLEAST < 0 OR BSAERO < 0
CALL NRREDUCE ([UGTKN], [PNSF(BC)], [UGTKF]);
ELSE
    NO_SPC_REDUCTION
[KFF] := [KNN];
IF BLOAD < 0 [PF] := [PG];
IF BMASS < 0 [MFF] := [MGG];

************** TAKEN OUT FOR EAERO **************

IF BSAERO < 0 THEN
    [GTFN] := [GTFK];
    [GSTFN] := [GTFK];
ENDIF;

************** TAKEN OUT FOR EAERO **************

IF BLUTLR < 0 OR BGUST < 0 OR BLEAST < 0 OR BSAERO < 0
    [UGTKF] := [UGTKG];
ENDIF;

IF NUMOPTBC > 1 CALL NULLMAT ([KAA], [PA], [KAA], [KAA], [PA], [UGTKA]);
IF NGDR < 0 THEN
    PERFORM THE GENERAL DYNAMIC REDUCTION WHICH IS DISCIPLINE
    INDNEPENT. THE RESULTING [GSUBO] MATRIX WILL BE USED BY
    ALL DISCIPLINES
ENDIF:
PRINT("LOG='DYNAMIC REDUCTION'");

OBTAIN THE OMITTED DOF PARTITION OF KF AND MFF;

CALL PARTN ([KF], [KOO], [KOA], [PFOA(BC)]);
CALL PARTN ([MFF], [MOO], [PFOA(BC)]);

ASIZE := GEIZE - MNPC - NSPC - NOMIT;
LSIZE := ASIZE - NSSET;
CALL GDR1 ([KOO], [MOO], [KSOO], [GGO], LSET, LSSET, NEIV, FMAX, BC, BGFDT(BC), USET(BC), NOMIT, LSIZE);

LSET <> 0
   MEANING
   APPOX. MODE SHAPES SELECTED
   = 0
   NO APPOX. MODE SHAPES IN GDR

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

CALL GDR3 ([MOO], [KOA], [MG], [PHIOK], [TMN(BC)], [GGO],
[HGM(BC)], [PNSF(BC)], [PFOA(BC)], [GSUBO(BC)],
BGFDT(BC), USET(BC),
LSET, LSSET, ASIZE, GNORM, BC);

CALL GDR4 (BC, GEIZE, PSIZE(BC), LSET, LSSET, NUMPTBC, NBNDCOND,
[HGM(BC)], [TMN(BC)], [PNSF(BC)], [PFOA(BC)],
[PARL(BC)], [PFDGK(BC)], [PAJK], [PFJK], BGFDT(BC), USET(BC));

ENDIF;

IF BLOAD <> 0 OR BNOHD <> 0 OR BFULTR <> 0 OR BDIN <> 0 THEN
   REDUCE THE MATRICES WITHOUT AEROLEASTIC CORRECTIONS
   IF NGDR <> 0 THEN
   IF NGMIM <> 0 THEN
   IF NDIM <> 0 THEN
   IF NBOUND <> 0 THEN
   IF NBMIS <> 0 THEN
   IF NRM <> 0 THEN
   IF NNSRT <> 0 THEN
   IF NBUL <> 0 OR BCGUST <> 0 OR BALLEL <> 0 THEN
   IF BBOUND <> 0 THEN
   IF NFALT <> 0 THEN
   NO F-SHAP REDUCTION
   [KAA] := [KFP];
   IF BLOAD <> 0 [PA] := [PF];
   IF BFULTR <> 0 OR BCGUST <> 0 OR BALLEL <> 0 [UGTKA] := [UGTKF];
IF BMAS <= 0 [MAA] := [MFF];
ENDIF;

IF NSET <= 0 THEN
PERFORM THE SUPPORT SET REDUCTION
PRINT("LOG='") SUPPORT REDUCTION"');

IF NITER = 1 THEN
CALL SDOMP ([KAA], [KRR], [KLR], [KLL], [PARL(BC)]);
CALL SLOMEX [KLL], [KLLNV(BC)], USET(BC), SINGLSET);
CALL FBS ([KLLNV(BC)], [KLL], [D(BC)], -1);
CALL SDOMP ([BC], USET(BC), BGSDF(BC), [D(BC)], [KLL], [KRR], [KLR]);
ELSE
IF BLOAD <= 0 THEN
CALL SDOMP ([KAA], [KLR], [KLL], [PARL(BC)]);
CALL SDOMP ([KLL], [KLLNV(BC)], USET(BC), SINGLSET);
ENDIF;
ENDIF;

CALCULATE THE REDUCED MASS MATRIX
CALL SDOMP ([MAA], [NRBAR], [MLR], [MII], [PARL(BC)]);
[IFR(BC)] := [NLL] * [D(BC)] + [MLR];
[MRB(BC)] := [NRBAR] + [FNS(BC)] * [IFR(BC)];
[TR] := [TRANS(D(BC))] * [IFR(BC)];
[Q22] := [TRANS(D(BC))] * [TR] + [NRBAR];

IF BLOAD <= 0 THEN
PROCESS STATICS WITH INERTIA RELIEF
PRINT("LOG='") >>>DISCIPLINE: STATIC(INERTIA RELIEF)'");
CALL ROWPART ([PA], [PR], [PLBAR], [PARL(BC)]);
[LH(BC)] := [MII] - [NRBAR];
[RH(BC)] := [TRANS(D(BC))] * [PLBAR] + [PR];
CALL INERTIA ([LH(BC)], [RH(BC)], [AR]);
[AL] := [D(BC)] * [AR];
CALL SIMERGE ([AA], [AR], [AL], [PARL(BC)]);
[RH(BC)] := [PLBAR] - [IFR(BC)] * [AR];
CALL FBS ([KLLNV(BC)], [RHS(BC)], [UL]);
CALL YSiMERGE ([UA], [UL], [PARL(BC)]);
ENDIF;
ENDIF;

IF BMODS <= 0 THEN
PRINT("LOG='") >>>DISCIPLINE: NORMAL MODES'");
CALL REG ([NTER, BC], USET(BC), [KAA], [NRBAR], [MII], [D(BC)], LAMBDA, [PHIA], [MII], [HEIZE(BC)]);
CALL OFFROOT ([NTER, BC], NUMOPTBC, LAMBDA);
CALL FCHEVAL ([NTER, BC], LAMBDA, CONST);
ENDIF;
ELSE
NO SUPPORT SET REDUCTION
ENDIF;

IF BLOAD <= 0 THEN
PRINT("LOG='") >>>DISCIPLINE: STATICS'");
CALL SDOMP ([KAA], [KLLNV(BC)], USET(BC), SINGLSET);
CALL FBS ([KLLNV(BC)], [PA], [UA]);
ENDIF;

IF BMODS <= 0 THEN
PRINT("LOG='") >>>DISCIPLINE: NORMAL MODES'");
CALL REG ([NTER, BC], USET(BC), [KAA], [NRBAR], [MII], [PHIA], [MII], [HEIZE(BC)]);
CALL OFFROOT ([NTER, BC], NUMOPTBC, LAMBDA);
CALL FCHEVAL ([NTER, BC], LAMBDA, CONST);
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
IF BFAERO <= 0 THEN
PERFORM STATIC AEROSLATIC ANALYSES
PRINT("LOG='") SAERO INITIALIZATION'");
CALL TRSFPO (GSKX);
SUB := 0;

WHILE LOOP DO

SUB := SUB + 1;

CALL SAERO2DR (BC, SUB, LOOP, MINDEX, SYM, MACH, QDF, 1); $!

ADJUST THE KFF MATRIX AND DETERMINE THE RIGID AIR LOADS $!

6156

6161

6167

6175

6179

6183

6189

6193

6195

6199

6201

6203

6205

6207

6209

6211

6215

6217

6219

6221

6223

6227

6229

6231

6235

6237

6239

6241

6243

6245

6247

6251

6253

6255

6257

6259

6261

6263

6265

6267

6269

6271

6273

6275

6277

6279

6281

6283

6285

6287

6289

6291

6293

6295

6297

6299

706

SUB := 0;

WHILE LOOP DO

SUB := SUB + 1;

CALL SAERO2DR (BC, SUB, LOOP, MINDEX, SYM, MACH, QDF, 1); $!

ADJUST THE KFF MATRIX AND DETERMINE THE RIGID AIR LOADS $!

6156

6161

6167

6175

6179

6183

6189

6193

6195

6199

6201

6203

6205

6207

6209

6211

6215

6217

6219

6221

6223

6227

6229

6231

6235

6237

6239

6241

6243

6245

6247

6251

6253

6255

6257

6259

6261

6263

6265

6267

6269

6271

6273

6275

6277

6279

6281

6283

6285

6287

6289

6291

6293

6295

6297

6299

706

SUB := 0;

WHILE LOOP DO

SUB := SUB + 1;

CALL SAERO2DR (BC, SUB, LOOP, MINDEX, SYM, MACH, QDF, 1); $!

ADJUST THE KFF MATRIX AND DETERMINE THE RIGID AIR LOADS $!

6156

6161

6167

6175

6179

6183

6189

6193

6195

6199

6201

6203

6205

6207

6209

6211

6215

6217

6219

6221

6223

6227

6229

6231

6235

6237

6239

6241

6243

6245

6247

6251

6253

6255

6257

6259

6261

6263

6265

6267

6269

6271

6273

6275

6277

6279

6281

6283

6285

6287

6289

6291

6293

6295

6297

6299

706

SUB := 0;

WHILE LOOP DO

SUB := SUB + 1;

CALL SAERO2DR (BC, SUB, LOOP, MINDEX, SYM, MACH, QDF, 1); $!

ADJUST THE KFF MATRIX AND DETERMINE THE RIGID AIR LOADS $!

6156

6161

6167

6175

6179

6183

6189

6193

6195

6199

6201

6203

6205

6207

6209

6211

6215

6217

6219

6221

6223

6227

6229

6231

6235

6237

6239

6241

6243

6245

6247

6251

6253

6255

6257

6259

6261

6263

6265

6267

6269

6271

6273

6275

6277

6279

6281

6283

6285

6287

6289

6291

6293

6295

6297

6299

706

SUB := 0;

WHILE LOOP DO

SUB := SUB + 1;

CALL SAERO2DR (BC, SUB, LOOP, MINDEX, SYM, MACH, QDF, 1); $!

ADJUST THE KFF MATRIX AND DETERMINE THE RIGID AIR LOADS $!
PRINT("LOG-\{'\"SAERO SUPPORT REDUCTION\'\}"));

IF MITER = 1 AND SUB = 1 AND BLOAD = 0 AND BMODES = 0 AND
BFLUR = 0 AND BDYN = 0 THEN
ENDIF:

CALL PARTN ( \{KAA, \{KRR, \{KLR, \{KLL, \{PABL(BC)\}\}\}\}\}\}\}\);}

CALL SDCOMP ( \{KLL, \{KLLINV(BC)\}, USET(BC), SINGLSET \} );

CALL FBS ( \{KLLINV(BC)\}, \{KLR\}, \{DBC\}, -1 \} );

CALL RACHECK ( BC, USET(BC), BGFDT(BC), \{DBC\}, \{KLL\},
\{KRR\}, \{KLR\} );

END:

CALCULATE THE REDUCED MASS MATRIX

CALL PARTN ( \{MAAA, \{MRRBAR\}, \{MLR\}, \{KLL\}, \{PABL(BC)\}\} );

\{R13(BC, SUB)\} = \{MGL\} \* \{DBC\} + \{MLR\};

\{R3\} = \{MRRBAR\} + TRANS ( \{MLR\} \* \{DBC\} + \{MRRBAR\} );

\{R3\} = TRANS ( \{DBC\} ) \* \{R13(BC, SUB)\} ;

\{R22\} = TRANS ( \{DBC\} ) \* \{MLR\} + \{MRRBAR\};

CALL TRNSPOSE ( \{R13(BC, SUB)\}, \{R21(BC, SUB)\} );

END:

PROCESS STEADY AEROELASTIC DISCIPLINE

PRINT("LOG-\{'\"DINEL: STEADY AERO\'\}"));

CALL PARTN ( \{KAAA, \{KRRR, \{R12(BC, SUB)\}, \{KAKL, \{R11, \{PABL(BC)\}\}\}\}\}\};

\{R32(BC, SUB)\} = TRANS(\{DBC\}) \* \{R12(BC, SUB)\} + \{KAKL\};

\{R31(BC, SUB)\} = TRANS(\{DBC\}) \* \{R11\} + \{KAKL\};

CALL DECOMP ( \{R11, \{R111(BC, SUB)\}, \{R111(BC, SUB)\}\} );

CALL ROWPART ( \{PAAL, \{PARBAR, \{PAL\}, \{PABL(BC)\}\}\}\} );

CALL GBFS ( \{K111(BC, SUB)\}, \{K111(BC, SUB)\}, \{PAAL\}, \{PAL\}, \{R111(BC, SUB)\}, -1\} );

\{FRIGID\} = \{PARBAR\} + TRANS(\{DBC\}) \* \{PAL\};

\{F1\} = \{R11(BC, SUB)\} \* \{R111PAL(BC, SUB)\};

\{F2\} = \{FRIGID\} \* \{R31(BC, SUB)\} \* \{R111PAL(BC, SUB)\};

CALC GBFS ( \{R11(BC, SUB)\}, \{R111(BC, SUB)\}, \{R12(BC, SUB)\}, \{R1112(BC, SUB)\}, -1\} );

CALC GBFS ( \{R111(BC, SUB)\}, \{K111(BC, SUB)\}, \{R13(BC, SUB)\}, \{K1113(BC, SUB)\}, -1\} );

\{K11\} = \{R22\} + \{R21(BC, SUB)\} \* \{R1112(BC, SUB)\};

\{K113(BC, SUB)\} = \{R21(BC, SUB)\} \* \{R1113(BC, SUB)\};

\{K21(BC, SUB)\} = \{R32(BC, SUB)\} \* \{R31(BC, SUB)\} \* \{R1112(BC, SUB)\};

\{K22\} = \{R33\} \* \{R31(BC, SUB)\} \* \{R1113(BC, SUB)\};

CALL DECOMP ( \{K11, \{K111(BC, SUB)\}, \{K111(BC, SUB)\}\} );

CALL GBFS ( \{K111(BC, SUB)\}, \{K111(BC, SUB)\}, \{F1\}, \{F2\}, \{PAR(BC, SUB)\} );

CALL GBFS ( \{R111(BC, SUB)\}, \{K111(BC, SUB)\}, \{K12(BC, SUB)\}, \{K1112(BC, SUB)\}, -1\} );

\{LHSA(BC, SUB)\} = \{K22\} + \{K21(BC, SUB)\} \* \{K1112(BC, SUB)\};

\{RHSA(BC, SUB)\} = \{F2\} - \{K21(BC, SUB)\} \* \{PAR(BC, SUB)\};

CALL SAERO NOW ! ***********************************************

CALL SAERO ( \{MITER, \{MINDER, \{SUB, \{SYM, \{GDP, \{STARC\}, \{BGDT(BC)\}, \{LHSA(BC, SUB)\}, \{RHSA(BC, SUB)\}, \{KAAA\}, \{PRIGID\}, \{R3\}, \{CONST\}, \{ASLG(SUB)\}, \{AARC\}, \{DEL\}\}\} );

CALL SEGL ( \{AARC\} );

CALL ROWMERGE ( \{AAA(SUB)\}, \{AARC\}, \{AAL\}, \{PABL(BC)\} );

\{VAR\} = \{K1112(BC, SUB)\} \* \{AARC\} + \{PAR(BC, SUB)\} \* \{DEL\} ;

\{VAR\} = \{R1112(BC, SUB)\} \* \{VAR\} + \{R1113(BC, SUB)\} \* \{AARC\};

\{VAR\} = \{R111PAL(BC, SUB)\} \* \{DEL\} ;

\{VAR\} = \{R1112(BC, SUB)\} \* \{VAR\} + \{R1113(BC, SUB)\} \* \{AARC\} - \{R111PAL(BC, SUB)\} \* \{DEL\} ;

61
CALL RORMERGE ( [UAAC(SUB)], [UAR], [UAL], [FARL(BC)])
ENDIF;
ELSE
NO SUPPORT SET REDUCTION
ENDIF;
ENDDO;
[GENF] := (BQDP) [FTF] * [BFRC];
[GENFA] := (BQDP) [FTF] * [MATS];
[GENQ] := (GENFA) * [DTSFL];
[GENQL] := (BQDP) [FTF] * [MATT];
CALL PARTN ( [GENQ], [QPR], , [QRE], [QEE], [MPART] );
CALL PARTN ( [GENQ], , , [KEI], [MPART] );
[KEQE] := [QEE] + [KEE];
CALL DECONP ( [KEQE], [LQG], [UKQ] );
CALL ROWPART ( [GENQ], [GFR], [GFE], [STEM] );
CALL GFRS ( [LQG], [UKQ], [GFE], [STEM] );
[DELN] := - [QRE] * [STEM] * [GFR];
CALL BLASTR9 ( BC, [DELN], [MRR(BC)], [URDB], [DELB] );
[ELAS] := [STEM] * [DELB];
[SLPMD] := TRANS ( [SLPMD] ) * [PHIE];
CALL BLASTDRV ( BC, [GENM], [GENK], [GENFA], [GENQL], [DELB],
[URDB], [DNWNSH], [SLPMD], [ELAS], [UBLAST] );
ENDIF;

BEGIN THE DATA OPERATIONS
PRINT("LOG('DATA RECOVERY']);
IF NIP Nate > 1 CALL NULLMAT ([UF], [AF], [PHIF], [UTRANP], [UFREQP]);
IF NGDR <= 0 THEN
DATA RECOVERY WITH GDR
APEND THE GER-GENERATED DOFS TO THE F-SET
PRINT("LOG('DYNAMIC REDUCTION RECOVERY'));
IF BLOAD <= 0 THEN
[UGDR] := (GRBU(Bc)) * [UA];
CALL ROWPART ( [UA], [UJK], , [PAJK] );
CALL ROMERGE ( [UF], [UJK], [UGDR], [PFJK] );
IF NRSET <= 0 THEN
[AFGR] := (GRBU(Bc)) * [AA];
CALL ROWPART ( [AA], [UJK], , [PAJK] );
CALL ROMERGE ( [AF], [UJK], [AFGR], [PFJK] );
ENDIF;
END;

IF BSAERO <= 0 THEN
FOR S = 1 TO SUB DO
[UGDR] := (GRBU(Bc)) * [UA(S)];
CALL ROWPART ( [UA(S)], [UJK], , [PAJK] );
CALL ROMERGE ( [UAFMP], [UJK], [UGDR], [PFJK] );
MERGE THE CURRENT SUBCASE DEPENDENT RESULTS INTO A SINGLE
MATRIX OF RESPONSE QUANTITIES FOR FURTHER RECOVERY
END;

CALL SAEROMRG ( BC, S, [UAFMP] );
IF NRSET <= 0 THEN
[AFGR] := (GRBU(Bc)) * [AA(S)];
CALL ROWPART ( [AA(S)], [UJK], , [PAJK] );
CALL ROMERGE ( [AAFMP], [UJK], [AFGR], [PFJK] );
CALL SAEROMRG ( BC, S, [AAFMP] );
ENDIF;

IF REFLG(S) THEN
[UGDR] := (GRBU(Bc)) * [UAAC(S)];
CALL ROWPART ( [UAAC(S)], [UJK], , [PAJK] );
CALL ROMERGE ( [UAAC(S)], [UJK], [UGDR], [PFJK] );
[AFGR] := (GRBU(Bc)) * [AAC(S)];
CALL ROWPART ( [AAC(S)], [UJK], , [PAJK] );
CALL ROMERGE ( [AAC(S)], [UJK], [AFGR], [PFJK] );
ENDIF;
END;

ENDDO;
ENDIF:
IF REMODES <= 0 THEN
[UGDR] := (GRBU(Bc)) * [PHIA];
CALL ROWPART ( [PHIA], [UJK], , [PAJK] );
CALL ROMERGE ( [PHIF], [UJK], [UGDR], [PFJK] );
ENDIF;
IF BDTR <= 0 OR BMTR <= 0 THEN
[UGDR] := (GRBU(Bc)) * [UTRANA];
CALL ROWPART ( [UTRANA], [UJK], , [PAJK] );
CALL ROMERGE ( [UTRANA], [UJK], [UGDR], [PFJK] );
ENDIF:
IF BDTR <= 0 OR BMFR <= 0 THEN
[UGDR] := (GRBU(Bc)) * [UFREQA];
CALL ROWPART ( [UFREQA], [UJK], , [PAJK] );
CALL ROMERGE ( [UFREQA], [UJK], [UGDR], [PFJK] );
ENDIF:
ELSE
IF NOMP <= 0 THEN
DATA RECOVERY WITH STATIC CONDENSATION

PRINT("LOG-'1 STATIC CONDENSATION RECOVERY'");

IF BLOAD <> 0 THEN
  CALL RECOVA [UA], [PO], [GSUBO(BC)], NRSET, [AA],
    [JIFM(BC)], [KIOINV(BC)], [FPDA(BC)], [UF];
  IF NRSET <> 0 CALL RECOVA [AA], [GSUBO(BC)],
    [PFDA(BC)], [AF];
ENDIF;

IF BSASR <> 0 THEN
  FOR S = 1 TO SUB DO
    CALL RECOVA [UA(S)], [PAO(S)], [GSUBO(BC,S)],
      NRSET, [AA(S)], [JIFM(BC,S)], BSASR,
      [KIOI(BC,S)], [KIOU(BC,S)],
      [PFDA(BC)], [UF];
  ENDF;

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

IF AEFLG(S) THEN
  CALL RECOVA [UAAC(S)], [PAOC(S)], [GSUBO(BC,S)],
    NRSET, [AAAC(S)], [JIFMA(BC,S)], BSASR,
    [KIOI(BC,S)], [KIOU(BC,S)],
    [PFDA(BC)], [UF];
  CALL RECOVA [AAAC(S)], [GSUBO(BC,S)];
ENDIF;

ENDIF;

ENDIF;

ENDIF;

IF BMODES <> 0 THEN
  (PHIO) := [GSUBO(BC)] * [PHIA];
  CALL RONERGE [PHIF, PHIO], [PHIA], [PFDA(BC)];
ENDIF;

IF BMTR <> 0 OR BMTR <> 0 THEN
  CALL RECOVA [UTRANA], [GSUBO(BC)],
    [PFDA(BC)], [UF];
ENDIF;

IF BMFR <> 0 OR BMFR <> 0 THEN
  CALL RECOVA [UFREQ], [GSUBO(BC)],
    [PFDA(BC)], [UFREQ];
ENDIF;

ENDIF;

ENDDO;

DATA RECOVERY WITHOUT F-SET REDUCTION

IF BLOAD <> 0 THEN
  [UF] := [UA];
  IF NRSET <> 0 [AF] := [AA];
ENDIF;

IF BSASR <> 0 THEN
  FOR S = 1 TO SUB DO
    CALL RECOVA [UA(S)], [AA(S)];
  ENDF;

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

IF AEFLG(S) THEN
  (PHIO) := [UAAC(S)] * [PHIA];
  CALL RONERGE [PHIF, PHIO], [PHIA], [PFDA(BC)];
ENDIF;

IF BMTR <> 0 OR BMTR <> 0 THEN
  CALL RECOVA [UTRANA], [UAAC(S)],
    [PFDA(BC)], [UF];
ENDIF;

IF BMFR <> 0 OR BMFR <> 0 THEN
  CALL RECOVA [UFREQ], [UAAC(S)],
    [PFDA(BC)], [UFREQ];
ENDIF;

ENDIF;

ENDDO;

IF BMODES <> 0 THEN
  (PHIO) := [PHIF];
  CALL RONERGE [PHIF], [PHIA];
ENDIF;

IF BMTR <> 0 OR BMTR <> 0 THEN
  CALL RECOVA [UTRANA], [UF];
ENDIF;

IF BMFR <> 0 OR BMFR <> 0 THEN
  CALL RECOVA [UFREQ], [UF];
ENDIF;

ENDIF;

IF NUNOPTEC > 1 CALL NULLMAT [UN], [AN], [PHIF];

IF NSFC <> 0 THEN
  DATA RECOVERY WITH SPC-REDUCTION
ENDIF;
PRINT("LOG='", SPC_RECOVERY");

IF BLOAD <> 0 THEN
  CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)] );
  CALL OFFSCP ( NITER, BC, 1, 1, GSIZF, ESIZF(BC), NGDF,
               [KFS], [KSS], [UF], [YS(BC)], [PS],
               [PNSF(BC)], [PNSF(BC)], [PFS],
               BGFDT(BC), OGRIDLO); ENDIF;

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

ENDIF;

IF BSAERO <> 0 THEN
  CALL YSMERGE ( [UAN], [YS(BC)], [UAF], [PNSF(BC)] );
  IF NSSET <> 0 CALL YSMERGE ( [AAN], [AAF], [PNSF(BC)] );
  FOR S = 1 TO SUB DO
    IF AEFLG(S) THEN
      CALL YSMERGE ( [UANC(S)], [YS(BC)], [UAFCS(S)], [PNSF(BC)] );
      CALL YSMERGE ( [AANC(S)], [AAFC(S)], [PNSF(BC)] );
    ENDIF;
  ENDDO;
ENDIF;

IF BMODES <> 0 THEN
  CALL YSMERGE ( [PHIN], [YS(BC)], [PHIF],
               [PNSF(BC)] );
  IF BMODES <> 0 CALL OFFSCP ( NITER, BC, 2, 1, GSIZF,
                               ESIZF(BC), NGDF,
                               [KFS], [PHIF], [PNSF(BC)], [PNSF(BC)],
                               BGFDT(BC), OGRIDLO); ENDIF;

IF BDIR <> 0 OR BMFR <> 0
  CALL YSMERGE ( [UTRANN], [YS(BC)], [UTRANN],
               [PNSF(BC)], [PNSF(BC)]) ;
ENDIF;

IF BDIR <> 0 OR BMFR <> 0
  CALL YSMERGE ( [UFREQN], [YS(BC)], [UFREQN],
               [PNSF(BC)], [PNSF(BC)]) ;
ENDIF;

IF BBLAST <> 0 THEN
  [UBLASTF] := [PHIF]*[UBLASTF];
  CALL OFFSCP ( NITER, BC, 6, 1, GSIZF, ESIZF(BC), NGDF,
               [KFS], [UBLASTF], [PNSF(BC)], [PNSF(BC)],
               [PFS], [PFS], BGFDT(BC), OGRIDLO); ENDIF;

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

  IF BSAERO <> 0 THEN
    [UAN] := [UAF];
    IF NSSET <> 0 [AAN] := [AAM];
    FOR S = 1 TO SUB DO
      IF AEFLG(S) THEN
        [UANC(S)] := [UAFCS(S)];
        [AANC(S)] := [AAFC(S)];
      ENDIF;
    ENDDO;
ENDIF;

  IF BMODES <> 0 [PHIN] := [PHIF];
  IF BDIR <> 0 OR BMFR <> 0 [UTRANN] := [UTRANN];
  IF BDIR <> 0 OR BMFR <> 0 [UFREQN] := [UFREQN];
ENDIF;

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

IF NMC <> 0 THEN
  DATA RECOVERY WITH MPC-REDUCTION
  PRINT("LOG='", MPC_RECOVERY");
  IF BLOAD <> 0 THEN
    [UN] := [TMM(BC)] * [UN];
  CALL ROWMERGE ( [UG(BC)], [UM], [UN], [PNSF(BC)] );
  IF NSSET <> 0 THEN
    [UM] := [TMM(BC)] * [AN];
  CALL ROWMERGE ( [AG(BC)], [UM], [AN], [PNSF(BC)] );
  ENDIF;
ENDIF;

IF BSAERO <> 0 THEN
  [UM] := [TMM(BC)] * [UAN];

END
CALL ROWMERGE ( [UAG(BC)], [UM], [UAN], [POMN(BC)] );

1200 6I IF NSET <> 0 THEN
1201 7I [UM] := [TWM(BC)] * [AAN];
1202 7I CALL ROWMERGE ( [AAG(BC)], [UM], [AAN], [POMN(BC)] );
1203 7I ENDIF;
1204 6I FOR S = 1 TO SUB DO
1205 7I IF AETLG(S) THEN
1206 8I [UM] := [TWM(BC)] * [UANC(S)];
1207 8I CALL ROWMERGE ( [UAGG(BC,S)], [UM], [UANC(S)], [POMN(BC)] );
1208 8I [UM] := [TWM(BC)] * [AANC(S)];
1209 8I CALL ROWMERGE ( [AAGC(BC,S)], [UM], [AANC(S)], [POMN(BC)] );
1210 8I ENDIF;
1211 7I ENDDO;
1212 6I ENDIF;
1213 6I IF BMODS <> 0 THEN
1214 6I [UM] := [TWM(BC)] * [PHIN];
1215 6I CALL ROWMERGE ( [PHIG(BC)], [UM], [PHIN], [POMN(BC)] );
1216 6I ENDIF;
1217 5I IF BDTR <> 0 OR BMTR <> 0 THEN
1218 6I [UM] := [TWM(BC)] * [UTRANN];
1219 6I CALL ROWMERGE ( [UTRANG], [UM], [UTRANN], [POMN(BC)] );
1220 6I ENDIF;
1221 5I IF BDPR <> 0 OR BMFR <> 0 THEN
1222 6I [UM] := [TWM(BC)] * [UPREQH];
1223 6I CALL ROWMERGE ( [UPREQH], [UM], [UPREQH], [POMN(BC)] );
1224 6I ENDIF;
1225 5I ELSE
1226 5I    DATA RECOVERY WITHOUT NPO-REDUCTION
1227 5I    
1228 5I    IF BLOAD <> 0 THEN
1229 6I [UG(BC)] := [UM];
1230 6I IF NRSET <> 0 THEN [ASG(BC)] := [AN];
1231 6I ENDIF;
1232 5I IF BSAERO <> 0 THEN
1233 6I [UAG(BC)] := [UAN];
1234 6I IF NRSET <> 0 THEN [AAG(BC)] := [AAN];
1235 6I FOR S = 1 TO SUB DO
1236 7I IF AETLG(S) THEN
1237 8I [UANC(BC,S)] := [UANC(S)];
1238 8I [AANC(BC,S)] := [AANC(S)];
1239 8I ENDIF;
1240 7I ENDDO;
1241 6I ENDIF;
1242 5I IF BMODS <> 0 THEN [PHIG(BC)] := [PHIN];
1243 5I IF BDTR <> 0 OR BMTR <> 0 THEN [UTRANG] := [UTRANN];
1244 5I IF BDPR <> 0 OR BMFR <> 0 THEN [UPREQH] := [UPREQH];
1245 5I ENDIF;
1246 5I
t 4I RECOVER PHYSICAL BLAST DISCIPLINE DISPLACEMENTS
1247 4I
1248 4I IF BBLAST <> 0 THEN [UBLASTG] := [PHIG(BC)] * [UBLASTI];
1249 4I
1250 4I PERFORM CONSTRAINT EVALUATION FOR STATIC DISCIPLINES
1251 4I
1252 4I PRINT("LOG1(' CONTAIENMENT')");
1253 4I
1254 4I IF BLOAD <> 0 THEN
1255 5I CALL DCEVAL ( NITER, BC, [UG(BC)], CON );
1256 5I CALL SCEVAL ( NITER, BC, [UG(BC)], [SNAT], TREF, [GLBSIG], CON );
1257 5I ENDIF;
1258 4I IF BSAERO <> 0 THEN
1259 5I CALL DCEVAL ( NITER, BC, [UAGG(BC)], CON, BSAERO, [BSNAT], TREF, [GLBSIG], CON, BSAERO );
1260 5I CALL SCEVAL ( NITER, BC, [UAGG(BC)], [SNAT], TREF, [GLBSIG], CON, BSAERO );
1261 5I ENDIF;
1262 4I
1263 4I HANDLE OUTPUT REQUESTS
1264 4I
1265 4I PRINT("LOG2(' OUTPUT PROCESSING')");
1266 4I IF BSAERO <> 0 THEN
1267 4I RECOVER STATIC AERODYNAMIC LOADS DATA
1268 4I
1269 4I LOOP := TRUE;
1270 5I SUB := 0;
1271 5I WHILE LOOP DO
1272 6I SUB := SUB + 1;
1273 6I CALL SAREDRV ( BC, SUB, LOOP, MINDX, SYM, MACH, QDP );
1274 6I CALL THE TRIMMED LOADS COMPUTATION WITH PROPER MATRICES
1275 6I
1276 6I
1277 6I
1278 6I
1279 6I

66
IF SYM = 1 THEN

CALL OFFLOAD ( NITER, BC, MIDDLEX, SUB, GSIE, BPDTC(BC),
               [GETK], [GETK], QDF, [AIRFRC(MIDDLEX)],
               [DELTA(SUB)], [AIRFCT(MIDDLEX)],
               [UAG(BC)], [ESIE(B)], [FSF],
               [KSS], [UAF], [YS(BC)], [FPSF(BC)],
               [PONM(BC)], [PFFK], NGDR, USET(BC),
               ORIDLOD );

ENDIF;

ENDIF;

ENDF;
CALL OFFLOAD ( NUMOPTBC, BC, NITER, GSIZE, BGPDT(BC), PSIЀE(BC), [PG]);
CALL OPDIFS ( NUMOPTBC, BC, NITER, GSIZE, BGPDT(BC), ESIZE(BC),
                PSIЀE(BC), OGRIDDSP, [UG(BC)], [AG(BC)], [UAG(BC)],
                [UAG(BC)], [UG(BC)], [UG(BC)], [UG(BC)], [UG(BC)], [UG(BC)]);
CALL EDTR ( NUMOPTBC, BC, NITER, NDV, GSIZE, EPRSHRY, EODTSC,
           GLEDES, LOCVAR, [PTRANS],
           [UG(BC)], [UG(BC)], [UG(BC)], [UG(BC)], [UG(BC)]);
CALL EPSLEVAL ( BC, NITER, NDV, GLEDES, LOCVAR, [PTRANS], PDIST,
              OPNSBUCK );
CALL EKLEVAL ( BC, NITER, NDV, GLEDES, LOCVAR, [PTRANS], OEULBUCK );
CALL OFFRED ( BC, HSIZE(BC), NITER );
ENDDO;

SELECT ACTIVE CONSTRAINTS
PRINT("LOG=" )
CALL ACTCON ( NITER, MAXITER, NREFAC, NDV, GLEDES, LOCVAR, [PTRANS],
              EPS, AFPCNVRG, GLBCNVRG,
              CTL, CTIMIN, CONST, [ANAT], DESHIST, PFLAG, OLOCALDV );
CALL DESPUNCH ( NITER, PFLAG, OLOCALDV );

IF GLBCNVRG OR NITER > MAXITER THEN
LAST ITERATION OUTPUT
FOR BC = 1 TO NUMOPTBC DO
CALL OPMBOOT ( NITER, BC, NUMOPTBC, LAMBD., 1 )
CALL OFFDIFS ( NUMOPTBC, BC, NITER, GSIZE, BGPDT(BC), ESIZE(BC),
               PSIЀE(BC), OGRIDDSP,,,,,LAMBD., 1 )
CALL OFFRED ( BC, HSIZE(BC), NITER, 1 );
ENDDO;
ENDIF;

IF NOT GLBCNVRG AND NITER < MAXITER THEN
USE APPROPRIATE RESIZING METHOD
IF NITER >= PSDE AND NITER < PSDE THEN
CALL PSDE ( NDV, NITER, PSDE, MFS, OCS, ALPHA,
           CNVRLIM, GLEDES, LOCVAR, [PTRANS], CONST,
           AFPCNVRG, CTL, CTIMIN, DESHIST );
ENDIF;
IF ( NITER >= MPS AND NITER < MPE ) OR
 ( NITER >= OCS AND NITER < OCE ) THEN
USE MATHEMATICAL PROGRAMMING OR OC METHODS
OBTAIN THE SENSITIVITIES OF THE CONSTRAINTS WRT THE
DESIGN VARIABLES
CALL MAKDEV ( NITER, NDV, [PMINT], [PMAXT], CONST, [ANAT] );
CALL LAMINSN ( NITER, NDV, GLEDES, LOCVAR, [PTRANS], CONST,
               [ANAT] );
SENSITIVITY EVALUATION FOR BOUNDARY CONDITION DEPENDENT CONSTRAINTS
FOR BC = 1 TO NUMOPTBC DO
CALL ABOUND ( NITER, BC, CONST, ACTBOUND, MAUS, NACSD, [PGAS],
              PCAS, ACTAERO, ACTDYN, ACTFLX, ACTSL, ACCTBR,
              NMPRC, NSPC, NONIT, NRESST, NGDV, 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.
THE INCOMPATIBILITY WILL NOT AFFECT THE SENSITIVITY ANALYSIS
AND WILL BE CORRECTED IN THE SUBSEQUENT ANALYSIS
IF NGDV <> 0 THEN
CALL MGSKR(BC, GSIZER, [YS(BC)], [YN(BC)], [PMGN(BC)],
           [PNSF(BC)], [PFOS(BC)], [PFLR(BC)], USET(BC) );
ENDIF:

Evaluate frequency constraint sensitivities

IF ACTDYN THEN

IF NGDR <> 0 THEN

CALL ROWPART ([PHIG(BC)], [GTMP], [PDGR(CB)]);

CALL FREQSNS (NITER, BC, NDV, GLEDES, CONST, LAMBDA,

GMECT, DKVI, GMECT, DKVI,

[GTMP], [AMAT]);

ELSE

CALL FREQSNS (NITER, BC, NDV, GLEDES, CONST, LAMBDA,

GMECT, DKVI, GMECT, DKVI,

[PHIG(BC)], [AMAT]);

ENDIF:

ENDIF:

Evaluate flutter constraint sensitivities

IF ACTFLUT THEN

SUB := 0;

IF NGDR <> 0 CALL ROWPART ([PHIG(BC)], [GTMP], [PDGR(CB)]);

WHILE LOOP DO

SUB := SUB + 1;

IF NGDR <> 0 THEN

CALL PLUSNS (NITER, BC, SUB, LOOP, GSIZER, NDV,

GLEDES, CONST, GMECT, DKVI, GMECT,

DKVI, CLANMB, LAMBDAB,

[GMFLH(BC,SUB)], [BMFLH(BC,SUB)],

[KKFLH(BC,SUB)], [AMAT],

AEROZ);

ELSE

CALL PLUSNS (NITER, BC, SUB, LOOP, GSIZER, NDV,

GLEDES, CONST, GMECT, DKVI, GMECT,

DKVI, CLANMB, LAMBDAB,

[GMFLH(BC,SUB)], [BMFLH(BC,SUB)],

[KKFLH(BC,SUB)], [PHIG(BC)], [AMAT],

AEROZ);

ENDIF:

ENDDO:

ENDIF:

Evaluate active displacement dependent constraints from the statics discipline

IF NAUS > 0 THEN

SENSITIVITIES OF CONSTRAINTS WRT DISPLACEMENTS FOR STATIC:

CALL NULLMAT ([DFDU], [DPGV]);

IF MAXS > NAUS * NDV THEN

USE GRADIENT METHOD

CALL MAKDFU (NITER, BC, GSIZER, [AMAT], [GLBSIG],

CONST, [DFDU]);

ELSE

USE VIRTUAL LOAD METHOD

CALL MAKDFU (NITER, BC, GSIZER, [AMAT], [GLBSIG],

CONST, [DPGV]);

ENDIF:

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

IF NGDR <> 0 THEN

CALL PARTN ([UG(BC)], [UGA], [PGAS], [PDGR(CB)]);

ELSE

CALL COLPART ([UG(BC)], [UGA], [PGAS]);

ENDIF:

Obtain the sensitivities of the design dependent loads

CALL DLLOAD(NDV, GSIZER, BC, EMGLOD, DDFLG, [PGAS], [DPV]);
CALL MAKDVF ( NITERS, NDVF, GLBDER, [UGA], [DUG],
        GMKCT, DGMV ) ;
CALL NULLMAT ( [DUG] ) ;
IF NRSET <> 0 THEN
  IF MGMR <> 0 THEN
    CALL PARTN ( [AG(BC)], ..., [AGA], [PGAS], [PGDRG(BC)] ) ;
  ELSE
    CALL COLPART ( [AG(BC)], ..., [AGA], [PGAS] ) ;
  ENDIF;
  CALL MAKDVF ( NITERS, NDVF, GLBDER, [AGA], [DMAG],
        GMKCT, DGMV ) ;
  [DUG] := [DUG] + [DMAG] ;
ELSE
  [DUG] := [DUG] ;
ENDIF;
ACCOUNT FOR VIRTUAL LOAD METHOD
IF NACSD > NAUS * NDVF THEN
  USE GRADIENT METHOD
  IF DDFLG > 0 THEN
    [DPGV] := [DPVJ] + [DUG] ;
  ELSE
    [DPGV] := [DUG] ;
  ENDIF;
ELSE
  USE VIRTUAL LOAD METHOD
  IF DDFLG > 0 THEN
    [DPFU] := [DPVJ] + [DUG] ;
  ELSE
    [DPFU] := [DUG] ;
  ENDIF;
ENDIF;
REDUCE THE RIGHT HAND SIDES TO THE L SET
CALL NULLMAT ( [DPVJ], [DMIN] ) ;
IF NMPC <> 0 THEN
  CALL GREduce ( [DPGV], [PGNS(BC)], [TMNS(BC)], [DPVJ] ) ;
ELSE
  [DPGV] := [DPGV] ;
ENDIF;
CALL NULLMAT ( [DPVJ], [DMUF] ) ;
IF NSPC <> 0 THEN
  CALL NREDuce ( [DPVJ], [PSNS(BC)], ..., [DPVJ] ) ;
ELSE
  [DPVJ] := [DPVJ] ;
ENDIF;
CALL NULLMAT ( [DPAV], [DMUA] ) ;
IF MGDR <> 0 THEN
  [DPAV] := TRANS ( [GSUBO(BC)] ) * [DPVJ] ;
ELSE
  IF NOMIT <> 0 THEN
    CALL PREDUCE ( [DPVJ], [PGOAS(BC)], ..., [KPIOINV(BC)], ..., [GSUBO(BC)], ...
    [DPAV], [DPVJ] ) ;
  ELSE
    [DPAV] := [DPAV] ;
  ENDIF;
ENDIF;
IF NRSET <> 0 THEN
  CALL ROWPART ( [DPAV], [DPVJ], [DPFLV], [PARLS(BC)] ) ;
  [DRLHS] := TRANS ( [D(BC)] ) * [DPFLV] + [DPVJ] ;
PROCESS ACTIVE CONSTRAINTS FOR STATIC DISCIPLINE
CALL INERTIA ( [MRR(BC)], [DRRHS], [DURD] ) ;
[DUFLD] := [D(BC)] * [DURD] ;
CALL COMMSIZE ( [DUAL], [DURD], [DULD], [PARLS(BC)] ) ;
[DPFLV] := [DPFLV] + [IFR(BC)] * [DURD] ;
CALL FBS ( [KLLINV(BC)], [DPFLV], [DULV] ) ;
CALL YMERGE ( [DUAL], [DULV], [PARLS(BC)] ) ;
ELSE
  CALL FBS ( [KLLINV(BC)], [DPAV], [DUAV] ) ;
ENDIF;
70
ENDIF;

RECOVER TO THE F SET

CALL NULLMAT ( [DFUV] );

IF NMDR <> 0 THEN

[DFUV] := [GSUBO(BC)] * [DUAV];

ELSE

IF NORMT <> 0 THEN

IF NRESET <> 0 THEN

[TMP1] := [DPOV] - [IFM(BC)] * [DUAD];

ELSE

[TMP1] := [DPOV];

ENDIF;

ENDIF;

CALL FBS ( [KCOINV(BC)], [TMP1], [UOO] );

[UO] := [GSUBO(BC)] * [DUAV] + [UOO];

CALL ROWMERGE ([DFUV], [UO], [DUAV], [PFOAS(BC)]);

ELSE

[DFUV] := [DUAV];

ENDIF;

ENDIF;

REDUCE THE LEFT HAND SIDE MATRIX

IF NNMR <> 0 THEN

CALL GREDUCE ([DFDU], [PGNS(BC)], [THE(BC)], [DFDUN]);

ELSE

[DFDUN] := [DFDU];

ENDIF;

ENDIF;

IF NSRC <> 0 THEN

CALL RNAPRT ( [DFDUN], [DFDUF], [RNSPS(BC)] );

ELSE

[DFDUF] := [DFDUN];

ENDIF;

ACCOUNT FOR VIRTUAL LOAD METHOD

IF NACSND > NAMS * NVO THEN

USE GRADIENT METHOD

CALL MKAMAT ([AMAT], [DFDU], [DFUV], PCAS, [PAGAS]);

ELSE

USE VIRTUAL LOAD METHOD

CALL MKAMAT ([AMAT], [DFUV], [DFDUF], PCAS, [PAGAS]);

ENDIF;

ENDIF; $ END IF IF ON ACTIVE APPLIED STATIC LOADS

EVALUATE ACTIVE CONSTRAINTS FROM

THE STATIC AERODYNAMICS DISCIPLINE

IF ACTASER THEN

LOOP := TRUE;

ACTUAG := FALSE;

SUB := 0;

CALL NULLMAT ( [DFUV] );

WHILE LOOP DO

SUB := SUB + 1;

CALL ARONSNDR ( NITER, BC, SUB, LOOP, MINDX, CONST, SIM, NGRD,

[PGDRC(BC)], [UAG(BC)], [AA(BC)],

ACTUAG, [UGA], [PGA], [PAGA],

PCAA, [UAGC(BC, SUB)], [AAGC(BC, SUB)],

ACTAEFF, [AUAGC], [AAAGC], PCAS );

IF ACTAEFF THEN

PROCESS PSEUDO DISPLACEMENTS FOR EFFECTIVENESS

CONSTRAINTS

CALL MAKDVO ( NITER, NVO, GLBDES, [AUAGC], [DKUG],

GMCT, DVVI );

IF NRESET <> 0 THEN

CALL MAKDVO ( NITER, NVO, GLBDES, [AAAGC], [IMAG],

GMCT, DMVI );

[DPGV] := [DKUG] + [IMAG];

CALL MAKDVO ( NITER, NVO, GLBDES, [AUAGC], [DMUG],

GMCT, DMVI );

71
ELSE
    [DPGV] := [DXUG];
ENDIF;

REDUCE THE RIGHT HAND SIDES TO THE L SET

CALL NULLMAT ( [DPGV], [DMUN] );

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

CALL NULLMAT ( [DPFV], [DMUF] );

IF NSFC <= 0 THEN
    CALL NREDUCE ( [DPFV], [PRNSFS(BC)], ..., [DPFV] );
    IF NRSET <= 0
        CALL NREDUCE ( [DMUN], [PRNSFS(BC)], ..., [DMUF] );
    ELSE
        [DPFV] := [DPGV];
        IF NRSET <= 0 [DMUF] := [DMUN];
    ENDIF;
ENDIF;

CALL NULLMAT ( [DDPA], [DMUA] );

IF NSFC <= 0 THEN
    [DPAP] := TRANS( [GSUB(BC)] ) * [DDPA];
    IF NRSET <= 0 [DMUA] := TRANS( [GSUB(BC)] ) * [DMUF];
ELSE
    IF NRSF <= 0 THEN
        CALL REDUCE ( [DDPA], [FGOS(BC)], 1,
                      [KOOD(BC, SUB)], [KOOU(BC, SUB)],
                      [KAO(BC, SUB)], [GASUB(BC, SUB)],
                      [DPAP], [DDPO] );
        IF NRSET <= 0
            CALL REDUCE ( [DMUA], [FGOS(BC)], 1,
                            [KOOL(BC, SUB)], [KOOU(BC, SUB)],
                            [KAO(BC, SUB)], [GASUB(BC, SUB)],
                            [DMUA], [DMUF] );
        ENDIF;
    ELSE
        [DPAP] := [DPAP];
        IF NRSET <= 0 [DMUA] := [DMUA];
    ENDIF;
ENDIF;

IF NRSET <= 0 THEN
    CALL ROWPART ( [DPAP], [DDPA], [DDL], [PARLS(BC)] );
    CALL ROWPART ( [DMUA], [DMUR], [DMUL], [PARLS(BC)] );
    CALL DBS ( [RK11(BC, SUB)], [RK11(BC, SUB)],
             [DDPA], [DL1DDL] );
    [DP1] := TRANS( [D(B)] ) * [DMUL] + [DMUR] -
             [RK11(BC, SUB)] * [DL1DDL];
    [DRHS] := TRANS( [D(B)] ) * [DPRE] + [DPVR] -
             [RK31(BC, SUB)] * [DL1DDL];
ENDIF;

PROCESS ACTIVE CONSTRAINTS FOR SAERO DISCIPLINE

CALL DBS ( [RK11(BC, SUB)], [RK11(BC, SUB)],
           [DP1], [DKIV] );
[DRHS] := [DRHS] - [RK11(BC, SUB)] * [DKIV];

CALL DECOMP ( [LMSPA(BC, SUB)], [LMSL], [LNUS] );
CALL DBS ( [LMSL], [LNUS], [DRDS], [DUG] );
[DUIR] := [DKIV] + [RK112(BC, SUB)] * [DUG];
[DUUL] := [DL1DDL] + [RK112(BC, SUB)] * [DUIR];
[EFFSENS] := - [RK31(BC, SUB)] * [DUUL] -
            [RK32(BC, SUB)] * [DUUR];

CALL AEROEFS ( MITER, BC, SUB, SYM, NOV, CONST, FCAE, EFFSENS, [AMU] );
ELSE
    NOTE THAT SAERO W/O SUPPORT IS NOT SUPPORTED
ENDIF;
ENDIF: $ END IF ON ACTAEFF
PROCESS ACTIVE CONSTRAINTS FOR SAERO DISCIPLINE

CALL GFB6 ( [K11 (BC,SUB)], [KU11 (BC,SUB)],
[DF1],
[DK11]);

[DRHS] := [DRHS] - [K11 (BC,SUB)] * [DK11];

CALL AERONSEN ( NITER, BC, MINDEX, SUB, CONST, SIM, NOV,
BOP, (BC,SUB), STABCF, [PGAA],
[LHSA (BC,SUB)], [RHS (BC,SUB)],
[DRHS], [AAR], [DEDELV], [AMAT]);

[DURV] := [K112 (BC, SUB)] * [AAR] +
[PAR (BC, SUB)] * [DEDELV] + [DK11];

[DULV] := [R112 (BC, SUB)] * [DUAV] +
[R113 (BC, SUB)] * [AAR] +
[R11PAL (BC, SUB)] * [DEDELV] + [R11DPL];

CALL ROWMERGE ([DUAV], [DUAV], [DUAV], [PARS (BC)]);
ELSE
NOTE THAT SAERO W/O SUPPORT IS NOT SUPPORTED
ENDIF;

ENDIF;

CONTINUE SENSITIVITIES TO THE F SET

CALL NULLMAT ( [UATMP]);
IF NOGDR <> 0 THEN
[UATMP] := [GASUBO (BC, SUB)] * [DUAV];
ELSE
IF NOMIT <> 0 THEN
IF MRSET <> 0 THEN
[TMP1] := [DFOV] + [PAR (BC, SUB)] * [DEDELV];
ELSE
[TMP1] := [DFDV];
ENDIF;
ELSE
CALL GFB6 ( [K112 (BC, SUB)], [KU112 (BC, SUB)],
[TMP1], [UOOG]);
[UF] := [GASUBO (BC, SUB)] * [DUAV] + [UOOG];
CALL ROWMERGE ([UATMP], [UF], [DUAV],
[PARS (BC)]);
ELSE
[UATMP] := [DUAV];
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
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 NMSC <> 0 THEN
CALL GREDOLE ( [DFDUN], [PMS (BC)], [TMN (BC)],
[DFDUN]);
ELSE
[DFDUN] := [DFDU];
ENDIF;

CALL NULLMAT ( [DFDU]);
IF NSPC <> 0 THEN
CALL ROWPART ([DFDUN], [DFDU], [PMS (BC)]);
ELSE
[DFDU] := [DFDUN];
ENDIF;

TAKE MERGED SENSITIVITIES OF DISPLACEMENTS AND
COMPUTE THE AMAT MATRIX TERMS FOR THE SAERO
CONSTRAINTS:

CALL MXAMAT ([AMAT], [DFDUF], [DFDU], PCAA, [PGAA]);
ENDIF; $ END IF ON ANY ACTIVE DISPLACEMENTS
ENDIF; $ END IF ON ACTIVE AEREOELASTIC CONSTRAINTS
EVALUATE PANEL BUCKLING CONSTRAINT SENSITIVITIES

74
IF ACTPNS THEN
   CALL PKLSENS (BC, NITER, NDV, GLBDSE, LOCLVAR, [TRANS],
                  POLIST );
ENDIF;
IF ACTBAR THEN
   CALL EKBSSENS ( BC, NITER, NDV, GLBDSE, LOCLVAR, [TRANS]);
ENDIF;
ENDIF; $ END IF ON ACTIVE BOUNDARY CONDITION
ENDO;$ END DO ON ACTIVE BOUNDARY CONDITIONS
CALL OFRGRAD ( NITER, NUMOPTBC, [AMAT], GLBDSE, CONST, GRADIENT );
IF NITER > NCS AND NITER < OCS THEN
   PRINT("LOG=" 'VANGO MODULE');
   CALL VANGO (NITER, NDV, APPCNVRG, MOVML, CHVRGLM,
                CTI, CTMIN, NUMOPTBC, GLBDSE, CONST, [AMAT],
                DESHIST );
ELSE
   IF NITER > MPS AND NITER < MPE THEN
      PRINT("LOG=" 'DESIGN MODULE');
      CALL DESIGN ( NITER, NDV, APPCNVRG, MOVML, CHVRGLM,
                    CTI, CTMIN, NUMOPTBC, GLBDSE, CONST, [AMAT],
                    DESHIST );
ENDIF;
ENDIF; $ END IF ON FSD METHOD
ENDIF; $ END IF ON TEST AFTER ACTCOM
ENDIF; $ END WHILE LOOP FOR GLOBAL CONVERGENCE
ENDIF; $ END IF ON OPTIMIZATION
BEGIN FINAL ANALYSIS LOOP
BEGINFINALANALYSISLOOP
BEGIN THE GLOBAL MATRICES
PRINT("LOG=" 'BEGIN FINAL ANALYSIS');
CALL ANALINIT;
CALL EMA2 ( , NDV, GSIZE, GLBDSE, GMCT, DKVI, [KGG],
          GMCT, DKVI, [MGG]);
FOR BC - NUMOPTBC + 1 TO NBDCOND DO
   PRINT("LOG=" 'BOUNDARY CONDITION ', I3', BC');
   ESTABLISH THE BASE SET AND PARTITIONING DATA FOR THE BC
   CALL MKUSET ( BC, GSIZE,YS(BC), TMM(BC), [POM(BC)], [PSN(BC)],
                 [PAO(BC)], [FARL(BC)], USET(BC) );
   MAKE B.C.-DEPENDENT BGPDT FROM BASE, ADDING THE EXTRA POINTS FOR
   THIS B.C.
   CALL CBGPDT ( BC, GSIZE, BGPDT(BC), ESIZE(BC) );
   PSIZE(BC) := ESIZE(BC) + GSIZE;
   PROCESS MATRICES, TRANSFER FUNCTIONS, AND INITIAL CONDITIONS FOR
   THIS B.C.
   CALL CBULK ( BC, PSIZE(BC), BGPDT(BC), USET(BC) );
   CALL BOUND ( BC, GSIZE, ESIZE(BC), USET(BC), BLOAD, BMASS, DMODES,
                EMODES, BSAERO, BFLUTR, BODIN, BDRS, BDR, BMTR, BDFP,
                BMTR, BSQST, BPPM, MNC, NSNC, NNTMT, NUMD, NGDR );
Determine if any M2GG/K2GG input data are to be added
CALL NULLMAT ( [KGG], [MGG] );
CALL MKGG ( BC, GSIZE, [MGG], M2GGFLAG, [KGG], K2GGFLAG );
IF M2GGFLAG THEN
   [MGG] := [MGG] + [MGG];
ELSE
   END;
ENDIF;
IF K2GOFLAG THEN 1
  [KGG] := [K1GG] + [K2GO]; 1
ELSE 1
  [KGG] := [K1GG]; 1
ENDIF; 1
CALL GRID POINT WEIGHT GENERATOR FOR THIS BOUNDARY CONDITION; 1
CALL GPWG (, BC, GPMGRID, [KGG], OSPWG ); 1
IF BLOAD <> 0 CALL GTLOAD (, BC, GSIZE, BSGD(BC), GLBDES, 1
  SMPLOAD, [GPWV], [GSPAN], [PG], OGRIDLOD); 1
PARTITION-REDUCTION OF GLOBAL MATRICES 1
IF NFMPC <> 0 THEN 1
PERFORM MPC REDUCTION 1
PRINT("LOG='MPC REDUCTION'"); 1
CALL GREDUCE ( [KGG], [PG], [PGMN(BC)], [TMN(BC)], [KNN], [PH]); 1
IF BMASS <> 0 CALL GREDUCE ([MGG], [PFGMN(BC)], [TM(BC)], [MNN]); 1
ENDIF; 1
IF BNEAERO <> 0 THEN 1
CALL GREDUCE (, [GIGNK], [PFGMN(BC)], [TM(BC)], , [GTMN]); 1
CALL GREDUCE (, [GIGNK], [PFGMN(BC)], [TM(BC)], , [GTMN]); 1
ENDIF; 1
IF BFLU < 0 OR BLOAD < 0 OR BNEAERO <> 0 THEN 1
CALL GREDUCE ([UGTKG], [PFGMN(BC)], [TM(BC)], , [UGTMN]); 1
ELSE 1
NO MPC REDUCTION 1
IF BLOAD <> 0 [PH] := [PG]; 1
IF BMASS <> 0 [MNN] := [MGG]; 1
ENDIF; 1
IF BNEAERO <> 0 THEN 1
[GTMN] := [GTIGNK]; 1
ENDIF; 1
IF BFLU < 0 OR BLOAD < 0 OR BNEAERO <> 0 THEN 1
[UFGMN] := [UGTMN]; 1
ENDIF; 1
PERFORM AUTOSPC CALCULATIONS ON THE EIGN MATRIX 1
PRINT("LOG='AUTOSPC COMPUTATIONS'"); 1
CALL GSPF (, BC, NDIR, [KNN], BSGD(BC), [YS(BC)], USET(BC), 1
  GPST(BC) ); 1
CALL MKFVECT ( USET(BC), [PFGMN(BC)], [PNSF(BC)], [PFOA(BC)], [PARL(BC)]: 1
  CALL BOUNDFUP ( BC, GSIZ, ESIZE(BC), USET(BC), NISP, NRO, NRSET ); 1
ENDIF; 1
IF NFMPC <> 0 THEN 1
PERFORM SPC REDUCTION 1
PRINT("LOG='SPC REDUCTION'"); 1
CALL NREDUCE ( [KNN], [PG], [PNSF(BC)], [YS(BC)], [KFF], [KFS], 1
  [KSS], [PF], [PS] ); 1
IF BMASS <> 0 CALL NREDUCE ( [MNN], [PNSF(BC)], [MFF] ); 1
ENDIF; 1
IF BNEAERO <> 0 THEN 1
CALL NREDUCE ( , [GIGNK], [PNSF(BC)], , , [GTMN] ); 1
CALL NREDUCE ( , [GIGNK], [PNSF(BC)], , , [GTMN] ); 1
ENDIF; 1
IF BFLU < 0 OR BLOAD < 0 OR BNEAERO <> 0 THEN 1
CALL NREDUCE (, [UGTKG], [PNSF(BC)], , , [UGTKF]); 1
ENDIF; 1
ENDIF; 1
ELSE 

NO SPC REDUCTION

[KFF] := [KNN];

IF BLOAD <> 0 [PF] := [PH];

IF BMASS <> 0 [MFF] := [MNN];

******************************************************* TAKEN OUT FOR SAERO ***************************************

IF SAERO <> 0 THEN

[GEKFT] := [GEKTN];

[GSTK] := [GSTKN];

ENDIF;

*******************************************************

IF NBLUTR <> 0 OR BGIUST <> 0 OR BLAST <> 0 OR SAERO <> 0

[UGKFT] := [UGTSH];

ENDIF;

IF NBNOCOND > 1 CALL NULLMAT ([KAA], [PA], [MAA], [KAAA], [PAA], [UGTKA]);

IF NGDR <> 0 THEN

PERFORM THE GENERAL DYNAMIC REDUCTION WHICH IS DISCIPLINE INDEPENDENT. THE RESULTING [GSUBO] MATRIX WILL BE USED BY ALL DISCIPLINES

PRINT("LOG=‘

DYNAMIC REDUCTION’");

OBTAIN THE OMITTED DOF PARTITION OF KFF AND MFF

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

CALL PARTH ([MFF], [MOO], [MFOA(BC)]);

ASIZE := GSIZE - NMPC - NSPC - NOMIT;

LSIZE := ASIZE - NSSET;

CALL GDR1 ([KOO], [KSOO], [GGO], LKSET, LSET, NEIV, FMAX, BC, BGPDF(BC), USET(BC), NOMIT, LSIZE);

LKSET

<& 0

APPROX. MODE SHAPES SELECTED

<& 0

NO APPROX. MODE SHAPES IN GDR

IF LKSET <> 0 THEN

CALL SDCOMP ([KSOO], [LSOO], USET(BC), SIMGOSET);

CALL GDR2 ([LSOO], [MOO], [PHIOK], LKSET, LSET, NEIV, FMAX, BC);

ENDIF;

CALL GDR3 ([KSOO], [KAOA], [PHIOK], [TMN(BC)], [GGO], [PFOA(BC)], [PSNF(BC)], [PFOA(BC)], [GSUBO(BC)],

BGPDF(BC), USET(BC),

LSET, LSET, ASIZE, GNORM, BC);

CALL GDR4 ([BC, GSIZE, PFSIZE(BC), LKSET, LSET, NUMOFBC, NBNOCOND, [PFOA(BC)], [PHIOK], [PSNF(BC)], [PFOA(BC)],

[PARL(BC)], [PSGRD(BC)], [PAJK], [PFJK], BGPDF(BC),

USET(BC)]);

ENDIF;

IF BLOAD <> 0 OR BMODES <> 0 OR BBLUTR <> 0 OR BGUIST <> 0 THEN

REDUCE THE MATRICES WITHOUT AEREOELASTIC CORRECTIONS

IF NGDR <> 0 THEN

PERFORM THE GENERAL DYNAMIC REDUCTION

PRINT("LOG=‘

SYMmetric DYNAMIC REDUCTION’");

[MAA] := TRANS ([GSUBO(BC)] *[MFF] *[GSUBO(BC)]);

[KAA] := TRANS ([GSUBO(BC)] *[KFF] *[GSUBO(BC)]);

IF BLOAD <> 0 [PA] := TRANS ([GSUBO(BC)] *[PF]);

IF BBLUTR <> 0 OR BGUIST <> 0 OR BMASS <> 0 THEN

[TMP1] := TRANS ([UGKFT] *[GSUBO(BC)]);

CALL TRANSPOSE ([TMP1], [UGTKA]);

ENDIF;

ELSE

IF NOMIT <> 0 THEN

PERFORM THE STATIC REDUCTION

PRINT("LOG=‘

STATIC CONDENSATION’");

CALL REDUCE ([KFF], [PF], [PFOA(BC)], [KOOINV(BC)],

[GSUBO(BC)], [KAA], [PA], [PS], USET(BC)];

ENDIF;

ENDIF;

77
IF BMSS <> 0 THEN
    PERFORM GUYAN REDUCTION OF THE MASS MATRIX

CALL PARMA ( [MFP], [MDO], [MAABAR], [FFPA(BC)] );

[MAABAR] := [MDOA] + TRANS([MDO]) * [GSUBO(BC)] +
TRANS([GSSUBO(BC)]) * [MDO] +
TRANS([GSSUBO(BC)]) * [MDO] * [GSUBO(BC)];

IF NRSET <> 0 [IPM(BC)] := [MDO] + [GSSUBO(BC)] + [MDO];
ENDIF;

IF BFLTR < 0 OR BUST < 0 OR BFLBAST <> 0 THEN
CALL ROWPARA ( [USTRF], [UGTRK], [UGTKAB], [FFPA(BC)] );

[TMP1] := TRANS([UGTRK]) * [GSSUBO(BC)];

CALL TRANSPOSE ([TMP1], [TMP2]);

[UGTKA] := [UGTKAB] + [TMP2];
ENDIF;

ELSE

NO F-SET REDUCTION

[KAA] := [KTF];

IF BLOAD <> 0 [PR] := [PF];

IF BFLTR < 0 OR BUST < 0 OR BFLBAST <> 0 [UGTKA] := [UGTRF];

IF BMSS <> 0 [MAF] := [MFF];

ENDIF;

IF NRSET <> 0 THEN
PERFORM THE SUPPORT SET REDUCTION

PRINT("LOG="
      " SUPPORT REDUCTION");

CALL PARMA ( [KAA], [KRR], [KLR], [KLL], [PARL(BC)] );

CALL SDCMP ( [KLL], [KLLINV(BC)], USET(BC), SINGSET );

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

CALL RCHECK ( BC, USET(BC), BOPD(BC), DBC, [KLL],

[KRR], [KLR] );

ENDIF;

ENDIF;

ENDIF;

ENDIF;

BEGIN

CALL PARMA ( [MAA], [MRRBAR], [MLR], [MLL], [PARL(BC)] );

[IFR(BC)] := [MLL] * [DFBC] + [MLR];

[MRB(BC)] := [MRRBAR] + TRANS([MLR]) * [DFBC] +
TRANS([DFBC]) * [IFR(BC)];

[MRB] := TRANS([DFBC]) * [MLR] + [MRRBAR];

ENDIF;

IF BLOAD <> 0 THEN

BEGIN

PROCEDURE STATICS WITH INERTIA RELIEF

PRINT("LOG="
      " >>>DISCIPLINE: STATICS(INERTIA RELIEF");

CALL ROWPARA ( [PA], [PR], [PLBAR], [PARL(BC)] );

[LSB(BC)] := [MRR(BC)];

[RSB(BC)] := TRANS([DFBC]) * [PLBAR] + [PR];

CALL INERTIA ( [LSB(BC)], [RSB(BC)], [ARR] );

[AL] := [DFBC] * [AL];

CALL RQMERGE ( [AA], [AR], [AL], [PARL(BC)] );

[RBS(BC)] := [PLBAR] + [IFR(BC)] * [AR];

CALL FBS ( [KLLINV(BC)], [RSB(BC)], [UL] );

CALL ISMERGE ( [UA], [UL], [PARL(BC)] );

ENDIF;

IF BMODES <> 0 THEN

BEGIN

CALL REIG ( [BC, USET(BC)], [KAA], [MAA], [MRR(BC)],

[DFBC], LAMBD, [PHIA], [MII], HSIZE(BC) );

CALL OMPROOT ( [BC, NUMOPTBC, LAMBDA] );

ENDIF;

ELSE

NO SUPPORT SET REDUCTION

IF BLOAD <> 0 THEN

PRINT("LOG="
      " >>>DISCIPLINE: STATICS");

CALL SDCMP ( [KAA], [KLLINV(BC)], USET(BC), SINGSET );

CALL FBS ( [KLLINV(BC)], [PA], [UA] );

ENDIF;

IF BMODES <> 0 THEN

BEGIN

PRINT("LOG="
      " >>>DISCIPLINE: NORMAL MODES");

CALL REIG ( [BC, USET(BC)], [KAA], [MAA], [LAMBD],

[PHIA], [MII], HSIZE(BC) );

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

ENDIF;

ENDIF;

IF BSAERO <> 0 THEN

PERFORM STATIC AEROELASTIC ANALYSES

PRINT("LOG("' SAERO_INITIALIZATION')");

TAKEN OUT FOR SAERO ***************

CALL TRANSPOSE ( [GSTTF], [GSTF] );

CALL TRANSPOSE ( [GAF], [GAF] );

LOOP := TRUE;

SUB := 0;

WHILE LOOP DO

SUB := SUB + 1;

CALL SAEROORV (BC, SUB, LOOP, MINDEX, SYM, MACH, QDF, 1 );

ADJUST THE KFF MATRIX AND DETERMINE THE RIGID AIR LOADS

IF SYM = 1 [AICS] := [GSTTF]*[TRANS([AICHAT(MINDEX)])]*[GSTF];

IF SYM = -1 [AICS] := [GSTTF]*[TRANS([-AICHAT(MINDEX)])]*[GSTF];

FAP := (QDF) [ [GSTF] * [AIRCFC(MINDEX)] ];

FAP := (QDF) [ [GAF] * [AIRCFC(MINDEX)] ];

REDUCE THE MATRICES WITH AEROSTATIC CORRECTIONS

SAVE THE SUBCASE/BC DEPENDENT DATA FOR SENSITIVITY ANALYSIS

IF NGDR <> 0 THEN

PERFORM THE GENERAL DYNAMIC REDUCTION

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

[MAAA] := [TRANS ([GSUBO(BC)]) * [AICS] * [GSUBO(BC)];

[MAAA] := [TRANS ([GSUBO(BC)]) * [KAA] * [GSUBO(BC)];

[MAAA] := [TRANS ([GSUBO(BC)]) * [FAP];

ELSE

IF NOMIT <> 0 THEN

PERFORM THE STATIC REDUCTION

PRINT("LOG("' SAERO STATIC CONDENSATION')");

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

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

CALL REDUCE ( [KFF] , [FPFOA(BC)] , [KCOINV(BC)] , ,
[GSUBO(BC)] , [KAA] , , USET(BC) );

ENDIF;

CALL REDUCE ( [KFF] , [FAP] , [FPFOA(BC)] , BSERO,
[KOOI(BC,SUB)] , [KOOU(BC,SUB)] ,
[KAO(BC,SUB)] , [GASUBO(BC,SUB)] , [KAA],
[KFA] , [POABO(BC,SUB)] , USET(BC) );

IF BMASS <> 0 THEN

PERFORM GUAYAN REDUCTION OF THE MASS MATRIX

CALL PARTN ( [MFF] , [MOO] , [MAO] , [MAABAR] ,
[MAAA] := [MAABAR] + TRANS ([MOA]) * [GASUBO(BC,SUB)] +
TRANS ([GASUBO(BC,SUB)]) * [MOA] +
TRANS ([GASUBO(BC,SUB)]) * [MOA] +
[GASUBO(BC,SUB)]);

IF NRSET <> 0

ENDIF;

ELSE

NO F-SET REDUCTION
IF NRSET <> 0 AND SUB = 1 AND BLOAD = 0 AND BMODES = 0 AND BBLTR = 0 AND BDYN = 0 THEN

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

[kaa] := [kff];

ENDIF;

[kaa] := [kff];

[m] := [mff];

[par] := [parf];

ENDIF;

ENDIF;

IF NRSET <> 0 THEN

PERFORM THE SUPPORT SET REDUCTION

PRINT("LOGOF \"SAERO SUPPORT REDUCTION\"");

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

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

CALL DECOMP ([KAA], [KRR], [KLR], [KLL], [PLAL]);

CALL DECMBP ([KLL], [KLLNV], [USET], [BC], [SINGSET]);

CALL FBSS ([KLLNV], [KLR], [D], [BC], [L]);

CALL RPCHEC ([BC], [USET], [BC], [BDFT], [D], [BC], [KLL], [F], [KLR], [KLL]);

ENDIF;

CALCULATE THE REDUCED MASS MATRIX

CALL PARTN ([MAAN], [MRRBAR], [MLR], [MRL], [PARL], [PARL]);

[R13(BC,SUB)] := [MLR] * [D(BC)] + [KLR];

[R35] := [MRRBAR] + TRANS([MLR]) * [D(BC)] + TRANS([D(BC)]) * [R13(BC,SUB)];

[R22] := TRANS([D(BC)]) * [MLR] + [MRRBAR];

CALL TRANSPE ([R13(BC,SUB)], [R21(BC,SUB)]);

PROCESS STEADY AEROELASTIC DISCIPLINE

PRINT("LOGOF \"DISCIPLINE: STEADY AERO\"");

CALL PARTN ([KAA], [KARL], [R12(BC,SUB)], [KRL], [R11], [PARL]);

[R32(BC,SUB)] := TRANS([D(BC)]) * [R12(BC,SUB)] + [KARL];

[R31(BC,SUB)] := TRANS([D(BC)]) * [R11] + [KARL];

CALL DECOMP ([R11], [R111(BC,SUB)], [R1111(BC,SUB)]);

CALL GBSS ([PAAR], [PARBAR], [PAL], [PARL]);

CALL GBSS ([R111(BC,SUB)], [R1111(BC,SUB)], [R12(BC,SUB)]);

CALL GBSS ([R111(BC,SUB)], [R1111(BC,SUB)], [R12(BC,SUB)]);

[R1113(BC,SUB)] := [R12(BC,SUB)] + [K112(BC,SUB)] * [R1113(BC,SUB)];

[K112(BC,SUB)] := [R12(BC,SUB)] * [R1113(BC,SUB)];

[K12(BC,SUB)] := [R12(BC,SUB)] + [R31(BC,SUB)] * [R1112(BC,SUB)];

[K22] := [R33] + [R31(BC,SUB)] * [R1112(BC,SUB)];

CALL DECOMP ([K11], [K111(BC,SUB)], [K1111(BC,SUB)]);

CALL GBSS ([K111(BC,SUB)], [K1111(BC,SUB)], [PAF], [PAR]);

CALL GBSS ([K11(BC,SUB)], [K111(BC,SUB)], [K112(BC,SUB)], [R1112(BC,SUB)]);

[LHSA(BC,SUB)] := [K22] + [K21(BC,SUB)] * [K1112(BC,SUB)];

[RHSA(BC,SUB)] := [PAF] - [R21(BC,SUB)] * [PAF];

CALL SAERO ([BC], [MINDEX], [SUB], [SYM], [QDF], [STARFC], [BGRFT], [LBHSA], [RHSA], [AAR]);

[DELTA] := [AAL] * [AAR];

CALL RORMERGE ([AAR], [AAR], [AAL], [PAR]);

[UAR] := [K1112(BC,SUB)] + [AAR] + [PAF];

[DELTA] := [AAL];
[UAL] := [R1112(BC,SUB)] * [VAR] + [R1113(BC,SUB)] * [AAR] | 2414 61
2415 61: [R1140(BC,SUB)] := [R1141(BC,SUB)] * [DLMAT(SUB)]; | 2416 61
2417 61: CALL RMMERGE ( [UAA(SUB)], [VAR], [UAL], [PAR(L,BC)] ); | 2418 61
2419 61: IF NOMIT <> 0 [PAO(SUB)] := [PAO(BC,SUB)] * [DLMAT(SUB)]; | 2420 61
2421 61: ELSE | 2422 61: ENDIF; | 2423 61
2424 61: [NO SUPPORT SET REDUCTION] | 2425 61
2426 61: ENDIF; | 2427 61
2428 41: ENDDO; | 2429 61
2430 31: [NO SUPPORT SET] | 2431 61
2432 31: PERFORM ANY DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT | 2433 61
2434 31: OF THE SUPPORT SET | 2435 61
2436 31: IF BDYN <> 0 THEN | 2437 61
2438 41: PRINT("LOG-*' | 2439 61
2440 41: ">>>DISCIPLINE: STEADY AERO''); | 2441 61
2442 41: SUB := 0; | 2443 61
2444 51: LOOP := TRUE; | 2445 61
2446 51: WHILE LOOP DO | 2447 61
2448 41: SUB := SUB + 1; | 2449 61
2450 41: CALL FLUTDRV ( BC, SUB, LOOP ); | 2451 61
2452 41: CALL FLUTQHKE ( [BC, SUB, E SIZE(BC), PSIZE(BC), [AJK], | 2453 61
2454 41: [SKE], [UGTKA], [PHIA], USET(BC), | 2455 61
2456 41: [TMB(BC)], [GSUB(BC)], NDDR, AECOMP, GEOMA, | 2457 61
2458 41: [PHIAM], [QHQLM(BC,SUB)], QAERDRP ); | 2459 61
2460 41: CALL FLUTMTA ( [BC, SUB, E SIZE(BC), PSIZE(BC), BGDPT(BC), | 2461 61
2462 41: USET(BC), [MAA], [KA], [TMB(BC)], [GSUB(BC)], | 2463 61
2464 41: NDDR, LAMBA, [PHIA], [MHH(BC,SUB)], | 2465 61
2466 41: [BHHL(BC,SUB)], [KHHL(BC,SUB)]]; | 2467 61
2468 41: CALL FLUTTRAS ( [BC, SUB, QHQLFM(BC,SUB)], LAMDA, KSIZE(BC), | 2469 61
2470 41: E SIZE(BC), [BHHT(BC,SUB)], [BHHL(BC,SUB)], | 2471 61
2472 41: [KHHL(BC,SUB)], LAMDA, AEROS ); | 2473 61
2474 61: ENDIF; | 2475 41
2476 51: IF BMTR <> 0 THEN | 2477 51:
2478 61: PRINT("LOG-*' | 2479 61
2480 61: ">>>DISCIPLINE: F L U T T R A S '); | 2481 61
2482 41: ENDDO; | 2483 51
2484 51: IF BMTR <> 0 THEN | 2485 51:
2486 61: PRINT("LOG-*' | 2487 61
2488 61: ">>>DISCIPLINE: TRANSIENT RESPONSE')); | 2489 61
2490 41: ENDDO; | 2491 51
2492 51: IF BMTR <> 0 THEN | 2493 51:
2494 61: PRINT("LOG-*' | 2495 61
2496 61: ">>>DISCIPLINE: FREQUENCY RESPONSE')); | 2497 61
2498 41: ENDDO; | 2499 51
2500 51: IF BMTR <> 0 THEN | 2501 51:
2502 61: PRINT("LOG-*' | 2503 61
2504 61: ">>>DISCIPLINE: BLAST'); | 2505 61
2506 41: CALL BLASTFIT ( [BC, [QJL]], [MATR], [MATSS], BGDPT, [BFRC], | 2507 61
2508 41: [DWNRS], HSIZE(BC), [IDZ], [MPART], [UGTKA], | 2509 61
2510 41: [BLSTJA]); | 2511 61
2512 41: CALL COLFARM ( [PHIA], [PHIA], [MPART] ); | 2513 61
2514 41: CALL RMMERGE ( [PHIR], [IDZ], [DBC], [PAR(L,BC)] ); | 2515 61
2516 41: CALL COLFARM ( [PHIR], [PHIR], [PHIR], [MPART] ); | 2517 61
2518 41: CALL DMFARM ( [PHIR], [MAA], [KA], [TMB(BC)], [GSUB(BC)], | 2519 61
2520 41: NDDR, [PHIAM], [QHQLM(BC,SUB)], QAERDRP ); | 2521 61
2522 41: CALL DYNLA ( [BC, O SIZE(BC), PSIZE(BC), BGDPT(BC), USET(BC), | 2523 61
2524 41: [TMB(BC)], [GSUB(BC)], NDDR, LAMDA, [PHIA], [QHQLM(BC,SUB)], | 2525 61
2526 41: [KHHL(BC,SUB)]; | 2527 61
2528 41: CALL DYNFA ( [BC, O SIZE(BC), [HDO], [KDO], [KDO], | 2529 61
2530 41: [MHH], [BBH], [BBH], [KHH], [KHH], | 2531 61
2532 41: [BBH], [BBH], [BBH], [KHH], [KHH], [BBH], [BBH], [BBH], [KHH], [KHH], | 2533 61
2534 41: [BBH], [BBH], [BBH], [KHH], [KHH], [BBH], [BBH], [BBH], [KHH], [KHH], | 2535 61
2536 41: [BBH], [BBH], [BBH], [KHH], [KHH], [BBH], [BBH], [BBH], [KHH], [KHH], | 2537 61
2538 41: [BBH], [BBH], [BBH], [KHH], [KHH], [BBH], [BBH], [BBH], [KHH], [KHH], | 2539 61
2540 41: [BBH], [BBH], [BBH], [KHH], [KHH], [BBH], [BBH], [BBH], [KHH], [KHH], | 2541 61
2542 41: [BBH], [BBH], [BBH], [KHH], [KHH], [BBH], [BBH], [BBH], [KHH], [KHH], | 2543 61
2544 41: [BBH], [BBH], [BBH], [KHH], [KHH], [BBH], [BBH], [BBH], [KHH], [KHH], | 2545 61
2546 41: [BHHL], [UTRRA], [UFREQA], [UTRANI], [UFREQJ], | 2547 61
2548 41: [U K T R A N I], [UFREQJ]; | 2549 61
2550 41: IF BMTR <> 0 THEN | 2551 61:
2552 61: PRINT("LOG-*' | 2553 61
2554 61: ">>>DISCIPLINE: FREQUENCY RESPONSE'); | 2555 61
2556 41: ENDDO; | 2557 51
2558 51: IF BMTR <> 0 THEN | 2559 51:
2560 61: PRINT("LOG-*' | 2561 61
2562 61: ">>>DISCIPLINE: BLAST'); | 2563 61
2564 41: ENDIF; | 2565 41
2566 31: IF BLAST <> 0 THEN | 2567 41:
2568 41: PRINT("LOG-*' | 2569 41
2570 41: ">>>DISCIPLINE: BLAST'); | 2571 41
2572 41:
2573 41:
2574 41:
2575 41:
2576 41:
2577 41:
2578 41:
2579 41:
2580 41:
2581 41:
2582 41:
2583 41:
2584 41:
2585 41:
2586 41:
2587 41:
2588 41:
2589 41:
2590 41:
2591 41:
2592 41:
2593 41:
2594 41:
BEGIN THE DATA RECOVERY OPERATIONS

IF NNSDCD > 1 CALL NULLMAT ([UF], [AF], [PHIF]);

IF NDRS <> 0 THEN
  CALL DATA RECOVERY WITH GDR;
  PRINT("'LOG(' DYNAMIC REDUCTION RECOVERY')");
  IF BLOAD <> 0 THEN
    [UFGRD] := [GSUBO(BC)] * [UA];
    CALL ROWPART ([UA], [UJK], [PAJK]);
    CALL ROWMERGE ([UF], [UJK], [UFGRD], [PPFK]);
    IF NRSET <> 0 THEN
      [AFGRD] := [GSUBO(BC)] + [AA];
      CALL ROWPART ([AA], [UJK], [PAJK]);
      CALL ROWMERGE ([AF], [UJK], [AFGRD], [PPFK]);
    ENDIF;
  ENDIF;
  IF BSAERO <> 0 THEN
    FOR S = 1 TO SUB DO
      [UFGRD] := [GSUBO(BC)] + [UAA(S)];
      CALL ROWPART ([UAA(S)], [UJK], [PAJK]);
      CALL ROWMERGE ([UAPTHP], [UJK], [UFGRD], [PPFK]);
    ENDIF;
    MERGE THE CURRENT SUBCASE DEPENDENT RESULTS INTO A SINGLE
    MATRIX OF RESPONSE QUANTITIES FOR FURTHER RECOVERY
    CALL SAEROCHR (BC, S, [UAPTHP]);
    IF NRSET <> 0 THEN
      [AFGRD] := [GSUBO(BC)] * [AAA(S)];
      CALL ROWPART ([AAA(S)], [UJK], [PAJK]);
      CALL ROWMERGE ([AARTMP], [UJK], [AFGRD], [PPFK]);
      CALL SAEROCHR (BC, S, [AFFP], [AARTMP]);
    ENDIF;
  ENDIF;
ENDDO;

ENDIF;

IF BMODES <> 0 THEN
  [UFGRD] := [GSUBO(BC)] * [PHIA];
  CALL ROWPART ([PHIA], [UJK], [PAJK]);
  CALL ROWMERGE ([PHIF], [UJK], [UFGRD], [PPFK]);
ENDIF;

IF BDMR <> 0 OR BMR <> 0 THEN
  [UFGRD] := [GSUBO(BC)] * [UTRAN];
  CALL ROWPART ([UTRAN], [UJK], [PAJK]);
  CALL ROWMERGE ([UUTMP], [UJK], [UFGRD], [PPFK]);
ENDIF;

IF BDMR <> 0 OR BMR <> 0 THEN
  [UFGRD] := [GSUBO(BC)] * [UFREQ];
  CALL ROWPART ([UFREQ], [UJK], [PAJK]);
  CALL ROWMERGE ([UFREQ], [UJK], [UFGRD], [PPFK]);
ENDIF;

ELSE
  IF NOMIT <> 0 THEN
    DATA RECOVERY WITH STATIC CONDENSATION
    PRINT("'LOG(' STATIC CONDENSATION RECOVERY')");
    IF BLOAD <> 0 THEN
      CALL RECOVA ([UA], [P0], [GSUBO(BC)], NRS, [AA],
                   [BP(BC)], [KOD(BC)], [PD(BC)], [UF]);
    ENDIF;
    IF NRSET <> 0 CALL RECOVA ([AA], [GSUBO(BC)], [AF]);
  ENDIF;
ENDIF;

82
ENDIF;
2577  51 IF BSAERO <> 0 THEN
2578     51 FOR S = 1 TO SUB DO
2579     71 CALL RECOVA ( [UAA(S)], [PAO(S)], [GASUBO(BC,S)],
2580     71 NRSET, [AAA(S)], [IFPA(BC,S)], BSAERO,
2581     71 [K0OL(BC,S)], [KODU(BC,S)],
2582     71 [PFoa(BC)], [UAFTMP] );
2583
2584  71$ MERGE THE CURRENT SUBCASE DEPENDENT RESULTS INTO A SINGLE
2585  71$ MATRIX OF RESPONSE QUANTITIES FOR FURTHER RECOVERY
2586  71$
2587  71 CALL SAEROMRG ( BC, S, [UAF], [UAFTMP] );
2588  71 IF NRSET <> 0 THEN
2589  81 CALL RECOVA ( [AAA(S)], [GASUBO(BC,S)],
2590  81 [PFoa(BC)], [AAFTMP] );
2591  81 CALL SAEROMRG ( BC, S, [AAF], [AAFTMP] );
2592  81 ENDIF;
2593  71
2594  61 ENDDO;
2595  61 ENDIF;
2596  61 IF BMODES <> 0 THEN
2597       61 [PHIO] := [GASUBO(BC)] * [PHIA];
2598  61 CALL ROXMERGE ( [PHIF], [PHIO], [PHIA], [PFoa(BC)] );
2599  61 ENDIF;
2600  61 IF BMFR <> 0 OR BMFR <> 0 THEN
2601       61 CALL RECOVA ( [UTRAN], [GASUBO(BC)],
2602       61 [PFoa(BC)], [UTRAN] );
2603  61 ENDIF;
2604  61 IF BMFR <> 0 OR BMFR <> 0 THEN
2605       61 CALL RECOVA ( [UFREQA], [GASUBO(BC)],
2606       61 [PFoa(BC)], [UFREQ] );
2607  61 ENDIF;
2608  51 ELSE
2609  51$ DATA RECOVERY WITHOUT F-SET REDUCTION
2610  51$
2611  51 IF BLOAD <> 0 THEN
2612      51 [UF] := [UA];
2613  51 IF NRSET <> 0 [AF] := [AA];
2614  51 ENDIF;
2615  51 IF BSAERO <> 0 THEN
2616      51 FOR S = 1 TO SUB DO
2617  51$ MERGE THE CURRENT SUBCASE DEPENDENT RESULTS INTO A SINGLE
2618  51$ MATRIX OF RESPONSE QUANTITIES FOR FURTHER RECOVERY
2619  51$
2620  51 CALL SAEROMRG ( BC, S, [UAF], [UA(S)] );
2621  51 IF NRSET <> 0 CALL SAEROMRG ( BC, S, [AAF], [AAA(S)] );
2622  51 ENDIF;
2623  71 ENDDO;
2624  61 ENDIF;
2625  61 IF BMODES <> 0 [PHIF] := [PHIA];
2626  61 IF BMFR <> 0 OR BMFR <> 0 ([UTRAN]) := ([UTRAN]);
2627  61 IF BMFR <> 0 OR BMFR <> 0 ([UFREQ]) := ([UFREQ]);
2628  61 ENDIF;
2629  61 ENDIF;
2630  61$ IF NBDNCOND > 1 CALL NULMAT ( [UN], [AN], [PHIN] );
2631  31 IF NSFC <> 0 THEN
2632  31$ DATA RECOVERY WITH SPO-REDUCTION
2633  31$
2634  41$ PRINT("LOG(\"SPC RECOVERY\")");
2635  41$ IF BLOAD <> 0 THEN
2636  41 CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)] );
2637  41 CALL OFFSPCF ( 0, BC, 1, 1, GSIZE, ESIZE(BC), NGDR,
2638  41 [KFS], [KSS], [UF], [YS(BC)], [PS],
2639  41 [PNSF(BC)], [PNSN(BC)], [PNSK],
2640  41 [BDPDT(BC)], OGRIDLOD );
2641  41 IF NRSET <> 0 CALL YSMERGE ( [AN], [AF], [PNSF(BC)] );
2642  41 ENDIF;
2643  41 IF BSAERO <> 0 THEN
2644  41 CALL YSMERGE ( [UN], [YS(BC)], [UAF], [PNSF(BC)] );
2645  41 IF NRSET <> 0 CALL YSMERGE ( [AAAN], [AAF], [PNSF(BC)] );
2646  41 ENDIF;
2647  41 IF BMODES <> 0 THEN
2648  41 CALL YSMERGE ( [PHIF], [YS(BC)], [PHIF],
2649  41 [PNSF(BC)] );
2650  41 IF DMODES <> 0 CALL OFFSPCF ( 0, BC, 2, 1, GSIZE,
2651  41 ESIZE(BC), NGDR,
2652  41 [KFS], [PHIF],
2653  41 [PNSF(BC)], [PNSN(BC)], [PNSK],
2654  41 [BDPDT(BC)], OGRIDLOD );
ENDIF;
2658 51 IF BSTR <> 0 OR BMTR <> 0
2659 51 CALL YMERGE ( [UPRAN], [YS(BC)], [UPRAF],
2660 51 [PNSF(BC)], BSTR );
2661 41 IF BDFR <> 0 OR BMTR <> 0
2662 51 CALL YMERGE ( [UFRQEN], [YS(BC)], [UFRQEF],
2663 51 [PNSF(BC)], BDFR );
2664 41 IF BFSUTR <> 0
2665 51 CALL OFFSPECF ( 0, BC, 4, 2, GSIZE, ESIZE(BC), NGDR, [KFS],,
2666 51 [PHIF],,, [PNSF(BC)], [PNUM(BC)], [PPFK],,
2667 51,, [BPDF(BC)], [GRIDLOD] );
2668 41 IF BBLAST <> 0 THEN
2669 51 [UBLAST] := [PHIF][UBLAST];
2670 41 CALL OFFSPECF ( 0, BC, 8, 1, GSIZE, ESIZE(BC), NGDR,
2671 51 [KFS],, [UBLAST],,, [PNSF(BC)], [PNUM(BC)],
2672 51 [PPFK],,, [BPDF(BC)], [GRIDLOD] );
2673 51 ENDIF;
2674 41 ELSE
2675 41$ DATA RECOVERY WITHOUT SPC-REDUCTION
2676 41$ ENDIF;
2677 41$ IF BLOAD <> 0 THEN
2678 51 [UN] := [UT];
2679 51 IF NRSET <> 0 [AN] := [AF];
2680 51 ENDIF;
2681 41$ IF BSAERO <> 0 THEN
2682 51 [UN] := [UAF];
2683 51 IF NRSET <> 0 [AN] := [AAF];
2684 51 ENDIF;
2685 41$ IF BMODES <> 0 [PHIM] := [PHIF];
2686 41$ IF BSTR <> 0 OR BMTR <> 0 [UPRAN] := [UPRAF];
2687 41$ IF BDFR <> 0 OR BMTR <> 0 [UFRQEN] := [UFRQEF];
2688 41$ IF BMODES <> 0 THEN
2689 41$ ENDIF;
2690 3$ IF NNDCOND > 1 CALL NULLMAY ( [UG(BC)], [AG(BC)], [UAG(BC)], [AAG(BC)],
2691 3$ [PHIG(BC)] );
2692 41$ IF NMPC <> 0 THEN
2693 41$ DATA RECOVERY WITH MPC-REDUCTION
2694 41$ PRINT("LOG=
2695 41$ MPS RECOVERY");
2696 41$ IF BLOAD <> 0 THEN
2697 41$ IF BLOAD <> 0 THEN
2698 41$ IF BMODES <> 0 THEN
2699 51 [UN] := [TM(BC)] * [UN];
2700 51 CALL RORWERG ( [UG(BC)], [UN], [UN], [PNUM(BC)] );
2701 51 IF NRSET <> 0 THEN
2702 51 [UN] := [TM(BC)] * [AN];
2703 61 CALL RORWERG ( [AG(BC)], [UN], [AN], [PNUM(BC)] );
2704 61 ENDIF;
2705 51 ENDIF;
2706 41$ IF BSAERO <> 0 THEN
2707 51 [UN] := [TM(BC)] * [UN];
2708 51 CALL RORWERG ( [UAG(BC)], [UN], [UN], [PNUM(BC)] );
2709 51 IF NRSET <> 0 THEN
2710 61 [UN] := [TM(BC)] * [AN];
2711 61 CALL RORWERG ( [AAG(BC)], [UN], [AN], [PNUM(BC)] );
2712 61 ENDIF;
2713 51 ENDIF;
2714 41$ IF BMODES <> 0 THEN
2715 51 [UN] := [TM(BC)] * [PHIN];
2716 51 CALL RORWERG ( [PHIG(BC)], [UN], [PHIN], [PNUM(BC)] );
2717 51 ENDIF;
2718 41$ IF BDFR <> 0 THEN
2719 51 [UN] := [TM(BC)] * [UFRQEN];
2720 51 CALL RORWERG ( [UFRQEN], [UN], [UFRQEN], [PNUM(BC)] );
2721 51 ENDIF;
2722 41$ IF BDFR <> 0 THEN
2723 51 [UN] := [TM(BC)] * [UFRQEN];
2724 51 CALL RORWERG ( [UFRQEN], [UN], [UFRQEN], [PNUM(BC)] );
2725 51 ENDIF;
2726 41$ ELSE
2727 41$ DATA RECOVERY WITHOUT MPC-REDUCTION
2728 41$ ENDIF;
2729 41$ IF BLOAD <> 0 THEN
2730 51 [UG(BC)] := [UN];
2731 51 IF NRSET <> 0 [AG(BC)] := [AN];
2732 51 ENDIF;
2733 41$ IF BSAERO <> 0 THEN
2734 51 [UAG(BC)] := [UN];
2735 51 IF NRSET <> 0 [AAG(BC)] := [AN];
2736 51 ENDIF;
IF BMODES <> 0 [PHIGH(BC)] := [PHIN];
IF BDR < 0 OR BMTR < 0 [UTRANG] := [UTRANN];
IF BDFR < 0 OR BMTR < 0 [UFREQS] := [UFREQN];
ENDIF;
RECOVER PHYSICAL BLAST DISCIPLINE DISPLACEMENTS
HANDLE OUTPUT REQUESTS
PRINT("LOG='\n OUTPUT PROCESSING'");
IF ISAERO <> 0 THEN
RECOVER STATIC AEROELASTIC LOADS DATA
LOOP := TRUE;
SUB := 0;
WHILE LOOP DO
SUB := SUB + 1;
CALL SAERODAV (BC, SUB, LOOP, MINDX, SYM, MACH, QD);
CALL THE TRIMMED LOADS COMPUTATION WITH PROPER MATRICES
ELSE
IF SYM = 1 THEN
CALL OFFALOAD (BC, MINDX, SUB, GSIDE, BGRID(BC));
CALL OFFALOAD (BC, MINDX, SUB, GSIDE, BGRID(BC));
CALL OFFALOAD (BC, MINDX, SUB, GSIDE, BGRID(BC));
ENDIF;
ELSE
IF SYM = -1 THEN
CALL OFFALOAD (BC, MINDX, SUB, GSIDE, BGRID(BC));
CALL OFFALOAD (BC, MINDX, SUB, GSIDE, BGRID(BC));
CALL OFFALOAD (BC, MINDX, SUB, GSIDE, BGRID(BC));
ENDIF;
CALL TO COMPUTE THE TRIMMED LOADS/DISPLACEMENTS ON THE AERODYNAMIC MODEL
IF SYM = 1 THEN
CALL OFFALEROM (NITER, BC, MINDX, SUB, GSIDE, GEOMSA);
CALL OFFALEROM (NITER, BC, MINDX, SUB, GSIDE, GEOMSA);
CALL OFFALEROM (NITER, BC, MINDX, SUB, GSIDE, GEOMSA);
ELSE
IF SYM = -1 THEN
CALL OFFALEROM (NITER, BC, MINDX, SUB, GSIDE, GEOMSA);
CALL OFFALEROM (NITER, BC, MINDX, SUB, GSIDE, GEOMSA);
CALL OFFALEROM (NITER, BC, MINDX, SUB, GSIDE, GEOMSA);
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
ENDIF;
IFDEF
ENDIF:
85
CALL OFFSETROM (MTER, BC, MINDX, SUB, GSIZ, GENMSA, RTOSS, [UG(BC)], [UAG(BC)], [UGM(@MINDX)], [UAGM(@MINDX)]);

ENDIF;

ENDIF;

ENDO;

ENDIF;

IF BDRSF <> 0 THEN
CALL OFFLOAD ( , BC, BGPDTH(BC), PSIZE(BC), ESIZE(BC), EPHIG(BC)),

IF EDTR <> 0 OR BMFR <> 0
CALL OFFSETCF ( , BC, 5, 1, GSIZ, ESIZE(BC),

NDR, [KFS], [UTRANS], [PHIG(BC)], [PHM(BC)], [PHM(BC)], [PPFJ],

[PHIG(BC)], [PHM(BC)], [PHM(BC)], [PPFJ],

BGPDTH(BC), ORGRIDL0,)

IF EDTR <> 0 OR BMFR <> 0
CALL OFFSETCF ( , BC, 6, 2, GSIZ, ESIZE(BC),

NDR, [KFS], [UTRANS], [PHIG(BC)], [PHM(BC)], [PHM(BC)], [PPFJ],

BGPDTH(BC), ORGRIDL0,)

ENDIF;

CALL OFFLOAD ( NUMOPBC, BC, GSIZ, BGPDTH(BC), PSIZE(BC),

[PSF]);

CALL OFFDISP ( NUMOPBC, BC, GSIZ, BGPDTH(BC), ESIZE(BC), PSIZE(BC),

ORGRIDDFS, [UGM(BC)], [UAGM(BC)], [UAGM(BC)], [UAGM(BC)],

[UGM(BC)], [UAGM(BC)], [UTRANS], [UTRANS], [UTRANS];

[PHIG(BC)]);
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>ACOORD</th>
<th>ID</th>
<th>XORIGN</th>
<th>YORIGN</th>
<th>ZORIGN</th>
<th>DELTA</th>
<th>THETA</th>
<th>XMNT</th>
<th>YMNT</th>
<th>CONT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>GT 0</td>
<td>0</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>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>
<tr>
<td>ACOORD</td>
<td>ID</td>
<td>XMNT</td>
<td>XBEND</td>
<td>YBEND</td>
<td>ZBEND</td>
<td>XTORQ</td>
<td>YTORQ</td>
<td>ZTORQ</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</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>CHECKS</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AERO2</th>
<th>ACSID</th>
<th>XSYM</th>
<th>ZMNT</th>
<th>XBEND</th>
<th>YBEND</th>
<th>ZBEND</th>
<th>XTORQ</th>
<th>YTORQ</th>
<th>ZTORQ</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<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>AESURF2</th>
<th>LABEL</th>
<th>TYPE</th>
<th>CID</th>
<th>SETK</th>
<th>SETG</th>
<th>CHAR</th>
<th>INT</th>
<th>INT</th>
<th>INT</th>
<th>CHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>-7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ATTACH</th>
<th>EID</th>
<th>MODEL</th>
<th>ISETK</th>
<th>ISETK</th>
<th>IREFGRID</th>
<th>FEEDBK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>GT 0</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>-6</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BODY7</th>
<th>LABEL</th>
<th>INDO</th>
<th>IPDO</th>
<th>ACOORD</th>
<th>INSEG</th>
<th>ID(1)</th>
<th>ID(2)</th>
<th>ID(3)</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>GT 0</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>BODY7</td>
<td>LABEL</td>
<td>INDO</td>
<td>IPDO</td>
<td>ACOORD</td>
<td>INSEG</td>
<td>INDO</td>
<td>IPDO</td>
<td>ACOORD</td>
<td>INSEG</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>------</td>
<td>------</td>
<td>--------</td>
<td>-------</td>
<td>------</td>
<td>------</td>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+BODY7</td>
<td>ID(5)</td>
<td>ID(6)</td>
<td>ID(7)</td>
<td>ID(8)</td>
<td>ID(9)</td>
<td>ID(10)</td>
<td>ID(11)</td>
<td>ID(12)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT2OB</td>
<td>GT2OB</td>
<td>GT2OB</td>
<td>GT2OB</td>
<td>GT2OB</td>
<td>GT2OB</td>
<td>GT2OB</td>
<td>GT2OB</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>-17</td>
<td></td>
</tr>
<tr>
<td>IDMESH1</td>
<td>IDMESH2</td>
<td>IDMESH3</td>
<td>IDMESH4</td>
<td>IDMESH5</td>
<td>IDMESH6</td>
<td>IDMESH7</td>
<td>IDMESH8</td>
<td>IDMESH9</td>
<td>IDMESH10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CEREO?</td>
<td>EID</td>
<td>LABELC</td>
<td>ACORED</td>
<td>NSPAN</td>
<td>NCROSS</td>
<td>LSSPAN</td>
<td>IZTAC</td>
<td>PAPFOL</td>
<td>CONT</td>
</tr>
<tr>
<td>--------</td>
<td>-----</td>
<td>--------</td>
<td>--------</td>
<td>-------</td>
<td>--------</td>
<td>--------</td>
<td>-------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>CHAR</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>DEFAULT</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>GE 0</td>
<td>GE 2</td>
<td>GE 2</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>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+CEREO?</td>
<td>XRL</td>
<td>YRL</td>
<td>ZRL</td>
<td>RCH</td>
<td>LACHD</td>
<td>IATTR</td>
<td>IATT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>INT</td>
<td></td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CHECKS</td>
<td></td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td></td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>+CEREO?</td>
<td>XTL</td>
<td>YTL</td>
<td>ZTL</td>
<td>TCH</td>
<td>LTCHD</td>
<td>ATT</td>
<td>ATTT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CHECKS</td>
<td></td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td>GE 0</td>
<td></td>
<td>INT</td>
<td>INT</td>
<td>INT</td>
</tr>
<tr>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>-21</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHORDCP</th>
<th>ID</th>
<th>IX</th>
<th>ICPU</th>
<th>ICFP</th>
<th>ICPL</th>
<th>IX</th>
<th>ICPU</th>
<th>ICPL</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>INT</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</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>GE 0.</td>
<td>2</td>
<td>15</td>
<td></td>
<td>3</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>-4</td>
<td>2</td>
<td>3</td>
<td>-4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+CHORDCP</td>
<td>X</td>
<td>CPU</td>
<td>CFL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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>REAL</td>
<td>REAL</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>GE 0.</td>
<td>GE 0.</td>
<td>2</td>
<td>15</td>
<td></td>
<td>3</td>
<td>16</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FLUTTER</th>
<th>SID</th>
<th>METHOD</th>
<th>IDENS</th>
<th>IDMK</th>
<th>IVEL</th>
<th>IMIST</th>
<th>KLIST</th>
<th>EFFID</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>INT</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>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
<td>PK</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>GT 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></td>
<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>
</tr>
<tr>
<td>+FLUTTER</td>
<td>SYMKZ</td>
<td>EPS</td>
<td>CURVFIT</td>
<td>PRINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>REAL</td>
<td>CHAR</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
<td>1.E-5</td>
<td>LINEAR</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td>IB</td>
<td>GT 0</td>
<td>FLIT</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>-14</td>
<td>PRINT</td>
</tr>
<tr>
<td>SYMKZ</td>
<td>SYMKZ</td>
<td>EPS</td>
<td>CURVFIT</td>
<td>PRINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GUST</th>
<th>SID</th>
<th>GLOAD</th>
<th>WSG</th>
<th>IXO</th>
<th>IV</th>
<th>IQDF</th>
<th>IDMK</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>INT</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>REAL</td>
<td>INT</td>
</tr>
<tr>
<td>DEFAULT</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>GT 0.</td>
<td>GT 0.</td>
<td>GT 0.</td>
<td>GT 0.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>GUST</td>
<td>SID</td>
<td>GLOAD</td>
<td>WSG</td>
<td>IXO</td>
<td>IV</td>
<td>IQDF</td>
<td>IDMK</td>
<td></td>
</tr>
<tr>
<td>GUST</td>
<td>SYMKZ</td>
<td>EPS</td>
<td>CURVFIT</td>
<td>PRINT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>INT</td>
<td></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>
</tr>
<tr>
<td>CHECKS</td>
<td>IB</td>
<td>-1</td>
<td>8</td>
<td>1</td>
<td>-9</td>
<td>0</td>
<td>SYMKZ</td>
<td>SYMKZ</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LOADMOD</th>
<th>LID</th>
<th>LABEL</th>
<th>CP</th>
<th>SETK</th>
<th>SETG</th>
<th></th>
<th></th>
<th></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>
</tr>
<tr>
<td>DEFAULT</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>GE 0</td>
<td>GE 0.</td>
<td>GT 0.</td>
<td>GT 0.</td>
<td>GT 0.</td>
<td>GT 0.</td>
<td>GT 0.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>-6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOADMOD</td>
<td>LID</td>
<td>LABEL</td>
<td>CP</td>
<td>SETK</td>
<td>SETG</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

89
<table>
<thead>
<tr>
<th>NAME</th>
<th>ID</th>
<th>IAFX</th>
<th>ITHR</th>
<th>ICAMR</th>
<th>ITHT</th>
<th>ICAMT</th>
<th>RADC</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAFD17</td>
<td>ID</td>
<td>IAFX</td>
<td>ITHR</td>
<td>ICAMR</td>
<td>RADR</td>
<td>ITHT</td>
<td>ICAMT</td>
<td>RADT</td>
</tr>
<tr>
<td>CHECKS</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>DEFAULT</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>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>ID</th>
<th>IAFX</th>
<th>ITHR</th>
<th>ICAMR</th>
<th>RADR</th>
<th>ITHT</th>
<th>ICAMT</th>
<th>RADT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PANLST1</td>
<td>SETID</td>
<td>MACROID</td>
<td>BOX1</td>
<td>BOX2</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>GT 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td>GT 0</td>
<td>GT 0</td>
<td>GT 0</td>
<td>GT 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>ID</th>
<th>IAFX</th>
<th>ITHR</th>
<th>ICAMR</th>
<th>RADR</th>
<th>ITHT</th>
<th>ICAMT</th>
<th>RADT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PANLST2</td>
<td>SETID</td>
<td>MACROID</td>
<td>BOX1</td>
<td>BOX2</td>
<td></td>
<td></td>
<td></td>
<td></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>GE 0</td>
</tr>
</tbody>
</table>

90
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PBODY7</td>
<td></td>
</tr>
<tr>
<td>WAKE</td>
<td></td>
</tr>
<tr>
<td>CPBASE</td>
<td></td>
</tr>
<tr>
<td>XSWAKE</td>
<td></td>
</tr>
<tr>
<td>XDWAKE</td>
<td></td>
</tr>
<tr>
<td>YWAKE</td>
<td></td>
</tr>
<tr>
<td>ZWAKE</td>
<td></td>
</tr>
<tr>
<td>INLET</td>
<td></td>
</tr>
<tr>
<td>CONT</td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS GT</td>
<td>0</td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 1.0</td>
<td></td>
</tr>
<tr>
<td>GE 1.1</td>
<td></td>
</tr>
<tr>
<td>GE 0.0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>IDP</td>
<td>9</td>
</tr>
<tr>
<td>FLOWRT</td>
<td></td>
</tr>
<tr>
<td>SEGMESH</td>
<td></td>
</tr>
<tr>
<td>IDMESH</td>
<td></td>
</tr>
<tr>
<td>IMESH</td>
<td></td>
</tr>
<tr>
<td>NAXIS</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS GT</td>
<td>0</td>
</tr>
<tr>
<td>GE 2</td>
<td></td>
</tr>
<tr>
<td>GE 2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>SEMGSH</td>
<td></td>
</tr>
<tr>
<td>IDMESH</td>
<td></td>
</tr>
<tr>
<td>NAXIS</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS GT</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>IDTYPE</td>
<td>X</td>
</tr>
<tr>
<td>CAM</td>
<td></td>
</tr>
<tr>
<td>YR</td>
<td></td>
</tr>
<tr>
<td>ZR</td>
<td></td>
</tr>
<tr>
<td>IDY</td>
<td></td>
</tr>
<tr>
<td>IDZ</td>
<td></td>
</tr>
<tr>
<td>SPINE1</td>
<td></td>
</tr>
<tr>
<td>EID</td>
<td></td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td>CP</td>
<td></td>
</tr>
<tr>
<td>SETK</td>
<td></td>
</tr>
<tr>
<td>SETG</td>
<td></td>
</tr>
<tr>
<td>IDZ</td>
<td></td>
</tr>
<tr>
<td>EPS</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS GT</td>
<td>0</td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>SPINE1 EID</td>
<td></td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td>CP</td>
<td></td>
</tr>
<tr>
<td>BOXSETIDOR</td>
<td></td>
</tr>
<tr>
<td>BOXSETIDFLEX</td>
<td></td>
</tr>
<tr>
<td>EPS</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
</tr>
<tr>
<td>SPINE2</td>
<td></td>
</tr>
<tr>
<td>EID</td>
<td></td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td>SETK</td>
<td></td>
</tr>
<tr>
<td>SETG</td>
<td></td>
</tr>
<tr>
<td>IDZ</td>
<td></td>
</tr>
<tr>
<td>DTOR</td>
<td></td>
</tr>
<tr>
<td>CID</td>
<td></td>
</tr>
<tr>
<td>DETHX</td>
<td></td>
</tr>
<tr>
<td>CONT</td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS GT</td>
<td>0</td>
</tr>
<tr>
<td>GT 0</td>
<td></td>
</tr>
<tr>
<td>GT 0</td>
<td></td>
</tr>
<tr>
<td>GT 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>SPINE2 EID</td>
<td></td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td>BOXSETIDOR</td>
<td></td>
</tr>
<tr>
<td>BOXSETIDFLEX</td>
<td></td>
</tr>
<tr>
<td>DTOR</td>
<td></td>
</tr>
<tr>
<td>CID</td>
<td></td>
</tr>
<tr>
<td>DETHX</td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>REAL</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS</td>
<td></td>
</tr>
<tr>
<td>-10</td>
<td></td>
</tr>
<tr>
<td>DTHY</td>
<td></td>
</tr>
<tr>
<td>SPINE3</td>
<td></td>
</tr>
<tr>
<td>EID</td>
<td></td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td>CP</td>
<td></td>
</tr>
<tr>
<td>SETK</td>
<td></td>
</tr>
<tr>
<td>SETG</td>
<td></td>
</tr>
<tr>
<td>IDZ</td>
<td></td>
</tr>
<tr>
<td>EPS</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS GT</td>
<td>0</td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>SPINE3 EID</td>
<td></td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td>BOXSETIDOR</td>
<td></td>
</tr>
<tr>
<td>BOXSETIDFLEX</td>
<td></td>
</tr>
<tr>
<td>EPS</td>
<td></td>
</tr>
<tr>
<td>SPINT</td>
<td></td>
</tr>
<tr>
<td>OID</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>INT/CHARINT</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>DEFAULT</td>
<td></td>
</tr>
<tr>
<td>CHECKS GT</td>
<td>0</td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>GE 0</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>SPINT</td>
<td>EXTID</td>
</tr>
<tr>
<td>TRIM</td>
<td>SETID</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>CHAR</td>
<td>INT</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
</tr>
<tr>
<td>TRIM</td>
<td>GT 0</td>
</tr>
<tr>
<td>+TRIM</td>
<td>REL/CHAR</td>
</tr>
<tr>
<td>CHAR</td>
<td>FIXI</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>RATE</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<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>INT</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>CHECKS</td>
<td>GT 0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>TRIMFLT</td>
<td>REL/CHAR</td>
<td>REL/CHAR</td>
<td>REL/CHAR</td>
<td>REL/CHAR</td>
<td>REL/CHAR</td>
<td>REL/CHAR</td>
<td>REL/CHAR</td>
<td></td>
</tr>
<tr>
<td>+TRIMF</td>
<td>VALUE1</td>
<td>VALUE2</td>
<td>VALUE3</td>
<td>VALUE4</td>
<td>VALUE5</td>
<td>VALUE6</td>
<td>VALUE7</td>
<td>VALUE8</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>REAL</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>CHECKS</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>STAIC</th>
<th>ID</th>
<th>NMLAP</th>
<th>MACCHP1</th>
<th>MACCHP2</th>
<th>MACCHP3</th>
<th>MACCHP4</th>
<th>MACCHP5</th>
<th>MACCHP6</th>
<th>CONT</th>
</tr>
</thead>
<tbody>
<tr>
<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>CONT</td>
<td>INT</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>1</td>
</tr>
<tr>
<td>STAIC</td>
<td>NMLAP</td>
<td>MACCHP1</td>
<td>MACCHP2</td>
<td>MACCHP3</td>
<td>MACCHP4</td>
<td>MACCHP5</td>
<td>MACCHP6</td>
<td>MACCHP6</td>
<td>MACCHP6</td>
</tr>
<tr>
<td>+STAIC</td>
<td>HINGE</td>
<td>INBDY</td>
<td>OUTBDY</td>
<td>LABEL</td>
<td>HINGE</td>
<td>INBDY</td>
<td>OUTBDY</td>
<td>ETC</td>
<td>ETC</td>
</tr>
<tr>
<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>INT</td>
<td>INT</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>GE 1</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>
</tr>
<tr>
<td>CHECKS</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>-12</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>-12</td>
<td>9</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>

92
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:

<table>
<thead>
<tr>
<th>RELATION ACOORD</th>
<th>RELATION BODY7</th>
<th>RELATION GEOSEA</th>
<th>RELATION PANO21</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID INT</td>
<td>IDBODY INT</td>
<td>MACROID INT</td>
<td>SETID INT</td>
</tr>
<tr>
<td>XORIG NSP</td>
<td>LABEL STR 8</td>
<td>ACHPMNT STR 8</td>
<td>MACROID INT</td>
</tr>
<tr>
<td>YORIG NSP</td>
<td>IBODY INT</td>
<td>NOOFP INT</td>
<td>ECKX1 INT</td>
</tr>
<tr>
<td>ZORIG NSP</td>
<td>ACORD INT</td>
<td>EXTID INT</td>
<td>ECKX2 INT</td>
</tr>
<tr>
<td>DELTA NSP</td>
<td>NS05 INT</td>
<td>INTID INT</td>
<td>END</td>
</tr>
<tr>
<td>THETA NSP</td>
<td>IDMESA INT</td>
<td>AREA INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>XMINT NSP</td>
<td>IDMESA8 INT</td>
<td>X INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>YMINT NSP</td>
<td>IDMESC INT</td>
<td>Y INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>ZMINT NSP</td>
<td>IDMESC9 INT</td>
<td>Z INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>XEND NSP</td>
<td>IDMESC1 INT</td>
<td>N1 INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>YEND NSP</td>
<td>IDMESC2 INT</td>
<td>N2 INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>ZEND NSP</td>
<td>IDMESC3 INT</td>
<td>N3 INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>XTROQ NSP</td>
<td>IDMESC4 INT</td>
<td>R1 INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>YTROQ NSP</td>
<td>IDMESC5 INT</td>
<td>R2 INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>ZTROQ NSP</td>
<td>IDMESC6 INT</td>
<td>R3 INT</td>
<td>-----------------</td>
</tr>
<tr>
<td>END</td>
<td>IDMESC7 INT</td>
<td>R4 INT</td>
<td>-----------------</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RELATION AGRID2</th>
<th>RELATION CAERO7</th>
<th>RELATION RELMOD</th>
<th>RELATION REUNMK</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXTID INT</td>
<td>EID INT</td>
<td>LID INT</td>
<td>IDMK INT</td>
</tr>
<tr>
<td>INTID INT</td>
<td>LABELL STR 8</td>
<td>LABEL STR 8</td>
<td>MACH INT</td>
</tr>
<tr>
<td>CORD INT</td>
<td>ACORD INT</td>
<td>CP INT</td>
<td>MACHCINT INT</td>
</tr>
<tr>
<td>X NSP</td>
<td>NSFAN INT</td>
<td>SETK INT</td>
<td>IMAK INT</td>
</tr>
<tr>
<td>Y NSP</td>
<td>MINCORD INT</td>
<td>SETG INT</td>
<td>MACHM INT</td>
</tr>
<tr>
<td>Z NSP</td>
<td>LSFAN INT</td>
<td>END</td>
<td>XNSMZ INT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RELATION AESURF3</th>
<th>RELATION CHORDC</th>
<th>RELATION RELMULT</th>
<th>RELATION SFLINE3</th>
</tr>
</thead>
<tbody>
<tr>
<td>LABEL STR 8</td>
<td>ID INT</td>
<td>ID INT</td>
<td>EID INT</td>
</tr>
<tr>
<td>TYPE STR 8</td>
<td>X NSP</td>
<td>X NSP</td>
<td>MODEL STR 8</td>
</tr>
<tr>
<td>CID INT</td>
<td>CPU NSP</td>
<td>CPU NSP</td>
<td>CP INT</td>
</tr>
<tr>
<td>SETK INT</td>
<td>CPL NSP</td>
<td>CPL NSP</td>
<td>BOXSTR INT</td>
</tr>
<tr>
<td>SETG INT</td>
<td>END</td>
<td>END</td>
<td>BOXSTID INT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RELATION AQUDAI</th>
<th>RELATION FLUTTER</th>
<th>RELATION RELPALL</th>
<th>RELATION RELUNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACROID INT</td>
<td>SETID INT</td>
<td>PDAPINT INT</td>
<td>ID INT</td>
</tr>
<tr>
<td>ACHPMNT INT</td>
<td>METHOD STR 4</td>
<td>SAVE STR 8</td>
<td>IAIST INT</td>
</tr>
<tr>
<td>NOOFP INT</td>
<td>DENS INT</td>
<td>FILE1 STR 8</td>
<td>ITMPC INT</td>
</tr>
<tr>
<td>EXTID INT</td>
<td>IDM00 INT</td>
<td>FILE2 STR 8</td>
<td>ICAMR INT</td>
</tr>
<tr>
<td>INTID INT</td>
<td>VEL INT</td>
<td>PRINT INT</td>
<td>ICAMR INT</td>
</tr>
<tr>
<td>AREA NSP</td>
<td>RELIST INT</td>
<td>RFSQ RSP</td>
<td>ICAMR INT</td>
</tr>
<tr>
<td>X NSP</td>
<td>EFLIST INT</td>
<td>END</td>
<td>ICAMR INT</td>
</tr>
<tr>
<td>Y NSP</td>
<td>SYNX INT</td>
<td>END</td>
<td>ICMINT INT</td>
</tr>
<tr>
<td>Z NSP</td>
<td>SYNX INT</td>
<td>END</td>
<td>ICINT INT</td>
</tr>
<tr>
<td>N1 NSP</td>
<td>EPS NSP</td>
<td>END</td>
<td>ICINT INT</td>
</tr>
<tr>
<td>N2 NSP</td>
<td>CURVFIT STR 8</td>
<td>END</td>
<td>ICINT INT</td>
</tr>
<tr>
<td>N3 NSP</td>
<td>MACHVAL RSP</td>
<td>END</td>
<td>ICINT INT</td>
</tr>
<tr>
<td>R1 NSP</td>
<td>PRINT INT</td>
<td>END</td>
<td>ICINT INT</td>
</tr>
<tr>
<td>R2 NSP</td>
<td>END</td>
<td>END</td>
<td>ICINT INT</td>
</tr>
<tr>
<td>R3 NSP</td>
<td>END</td>
<td>END</td>
<td>ICINT INT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RELATION AQUGAI</th>
<th>RELATION NGAERO5</th>
<th>RELATION RELWAVE</th>
<th>RELATION SEAGMESH</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACROID INT</td>
<td>ID INT</td>
<td>ID INT</td>
<td>IDMESH INT</td>
</tr>
<tr>
<td>ACHPMNT INT</td>
<td>X NSP</td>
<td>MACH INT</td>
<td>NAKIS INT</td>
</tr>
<tr>
<td>NOOFP INT</td>
<td>CPU NSP</td>
<td>IGROD INT</td>
<td>NID INT</td>
</tr>
<tr>
<td>EXTID INT</td>
<td>CPL NSP</td>
<td>IINDICIA INT</td>
<td>ITYPE INT</td>
</tr>
<tr>
<td>INTID INT</td>
<td>END</td>
<td>SPANID INT</td>
<td>X NSP</td>
</tr>
<tr>
<td>AREA NSP</td>
<td>END</td>
<td>CHORDC INT</td>
<td>CAM NSP</td>
</tr>
<tr>
<td>X NSP</td>
<td>END</td>
<td>END</td>
<td>YR NSP</td>
</tr>
<tr>
<td>Y NSP</td>
<td>END</td>
<td>END</td>
<td>ZR NSP</td>
</tr>
<tr>
<td>Z NSP</td>
<td>END</td>
<td>END</td>
<td>IDY INT</td>
</tr>
<tr>
<td>N1 NSP</td>
<td>END</td>
<td>END</td>
<td>ID2 INT</td>
</tr>
<tr>
<td>N2 NSP</td>
<td>END</td>
<td>END</td>
<td>ID2 INT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RELATION AQUGAI</th>
<th>RELATION RELWAVE</th>
<th>RELATION SFLINE3</th>
<th>RELATION SEAGMESH</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACROID INT</td>
<td>ID INT</td>
<td>EID INT</td>
<td>IDMESH INT</td>
</tr>
<tr>
<td>ACHPMNT INT</td>
<td>X NSP</td>
<td>MODEL STR 8</td>
<td>NAKIS INT</td>
</tr>
<tr>
<td>NOOFP INT</td>
<td>CPU NSP</td>
<td>CP INT</td>
<td>NID INT</td>
</tr>
<tr>
<td>EXTID INT</td>
<td>CPL NSP</td>
<td>BOXSTR INT</td>
<td>ITYPE INT</td>
</tr>
<tr>
<td>INTID INT</td>
<td>END</td>
<td>BOXSTID INT</td>
<td>X NSP</td>
</tr>
<tr>
<td>AREA NSP</td>
<td>END</td>
<td>BOXSTID INT</td>
<td>CAM NSP</td>
</tr>
<tr>
<td>X NSP</td>
<td>END</td>
<td>BOXSTID INT</td>
<td>YR NSP</td>
</tr>
<tr>
<td>Y NSP</td>
<td>END</td>
<td>BOXSTID INT</td>
<td>ZR NSP</td>
</tr>
<tr>
<td>Z NSP</td>
<td>END</td>
<td>BOXSTID INT</td>
<td>IDY INT</td>
</tr>
<tr>
<td>N1 NSP</td>
<td>END</td>
<td>BOXSTID INT</td>
<td>ID2 INT</td>
</tr>
<tr>
<td>N2 NSP</td>
<td>END</td>
<td>BOXSTID INT</td>
<td>ID2 INT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RELATION AQUGAI</th>
<th>RELATION RELWAVE</th>
<th>RELATION SFLINE3</th>
<th>RELATION TRIMFL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACROID INT</td>
<td>ID INT</td>
<td>IAIF INT</td>
<td>IDINT INT</td>
</tr>
<tr>
<td>ACHPMNT INT</td>
<td>ITHM INT</td>
<td>ITIL INP</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>NOOFP INT</td>
<td>ICAMR INT</td>
<td>ICAMR INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>EXTID INT</td>
<td>ICAMR INT</td>
<td>ICAMR INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>INTID INT</td>
<td>RADM INT</td>
<td>RADM INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>AREA INT</td>
<td>ICHNT INT</td>
<td>ICHNT INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>X INT</td>
<td>ICINT INT</td>
<td>ICINT INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>Y INT</td>
<td>ICINT INT</td>
<td>ICINT INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>Z INT</td>
<td>ICINT INT</td>
<td>ICINT INT</td>
<td>TILTM INT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RELATION AQUGAI</th>
<th>RELATION RELWAVE</th>
<th>RELATION SFLINE3</th>
<th>RELATION TRIMFL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACROID INT</td>
<td>ID INT</td>
<td>IAIF INT</td>
<td>IDINT INT</td>
</tr>
<tr>
<td>ACHPMNT INT</td>
<td>ITHM INT</td>
<td>ITIL INP</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>NOOFP INT</td>
<td>ICAMR INT</td>
<td>ICAMR INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>EXTID INT</td>
<td>ICAMR INT</td>
<td>ICAMR INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>INTID INT</td>
<td>RADM INT</td>
<td>RADM INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>AREA INT</td>
<td>ICHNT INT</td>
<td>ICHNT INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>X INT</td>
<td>ICINT INT</td>
<td>ICINT INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>Y INT</td>
<td>ICINT INT</td>
<td>ICINT INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>Z INT</td>
<td>ICINT INT</td>
<td>ICINT INT</td>
<td>TILTM INT</td>
</tr>
<tr>
<td>RELATION</td>
<td>ZTAIC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACHCP1</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACHCP2</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACHCP3</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACHCP4</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACHCP5</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MACHCP6</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LABEL</td>
<td>STR 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HINGE</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INBDY</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTBDY</td>
<td>INT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>END</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX E

ZAERÔ 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 ZOA'S AEROGM MODULE MESSAGES**

'NO $ BULK DATA ENTRY IS DEFINED, BUT BODY? BULK DATA EXISTS IN THE INPUT.'

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

'BULK DATA ENTRY IS REFERRED 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 SPANNWISE DIVISIONS DEFINED, BUT THERE ARE $ NUMBER OF VALUES.'

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

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

'THE SPANNWISE 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 $.'

'CAREOTHER WITH WID: $, HAS NO STEADY PRESSURE INPUT ON SPANNWISE STRIP INDEX = $ AND MACH NUMBER = $.'

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

'CAREOTHER WITH WID: $, HAS MORE THAN ONE SPANNWISE STRIP INDEX DEFINED FOR A MACHCP BULK DATA ENTRY.'

'FOR SPANNWISE STRIP INDEX = $ AND MACH NUMBER = $.'

'AEROODYNAMIC $ ID: IS TOO LARGE TO AVAILABLE MEMORY.'

'A DPLICATE AEROODYNAMIC $ EXISTS WITH ID: $.'

'A SEGMENT BULK DATA CARD WITH IMESH: $ HAS $ NUMBER OF D-VALUE CIRCUMFERENTIAL POINTS (NRAD) DEFINED,'

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

'WITH WID: $ HAS A BOX OF ZERO AREA WITH ID: $.'

'ERROR IN $ WITH ID: $ INCOMPLETE LIST OF LABEL-NINGE-INDIB-OUTDY PAIRS FOR NFPAL = $.'

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

'ERROR IN $ WITH ID: $ ENTRY HINGE = $ IS GREATER THAN 1 AND LESS THAN $ (NCHORD).'

'ERROR IN $ WITH ID: $ ENTRY INDBY = $ IS GREATER THAN 1 AND LESS THAN $ (NSPAN).'

'ERROR IN $ WITH ID: $ ENTRY OUTDY = $ IS GREATER THAN 1 AND LESS THAN $ (NSPAN).'

'ERROR IN $ WITH ID: $ ENTRY INDBY = $ IS GREATER THAN OR EQUAL TO ENTRY OUTDY = $.'

'$ BULK DATA CARD WITH ID: $ HAS A SPANNWISE INDEX (SPANID) = $ WHICH IS LESS THAN 1 OR GREATER THAN THE.'

'NUMBER OF SPANNWISE BOXES (NSPAN) = $.'

'$ BULK DATA CARD WITH ID: $ DOES NOT HAVE COMPLETED X-CPU-CPL PAIRS (i.e., IN THREEES).'

'$ BULK DATA CARD WITH ID: $ HAS A X-LOCATION VALUE GREATER THAN 100 PERCENT CHORD.'

'$ BULK DATA CARD WITH ID: $ HAS X-LOCATION VALUES THAT ARE NOT IN ASCENDING ORDER.'

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

'$ BULK DATA CARD WITH ID: $ HAS CHORDWISE X-VALUES THAT DO NOT START WITH 0.0 OR END WITH 100.0.'

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

'$ 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: $.'

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

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

'AND $ NUMBER OF CHORDWISE DIVISIONS (NCHORD) SPECIFIED,'

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

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

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

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

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

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

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

**MODULE 36 ZOA'S SPLINI MODULE MESSAGES**

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

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

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

'ENTRY $ REFERENCES STRUCTURAL SET DEFINITION $ THAT DOES NOT EXIST.'

'THE STRUCTURAL SET DEFINED BY SET2 ENTRY $, REFERENCED ON $ ENTRY $, IS EMPTY.'

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

'$ ENTRY $ RESULTS IN A SINGULAR TRANSFORMATION MATRIX.'

'AEROODYNAMIC BOX WITH INTERNAL IDENTIFICATION NUMBER $ HAS BEEN SPLINED MORE THAN ONCE.'

'ANALYSES ARE REQUESTED IN SOLUTION CONTROL BUT NO SPLINE OR ATTACH ENTRIES EXIST.'

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

'WHEN USED ON A LIFTING SURFACE A CID MUST BE SUPPLIED.'

'$ SPECIFIES NON-EXISTENT MACRO-ELEMENT $.$'

'$ SETID $ SPECIFIES NON-EXISTENT AEROODYNAMIC BOXES FOR MACRO-ELEMENT $.'

'THE RECTANGULAR REGION SPECIFIED BY BOX1 AND BOX2 ON SETID $ CONTAINS NO AEROODYNAMIC BOXES.'

'$ SETID $ SPECIFIES MORE BOXES THAN EXIST IN THE AEROODYNAMIC MODEL.'

'$ SETID $ SPECIFIES DUPLICATE AEROODYNAMIC BOXES MACROID $, SETID $.'
"$ 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.
"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.
"S WITH ID: $ REFERS TO A SETI THAT HAS LESS THAN $ GRIDS.
"S WITH ID: $ REFERS TO A SETI THAT HAS ALL GRIDS ALIGNED ALONG A LINE.
"S WITH ID: $ REFERS TO A SETI THAT HAS TWO GRIDS AT THE SAME LOCATION.
"S 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 ZONAS ZAERO MODULE MESSAGES
"S 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 WING 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: $.