SANDIA REPORT SAND80-2056 • Unlimited Release • UC-32
Printed October 1981

A User's Manual for the AMEER Flight Path-Trajectory Simulation Code

Eugene J. Meyer

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-76DP00789

SF2900Q(8-81)

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof or any of their contractors or subcontractors.

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes
Printed copy: $12.00
Microfiche copy: A01

CHANGES TO SANDIA REPORT -SAND80-2056-

A USER'S MANUAL FOR THE AMEER FLIGHT PATH-TRAJECTORY SIMULATION CODE

DATE:FEB 1983

REMOVE PAGES	REPLACE WITH PAGES
19,20	19,20
27 THRU 30	27 THRU 30
37 THRU 42	37 THRU 42
57,58	57,58
81 THRU 86	81 THRU 86
183 THRU 186	183 THRU 186
191,192	191,192

DONE
3/17/83

SAND80-2056
Unlimited Release
Printed October 1981

Distribution
Category UC-32

A User's Manual For The AMEER Flight Path-Trajectory Simulation Code

Eugene J. Meyer
Aerodynamics Simulation Division 5636
Sandia National Laboratories
Albuquerque, New Mexico 87185

Abstract

A guide to the use of the AMEER (*A*ero-*M*echanical *E*quations *E*valuation *R*outines) flight path-trajectory simulation code is presented. The input data requirements, computed output data available, code control features, and code flow logic are described for a rigid-body six-degree-of-freedom or point mass simulation.

Acknowledgments

The many formulas and concepts incorporated in the AMEER code are readily obtained from available literature. However, special insight into the problem of developing a generalized flight-path trajectory code was furnished by the Problem Formulation Manual for the Six-Degree-of-Freedom Flight-Path Study, Generalized Computer Program prepared by McDonnell Aircraft Corporation, St. Louis, MO, under contract to the U.S. Air Force.[1]

The author expresses his appreciation for the support and contribution furnished by the following Sandia National Laboratories personnel:

Organization 5630 - A. E. Hodapp for his formulation of the six-degree-of-freedom equations
W. E. Williamson for his formulation of the point-mass equations and point-mass controller
H. A. Wente for his support in encoding the earth model, the atmosphere models, and the propulsion system
W. H. Curry for his polynominal formulation of the aerodynamic force and moment coefficients
The committee members for their specification of the code requirements.

Organization 2600 - R. E. Jones for his development of the input and table handling package
H. A. Watts for his aid in selecting and implementing the integration package.

Contents

SUMMARY 7
1. INTRODUCTION 9
2. DISCUSSION 11
2.1 General Code Description 11
2.2 Operational Procedure 11
2.3 User Perogatives 11
2.4 Data Preparation 12
2.5 Main Programs 12
2.5.1 Six-Degree-of-Freedom (SIXDOF) 12
2.5.2 Point Mass (PTMASS) 12
2.6 Support Routines 13
2.6.1 Aerodynamic (AERO) 13
2.6.2 Atmosphere (ATMS) 13
2.6.3 Control 13
2.6.3.1 Six-Degree-of-Freedom (CONTRL) 13
2.6.3.2 Point Mass (PTMSCO) 13
2.6.4 Station/Radar (RADAR) 13
2.6.5 Propulsion System (THRUST) 14
2.6.6 Vehicle Mass Properties (VEHMSP) 14
2.6.7 Gravity Model (GRAV) 14
2.7 Data Output 14
2.7.1 Block Print 14
2.7.2 Tabular Lists 15
2.7.3 Y vs X Plots 15
2.7.4 TAPE20 File 15
2.8 Description of Appendix A 15
2.9 Description of Appendix B 15
2.10 Description of Appendix C 15
3. INPUT DATA DECLARATION AND SPECIFICATION 17
3.1 General Data Card Information 17
3.2 Input Variable Name Forms 18
3.3 Input Value Forms 18
3.4 Data Definition Forms 19
3.5 Variable and Array Forms 19
3.6 Hollerith Data Forms 19
3.7 Table Data Forms 20
3.7.1 TABLE Card Form 20
3.7.2 BREAKS Card Form 21
3.7.3 BODY Card Form 21
3.8 DELETE Card Form 22
3.9 FORMAT Card Form 22
3.10 Data Editing 22
3.11 Data Equivalencing 23
4. REQUIRED INPUT DATA 25
4.1 General Code Items 25
4.2 General Problem Items 26
4.3 Specific Problem Items: Main Program 30
4.3.1 Six-Degree-of-Freedom (SIXDOF) 30
4.3.2 Point Mass (PTMASS) 32

Contents (cont)

4.4 Specific Problem Items: Support Routines 33
4.4.1 Aerodynamics 33
4.4.2 Atmosphere 36
4.4.3 Control 38
4.4.3.1 Six-Degree-of-Freedom 38
4.4.3.2 Point Mass 39
4.4.4 Propulsion System 41
4.4.5 Vehicle Mass Properties 42
4.4.6 Gravity Model 43
4.5 General Output Requests 44
4.5.1 Tabulated Data 44
4.5.2 Plotted Data 45
4.5.3 TAPE20 File 45
5. CODE CONTROL 47
5.1 Case or Problem Control 47
5.1.1 Staging - Input Data 47
5.2 Execution Control: *CONTROL Cards 49
5.2.1 Base Case 50
5.2.2 Merging Cases 50
5.2.3 Stacking Cases 51
5.2.4 *SAVE, *RESTORE Utility Feature 52
5.2.5 *END Card 52
References 53
APPENDIX A — Logic Flow Diagram and Examples 55
APPENDIX B — Flow Charts and Dimensional Data 69
APPENDIX C — Variables: Program, Output Block, Data File, and Tabular Lists 181

Summary

The AMEER (*A*ero-*M*echanical *E*quations *E*valuation *R*outines) code is a collection of routines that predict the trajectory-flight path of an object over a planet under the influence of the planet atmosphere and gravitational field. The path may be further modified by introducing forces and moments on the object through the use of a thrusting and swiveling propulsion system, an attitude control system, an aerodynamic force and moment evaluator, and by varying the center-of-mass position and mass properties.

Input data requirements, whether numeric or Hollerith, are presented relative to the routine with which they are logically associated. The computed data items available for output are given and the available methods of presentation are discussed.

Internal code control through the integration routine and the external control by data deck structure are described. Sample input data decks are presented in Appendix A. Code flow logic diagrams for the major routines are given in Appendix B, with program variable name descriptions listed in Appendix C.

A User's Manual For The AMEER Flight Path-Trajectory Simulation Code

1. Introduction

A seven-member committee was designated in November 1975, to investigate the need for a general purpose flight-path trajectory calculation code that would replace the various existing codes being used by the Aerodynamics Department. The code specifications included:

a. The special features of the existing codes
b. A numerical integrator that is constructed to the current state-of-the-art and that could be easily replaced when solving different types of problems
c. A data table storage and retrieval package for handling large and multi-independent-valued tables
d. An input card reader and decoder capable of accepting and storing various forms of data
e. A set of six-degree-of-freedom and point-mass motion equations that incorporate the effects of an offset center-of-mass, a thrusting and swiveling propulsion system, a control system, and an aerodynamic force and moment evaluation
f. Modular construction for ease of modification.

The investigating committee decided that it would be possible to incorporate these requirements into a workable general purpose code. Consequently, in January 1976, a nine-member working group was appointed to develop the design philosophy for the proposed code.

The AMEER code was then formulated and written, principally by the author with support on special sections from various members of the working group.

The code is currently operational on the Sandia CDC 7600 computing facility.

2. Discussion

2.1 General Code Description

The AMEER code is a general purpose trajectory-flight path calculator that predicts the path of an object as it moves over an oblate-rotating planet under the influence of the planetary atmosphere and gravitational field. The referenced Geoid is by Fischer[2] with a fourth spherical harmonic polynomial evaluation for the gravitational components. The atmospheres are discussed below under a separate heading.

Two sets of differential equations of motion are available to evaluate the vehicle motion as a function of time. One is a six-degree-of-freedom set formulated by Hodapp[3] and the other is a point mass formulated by W. E. Williamson. These are described below under separate headings. The code is so structured that other sets of motion equations can easily be substituted and one can take advantage of the modular construction of the supporting routines; i.e., each routine performs a single task and returns one or more related values.

The integration routine incorporated in the code is the Sandia Mathlib RKF[4] (Fehlberg fourth-fifth order Runge-Kutta method) routine designed primarily to solve nonstiff or mildly stiff differential equations, and will provide fast execution time and good accuracy for most trajectory calculations. For higher accuracy, stiff equations, or faster execution, other Mathlib routines are available and can be easily substituted because of the nearly identical argument lists and structure.

2.2 Operational Procedure

The code operates in a phased mode in which three separate and distinct phases are available for code execution: initialization, computation, and formatted output. The passage of information (variable values) between the initialization and computational phases is through the use of labeled COMMON blocks, whereas, a file of data (TAPE20) from the computational phase is used for the output phase.

During initialization and before data read-in, core storage is "zeroed" out and a set of nominal (default) values stored to assure that there are no extraneous (garbage) values for code initial values.

After data read-in, data tables are checked for action status (input), atmospheric-properties data arrays are selected, and core locations are determined for the variable values associated with the time-history print-block array, the staging-variable array, and the integration-variable arrays. The main-program and support-routine initial values are computed and stored with the integrator-state variable values stored in the integration array.

During the computational phase, the evaluation of code variables is divided into two categories: those that have a direct effect on the main-program derivative evaluations and are computed while the integrator is trying to establish a good time step; and those that are informational in nature and are computed after a good time step has been established. This division of computation reduces running time and establishes a convenient location for insertion of other computations.

There are three methods of displaying available computed data. A time history block print of code items that is written to the output file at selected intervals during execution—this is the default type of operation. After execution to the stop criteria, tabular listings of related data and X-Y plots of selected items are generated at user request.

2.3 User Perogatives

The code is user-oriented in that the user must specify the main program (set of differential motion equations), the support routines, and the initial values for problem start that differ from the preset nominal (default) values. Execution request flags indicate the main program and support routines necessary for solving the problem. Statement type card images specify the initial conditions. These make up the bulk of the data deck.

The code will accept multiple problem-solving requests, whether they are completely independent problems ("Cases") or perturbations of one problem (the "Base Case"). Additions of data from the INPUT file to change or extend the "Base Case" is referred to as "Merging." The grouping of multiple "Cases" as input data is called "Stacking."

Within the "Base Case," values for as many as four variables will be accepted as code-execution stopping criteria, after which additional data may be read in from the "Base Case" file and execution allowed to continue. This stopping and restarting of the code execution is called "Staging." Consequently, the values and variables mentioned above are called "Staging Values" and "Staging Variables," respectively. Some aspects of the code-input data deck have been referred to above, however, one feature of the data input that deserves special attention is the data table makeup. The user has complete control when specifying the table independent parameters. These may be any code-defined variable name in any order and, during execution, they may be changed or reordered to specify new tables.

2.4 Data Preparation

A description of the card images that make up the data deck is given in detail in the report by Jones.[5] A version pertinent to this code is given in Section 3, "Input Data Declaration and Specification."

2.5 Main Programs

The following description of the main programs currently available are intended to familiarize the user with their capabilities.

2.5.1 Six-Degree-Of-Freedom (SIXDOF)

The main program set of motion equations selected by the $SIXDOF$ designator computes a time-dependent history of the motion of a translating and rotating rigid-body vehicle as it moves over an oblate-rotating planet. The derived six-degree-of-freedom equations of motion include the following effects:

a. A movable offset center-of-mass
b. Forces and moments from control-surface misalignments
c. Wind shear and gusts
d. Aerodynamic forces and moments
e. Propulsion-system forces and moments
f. Gravitational accelerations as a function of altitude
g. Atmospheric properties as a function of attitude

2.5.2 Point Mass (PTMASS)

The main program set of motion equations selected by the $PTMASS$ designator computes a time-dependent history of the motion of a point source as it moves over an oblate-rotating planet. The derived translational equations of motion include the following effects:

a. Forces due to commanded control items
b. Wind shear and gusts
c. Forces due to aerodynamic considerations
d. Forces due to a propulsion system
e. Gravitational accelerations as a function of altitude
f. Atmospheric properties as a function of altitude.

2.6 Support Routines

The description of the following support routines are intended to familiarize the user with their capabilities and are to be used as an aid in selecting the correct combination of routines to solve the problem.

2.6.1 Aerodynamics (AERO)

Aerodynamic forces and moments are computed for the six-degree-of-freedom evaluation. Forces only are computed for the point-mass calculations. The forces and moments are evaluated from total coefficients that are obtained from a polynomial summation of aerodynamic coefficients. These coefficients are determined by linear table lookup of tabulated input data or by an input constant. The polynomial terms indicate the effects of angular rotation, trim angle, angle-of-attack, angle-of-sideslip, control surface misalignments, translational and rotational damping, and magnus effects.

A sketch of the axis system for the coefficient sign convention is shown in Appendix B, Figure B-1. The default-axis system is the body axis; however, a wind-axis system may be substituted if the appropriate flag is set and data submitted.

2.6.2 Atmosphere (ATMS)

Atmospheric properties as a function of geodetic altitude are provided in data-array form for the 1962 US Standard atmosphere,[6] the 1966 Supplemental atmospheres[7] ranging from 15° to 75° north latitude for various monthly conditions and temperature variations, and four seasonal Tonopah, Nevada atmospheres. Space is provided for the user to supply an atmosphere or to accept measured altitude, pressure, and density values from a site. A list of the available atmospheres is given in the atmosphere section of the required-input data.

2.6.3 Control

2.6.3.1 Six-Degree-Of-Freedom (CONTRL)

Attitude control is provided for the vehicle whose path is being simulated by the SIXDOF set of equations. A limited amount of control is permitted through lookup of tabular values of fin/tab deflections. Full control of the vehicle is not provided because a generalized-feedback loop-control package is very much vehicle-problem dependent. However, a control package could be substituted for the table lookup coding with very little modification.

2.6.3.2 Point Mass (PTMSCO)

For the PTMASS equations evalution, control is provided in the wind-axis reference system in the form of tables of commanded items. Newton's method is used to predict the aero angles necessary to fly the correct path. Aerodynamic forces due to the aero angles provide the required inputs to the motion equations. The five sets of commanded items are listed under the control section of the required input data.

2.6.4 Station/Radar (RADAR)

Radar "look" items: azimuth, elevation, and slant range are computed relative to a designated station position. The azimuth is measured relative to an initial heading defined for the station. The station position is defined in geodetic coordinates of latitude, longitude, and altitude.

2.6.5 Propulsion System (THRUST)

A thrusting- and swiveling-propulsion system of one to five motors is available for the SIXDOF or PTMASS evaluations. Vacuum thrust is specified for each motor during input, and is corrected for pressure-altitude effects before the resultant forces and moments are computed. The time rate of changes of mass and the acceleration of the mass due to fuel flow are obtained by table lookup. The mass flow is integrated to determine the current mass loss and the instantaneous mass of the vehicle. The mass flow and acceleration effects are accounted for in the SIXDOF evaluations as terms in the force and moment equations, but are not included in the PTMASS evaluations.

The thrust force and moment components are referenced to the body-axis system in the SIXDOF evaluations whereas in the point-mass evaluations they are referenced to the wind- or velocity-axis system. Thrust misalignment effects due to motor-nozzle cant or nozzle swivel are transformed to the reference axis through the use of a swivel angle—measured with respect to the longitudinal axis—and a rotation angle of the plane that contains the angle line segments.

The effects of a rate of change of the nozzle exit point (swivel) are included in the SIXDOF equations.

A sketch showing the pertinent parameter data, the swivel angle, and the axis-system orientation is given in Appendix B, Figure B-2.

2.6.6 Vehicle Mass Properties (VEHMSP)

The vehicle mass-properties routine provides a method of introducing the various vehicle physical characteristics and mass properties into the problem solution.

Mass properties—moments of inertia and moments of inertia rates—are determined by linear table lookup of input-data tables. The center-of-mass position referenced to datum coordinates is also determined by table lookup and then transferred to values with respect to the body-reference point.

A sketch showing the pertinent parameters and the reference axis systems is given in Appendix B, Figure B-3.

2.6.7 Gravity Model (GRAV)

The gravity model assumes the planet earth is the geoid being traversed by the vehicle during its trajectory. Gravitational acceleration components in the radial (increasing latitude) and tangential (toward planet center) are computed from the gravity potential equation using the fourth spherical harmonic to define the gravity potential of the planet. Components of gravitational acceleration along the X and Z axes of the local geocentric system are returned to the main program for inclusion in the equations of motion. The Y-component, of course, is zero.

2.7 Data Output

As mentioned in Section 2.2, there are three methods of displaying computed data available to the code user: the time-history block print (default method), the tabular lists of selected items, and Y vs X plots of selected data values.

2.7.1 Block Print

The time-history print block is made up of selected variable values from the main program and the auxiliary calculations routine. Associated with the print block are lists of variable values computed by the active support routines. These data are dumped to output at user-specified intervals.

2.7.2 Tabular Lists

When the code solution has progressed to the stopping criteria, sets of preselected variable values are made available to the user for listing in tabular form. These sets consist of related trajectory items, stability items, atmospheric properties and air loads, mass properties, g-loadings, and accelerations. Space has also been provided for the user to specify a set of variables to be listed along with the format and column headings. The table lists are presented in Appendix C.

2.7.3 Y vs X Plots

With the code-stopping criteria met, graphic displays of computed variable data in the form of Y vs X plots are available for user selection. No preselection of plots have been made because of the infinite number of choices possible. The user must specify the dependent variable and one or more independent variables for each plot, as well as the stacking request.

2.7.4 TAPE20 File

A data file (TAPE20) consisting of blocks of code-computed variable values is generated during code execution. The file is made up of a header-block of Hollerith names followed by blocks of values written in binary mode. There is a one-to-one correspondence between the header-block names and the values in the following blocks. The binary blocks are written at the selected print intervals. The plot arrays are filled and the table lists are generated using this file. By saving the file after execution,the trajectory data would be available for other uses. A list of the block variable names is given in Appendix C.

2.8 Description of Appendix A

Appendix A presents a logic-flow chart to be used as an aid in selecting the main program and its supporting routines when assembling the data into the data deck for program start. It is also intended to familiarize the novice user with the code requirements when used in a question-and-answer mode. For the more experienced user, using the flow chart in conjunction with the required input-data section will allow him to formalize the structure of his data deck. There is a one-to-one correspondence between the logic chart and the input-data section.

Sample data decks are included for reference.

2.9 Description of Appendix B

Assembled in Appendix B are the computational-logic flow charts of all the major routines associated with the code and, where relevant, the equations evaluated by the routine during execution. Sketches are provided for those routines that required dimensional data for computation initialization. The routines are grouped according to the functional task they perform during execution. A brief description of the functional relationship between the routines, main program, and code is given at the beginning of each grouping. This grouping of routines is intended as an aid in understanding the sequence of events when it becomes necessary to modify the code.

2.10 Description of Appendix C

Appendix C presents a listing along with a short description of the code variable names (mnemonics), the variable names associated with the block output, and the names of the variables identified with each table available for listing from the tabular output routine.

3. Input Data Declaration and Specification

3.1 General Data Card Information

Data are supplied to the code in line (card image) form. The format is "free form"; i.e., data items can be presented in any practical form, as long as the individual values are delimited by a separator symbol. The allowable symbols are:

a. One or more blanks
b. A comma
c. An equal sign
d. Any number of blanks, followed by a comma or equal sign, followed by any number of blanks (where "any number" includes zero)

Special delimiters: $, /, // are used, in context below, to specify various types of data.

Input lines are classified as initial lines or continuation lines. The general form of the initial line is: code designator or variable symbol (set of characters), subscript indicator (if required), a delimiter-separator, the variable value, and a delimiter-separator or the termination symbol. The variable value-delimiter combination may be repeated as required to satisfy the storage list.

A symbolic representation of the general form is

SYMBOL [=] $VALUE_1$ [,] $VALUE_2$ [,] $VALUE_3$

where [ ] denotes any delimiter-separator. This form is used below to describe the specification forms associated with the various types of input data and, with judicious use of the code symbol, the allowable separators, and the value, the data line can be made to resemble the FORTRAN statement or any other readable statement.

The reading of the data line is terminated when all 80 columns have been read or when the terminator symbol "//" (two immediately adjacent slashes) is encountered. Therefore, comments about the card can be placed after the terminator. This feature is used below when the card examples require an explanation.

When the data value list exceeds the card space available, continuation lines (cards) may be used. They are of the same form and usually immediately follow the initial card. The variable symbol with the appropriate subscript may be used; however, an alternate method is to leave at least the first six columns blank—generally the variable symbol is omitted. If the alternate method is used, the lines of data must be sequential and any number of continuation lines may be used. The requirement for leaving the first six colunms blank also implies the initial line should begin before column 6.

An acceptable first card (initial line) must contain a minimum amount of information. The minimum for each input-data type is listed below.

Types of Input Data	Minimum Information
Variable definition	The variable name
Array definition	The array name and subscript (if used)
TABLE declaration	Name of table
BREAKS definition	Name of the table and the independent variables
BODY definition	Name of table and subscript (if any)
DELETE	No minimum
FORMAT	No minimum, but the format specifications must be completed on one card.

The following sections describe the use of the various data definition types.

3.2 Input Variable Name Forms

The program variable name (SYMBOL) recognized by the code-read routine is made up of a maximum of six characters that denote the engineering notation, the reference frame, and the units associated with the variable.

Characters 1-3: represent the engineering or mathematical notation
2-4: the reference frame
5-6: the units

Example: VGD8M: V_{gd} - Geodetic velocity (m/s)
$\dot{X}$I88M1: $\dot{X}_i$ - X-direction inertial velocity (m/s)
GAMGCD: γ_{gc} - Elevation flight path angle, geocentric reference (deg)

NOTE: If the notation characters begin with an integer-implying character (L thru N) for a floating point variable name, the character F precedes the character set.

Ex: FMACH - Mach number

Support routines are requested and activated by a variable name that begins with the characters IX and is followed by a character set that denotes the support routine it activates.

Ex: IXTHRS activates the thrust routine.

The main programs or those support routines that offer a choice of evaluations begin with the characters IM or IS followed by the character combination PROG.

Ex: IMPROG = $SIXDOF$

3.3 Input Value Forms

The numeric "integer" value is any string of digits with no embedded blanks or special characters; the exception is a plus or minus sign that immediately precedes the first digit. For example:

12, +12, -12 0012

The numeric "floating point" value is any string of digits containing an embedded decimal point, but no embedded blanks. It is of the same form as the FORTRAN floating-point constant.
For example, the floating point 1 may be represented thusly:

1. 1.0 +1.0 1.0E0 etc

The "Hollerith" data value is any string of alphanumeric characters including blanks that is begun and ended with the dollar sign character, $. This character flags the reader routine to handle the delimited data accordingly.
For example:

$ THIS IS HOLLERITH DATA $

3.4 Data Definition Forms

Under FORTRAN specifications, data values are recognized as single valued or an array of values. Data items (values) that are to be submitted to the code must be defined to the code-reader routine in such a way as to satisfy the specifications. Therefore, three methods of defining the input data have been chosen which are synonymous with the storage criteria. They are single variable, array, and table definition, with the table definition a special form of the array definition.

3.5 Variable and Array Forms

All variables and arrays must have the default type implied by the usual FORTRAN naming convention (i.e., all names beginning with I through N are type integer; all other names are type real).

The single-valued variable is defined by the statement form:

```
Integer:        I  [=]  integer
Floating Point: A  [=]  value
```

Examples:

Integer		Floating Point	
I = 0		A = 1	
K,7	// same as K = 7	AK, 7.	// same as AK = 7.
LL 50	// same as LL = 50	AL 1.11E1	

The array values, whether integer or floating point, are defined by a similar statement form:

```
Integer:        I(i,j,...n)  [=]  integer  [,]  integer[,]  etc
Floating Point: A(i,j,...n)  [=]  value    [,]  value    [,]  etc
```

The storage start of the first item of the array is indicated by the name, a null subscript, or all subscript entries equal to 1. For subsequent first item storage, a location indicated by the subscript denotes the start; storage then proceeds according to the usual FORTRAN rules of storage (first subscript varies the fastest, etc).

Some examples:

```
X = 1. 2. 3.
X(1) = 1., 2., 3.   // same as above
IX(1) = 1, 2, 3
Y(1,1) = 1. 2., 3.  4.0E-4
Y(2,3) = 4.44E4, 3.33E3, 2.22E2, 1.11E1
DOG(1, 7, 2, 1) = 0., 0., 0.
```

3.6 Hollerith Data Forms

Hollerith data must be assigned to integer-type variables only. The single variable and array definition form is as follows:

```
I          [=] $Hollerith data$
I(i,j,...n) [=] $Hollerith data$
```

Data following the second dollar sign is ignored. If the second dollar sign or the "//" terminator symbol is not present, all characters to the end of the card are read and stored, six characters per word. The first character, including blanks, that follows the first dollar sign is stored as the first character of the variable or array element. When defining a single-valued variable, the Hollerith field should contain no more than six characters to prevent overstorage. For array storage, the string of Hollerith characters is read and stored, six characters per array element. Continuation of Hollerith data onto the next card is allowed and follows the continuation card rules explained above, with the exception that the dollar sign must precede the start of the new data. It should be noted that the packing of the Hollerith data, from the continuation card into the array being filled, continues at the very next character position in the array element and *not* at the beginning of the next array element. To specify the array element, use the array subscript on the continuation card.

Some examples:

```
IMPROG  = $SIXDOF$
ITITLE(1) = $TITLE OF PROGRAM$
NAME      $FIRST $
          $MIDDLE$          Notice continuation
NAME (3) = $ LAST $  // notice continuation
NAME (1) = $FIRST MIDDLE LAST$  // same as above three lines
```

3.7 Table Data Forms

Tables of data values are defined for the code through a TABLE declaration card, a table BREAKS card set, and a table BODY card set. All possible combinations associated with the table card set are not given here—only the major points. For more detail, the user is referred to the report by Jones.[4]

3.7.1 TABLE Card Form

The TABLE card requests space for the table (whose name follows), specifies the table's dimensions, designates the independent variables, and allows table data array value modifiers to be specified. It has the following form:

TABLE name $spec_1$ $spec_2$ $spec_3$ etc ,

where

"name" is the table name with no embedded blanks.

"Dimension" specification ($$(m)i,j,...,k) implies the table is dimensioned i by j by...,by k (total six) with m values for each entry. If m is one, "m" may be omitted. Dollar signs ($) denote where data are to be stored: no dollar sign—table stored in memory, one dollar sign—table stored in extended memory, two dollar signs—data storage is on disk or tape (file ITABL). If possible, the table should be stored so that the last dimension is associated with a program variable that varies the slowest during execution.

"Independent Variable" specification/$variable_i$, $variable_j$, ..., $variable_k$/ specifies the name of the program variable associated with the dimension.

"Scaler Multiplier" specification *value causes the result of the table interpolation to be multiplied by the value.

"Scaler Adder" specification ± value causes the result of the table interpolation to be increased or decreased by the value.

3.7.2 BREAKS Card Form

The BREAKS card specifies the list of values—"breakpoint" list—associated with the independent variable. It has the following form:

BREAKS name /variable name [index]/value [,] ... [,] value

where

"name" is the table name
"variable name" is the name of the independent variable
"index" is the position within the array to which the first value belongs
"value" is the numeric value.

3.7.3 BODY Card Form

The BODY card defines the body or contents of a table. It has the following form:

BODY name(i,j,...,k) [=] value [,] value [,] ... [,] value

where

"name" is the table name
"(i,j,...k)" is the subscript corresponding to the first value.

NOTE: Form is the same as the array definition form.

Example:

The three (3) dimensional table, TCOEF1, has dimensions 4 by 3 by 2 with ALPHA (α), BETA (β), and MACH NO (M) as the independent variable names. The dependent coefficient (C) is tabulated below as a function of the independent variables. The coefficient is to be decreased by 10% and offset by a factor of 0.05.

Table: TCOEF1

M_1 = 0.0

α \ β	0.0	5.0	10.0
0.0	C_{111}	C_{121}	C_{131}
10.0	C_{211}	C_{221}	C_{231}
20.0	C_{311}	C_{321}	C_{331}
30.0	C_{411}	C_{421}	C_{431}

M_2 = 10.0

α \ β	0.0	5.0	10.0
0.0	C_{112}	C_{122}	C_{132}
10.0	C_{212}	C_{222}	C_{232}
20.0	C_{312}	C_{322}	C_{332}
30.0	C_{412}	C_{422}	C_{432}

The TCOEF1 table would be described as follows:

```
TABLE     TCOEF1(4,3,2)  /ALPHA, BETA, FMACH/  *0.9 +0.05
BREAKS    TCOEF1  /ALPHA/  0.0 10.0 20.0 30.0
BREAKS    TCOEF1  /BETA/  0.0 5.0 10.0
BREAKS    TCOEF1  /FMACH/  0.0 10.0
BODY      TCOEF1          C111 C211 C311 C411  //  FORTRAN array storage
                          C121 C221 C321 C421
                          C131 C231 C331 C431
```

BODY	TCOEF1(1,1,2)	C_{112}	C_{212}	C_{312}	C_{412}	//BODY	TCOEF1 need not be repeated
		C_{122}	C_{222}	C_{322}	C_{422}		
		C_{132}	C_{232}	C_{332}	C_{432}		

NOTE: If ALPHA and BETA vary at the same rate, reordering the variables and subscripts would produce a list similar to table above.

Two cards associated with the table description are the DELETE and FORMAT cards.

3.8 DELETE Card Form

The DELETE card is a request to purge all items associated with the named table. The allocated storage space is freed, and the table name is made available for reuse. It has the following form:

DELETE [ ] name [,] name [,]... [,] name

where
"name" is the name of the table to be "DELETE"ed.

3.9 FORMAT Card Form

The FORMAT card specifies a format by which the bodies of certain tables are to be read. The tables are those which have the "FORMAT" specification on their TABLE card. It can have the following forms:

FORMAT n
FORMAT (format specification)
FORMAT n (format specification)

where
"n" is the number of items to be read
"format specification" is of the same form as the FORTRAN format statement

The advantage of using the FORMAT mode is that keypunch or data errors are more likely to be diagnosed by the system than the more flexible code-reader routine.

3.10 Data Editing

Stored data editing is accomplished by "overstoring" the new value onto the old value. That is, redefining the single variable with a new statement of the value or by selectively redefining an array as table data values through the use of array subscripts.

Example:

```
X(1)  =  1., 2., 3., 4., 7., 9.5, 11.
  .
  .
  .
X(7)  =  3.7    // X(7) was 11., is now 3.7
  .
  .
  .
X(7)  =  2.9    // X(7) is now 2.9
```

3.11 Data Equivalencing

A data submission feature that is akin to "Data Editing" of an array is available to the user through the EQUIVALENT ARRAY name that is associated with sequentially stored variables (i.e., LABELED COMMON BLOCK variables). Use of the EQUIVALENT ARRAY name on the defining statement card allows the user to make use of the array form of data entry to store values for the single variable items (i.e., overstore).

Example:

```
FIXXK      = 4.89  // single valued variable input
FIYYK      = 8.62
FIZZK      = 8.62
XYZMIS(1)  = 4.89, 8.62, 8.62 //Equivalent Storage
```

It should be noted that if the EQUIVALENT ARRAY name is used to store data originally, it does not inhibit the use of the single valued variable name to "data edit."

4. Required Input Data

The required input data has been divided into categories of general code items, general problem items, specific problem items, general output requests, and code and program execution control.

The general code items allow the specification of planet properties and integration control. The general problem items specify the initial conditions: vehicle position, velocity vector, time, mass, etc, and the selection of the main program and support routines. The specific problem items are the result of the main program and support routine selection. The general output requests allow the user to designate the disposition and display of the computed data. Program execution control is specified by the staging parameters whereas code control is accomplished by specifying the *CONTROL cards.

The following lists provide the user a choice of at least one nominal (default) value for each item which, if accepted, requires no data specification statement for that item.

4.1 General Code Items

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Reference Geoid (Fischer 1960-Mercury)	IEARTH				$OBLATE$	
Spin Rate	OMGPLR			rad/s	7.29211508E–5	ω_p
Equatorial Radius	REQ8M			m	6.3781660E+6	R_{eq}
Polar Radius	RPOL8M			m	6.35678428E+6	R_{pol}
Flattening Factor	BETER			-	~~0.0067385~~	B_f
Reference Gravity	GREF			m/s^2	9.80665	g_{ref}
Gravity Coefficients (4th Spherical Harmonic)	H1			-	1.62341E–3	H_1
	H2			-	6.04E–6	H_2
	H3			-	6.37E–6	H_3
Geocentric Constant	FMU			m^3/s^2	3.98615683E+14	μ
Integration Control:						
Absolute Error	ABSERR			-	1.0E–7	ϵ_a
Relative Error	RELERR			-	1.0E–4	ϵ_r
Input Data Conversion Choice: 0-No (Metric) 1-Yes (English to Metric)	ICNVRT				0	

4.2 General Problem Items

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Identification:						
Report Title	ITITLE (30)				$BLANK$	
Problem Ident	ICASE (1)				$BLANK$	
Planet Location:						
Surface Reference	IDLAT (1)				$GEODET$	
Choice: $GEODETIC$	IDLAT (2)				IC	
$GEOCENTRIC$						
$GEODETIC$ Reference:						
Initial Altitude	HGDZM	POSZGD(1)		m	0.0	h_{gd_o}
Initial Latitude	PHGDZD	(2)		deg	0.0	ϕ_{gd_o}
Initial Longitude	THTLZD	(3)		deg	0.0	θ_{L_o}
Current Altitude	HGD8M	POSCGD(1)		m	0.0	h_{gd}
Current Latitude	PHIGDD	(2)		deg	0.0	ϕ_{gd}
Current Longitude	THTLD	(3)		deg	0.0	θ_L
Velocity Vector:						
Total Velocity	VTGDM	VELVGD(1)		m/s	0.0	$V_{T_{gd}}$
Elevation F.P.A.*	GAMGDD	(2)		deg	0.0	γ_{gd}
Horizontal F.P.A.	SIGGDD	(3)		deg	0.0	σ_{gd}
$GEOCENTRIC$ Reference:						
Initial Longitude	THTLZD	POSZGC(1)		deg	0.0	θ_{L_o}
Initial Latitude	PHGCZD	(2)		deg	0.0	ϕ_{gc_o}
Initial Altitude	HGCZM	(3)		m	0.0	h_{gc_o}
Current Longitude	THTLD	POSCGC(1)		deg	0.0	θ_L
Current Latitude	PHIGCD	(2)		deg	0.0	ϕ_{gc}
Current Altitude	HGC8M	(3)		m	0.0	h_{gc}
Velocity Vector:						
Total Velocity	VTGCM	VELVGC(1)		m/s	0.0	$V_{T_{gc}}$
Elevation F.P.A.	GAMGCD	(2)		deg	0.0	γ_{gc}
Horizontal F.P.A.	SIGGCD	(3)		deg	0.0	σ_{gc}

*Flight Path Angle

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Initial Heading	SIGZD			deg	0.0	σ_o
Initial Time	TIMEZS			s	0.0	t_o
Print Interval	DELTS			s	0.0	Δt
Maximum Time	TMAXS			s	0.0	t_{max}
Stage Time Increment	TIMSDX			s	0.0	Δt_s
Number of Good Time Steps/Print	NTSTEP			–	0	N_{TS}
Initial Mass	FMASZK			kg	0.0	m_o
Adder-Init Mass	EAMASZ			kg	0.0	ϵ_{m_o}
Gravity Option:						
Execution	IXGRAV				0	
Choice: 0-No Execution						
1-Yes (see routine GRAV requirements)						
Wind Option:						
Execution	IXWIND				0	
Choice: 0-No						
1-Yes (enter tables below)						
X-Direction (+S→N)	XWGDM1		TXWIND	m/s	0.0	$\dot{X}_{wgd}$
Y-Direction (+W→E)	YWGDM1		TYWIND	m/s	0.0	$\dot{Y}_{wgd}$
Z-Direction (+U→D)	ZWGDM1		TZWIND	m/s	0.0	$\dot{Z}_{wgd}$
Atmosphere Option:						
Selection	IATMNM				$62STD$	
Choice: (see routine ATMOSPHERE choices)						
Main Program:						
Selection	IMPROG				$SIXDOF$	
Choice: $SIXDOF$						
$PTMASS$						
Number of Bodies:	NBODY				1	

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Main Program Auxillary Calculations:						
Selection:						
Load Factors:	IXLODF				0	
Choice: 0-No calculations						
1-Calculate and print: XYZ-Axis loads (g's)						
Ranges	IXRNGE				0	
Choice: 0-No calculations						
1-Calculate and print: Down range						
Cross range						
Great circle range						
Total path traveled						
Surface Path Traveled						
(if initial ≠ current)						
Total Path	TPATHM			m	0.0	T_{PATH}
Station/Radar	IXSTA				0	
Choice: 0-No calculations						
1-Calculate and print station-relative items:			Slant range			
			Azimuth			
			Elevation			
			Velocity components			
(if IXSTA = 1)						
Station Altitude	HGDSTM	STALOC(1)		m	0.0	$h_{s_{gd}}$
Station Latitude	PHGDSD	(2)		deg	0.0	$\phi_{s_{gd}}$
Station Longitude	THTSTD	(3)		deg	0.0	$\theta_{s_{gd}}$
Azimuth Heading	PSISTD	(4)		deg	0.0	$\Psi_{s_{gd}}$
Support Routine Selection:						
Aerodynamic (see routine AERO)						
Selection	ISPROG				$SDAERO$	
Choice: $SDAERO$						
$PMAERO$						
Execution	IXAERO				0	
Choice: 0-No						
1-Yes (compute forces and moments)						
Thrust (see routine THRUST)						
Selection	IXTHRS				0	
Choice: 0-No						
1-Yes (Execution)						

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Vehicle Mass Properties (see routine VEHMSP)						
Selection	IXVMP				0	
Choice: 0-No						
1-Yes (Execution)						
Control						
IMPROG = $SIXDOF$ Requested						
Selection	IXSDC				0	
Choice: 0-No						
1-Yes (Execution: see routine CONTRL)						
IMPROG = $PTMASS$ Requested						
Selection	IXPTMC				0	

Choice: 0 - No
Yes (Execution: see routine PTMSCO)
= 1, Aerodynamic coefficient control (CD, CL, CY)
= 2, Commanded aero-angle control (α_c, β_c)
= 3, Commanded pitch-and-yaw angle control (θ_c, ψ_c)
= 4, Load factor control
= 5, Flight-path angle control (γ,σ)
= 6, User supplied routine-required

4.3 Specific Problem Items: Main Program

4.3.1 Six-Degree-of-Freedom (SIXDOF)

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Datum Reference	IDATUM				$NOSE$	
Datum Coordinates						
X-Direction	XDTM8M	XYZDTM(1)		m	0.0	X_{DTM}
Y-Direction	YDTM8M	(2)		m	0.0	Y_{DTM}
Z-Direction	ZDTM8M	(3)		m	0.0	Z_{DTM}
Body Reference Point (BRP) Coordinates (w.r.t. DATUM)						
X-Direction (+FWD)	XBRP8M	XYZBRP(1)		m	0.0	X_{BRP}
Y-Direction (+RIGHT)	YBRP8M	(2)		m	0.0	Y_{BRP}
Z-Direction (+DOWN)	ZBRP8M	(3)		m	0.0	Z_{BRP}
Aero-Angle Start Option Selection	IDANGL				$EULFPA$	
Choice: $EULFPA$ (see general item: Velocity Vector) $AROFPA$						
Angle-of-Attack	ALPHAD			deg	0.0	α
Angle-of-Sideslip	BETAD			deg	0.0	β
Body Roll Angle	PHIVD			deg	0.0	ϕ_v
Euler Angle Sequence Selection	IBODSQ				YPR	
Choice: YPR, PRY, PYR, RPR						
$GEODETIC$						
Yaw	PSBGDD	EULRGD(1)		deg	0.0	$\Psi_{B_{gd}}$
Pitch	THBGDD	(2)		deg	0.0	$\theta_{B_{gd}}$
Roll	PHBGDD	(3)		deg	0.0	$\phi_{B_{gd}}$
$GEOCENTRIC$						
Yaw	PSBGCD	EULRGC(1)		deg	0.0	$\Psi_{B_{gc}}$
Pitch	THBGCD	(2)		deg	0.0	$\theta_{B_{gc}}$
Roll	PHBGCD	(3)		deg	0.0	$\phi_{B_{gc}}$
Body Rotation Rates Selection		IDPQR(1)			$INERTI$	
		(2)			AL	

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Choice: $INERTIAL$						
Roll Rate	PIBPR	PQRIBR(1)		rad/s	0.0	p_i
Pitch Rate	QIBPR	(2)		rad/s	0.0	q_i
Yaw Rate	RIBPR	(3)		rad/s	0.0	r_i
Choice: $PLANET$						
Roll Rate	PEBPR	PQREBR(1)		rad/s	0.0	p_e
Pitch Rate	QEBPR	(2)		rad/s	0.0	q_e
Yaw Rate	REBPR	(3)		rad/s	0.0	r_e
Choice: $CONSTANTS$						
Roll Rate	PIBPR			rad/s	0.0	p_i
Auxillary Calculations						
Yawsonde Selection	IXYAWS				0	
Choice: 0-No 1-Yes						
Sun Elevation	TSUNGD			deg	0.0	θ_{SUN}
Sun Azimuth	PSUNGD			deg	0.0	Ψ_{SUN}
Critical Frequency - Pitch Selection	IXCRFQ				0	
Choice: 0-No 1-Yes						
NOTE: C_{m_α}, C_{m_q} VALUES MUST BE SPECIFIED						
Accelerometer Selection	IXACL				0	
Choice: 0-No 1-Yes (see routine ACCELM)						

4.3.2 Point Mass (PTMASS)

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Roll-Bank Angle Selection Choice: 0-No 1-Yes	IXROLT				0	
Roll Angle	PHIA8D		TROLLA	deg	0.0	ϕ_a

4.4 Specific Problem Items: Support Routines

4.4.1 Aerodynamics

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Selection Choice: $SDAERO$ $PMAERO$	ISPROG				$SDAERO$	
Execution Choice: 0-No 1-Yes	IXAERO				0	
Axis System Choice: 0-BODY 1-WIND	IXAROB				0	
Reference Dimensions						
Area	SREFAM	SXYZRF(1)		m^2	0.0	S_{ref}
X-Length	DREFXM	(2)		m	0.0	ℓ_{R_x}
Y-Length	DREFYM	(3)		m	0.0	ℓ_{R_y}
Z-Length	DREFZM	(4)		m	0.0	ℓ_{R_z}
Force Coefficients						
X-Direction	CXZERO	CX(1)	TCX0		0.0	C_{x_o}
	CXALPH	(2)	TCXA	1/deg	0.0	C_{x_α}
	CXALP2	(3)	TCXA2	$1/deg^2$	0.0	$C_{x_\alpha}{}^2$
	CXBETA	(4)	TCXB	1/deg	0.0	C_{x_β}
	CXBET2	(5)	TCXB2	$1/deg^2$	0.0	$C_{x_\beta}{}^2$
	CXDELQ	(6)	TCXDQ	1/deg	0.0	$C_{x_{\delta q}}$
	CXDELR	(7)	TCXDR	1/deg	0.0	$C_{x_{\delta r}}$
Y-Direction	CYZERO	CY (1)	TCY0		0.0	C_{y_o}
	CYBETA	(2)	TCYB	1/deg	0.0	C_{y_β}
	CYBET2	(3)	TCYB2	$1/deg^2$	0.0	$C_{y_\beta}{}^2$
	CYBET3	(4)	TCYB3	$1/deg^3$	0.0	$C_{y_\beta}{}^3$
	CYDELR	(5)	TCYDR	1/deg	0.0	$C_{y_{\delta r}}$
	CYP	(6)	TCYP	1/rad	0.0	C_{y_p}
	CYR	(7)	TCYR	1/rad	0.0	C_{y_r}
	CYBDOT	(8)	TCYBDT	1/rad	0.0	$C_{y_{\dot\beta}}$

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Z-Direction	CZZERO	CZ (1)	TCZ0		0.0	C_{z_o}
	CZALPH	(2)	TCZA	1/deg	0.0	C_{z_α}
	CZALP2	(3)	TCZA2	$1/deg^2$	0.0	$C_{z_\alpha}{}^2$
	CZALP3	(4)	TCZA3	$1/deg^3$	0.0	$C_{z_\alpha}{}^3$
	CZDELQ	(5)	TCZDQ	1/rad	0.0	$C_{z_{\delta_q}}$
	CZP	(6)	TCZP	1/rad	0.0	C_{z_p}
	CZQ	(7)	TCZQ	1/rad	0.0	C_{z_q}
	CZADOT	(8)	TCZADT	1/rad	0.0	$C_{z_{\dot\alpha}}$
Moment Coefficients						
About X-X (roll)	CLZERO	CL (1)	TCL0		0.0	C_{ℓ_o}
	CLALPH	(2)	TCLA	1/deg	0.0	C_{ℓ_α}
	CLBETA	(3)	TCLB	1/deg	0.0	C_{ℓ_β}
	CLDELP	(4)	TCLDP	1/deg	0.0	$C_{\ell_{\delta_p}}$
	CLP	(5)	TCLP	1/rad	0.0	C_{ℓ_p}
	CLR	(6)	TCLR	1/rad	0.0	C_{ℓ_r}
	CLBDOT	(7)	TCLBDT	1/rad	0.0	$C_{\ell_{\dot\beta}}$
About Y-Y (pitch)	CMZERO	CM (1)	TCM0		0.0	C_{m_o}
	CMALPH	(2)	TCMA	1/deg	0.0	C_{m_α}
	CMALP2	(3)	TCMA2	$1/deg^2$	0.0	$C_{m_\alpha}{}^2$
	CMALP3	(4)	TCMA3	$1/deg^3$	0.0	$C_{m_\alpha}{}^3$
	CMDELQ	(5)	TCMDQ	1/deg	0.0	$C_{m_{\delta_q}}$
	CMP	(6)	TCMP	1/rad	0.0	C_{m_p}
	CMQ	(7)	TCMQ	1/rad	0.0	C_{m_q}
	CMADOT	(8)	TCMADT	1/rad	0.0	$C_{m_{\dot\alpha}}$
About Z-Z (yaw)	CNZERO	CN (1)	TCN0		0.0	C_{n_o}
	CNBETA	(2)	TCNB	1/deg	0.0	C_{n_β}
	CNBET2	(3)	TCNB2	$1/deg^2$	0.0	$C_{n_\beta}{}^2$
	CNBET3	(4)	TCNB3	$1/deg^3$	0.0	$C_{n_\beta}{}^3$
	CNDELR	(5)	TCNDR	1/deg	0.0	$C_{n_{\delta_r}}$
	CNR	(6)	TCNR	1/rad	0.0	C_{n_r}
	CNBDOT	(7)	TCNBDT	1/rad	0.0	$C_{n_{\dot\beta}}$

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
If IXAERO = 0, constant forces and moments can be entered.						
Force						
X-Direction	XFARON			N	0.0	F_{xA}
Y-Direction	YFARON			N	0.0	F_{yA}
Z-Direction	ZFARON			N	0.0	F_{zA}
Moment						
About X-X	XMARON			N·m	0.0	M_{xA}
About Y-Y	YMARON			N·m	0.0	M_{yA}
About Z-Z	ZMARON			N·m	0.0	M_{zA}

4.4.2 Atmosphere

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Headers						
Selection	IATMNM				$62STD$	
Choice: $ZERO $	- Atmospheric Properties = 0.0					
$62STD $	- 1962 U.S. Standard					
$15NANL$	- 15°N Annual					
$30NJAN$	- 30°N January					
$30NJUL$	- 30°N July					
$45NJAN$	- 45°N January					
$45NJUL$	- 45° July					
$45NSF $	- 45°N Spring/Fall					
$60NJAN$	- 60°N January					
$60NJC $	- 60°N Janaury (Cold)					
$60NJW $	- 60°N January (Warm)					
$60NJUL$	- 60°N July					
$75NJAN$	- 75°N January					
$75NJC $	- 75°N Janaury (Cold)					
$75NJW $	- 75°N January (Warm)					
$75NJUL$	- 75°N July					
$TTRWIN$	- Tonopah Winter					
$TTRSPR$	- Tonopah Spring					
$TTRSUM$	- Tonopah Summer					
$TTRFAL$	- Tonopah Fall					
$USER $	- User Supplied Atmosphere					
$SITE $	- Site Measured Atmosphere					
User Supplied Atmosphere						
Selection	IATMNM				$USER$	
Molecular scale temperature-altitude gradient		FK1(25)		K/m	0.0	K_1
Molecular weight of air-altitude gradient		FK2(25)		kg/kg-mol-m	0.0	K_2
Mean molecular weight of air		FMB(25)		kg/kg-mol	0.0	M_B
Molecular weight of air	FMO			kg/kg-mol	0.0	M_O
Gravity at user latitude	GPHI			m/s^2	0.0	g_ϕ
Base of altitude layer		HB(25)		m	0.0	h_B
No. of geopotential layers	NLAYRS				0	N_L

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
No. of geometric layers	MLAYRS				0	M_L
Pressure at altitude layers		PB(25)		Pa	0.0	P_B
Earth radius at user latitude	RPHI			m	0.0	R_ϕ
Molecular scale temperature at altitude layer		TMB(25)		K	0.0	T_{M_B}
Geometric altitude layers		ZB(25)		m	0.0	Z_B
Maximum height of atmosphere	ZMAXM			m	0.0	h_{MAX}
Site Measured Atmosphere						
Selection	IATMNM				$SITE$	
Measured density at altitude		FK1(50)		kg/m^3	0.0	ρ_M
Measured pressure at altitude		PB(50)		Pa	0.0	P_M
Measured geometric altitude		ZB(50)		m	0.0	Z_B
Molecular Weight of air		FMO		kg/kg mol	0.0	M_O
No of geometric Layers		MLAYRS			0	M_L

4.4.3 Control

4.4.3.1 Six-Degree-of-Freedom

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Execution Choice: 0 - No 1 - YES	IXSDC				0	
Control Surface Deflection						
Rate About X-Y	DELPD	TDELP		deg	0.0	δ_p
Rate About Y-Y	DELQD	TDELQ		deg	0.0	δ_q
Rate About Z-Z	DELRD	TDELR		deg	0.0	δ_r

4.4.3.2 Point Mass

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Execution Choice: 0 - NO Yes - 1, Control by Aerodynamic coefficient (CD, CL, CY) - 2, Control by commanded aero-angles (α,β) - 3, Control by commanded pitch-and-yaw angles (θ,ψ) - 4, Control by wind-axis normal load factors (N_γ, N_σ) - 5, Control by elevation flight-path angle (γ) - 6, User specified control	IXPTMC				0	
IXPTMC = 1						
Drag Coefficient	CD		T1PMC		0.0	C_D
Lift Coefficient	CL		T2PMC		0.0	C_L
Yaw Coefficient	CY		T3PMC		0.0	C_Y
IXPTMC = 2						
Commanded Angle of Attack	ALPHAD		T1PMC	deg	0.0	α
Commanded Angle of Sideslip	BETAD		T2PMC	deg	0.0	β
IXPTMC = 3						
Commanded Yaw	PSICD		T1PMC	deg	0.0	Ψ_C
Commanded Pitch	THTCD		T2PMC	deg	0.0	θ_C
Error Criteria	EPSPMC				0.0	ϵ_{PMC}
IXPTMC = 4						
Elevation Load Factor	FLDGMX		T1PMC	g	0.0	N_γ
Horizontal Load Factor	FLDSGX		T2PMC	g	0.0	N_σ
Error Criteria	EPSPMC				0.0	ϵ_{PMC}
Angle of Attack Increment	DLAPMC			deg	0.0	$\Delta\alpha_{PMC}$
Angle of Sideslip Increment	DLBPMC			deg	0.0	$\Delta\beta_{PMC}$

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Angle-of-Attack Limits						
Minimum	ALPMIN	ABMNMX(1)		deg	0.0	α_{MIN}
Maximum	ALPMAX	(2)		deg	0.0	α_{MAX}
Angle-of-Sideslip Limits						
Minimum	BETMIN	ABMNMX(3)		deg	0.0	β_{MIN}
Maximum	BETMAX	(4)		deg	0.0	β_{MAX}
IXPTMC = 5						
Commanded Elevation Flight-Path Angle	GAMASX (GAMANX)		T1PMC*	deg	0.0	γ_{CS}
		*f (Altitude) Only				
Angle-of-Attack Increment	DLAPMC			deg	0.0	$\Delta\alpha_{PMC}$
Altitude Increment	DLHPMC			m	0.0	Δh_{PMC}
Gain Factor For Change of Gamma	GANPMC			rad-s/m	0.0	Γ_{γ}

IXPTMC = 6

User Supplied Controller - (routine USRPMC provided)

4.4.4 Propulsion System

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Execution Choice: 0 - NO 1 - YES	IXTHRS				0	
Number of Motors	NMTR (MAX = 9)				1	
Enter The Following For Each Motor (see Figure B2) (I = 1,NMTR)						
Nozzle Exit Point: Displacement						
X-Direction	XNOZEM(I)			m	0.0	X_{N_i}
Y-Direction	YNOZEM(I)			m	0.0	Y_{N_i}
Z-Direction	ZNOZEM(I)			m	0.0	Z_{N_i}
Nozzle Exit Point: Velocity						
X-Direction	XETHM1(I)			m/s	0.0	$\dot{X}_{e_i}$
Y-Direction	YETHM1(I)			m/s	0.0	$\dot{Y}_{e_i}$
Z-Direction	ZETHM1(I)			m/s	0.0	$\dot{Z}_{e_i}$
Nozzle Exit Area	AENOZM(I)			m^2	0.0	$A_{e_{n_i}}$
Thrust Swivel Angle	XIMTRD(I)			deg	0.0	ξ_{T_i}
Rotation Angle - Plane of Swivel	PHMTRD(I)			deg	0.0	ϕ_{T_i}
Vacuum Thrust	TVACI(I)		TVACTn (n=1,9)	N	0.0	T_{VAC_i}
Mass Flow	FMASK1(I)		TMASn1 (n=1,9)	k/s	0.0	$\dot{m}_i$
Mass Flow Rate	FMASK2(I)		TMASn2 (n=1,9)	k/s^2	0.0	$\ddot{m}_i$
If IXTHRS = 0, constant forces and moments can be entered						
Constant Force:						
X-Direction	XFTHRN			N	0.0	F_{xT}
Y-Direction	YFTHRN			N	0.0	F_{yT}
Z-Direction	ZFTHRN			N	0.0	F_{zT}
Constant Moment						
About X-X	XMTHRN			N·m	0.0	M_{xT}
About Y-Y	YMTHRN			N·m	0.0	M_{yT}
About Z-Z	ZMTHRN			N·m	0.0	M_{zT}

4.4.5 Vehicle Mass Properties

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Execution Choice: 0-No 1-Yes	IXVMP				0	
Center-of-Mass (see Figure B3) (w.r.t. DATUM)						
X-Direction	XCMRFM	XYZCM(1)	TXCM	m	0.0	X_{cm}
Y-Direction	YCMRFM	(2)	TYCM	m	0.0	Y_{cm}
Z-Direction	ZCMRFM	(3)	TZCM	m	0.0	Z_{cm}
Moment of Intertia	FIXXK	XYZMIS(1)	TIXX	$K \cdot m^2$	0.0	I_{xx}
	FIYYK	(2)	TIYY	$K \cdot m^2$	0.0	I_{yy}
	FIZZK	(3)	TIZZ	$K \cdot m^2$	0.0	I_{zz}
	FIXYK	(4)	TIXY	$K \cdot m^2$	0.0	I_{xy}
	FIXZK	(5)	TIXZ	$K \cdot m^2$	0.0	I_{xz}
	FIYZK	(6)	TIYZ	$K \cdot m^2$	0.0	I_{yz}
Moment of Inertia Rate	FIXXK1	XYZMIS(7)	TIXX1	$(K \cdot m^2)/s$	0.0	$\dot{I}_{xx}$
	FIYYK1	(8)	TIYY1	$(K \cdot m^2)/s$	0.0	$\dot{I}_{yy}$
	FIZZK1	(9)	TIZZ1	$(K \cdot m^2)/s$	0.0	$\dot{I}_{zz}$
	FIXYK1	(10)	TIXY1	$(K \cdot m^2)/s$	0.0	$\dot{I}_{xy}$
	FIXZK1	(11)	TIXZ1	$(K \cdot m^2)/s$	0.0	$\dot{I}_{xz}$
	FIYZK1	(12)	TIYZ1	$(K \cdot m^2)/s$	0.0	$\dot{I}_{yz}$

4.4.6 Gravity Model

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Reference Gravitational Acceleration	GREF			m/s^2	9.80665	g_{ref}
Geocentric Gravitational Constant	FMU			m^3/s^2	3.98615683E14	μ
Coefficients						
2nd Harmonic	H1				1.62341E–3	H_1
3rd Harmonic	H2				6.04E–6	H_2
4th Harmonic	H3				6.37E–6	H_3

4.5 General Output Requests

4.5.1 Tabulated Data

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Tabulated Data - Standard Set						
Execution	IXTBLS				0	
Choice: 0 - NO						
1 - YES						
Set Request	NTABLE(I), I=1,7				0	
Choice: 0 - No Table						
= 1 - Trajectory Summary						
= 2 - Trajectory - Tangent Plane						
= 3 - Stability						
= 4 - Air Properties and Loads						
= 5 - Mass Properties and Load Factors						
= 6 - Station/Radar Look Values						
= 7 - Accelerations and Accelerometer Values						
Tabular Data - User Defined Set						
Execution	IXTBLU				0	
Choice: 0 - NO						
1 - YES						
Table Names	ITBLUN(15)				$BLANK$	
Table Headings	ITBLUH(15,3)				$BLANK$	
Table Format	ITBLUF(15)				$BLANK$	

NOTE: TAPE20 file must be requested.

4.5.2 Plotted Data

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Plotted Data Requested						
Execution Choice: 0 - NO 1 - YES	IXPLOT				0	
Y vs X Plots	IXYPLT(7,5)				$BLANK$	

Example:
IXYPLT(1,1) = $TIME HGD8M DYNP8KXGD8M1$

Stack Request: IXYPLT(1,2) = $TIME PI88R OMGCYRSTACK $
NOTE: STACK is used as the last request

Case Identifier	IDCASE(5)				$BLANK$	

Example:
IDCASE(1) = $497-04APPLE ARM-1X498-05PEACH $

Stack Request: $497-04APPLE ARM-1X498-05STACK $
NOTE: STACK is used as the last request.
NOTE: TAPE20 file must be requested.

4.5.3 TAPE20 File

Execution Choice: 0 - NO 1 - YES - must be used if tables or plots are requested	IXTP20				0	

5. Code Control

5.1 Case or Problem Control

As discussed in Section 2.3, partitioning of the base case (problem) into sections or stages is permitted. The stages are usually patterned after the problem; i.e., one or more of the support routines or main programs may be activated or deactivated.

Two types of stages are recognized: a minor action type in which new data are read in that causes no discontinuity to the integration procedure, and the major action type where the main program or support routines are affected, or when new data cause a discontinuity in the problem solution.

For case control (staging), therefore, the user must request a program-computation stop based upon some computed variable value, supply new data, and request problem continuation.

A major action request is initiated by flagging the support routine execution flag with a minus (–) sign or by specifying a new main program designator.

5.1.1 Staging - Input Data

Item	Code Mnemonic	Equivalent Array	Table Name	Units	Nominal Value	Sym
Requesting A Program Computation Halt						
Staging Variable						
Name	NAMSTG(1–4)				$BLANK$	
Value	VALSTG(1–4)				0.0	
Default Stop						
Name	NAMSTG(5)				$HGD8M$	h_{gd}
Value	VALSTG(5)				0.0	
Major Action Indicator						
Support Routine Execution Flag	IX----				–N	
where: N = 0 - Return to intial values with execution deleted						
N = 1 - Reinitialize with new values and execution requested						
Main Program Designator	IMPROG				$SIXDOF$	

Example:

Minor Action

```
Times   = 0.5
Delts   = 0.2
  .
  .
  .
NAMSTG(1)   = $TIMES$
VALSTG(1)   = 2.3
*RUN
Delts   = 1.0 //Resets print internal
  .
  .
  .
*RUN
```

Major Action

```
Times   = 0.5
Delts   = 0.2
  .
  .
  .
NAMSTG(1)   = $TIMES$
VALSTG(1)   = 2.3
*RUN
IXTHRS   = -1 //Reinitializes and turns on THRUST routine
Delts   = 1.0
TIMSDX   = 2.3 //Returns stage time to 0.0
  .
  .
  .
*RUN
```

NOTE: *RUN card described in Section 5.2

5.2 Execution Control: *CONTROL Cards

Control of code flow through the input data deck is achieved through the use of a set of key words which, when preceded by an asterisk (*) and used on an input card, causes the card to become a control card.
The code allows the following set:

*COMMENT
*COPY
*STOP
*MERGE
*SAVE
*RESTORE
*RUN
*OUTPUT
*CLEAR
*CLEAR KEEPTABLES
*END

The cards may be used singly or in pairs to instruct the code to perform a specific task. A description of the tasks performed by the control cards and the method by which the code accomplishes them is described in the report by Jones[4] and will not be reiterated here. However, brief descriptions of the cards which are used singly are given below. The paired cards are described in the following sections with examples to clarify their use.

*COMMENT	- Allows comments to be inserted into the input deck.
*RUN	- Signals the reader routine to stop processing input data cards and return control to the calling routine.
*OUTPUT	- Instructs the code to proceed immediately to the output phase and display the computed data.
*CLEAR	- Instructs the code to clear all labeled COMMON blocks and table storage location, and resets the code default values.
*CLEAR KEEPTABLES	- Same as *CLEAR except table storage locations remain unchanged.

5.2.1 Base Case

A "Base" case is a set of input data cards plus control cards that describe the conditions under which the current problem is to be solved. It may consist of a single stage or be made up of multiple stages.

The control card set to describe a typical three-stage base-case execution could be set up as follows:

```
*COMMENT        SIMPLE THREE STAGE PROBLEM
...             //Problem Data
*RUN
...             //Problem Data
*RUN
...             //Problem Data
*RUN
*END
```

5.2.2 Merging Cases

It is sometimes desirable to run a series of problems in which successive problems are only a minor variation of the base problem. A procedure to accomplish this is called "Merging"; i.e., the data of the succeeding cases are merged with the base case and executed. The *COPY, *STOP, and *MERGE control cards are used to control the code for this procedure. The following example shows a possible combination of data and control cards that would excecute a two-stage base case and then execute a modification of the base case.

```
*COMMENT    TWO MERGE CASES, ONE BEING THE BASE CASE ITSELF
*COPY
...
            // Problem Data 1st Stage
...
                                        //Base Case
*RUN
...         // Problem Data 2nd Stage
...
*RUN
*STOP
*MERGE
                                        //Run the base case
*RUN
*RUN
*MERGE
...         //1st Stage data modification
...
                                        //Run the modified case
*RUN
...         //2nd Stage data modification
...
*RUN
*END
```

5.2.3 Stacking Cases

When it is necessary to run a series of unrelated cases, a procedure called "Stacking" is available. The *CLEAR card is used, either in the full clearance mode or in the KEEPTABLE mode, to prepare the code for the input of data related to the next problem.

A two-problem example is given below, but could be extended to many problems.

```
*COMMENT      PROBLEM 1
...
*RUN
...
*RUN
*OUTPUT
*CLEAR
*COMMENT      PROBLEM 2
...
*RUN
...
*RUN
*OUTPUT
*END
```

5.2.4 *SAVE, *RESTORE Utility Feature

The *SAVE control card permits the current state of the problem solution (contained in the labeled COMMON blocks) and the input tabular data (located in blank COMMON, LCM, or mass storage) to be saved (file: ISAVE=8) for later use on a code restart. The *RESTORE is the companion control card which reverses the procedure to restart execution. Other usages of the *SAVE/*RESTORE feature are discussed in the report by Jones.[4]

The following example of the *SAVE/*RESTORE cards shows their use for continuing execution at a later date.

```
*COMMENT      MONDAY JOB
...
*COMMENT      FIRST STAGE
*RUN
...
*COMMENT      SECOND STAGE
*RUN
*OUTPUT
*SAVE
*END
```

If file (TAPE8) is saved on a permanent file or magnetic tape file, the next day's control card stream could be as follows:

```
*COMMENT      TUESDAYS JOB
*RESTORE
...
*COMMENT      THIRD STAGE
*RUN
...
*RUN
*END
```

5.2.5 *END Card.

To end the execution of a problem case or a series of problem cases, the *END card is used. When used within the data deck, it has the same effect as an end-of-file mark encountered on the input file at that point. This card may appear in the base case, at the end of a base case, anywhere within the primary input, and especially at the end of input. Cards appearing after this card are ignored, so temporarily unneeded cards may be stored here. As with the *CLEAR command, any output data not yet printed or plotted will be processed according to the output phase when the *END card is encountered.

References

1. R. C. Brown, R. V. Brulle, A. E. Combs, and G. D. Griffin, *Six-Degree-of-Freedom Flight-Path Study Generalized Computer Program,* Part 1, Vol 1, Problem Formulation FDL-TDR-64-1, October 1964.

2. I. Fischer, "An Astrogeodetic World Datum From Geoidal Heights Based on Flattening f-l/298.3," *J Geophysical Res,* 65; 7, 2067, July 1960.

3. A. E. Hodapp, Jr., *Equations of Motion for Free-Flight System of Rotating-Translating Bodies Having Variable Mass and Configuration,* SAND77-0606, Albuquerque: Sandia Laboratories, 1978.)

4. L. F. Shampine, and H. A. Watts, *Practical Solution of Ordinary Differential Equations By Runge-Kutta Methods,* SAND76-0585, (Albuquerque: Sandia Laboratories, 1976.)

5. R. E. Jones, *An Input and Table Handling Facility for Fortran Programs,* SAND78-ll65, (Albuquerque: Sandia Laboratories, 1979.)

6. *U.S. Standard Atmosphere, 1962,* Superintendent of Documents, U.S. Government Printing Office, Washington, DC, 1962.

7. *U.S. Standard Atmosphere Supplements, 1966,* Superintendent of Documents, U.S. Government Printing Office, Washington, DC.

APPENDIX A

Logic Flow Diagram and Examples

This appendix contains a logic flow diagram (to aid the use in assembling the input data) and several examples of executable decks.

Contents

Input Data Selection Logic Chart 57
General Code Items 57
General Problem Items 57
Example 1 - A Point Mass Reentry With Radar Look Calculations 59
Example 2 - Staging With Main Program and Support Routine Interchange 60
Example 3 - Six-Degree-of-Freedom-Point-Mass Interchange With Table Value Options 64

INPUT DATA SELECTION LOGIC CHART

General Code Items

NO	Is the reference GEOID acceptable?	YES
	ENTER: Geoid shape, spin rate, radii, flattening factor	
NO	Is the reference GRAVITY acceptable?	YES
	ENTER: Reference gravity, gravity constants, geocentric constant	
NO	Are the integration controls acceptable?	YES
	ENTER: Absolute error, relative error	
NO	Is the input data METRIC?	YES
	SET: Conversion flag	

General Problem Items

NO	Is report titling wanted?	YES
	ENTER: 30 words of title	
NO	Is case identifier required?	YES
	ENTER: 1 word case identifier	
GEODETIC	Select planet surface location	GEOCENTRIC
	ENTER APPROPRIATE VALUES: a, Initial Position: latitude, longitude, altitude b, Current Position: latitude, longitude, altitude c, Velocity Vector: total velocity, elevation angle, horizontal angle	
NO	Is initial reference heading NORTH?	YES
	ENTER: Initial azimuth heading angle	
NO	Are time items acceptable?	YES
	ENTER: Initial time, print internal, maximum time, stage time increment, number of good time steps/print	
NO	Are mass items acceptable?	YES
	ENTER: Initial mass, initial mass adder	
NO	Is gravity model acceptable?	YES
	ENTER: GRAVITY items (Section 4.4.6)	
NO	Are wind tables required?	YES
	SET: Execution request flag ENTER: Three wind tables	

General Problem Items (cont)

NO	Is 1962 standard atmosphere acceptable?		YES
	ENTER: Atmosphere selection (Section 4.4.2)		
	Select Main Program		
	SIXDOF: See Section 4.3.1	PTMASS: See Section 4.3.2	
NO	Is there only one body?	YES	
	ENTER: Number of bodies		
NO	Are load factors required?		YES
	SET: Execution request flag		
NO	Are range calculations required?		YES
	SET: Execution request flag		
NO	Are RADAR look values required?		YES
	SET: Execution request flag ENTER: Station: latitude, longitude, altitude, heading		
NO	Are aerodynamic items required?		YES
	Select aerodynamic routine		
	SDAERO See Section 4.4.1 ENTER: Force and moment coefficient tables	PMAERO: See Section 4.4.1 ENTER: Force coefficient tables	
	SET: Execution request flag		
NO	Is a propulsion system requested?		YES
	SET: Execution request flag ENTER: Propulsion System Data (see Section 4.4.4)		
NO	Are vehicle mass properties requested?		YES
	SET: Execution request flag ENTER: Vehicle Mass Properties Data (see Section 4.4.5)		
NO	Is vehicle control required?		YES
	Which main program selected?		
	SIXDOF: (see Section 4.4.3.1)	PTMASS: (see Section 4.4.3.2)	
	SET: Execution request flag	SET: Execution request flag	
NO	Is staging required?		YES
	ENTER: Stage Request Data (see Section 5.1.1)		
	SET: Execution request (*RUN)		
NO	Is problem finished?		YES
	RETURN TO START	SET: *END	

Example 1. A Point Mass Reentry With Radar Look Calculations

```
*COMMENT    $$$$    SAMPLE CASE    $$$$
ITITLE(1)    $TEST TRIAL OF POINT MASS  COMPUTATIONS          $
ITITLE(13) =  $  REENTRY  FROM 50000 FT $
ITITLE(29) =  $MARCH 1980$
ICASE      $PM-TST$
PHGDZD = 0.0
THTLZD = 0.0
HGDZM  = 15240.0
PHIGDD = 0.0
THTLD  = 0.0
VTGDM  = 6705.6
GAMGDD = -30.0
SIGGDD = 45.0
SIGZD  = 45.0
TIMEZS = 0.0
DELTS = 0.10
TMAXS  = 10.0
FMASZK = 37.42137548
IXGRAV = 1
IATMNM    $15NANL$
IMPROG     $PTMASS$
ITITLE(21) = $ RADAR STATION EXECUTION REQUIRED    $
IXSTA  = 1
HGDSTM = 0.0
PHGDSD = 0.0
THTSTD = 0.0
PSISTD = 0.0
ISPROG     $PMAERO$
IXAERO = 1
SREFAM = 0.0506693
TABLE  TCXO(4)  /HGD8M/ EXTRAP
BREAKS  TCXO  /HGD8M/    0.0,0.3048E4,3.048E4,30.48E4
BODY TCXO  0.06,0.06,0.06,0.06
NAMSTG(1) = $TIME  $
VALSTG(1) = 0.985
*RUN
NAMSTG(1) = $TIME  $
VALSTG(1) = 1.0
*RUN
NAMSTG(1) = $HGD8M $
VALSTG(1) = 0.0
*RUN
```

Example 2. Staging With Main Program and Support Routine Interchange

```
*COMMENT     $$$$  SAMPLE CASE    $$$$
ITITLE(2)  $TEST TRAJECTORY FOR AMEER CHECK OUT $
ITITLE(11) $MILLARDS SINGLE STAGE MALEMUTE AT TTR $
ITITLE(21) $ QE=87.0 DEG   AZ=140.0 DEG   $
ICASE      $497-04$
PHGDZD     = 37.84608
THTLZD     = -116.70634
HGDZM      = 1644.7008    // 5396.0 FT AT TTR  //
PHIGDD     = 37.84608
THTLD      = -116.70634
VTGDM      = 22.2504       // 73.0 FPS AT EOL 17.2FT  17641. LBS //
GAMGDD     = 87.0
SIGGDD     = 140.0
SIGZD      = 90.0
TIMEZS     = 0.52
DELTS      = 1.0
TMAXS      = 500.0
FMASZK     = 791.135319     // 54.21 SLUGS - 1744.14 LBS //
IXGRAV     =1
IATMNM     $TTRSUM$
IATMNM     $62STD $
IMPROG     $SIXDOF$
IDATUM     $NOSE  $
XYZDTM(1)  = 0.0,0.0,0.0
XYZBRP(1)  = -3.429,0.0,0.0
PSBGDD     = 140.0
THBGDD     = 87.0
PHBGDD     = 0.0
IDPQR(1)  = $PLANET RATES$
IXLODF   = 1
IXRNGE   = 1
ISPROG     $SDAERO$
IXAERO     = 1
SREFAM     = 0.1296926438  // 1.396 FT SQ //
DREFXM     = 6.8580         // 22.5 FT  //
DREFYM     = 6.8580         // 22.5 FT  //
TABLE  TCXO(15,2) /FMACH,REYNOM/     // TWO DIMENSIONAL TABLE //
BREAKS TCXO   /FMACH /  0.0,0.70,0.80,0.90,0.95,1.00,1.05
                       1.10,1.20,1.56,1.96,2.44,4.94,7.32,14.42
BREAKS TCXO   /REYNOM/  0.0,1.0E9
BODY TCXO     -0.505,-0.505,-0.530,-0.625,-0.780,-0.928,-0.924        //RN = 1 //
              -0.900,-0.841,-0.678,-0.557,-0.460,-0.270,-0.198,-0.120
              -0.505,-0.505,-0.530,-0.625,-0.780,-0.928,-0.924
              -0.900,-0.841,-0.678,-0.557,-0.460,-0.270,-0.198,-0.120
TABLE  TCZA(15)  /FMACH /
BREAKS TCZA /FMACH /  0.00,0.70,0.80,0.90,0.95,1.00,1.05
                      1.10,1.20,1.56,1.96,2.44,4.95,7.31,14.4
BODY   TCZA    -0.295,-0.295,-0.315,-0.318,-0.322,-0.324,-0.325
               -0.325,-0.321,-0.270,-0.234,-0.214,-0.168,-0.147,-0.095
```

Example 2 (cont)

```
TABLE   TCYB(15)   /FMACH /
BREAKS  TCYB /FMACH /   0.00,0.70,0.80,0.90,0.95,1.00,1.05
                        1.10,1.20,1.56,1.96,2.44,4.95,7.31,14.4
BODY    TCYB   -0.295,-0.295,-0.315,-0.318,-0.322,-0.324,-0.325
               -0.325,-0.321,-0.270,-0.234,-0.214,-0.158,-0.147,-0.095
TABLE   TCMA(15)   /FMACH /
BREAKS  TCMA   /FMACH / 0.00,0.70,0.80,0.90,0.95,1.00,1.05
                        1.10,1.20,1.56,1.96,2.44,4.95,7.31,14.42
BODY    TCMA   -.1005,-.1005,-.1105,-.1130,-.1140,-.1150,-.1155
               -.1145,-.1125,-.0880,-.0670,-.0550,-.0310,-.0245,-.0095
TABLE   TCMQ(12)    /FMACH /
BREAKS  TCMQ   /FMACH /   0.00,0.60,0.80,0.90,1.00,2.00
                        3.00,4.00,6.00,8.00,10.0,12.0
BODY    TCMQ   -13.13,-14.80,-16.80,-18.64,-22.07,-21.07
               -13.92,-11.03,-08.45,-07.29,-06.66,-06.31
TABLE   TCNB(15)   /FMACH /
BREAKS  TCNB   /FMACH / 0.00,0.70,0.80,0.90,0.95,1.00,1.05
                        1.10,1.20,1.56,1.96,2.44,4.95,7.31,14.42
BODY    TCNB    0.1005,0.1005,0.1105,0.1130,0.1140,0.1150,0.1155
               0.1155,0.1125,0.0880,0.0670,0.0550,0.0310,0.0245,0.0095
TABLE   TCNR(12)   /FMACH /
BREAKS  TCNR   /FMACH / 0.00,0.60,0.80,0.90,1.00,2.00
                        3.00,4.00,6.00,8.00,10.0,12.0
BODY    TCNR   -13.13,-14.80,-16.80,-18.64,-22.07,-21.07
               -13.92,-11.03,-08.45,-07.29,-06.66,-06.31
IXTHRS     = 1
NMTR       = 1
XNOZEM(1)  = -6.4008        // 21.0 FT   //
AENOZM(1)  = 0.084380803    // 0.908 FT SQ  12.9 IN DIA  //
XIMTRD(1)  = 0.0
PHMTRD(1)  = 0.0
TABLE   TVACT1(28)   /TSTAGE/
BREAKS  TVACT1 /TSTAGE/   0.00,0.16,1.00,2.00,3.00,4.00,5.00
                          6.00,7.00,8.00,9.00,10.0,11.0,12.0
                          13.0,14.0,15.0,16.0,17.0,18.0,19.0
                          20.0,21.0,22.0,22.16,22.34,22.59,100.0
BODY    TVACT1   00000.0,47262.4,49909.1,52555.7,54624.2,56581.4,58249.5
                 59628.4,60895.2,62275.1,63520.6,64632.7,65611.3,66434.2
                 66990.2,67568.5,67902.1,68258.0,68391.4,68814.0,69214.3
                 69636.9,70192.9,70904.7,17792.9,04448.2,00000.0,00000.0
TABLE   TMAS11(28)   /TSTAGE/
BREAKS  TMAS11 /TSTAGE/   0.00,0.16,1.00,2.00,3.00,4.00,5.00
                          6.00,7.00,8.00,9.00,10.0,11.0,12.0
                          13.0,14.0,15.0,16.0,17.0,18.0,19.0
                          20.0,21.0,22.0,22.16,22.34,22.59,100.0
BODY    TMAS11   -00.0000,-17.2792,-18.2424,-19.2056,-19.9645,-20.6796,-21.2925
                -21.7887,-22.2557,-22.7565,-23.2189,-23.6275,-23.9778,-24.2842
                -24.4886,-24.6929,-24.9410,-24.9994,-25.1453,-25.2912,-25.2912
                -25.4518,-25.6561,-25.7728,-06.5089,-01.6199,00.0000,00.0000
```

Example 2 (cont)

```
IXVMP      = 1
XCMRFM     = -3.429
TABLE   TXCM(13)   /FMASSK/
BREAKS  TXCM    /FMASSK/   000.00,287.21,316.69,375.36,435.63,494.59
                           552.09,607.54,660.52,712.47,757.13,800.18,1444.8
BODY    TXCM          -3.3833,-3.3833,-3.5022,-3.6881,-3.8313,-3.9441
                      -4.0386,-4.1209,-4.1910,-4.2550,-4.3068,-4.3586,-4.3586
TABLE   TIXX(13)   /FMASSK/
BREAKS  TIXX    /FMASSK/   000.00,287.21,316.69,375.36,435.63,494.59
                           552.09,607.54,660.52,712.47,757.13,800.18,1444.8
BODY    TIXX           7.0503,7.0503,8.0536,10.1003,11.9175,13.4497,14.7106
                  15.7410,16.5816,17.2731,17.7883,18.1815,18.1815
TABLE   TIYY(13)   /FMASSK/
BREAKS  TIYY    /FMASSK/   000.00,287.21,316.69,375.36,435.63,494.59
                           552.09,607.54,660.52,712.47,757.13,800.18,1444.8
BODY    TIYY       1340.30,1340.30,1400.00,1515.70,1640.70,1777.33
                   1924.36,2082.94,2247.73,2423.31,2587.99,2764.78
                   2764.78
TABLE   TIZZ(13)   /FMASSK/
BREAKS  TIZZ    /FMASSK/   000.0,287.21,316.69,375.36,435.63,494.59
                           552.09,607.54,660.52,712.47,757.13,800.18,1444.8
BODY    TIZZ       1340.30,1340.30,1400.00,1515.70,1640.70,1777.33
                   1924.36,2082.94,2247.73,2423.31,2587.99,2764.78
                   2764.78
NAMSTG(1) = $TIME  $
VALSTG(1) = 22.59
*RUN
IXTHRS     = -0
DELETE    TVACT1,TMAS11
DELTS      = 2.0
TABLE    TCX0(15,2)   /FMACH,REYNOM/
BREAKS  TCX0   /FMACH/   0.0,0.7,0.80,0.90,0.95,1.00,1.05
                         1.10,1.20,1.56,1.96,2.44,4.94,7.32,14.42
BREAKS  TCX0   /REYNOM/   0.0,1.0E9
BODY    TCX0   -0.745,-0.745,-0.810,-0.940,-1.100,-1.290,-1.230,-1.190    //RN = 1
               -1.125,-0.835,-0.692,-0.575,-0.302,-0.234,-0.120
               -0.745,-0.745,-0.810,-0.940,-1.100,-1.290,-1.230      //RN = 2 //
               -1.190,-1.125,-0.835,-0.692,-0.575,-0.302,-0.234,-0.120
NAMSTG(1) = $HGD8M $
VALSTG(1) = 91440.0
*RUN
```

Example 2 (cont)

```
IDLAT(1) = $GEODET$
IDLAT(2) = $IC    $
DELTS      = 5.0
IMPROG     = $PTMASS$
ISPROG     = $PMAERO$
IXAERO     = -1
DELETE    TCX0,TCZA,TCYB,TCMA,TCMQ,TCNR,TCNB
TABLE    TCX0(15,2)  /FMACH,REYNOM/
BREAKS TCX0  /FMACH/  0.0,0.7,0.80,0.90,0.95,1.00,1.05
                      1.10,1.20,1.56,1.96,2.44,4.94,7.32,14.42
BREAKS TCX0  /REYNOM/  0.0,1.0E9
BODY   TCX0  -0.745,-0.745,-0.810,-0.940,-1.100,-1.290,-1.230,-1.190   //RN = 1
             -1.125,-0.835,-0.692,-0.575,-0.332,-0.234,-0.120
             -0.745,-0.745,-0.810,-0.940,-1.100,-1.290,-1.230     //RN = 2 //
             -1.190,-1.125,-0.835,-0.692,-0.575,-0.332,-0.234,-0.120
IXVMP      -0
DELETE     TXCM,TIXX,TIYY,TIZZ
NAMSTG(1) = $HGD8M $
VALSTG(1) = 1638.3
*RUN
*END
```

Example 3. Six-Degree-of-Freedom-Point-Mass Interchange With Table Value Options.

```
*COMMENT     $$$$  SAMPLE CASE    $$$$
*COMMENT  SECORS ACTIVE P/L NO. 1
ITITLE(1)  $SECORS POP ACTIVE ONE//
ITITLE(11) $ROLLSTIN 3633  11-79  W/1.030 T MULT AND QE 85.0$
ITITLE(21) $QE=85.0 DEG  AZ=170.0 DEG  P/L WT=51.0 LBM//
ICASE      $SEC023$
IMPROG     $SIXDOF$
ISPROG     $SDAERO$
IDLAT(1)   = $GEODETIC     $
IATMNM     $TTRSUM$
IDATUM     $ZUNNEP$
XYZDTM(1)  = 0. 0. 0.          //XYZ-DATUM POINT ARRAY//
XYZBRP(1)  = 1.938 0. 0.       //XYZ-DIST TO BRP ARRAY//
IXGRAV     =1                  //GRAVITY HARMONIC EXEQ FLAG//
IXRNGE   = 1                   //RANGE CALCULATION FLAG//
IXLODF   = 1                   //LOAD FACTOR CALCS - FLAG//
IXCRFQ     = 1
PHIGDD     = 37.84632         //GEODETIC LATITUDE  TTR-L1//
PHGDZD     = 37.84632         //INITIAL GEODETIC LATITUDE  TTR-L1//
THTLD      = -116.70567       //LONGITUDE - LOCAL  TTR-L1//
THTLZD     = -116.70567       //INITIAL LONGITUDE - ORIGIN  TTR-L1//
SIGGDD     = 170.0            //GEODETIC HORIZ F.P. ANGLE//
SIGZD      = 170.0            //INITIAL HORIZ F.P. ANGLE//
GAMGDD     = 85.0             //GEODETIC ELEV F.P. ANGLE INRT 2//
PSBGDD     = 170.0            //EULER ANGLE-YAW  GD REF//
THBGDD     = 85.0             //EULER ANGLE-PITCH  GD REF INRT 2//
PHBGDD     = 0.               //EULER ANGLE-ROLL  GD REF//
TIMEZS     = .11              //TIME ZERO - START PROG  TUBE EXIT//
TMAXG      = 25.              //MAXIMUM PROGRAM RUN TIME//
TMAXS      = 5.6
TMAXS      = 10000.           //MAXIMUM PROGRAM RUN TIME//
DELTS      = .2               //TIME INCREMENT FOR PRINT-SEC//
DELTS      = 0.1
VTGDM      = 40.23 //M/S 132 FPS AT EO 6-FT LAUNCH TUBE //
HGDZM      = 1646.     // M LB FWD END + 6 FT TUBE//
FMASZK     = 58.0264   // KG 127.9 LB LNCH ACT 1 51 P/L//
IXAERO     = 1         // AERO EXECUTION FLAG //
SREFAM     = .012668   // M2 .1364 FT2 BODY X-SECT AREA//
DREFXM     = .127000   // M .4167 FT BODY DIAM//
DREFYM     = .127000   // M .4167 FT BODY DIAM//
RELERR     = 1.0E-3
ABSERR     = 1.0E-5
ABSERR     = 1.0E-4
PIBPR      = 26.389
IXWIND     = 1
TABLE  TXWIND(27) /HGDBM/
BREAKS  TXWIND  /HGDBM/        0.   1595.   1650.   1665.   1698.   1703.
                            1741.   1746.   1976.   1981.   2433.   2438.
                            3043.   3048.   4567.   4572.   6091.   6096.
                            7615.   7620.   9139.   9144.  10663.  10668.
                           12187.  12192.  20000.
```

Example 3 (cont)

```
BODY   TXWIND      0.0  0.5  0.5  0.6  0.5 -1.0 -1.0  0.5  0.5  2.7  2.7
                   2.2  2.2  3.5  3.5  3.3  3.3  2.3  2.3  4.7  4.7  5.2
                   5.2  0.4  0.4 -2.6 -2.6
TABLE  TYWIND(27) /HGD8M/
BREAKS  TYWIND  /HGD8M/        0.   1595.   1660.   1665.   1698.   1703.
                            1741.   1746.   1976.   1981.   2433.   2438.
                            3043.   3048.   4567.   4572.   6091.   6096.
                            7615.   7620.   9139.   9144.  10663.  10668.
                           12187.  12192.  20000.
BODY   TYWIND      0.0 -0.2 -0.2  2.2  2.2 -0.1 -0.1 -1.1 -1.1 -1.0 -1.0
                  -0.9 -0.9  0.1  0.1  5.6  5.6  7.1  7.1  9.8  9.8 12.3
                  12.3 17.5 17.5 25.1 25.1
TABLE  TCXO(23)  /FMACH/  +.04
TABLE   TCXO  EXTRAPOLATION
BREAKS TCXO  /FMACH /  0.00 0.25 0.50 0.76 0.91 1.00 1.10 1.16 1.20//5-18-79  //
                       1.25 1.29 1.40 1.50 1.60 1.70 1.80 2.00      //FROM NOS //
                       2.20 2.40 2.60 2.80 3.01 3.25                // IH MD   //
BODY   TCXO    -.470 -.470 -.474 -.498 -.604 -.690 -.863 -.905 -.916 -.903
               -.882 -.837 -.792 -.755 -.718 -.693 -.639 -.600 -.562 -.530
               -.502 -.480 -.447
CXALP2     =  -0.004
CXBET2     =  -0.004
TABLE  TCZA(5)  /FMACH/
BREAKS TCZA  /FMACH/  0.0 1.0 2.5 5.0 10.
BODY   TCZA  -.104 -.177 -.083 -.057 -.045   //SS ZUNI WT W/RVRTV TREND//
CZALP2     =  -0.0103
TABLE  TCZQ(5)  /FMACH/
BREAKS TCZQ  /FMACH/  0.0 1.0 2.5 5.0 10.
BODY   TCZQ  -144.4 -266.4 -139.2 -78.5 -75.5
TABLE  TCYB(5)  /FMACH/
BREAKS TCYB  /FMACH/  0.0 1.0 2.5 5.0 10.
BODY   TCYB  -.104 -.177 -.083 -.057 -.045   //SS ZUNI WT W/RVRTV TREND//
CYBET2     =  -0.0103
TABLE  TCYR(5)  /FMACH/
BREAKS TCYR  /FMACH/  0.0 1.0 2.5 5.0 10.
BODY   TCYR  144.4 266.4 139.2 78.5 75.5
TABLE  TCLO(11)  /TSTAGE/       //FLUTE TORQUE FOR 12.5 HZ AND SS ZUNI WT//
BREAKS TCLO  /TSTAGE/  0. .044 .12 .386 .744 1.56 1.644 1.7 1.776 1.9 10.
BODY   TCLO  .0067 19.90 5.15 .385 .111 .033 .028 .0181 .0137 .0067 .0067
TABLE  TCLA(19)  /ALPHAD/ EXTRAP      //SS ZUNI WT DATA//
BREAKS TCLA  /ALPHAD/ -180. -10. -7. -6. -5. -4. -3. -2. -1.
                       0. 1. 2. 3. 4. 5. 6. 7. 10. 180.
BODY   TCLA  0.00 -.005 -.0012 0. .001 .00145 .00215 .0025 .003
             .0 -.003 -.0025 -.00215 -.00145 -.001 0. .0012 .005 0.0
TABLE  TCLB(19)  /BETAD/  EXTRAP      //SS ZUNI WT DATA//
BREAKS TCLB  /BETAD/  -180. -10. -7. -6. -5. -4. -3. -2. -1.
                       0. 1. 2. 3. 4. 5. 6. 7. 10. 180.
BODY   TCLB  0.00 -.005 -.0012 0. .001 .00145 .00215 .0025 .003
             .0 -.003 -.0025 -.00215 -.00145 -.001 0. .0012 .005 0.0
```

Example 3 (cont)

```
TABLE   TCLDP(4)  /FMACH/
BREAKS  TCLDP  /FMACH/  0. 2. 2.5 10.
BODY    TCLDP .04346 .04346 .03070 .03070
TABLE   TCLP(4)  /FMACH/
BREAKS  TCLP  /FMACH/  0. 2. 2.5 10.
BODY    TCLP  -6.1631 -6.1631 -4.3529 -4.3529
TABLE   TCMA(5)  /FMACH/
BREAKS  TCMA  /FMACH/  0. 1. 2.5 5. 10.
BODY    TCMA  -.568 -1.055 -.362 +.093 +.614   //NOSIH XCP PATTERN W/SLA MOD//
CMALP2     =  -0.028
TABLE   TCMQ(5)  /FMACH/
BREAKS  TCMQ  /FMACH/  0.,1.,2.5,5.,10.
BODY    TCMQ  -2200. -4074. -2131. -1467. -1135. //COEF TO CNA  SACKS//
TABLE   TCNB(5)  /FMACH/
BREAKS  TCNB  /FMACH/  0. 1. 2.5 5. 10.
BODY    TCNB  .568 1.055 .362 -.093 -.614      //NOSIH XCP PATTERN W/SLA MOD//
CNBET2     = 0.028
TABLE   TCNR(5)  /FMACH/
BREAKS  TCNR  /FMACH/  0. 1. 2.5 5. 10.
BODY    TCNR  -2200. -4074. -2131. -1467. -1135. //COEF TO CNA  SACKS//
IXTHRS     = 1
NMTR       = 1
AENOZM(1) = .0066     //.07 FT2 NOZZ EXIT AREA//
XNOZEM(1) = .204      //THREADED JOINT DIST FROM BASE 8 IN//
XIMTRD(1) = 0.0
PHMTRD(1) = 0.0
TABLE   TVACT1(11)  /TSTAGE/  *1.030
BREAKS  TVACT1 /TSTAGE/  0. .044 .12 .386 .744 1.56 1.644 1.7 1.776 1.9 5.
BODY    TVACT1  0. 27005. 27868. 23980. 24412. 28299. 24412. 11454. 4978. 0. 0.
TABLE   TMAS11(11)  /TSTAGE/
BREAKS  TMAS11 /TSTAGE/  0. .044 .12 .386 .744 1.56 1.644 1.7 1.776 1.9 5.
BODY    TMAS11  0. -12.82 -13.24 -11.36 -11.56 -13.45 -11.56 -5.25 -2.10 0. 0.
IXVMP      = 1
TABLE   TXCM(3)  /TSTAGE/
BREAKS  TXCM   /TSTAGE/  0. 1.776 10.
BODY    TXCM   1.519 1.797 1.797        //8-20-79 INERT ONE//
TABLE   TIXX(3)  /TSTAGE/
BREAKS  TIXX   /TSTAGE/  0. 1.776 10.
BODY    TIXX   .1514 .1074 .1074        //8-20-79 INERT ONE//
TABLE   TIYY(3)  /TSTAGE/
BREAKS  TIYY   /TSTAGE/  0. 1.776 10.
BODY    TIYY   45.97 33.14 33.14        //8-20-79 INERT ONE//
TABLE   TIZZ(3)  /TSTAGE/
BREAKS  TIZZ   /TSTAGE/  0. 1.776 10.
BODY    TIZZ   45.97 33.14 33.14        //8-20-79 INERT ONE//
NAMSTG(1) = $TIME $
VALSTG(1) = 1.9          //ZUNI 80//
TABLE    TCYB   EXTRAPOLATION
TABLE    TCZA   EXTRAPOLATION
TABLE     TCZQ  EXTRAPOLATION
```

Example 3 (cont)

```
TABLE     TCMA    EXTRAPOLATION
TABLE     TCMQ    EXTRAPOLATION
TABLE     TCNB    EXTRAPOLATION
TABLE     TCNR    EXTRAPOLATION
TABLE      TCYR   EXTRAPOLATION
TABLE     TCL0    EXTRAPOLATION
TABLE     TCLDP   EXTRAPOLATION
TABLE     TCLP    EXTRAPOLATION
*RUN
IXTHRS     = -0
DELETE        TVACT1 TMAS11
IXVMP     =   -0
DELETE        TXCM TIXX TIYY TIZZ
FMASSK    =   38.3899     //KG  84.6 LB CST ACT 1 51 P/L//
XCG8M     =   -.116
FIXXK     =   .0943
FIYYK     =   33.98
FIZZK     =   33.98
DELTS      = 0.2
DELETE        TCXC TCL0
TABLE  TCX0(23)  /FMACH/  *1.00 +.04  EXTRAP
BREAKS TCX0  /FMACH/  0.00 0.24 0.50 0.76 0.90 1.00 1.09 1.15 1.20 1.25
                      1.30 1.40 1.50 1.60 1.70 1.80 2.00 2.20 2.40 2.60
                      2.80 3.00 3.25
BODY   TCX0  -.545 -.539 -.552 -.573 -.704 -.812 -.991 -1.039 -1.050 -1.035
             -1.013 -.962 -.924 -.880 -.838 -.805 -.741 -.690 -.644 -.600
             -.568 -.536 -.495
CLZERO    =  0.0067
NAMSTG(1)    = $TIME $
VALSTG(1)    = 10.
VALSTG(1)    = 5.
VALSTG(1) = 3.5
*RUN
DELTS     = 0.5
NAMSTG(1) = $TIME  $
VALSTG(1) = 20.0
*RUN
DELTS     = 1.0
NAMSTG(1) = $GAMGDD$
VALSTG(1) = -.01
*RUN
NAMSTG(1) = $TIME $
VALSTG(1) = 50.
*RUN
IXVMP        -0
IMPROG    = $PTMASS$
ISPROG    = $PMAERO$
IXAERO    = -1
DELETE        TCX0 TCZA TCZQ TCYB TCYR
DELETE        TCMA TCMG TCNB TCNR
```

Example 3 (cont)

```
IDLAT(1)   = -1
SREFAM     = .0929034  //M2 1 FT2 CHUTE REF AREA//
FMASSK     = 11.79  //KG 26 LB NC ON PILOT CHT//
TABLE  TCX0(5)  /FMACH/
BREAKS TCX0  /FMACH/  0.00 1.00 2.00 3.00 4.00
BODY   TCX0  -25.0 -25.0 -25.0 -25.0 -25.0  //8-FT CROSS CDS 25//
DELTS      = 2.5
DELTS      = 5.0
DELETE        TCLA TCLB TCLDP TCLP
IDPQR      = $CONSTA$
PIBPR      = 0.
NAMSTG(1)  = $HGDBM $
VALSTG(1)  = 10000.
*RUN
DELTS      = 10.
NAMSTG(1)  = $HGDBM $
VALSTG(1)  = 1600.
*RUN
*END
```

APPENDIX B

Flow Charts and Dimensional Data

This appendix contains the logic flow charts, equations evaluated, and sketches showing dimensional data that are pertinent to the various code routines. These routines are functionally grouped, where possible, to show code continuity.

Contents

AMEER - Main Program Overview 73
PHASE 1 - Data Read-In and Store Phase 75
BASERD - Reads "Base Case" File 76
READER - Main Card Read Routine 78
PHASE 2 - Computation Phase Control 79
PHASE 3 - Formatted Output Phase 80
SIXDOF - Six-Degree-of-Freedom Evaluation Designation 81
SDINLZ - Six-Degree-of-Freedom Initialization 82
SDDERV - Six-Degree-of-Freedom-Evaluator 86
TRNACL - Acceleration Computation Routine 94
TRNMOM - Moment Transfer Routine 95
PTMASS - Point Mass Evaluation Designation 96
PMINLZ - Point Mass Initialization 97
PMDERV - Point Mass Evaluator 100
AERODYNAMICS 106
AROINL - Aerodynamic Evaluation Initialization 107
AERO - Aerodynamic Equations Evaluator 108
ATMOSPHERE 111
ATMSET - Selects Requested Atmosphere Properties 112
ATMS - Evaluates Atmosphere Properties 114
AUXILLARY CALCULATIONS 117
AUXCLC - Evaluates Miscellaneous Items 118
TQ2A - Transforms Quaternions to Inertial Angles 121
CONTROL 123
CONTRL - Six-Degree-of-Freedom Control Package 124
PTMSCO - Point Mass Control Package 125
SOLALP - Evaluates Aero Angle (α) for Control 129
SOALBT - Evaluates Aero Angles (α,β) for Control 131
USRPMC - User Written Control Routine 133
ROLANG - Point Mass Roll Angle Evaluator 133
GRAVITY 134
GRAV - Evaluates Gravity Components 134
INITIALIZATION 136
INITLZ - Post-Data Read-In Initialization 137
ZERCOM, ZERTAB/RINIT, DEFALT - Pre-Data Read-In Initialization 139
INTEGRATION 140
INTARA - Sets-Up the Integration Array 141
PUTNRA - Places State/Rate Values in Integration Array 142
TAKFRA - Returns State Values to Program Storage 142
INGRTR - Controls Integration Procedure 143
DERIV - Selects the Derivative Evaluation 144
LATITUDE CONVERSION 145
GD2GC - Geodetic to Geocentric Conversion 146
GC2GD - Geocentric to Geodetic Conversion 148
RADIUS - Evaluates Earth Radius at Latitude 150
MATRIX OPERATION 151
OUTPUT 152
WRTINL - Writes the Initial Report Summary 153
BOPINL - Initializes Block Output Array 154
BOPWRT 156

Contents (cont)

HEADER - Writes Report Page Banner 156
T20INL - Sets up TAPE20 File Data Block 157
T20WRT - Writes TAPE20 Data Block 157
TABLOP - Writes Computed Data in Table Format 158
PLTMKR - Generates Computed Data Plots 159
STRDAT - Selects and Stores Plot Request Data 161
PLOTDT - Calls Plotting Routines 162
STAGING 163
STGINL - Initializes Staging Request Items 164
STGTST - Stage Testing Control 165
STGEQN - Evaluates the Staging Equation 166
STGEQR - Determines Root of Stage Equation 167
STATION/RADAR 168
STAINL - Initializes the Station Position Items 168
RADAR - Calculates Radar "Look" Items 170
THRUST 172
THRINL - Initializes Thrusting Items 172
THRUST - Evaluates Thrusting Items 173
VEHICLE MASS PROPERTIES 176
VMPINL - Initializes the Vehicle Mass Properties Items 176
VEHMSP - Evaluates Vehicle Mass Properties 177
WIND 179
WIND - Determines Wind Velocity Components 179

Illustrations

Figure

B-1 Aerodynamic Parameters and Axis System 110
B-2 Thrust Parameters and Axis System 175
B-3 Vehicle Mass Properties Parameters and Axis System 178

AMEER - Main Program Overview

The AMEER (*A*ero-*M*echanical *E*quations *E*valuation *R*outines) code is structured logically into three phases: (1) data read-in, (2) motion equations evaluation, and (3) formatted output. Associated with the data read-in phase (PHASE1) are the pre- and post-initializations of the code. Preinitialization assigns a set of nominal values to the code variables, while postinitialization computes the code startup values for the various program variables and those state variables that are associated with the integration array.

PHASE2 evaluates the derivatives of the user-selected set of motion equations, and computes program variable values for output and for testing against user specified variable values related to code control.

PHASE3, output, provides for the listing of PHASE2 computed variable values in several different tabular formats and for graphic presentation of data in the form of X-Y plots.

LOGIC FLOW CHART

AMEER:

<table>
<tr><td colspan="6">CALL ZERCOM: Zero-out all /COMMON/ blocks</td></tr>
<tr><td colspan="6">CALL ZERTAB: Zero-out all core TABLE storage</td></tr>
<tr><td colspan="6">CALL DEFALT: Set the code nominal values</td></tr>
<tr><td colspan="6">CALL PHASE1: Reads in problem data and sets the procedure flag: IFLAG</td></tr>
<tr><td colspan="6">IFLAG</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr>
<tr><td>INPUT ERROR</td><td>*RUN</td><td>*OUTPUT</td><td>*CLEAR</td><td>*CLEAR KEEPTABLES</td><td>E-O-F</td></tr>
<tr><td rowspan="5">CALL CSAVE (ISAVE): Dumps the contents of all /COMMON/ blocks onto file ISAVE</td><td rowspan="2">CALL INITLZ: Compute initial values for the main program and related routines</td><td colspan="4">CALL PHASE 3: List the computed data in tabular form and/or plot selected variables</td></tr>
<tr><td rowspan="4"></td><td colspan="2">CALL ZERCOM: Zero-out all /COMMON/ blocks</td><td rowspan="4"></td></tr>
<tr><td>CALL WRTINL: Write the problem report summary page</td><td>CALL ZERTAB: Zero-out TABLE storage</td><td></td></tr>
<tr><td>CALL PHASE 2: Evaluate the main program set of equations</td><td colspan="2">CALL DEFAULT: Set the code nominal values</td></tr>
<tr><td colspan="4">Return to PHASE1 call</td></tr>
<tr><td colspan="6">END</td></tr>
</table>

LOGIC FLOW CHART

PHASE1:

Causes all input data cards to be read until a CONTROL card is encountered.

		Initialize read error flag (IERR=0)	
		Initialize read control flag (IFLAG=0)	
		Do until IFLAG ≠ 0	
	NO	Is merge flag (MERGE) set?	YES
		CALL BASERD: Read base case data (file IMERG) to next: *RUN, *OUTPUT, *END card, or E-O-F. Set: IFLAG	
	NO	Is IFLAG zero?	YES
		CALL PRIMRD: Read the INPUT file to the next: *MERGE, *RUN, *OUTPUT, *CLEAR card, or E-O-F. Set: IFLAG	
	NO	Is IFLAG zero?	YES
		CALL ZERCOM: Zero out all /COMMON/ blocks	
		CALL ZERTAB: Zero out all TABLE storage	
		CALL DEFALT: Set the code nominal values	
		IFIRST=0	
		CALL TABCMP: Check all TABLE descriptions for completeness	
NO		Are all tables complete?	YES
		Set: IFLAG=0	

LOGIC FLOW CHART

BASERD:

Reads card image data from the "base case" file (TAPE9/ IMERG) for the data merging option.

	IFLAG=0	
NO	Is the base case at End-of-File?	YES
	Do until *RUN, *OUTPUT, *END, or E-O-F is encountered.	
	CALL READER: Read input file until an unrecognized card or control card is encountered	
	NO Was E-O-F encountered by READER?	YES
	Extract keyword and compare against keyword list.	
	Do Proper Case	

UNRECOGNIZED KEYWORD	ILLEGAL KEYWORD	*COMMENT	*SAVE	*RESTORE	*RUN	*OUTPUT	*END
Print message and set IERR=1		↓	CALL CSAVE	CALL CREAD	↓	Set: IFLAG=2	Set: IBCEOF=1

NO	Is IFLAG still zero?	YES
	Print message that base has been fully read.	
	MERGE = -1 (base case has been read for now).	

LOGIC FLOW CHART

PRIMRD:

Reads card image input data from the "Primary" (input) file

Do until *MERGE, *RUN, *OUTPUT, or E-O-F is encountered.

CALL READER: Read input until unrecognized card or control card is encountered.

Extract keyword and compare against keyword list.

Do for the keyword case

UNRECOGNIZED WORD	Print Diagnostic; IERR=1
*COMMENT	
*COPY	Copy cards up to next *STOP card to file IMERG.
*STOP	Print: Card out of sequence, IERR=1
*MERGE	Rewind IMERG file, MERGE=1, IBCEOF=0, IFLAG=0
*SAVE	CALL CSAVE to write all table data to file ISAVE
*RESTORE	CALL CREAD to read all table data from file ISAVE
*RUN	IFLAG=1; MERGE=IABS (MERGE)
*OUTPUT	IFLAG=2
*CLEAR	IFLAG=3; if word "KEEPTABLE" follows, IFLAG=4
*END	IFLAG=5

LOGIC FLOW CHART

READER:

Main routine for processing card image input data for the input and table handling package.

- Initialize the reader flags
- Do until an unrecognized card is encountered.
  - Read a card.
  - If E-O-F, print diagnostics, set IERR=1, and RETURN.
  - Print the card.
  - Find last effective column (col 80 unless // occurs).
  - If first nonblank is before col 7, read keyword; otherwise use old keyword.
  - Process proper keyword....

Variable or Array Element Definition

- Is first nonblank before col 7?
  - NO: Is "$" Hollerith indicator present?
    - NO: CALL READV1: Read each value in turn, and store it in the next word of the array. Quit when all data is read.
    - YES: For all successive characters, compute its word and character position and store it there.
  - YES:
    - Read subscript, if any.
    - Translate subscript into array position.
    - Determine whether name implies real or integer type.

TABLE

- If first nonblank is before col 7, CALL READTN to read the name of the table.
- CALL READTC to read the specifications on the card.

BREAKS

- Is first nonblank before col 7?
  - NO: CALL READV1 to read each value in turn, then store it in the next word of the breakpoint list. Quit when all data on the card is read.
  - YES:
    - Read name of table.
    - Read independent variable names.
    - Determine which demension the name goes with.
    - Read subscript, if any, and translate it into an array position.

BODY

- Is first nonblank before col 7?
  - NO: If table is formatted, decode whole card and save.
  - YES:
    - Read name of table.
    - Read subscript, if none, presume all indices = 1.
    - Translate the subscripts into a position within the body.
    - Blank out the portion of the card now processed.
- DO WHILE data remains.
  - Get next value from READV1 as saved list, if formatted.
  - Store value in main or extended memory.
  - Increment counter of values in this table body.
  - If this is last element of $$ table save subspace on file ITABL.

DELETE

- Read each name and CALL TABDEL to delete that table

FORMAT

- Read number of items. Copy into format array.

Unrecognized

LOGIC FLOW CHART

PHASE2:

Provides control of the integration and computation phase.

NO	Is latitude position geocentric?				YES
	IDLAT=$GEOCENTRIC$				
	CALL DERIV: Evaluate derivatives for initial values				
	CALL AUXCLC: Evaluate auxillary items for initial values				
	CALL BOPWRT: Write the initial values block				
NO	Is file TAPE20 to be written?				YES
	CALL T20WRT: Write the initial value block to file (TAPE20)				
	Do for each STAGING variable.				
	Evaluate the staging equation (STGEQN) for STGEQV value				
	Do for number of equations (NEQN)				
	Save the current state and rate array values				
	Set the integrator control items				
	Do until TOUT reaches staging time or TMAX				
	Set TOUT for one integration interval				
	CALL INGRTR: Advance problem solution one good interval				
	NO	Is integration start up procedure in effect?			YES
		Set: Program TIME to TOUT			
	NO	Is TAPE20 file to be written?		YES	
		CALL T20WRT: Write current time history block			
	NO	Has staging criteria been met?			YES
	NO	Has program TIME exceeded TMAXS?		YES	
	NO	Is integration start up procedure in effect?	YES		
		Reset program TIME to start TIME			
		Turn off integration start up flag			

LOGIC FLOW CHART

PHASE3:

Displays the data computed in PHASE 2 by means of a set of tables or a series of graphic plots. The plots can be stacked wrt cases or according to compatible variables.

NO	Have tabular listings been requested?		YES
	CALL TABLOP: To generate tabular listings of PHASE2 data stored on file: TAPE20		
NO	Have graphic plots of data been requested?		YES
	CALL DSTART: Initialize DISSPLA package		
	Do for 5 plot requests		
	NO	Is plot request blank?	YES
	Set: Number of plots to 5	Set: Number of plots to counter	
		NO Is counter zero?	YES
	CALL PLTMKR: Generate the requested plots		Print: no plot message
	CALL DONEPL: Terminate DISSPLA plotting package		
	Blank out plot request storage arrary for next plot		

SIXDOF:

The main program designator, SIXDOF, calls into execution two routines, SDINLZ and SDDERV, which initialize and evaluate the motion of a rigid-body flight vehicle over a rotating-oblate planet. The evaluation of the motion equations includes the effects of an offset center-of-mass, a thrusting and swiveling propulsion system, varying mass properties, wind shear and gusts, and aerodynamic force and moment components.

The problem-starting position, velocity, and angular data are usually given to the code relative to a geodetic axis system; these data are then transformed to a geocentric system for continuing computation. However, geocentric data can be supplied if the user desires.

Forces and moments are summed in the vehicle body-axis system and transferred to the inertial system. Integration of the motion derivatives takes place in the inertial frame. Transformations in the form of direction cosine matrices are used throughout to relate the various axis systems. However, quaternions, which represent the rigid-body rotations, are initially derived from the inertial to body transformation components and given to the integrator for updating during code execution to eliminate the singularities inherent in Euler angle integration. After integration, a direction cosine matrix is computed from the quaternion components to relate inertial to body items.

Two routines, TRNACL and TRNMOM, support the SDDERV evaluation of the six-degree-of-freedom motion derivatives. TRNACL calculates the accelerations acting along the xyz axes at the body reference by summing and transferring the body forces to the reference point. TRNMOM computes the moments about the body reference point by summing and transferring the various moment components to the reference point.

LOGIC FLOW CHART

SDINLZ:

Problem input data, related to a geodetic or geocentric axis system, are transformed into inertially referenced values for insertion into the integrator state array as initial values for code start up.

- Compute problem start longitude change, B_i
- Is IFIRST equal to zero?
  - NO:
    - Is input data GEOCENTRIC?
      - NO:
        - CALL GD2GC: Compute geocentric latitude (ϕ_{gc}), altitude (h_{gc}), and inertial distance to body (R_i)
        - Are flight path angles given?
          - NO: Compute flight path angles (γ,σ)
          - YES: Compute aero angles (α,β, ϕ_v)
        - Compute geodetic velocity components ($\dot{X}_{gd}$, $\dot{Y}_{gd}$, $\dot{Z}_{gd}$)
        - Convert geodetic velocity components to geocentric components ($\dot{X}_{gc}$, $\dot{Y}_{gc}$, $\dot{Z}_{gc}$)
        - Compute geodetic to body transformation (TGD2BP)
        - Compute geocentric to geodetic transformation (TGC2GD)
        - Compute geocentric to body transformation (TGC2BP) from geodetic items
      - YES:
        - CALL RADIUS: Compute geocentric earth radius (R_{gc})
        - Compute inertial radius to body (R_i)
        - Compute geocentric to body transformation (TGC2BP) from current values of geocentric items
    - Compute inertial to geocentric transformation (TI2GC)
    - Compute inertial to body transformation (TI2BP)
    - Compute inertial Euler angles (θ_i, ψ_i, ϕ_i)
    - Compute quaternion components (λ_0, λ_1, λ_2, λ_3)
    - Compute inertial displacements of body (X_i, Y_i, Z_i)
    - Compute inertial velocity components ($\dot{X}_i$, $\dot{Y}_i$, $\dot{Z}_i$)
    - Compute body components of inertial velocity (U, V, W)
    - Are p, q, r spin rates inertially referenced?
      - NO:
        - Compute inertial p, q, r values (p_i, q_i, r_i) from earth referenced values (p_e, q_e, r_e)
        - Has a zero roll rate been requestd?
          - NO:
          - YES:
            - Set roll rate (p_i) = 0.0
        - Set p, q, r flag (IDPQR) to INERTIAL
      - YES: IDPQR = $INERTIAL$
    - Save the current time
  - YES:

Equations Evaluated

SDINLZ:

PROBLEM START LONGITUDE CHANGE

$$B_i = \theta_{L_o} - \omega_p t - \theta_L$$

PROBLEM START WITH AERO AND FLIGHT PATH ANGLES

TRANSFORMATION GEODETIC TO VELOCITY

TGD2V $[T_{V/GD}]$ TGD2V

$$\begin{aligned}
(1,1) &= \cos\sigma_{gd}\cos\gamma_{gd}\\
(2,1) &= -\sin\sigma_{gd}\\
(3,1) &= \cos\sigma_{gd}\sin\gamma_{gd}\\
(1,2) &= \sin\sigma_{gd}\cos\gamma_{gd}\\
(2,2) &= \cos\sigma_{gd}\\
(3,2) &= \sin\sigma_{gd}\sin\gamma_{gd}\\
(1,3) &= -\sin\gamma_{gd}\\
(2,3) &= 0.0\\
(3,3) &= \cos\gamma_{gd}
\end{aligned}$$

TRANSFORMATION VELOCITY TO BODY

TV2B $[T_{B/V}]$ TV2B

$$\begin{aligned}
(1,1) &= \cos\alpha\cos\beta\\
(2,1) &= -\cos\phi_v\sin\beta - \sin\phi_v\sin\alpha\cos\beta\\
(3,1) &= \sin\phi_v\sin\beta - \cos\phi_v\sin\alpha\cos\beta\\
(1,2) &= \cos\alpha\sin\beta\\
(2,2) &= \cos\phi_v\cos\beta - \sin\phi_v\sin\alpha\sin\beta\\
(3,2) &= -\sin\phi_v\cos\beta - \cos\phi_v\sin\alpha\sin\beta\\
(1,3) &= -\sin\alpha\\
(2,3) &= \sin\phi_v\cos\alpha\\
(3,3) &= \cos\phi_v\cos\alpha
\end{aligned}$$

TRANSFORMATION GEODETIC TO BODY

$$|T_{B/GD}| = |T_{B/V}|' \cdot |T_{V/GD}|$$

GEODETIC EULER ANGLES

$$\psi_{B_{gd}} = \tan^{-1}(TGD2B(1,2)/TGD2B(1,1))$$
$$\theta_{B_{gd}} = \sin^{-1}(-TGD2B(1,3))$$
$$\phi_{B_{gd}} = \tan^{-1}(TGD2B(2,3)/TGD2B(3,3))$$

GEODETIC VELOCITY COMPONENTS

$$\dot{X}_{gd} = V_{T_{gd}}\cos\gamma_{gd}\cos\sigma_{gd}$$
$$\dot{Y}_{gd} = V_{T_{gd}}\cos\gamma_{gd}\sin\sigma_{gd}$$
$$\dot{Z}_{gd} = -V_{T_{gd}}\sin\gamma_{gd}$$

GEOCENTRIC VELOCITY COMPONENTS

$$\Delta\phi = \phi_{gd} - \phi_{gc}$$
$$\dot{X}_{gc} = \dot{X}_{gd}\cos\Delta\phi - \dot{Z}_{gd}\sin\Delta\phi$$
$$\dot{Y}_{gc} = \dot{Y}_{gd}$$
$$\dot{Z}_{gc} = \dot{X}_{gd}\sin\Delta\phi + \dot{Z}_{gd}\cos\Delta\phi$$

TRANSFORMATION GEODETIC TO BODY

TGD2BP $(1,1) = \cos\theta_{B_{gd}} \cos\psi_{B_{gd}}$

$(2,1) = -\sin\psi_{B_{gd}} \cos\phi_{B_{gd}} + \cos\psi_{B_{gd}} \sin\theta_{B_{gd}} \sin\phi_{B_{gd}}$

$[T_{B/GD}]$ $(3,1) = \sin\psi_{B_{gd}} \sin\phi_{B_{gd}} + \cos\psi_{B_{gd}} \sin\theta_{B_{gd}} \cos\phi_{B_{gd}}$

$(1,2) = \cos\theta_{B_{gd}} \sin\psi_{B_{gd}}$

$(2,2) = \cos\psi_{B_{gd}} \cos\phi_{B_{gd}} + \sin\psi_{B_{gd}} \sin\theta_{B_{gd}} \sin\phi_{B_{gd}}$

$(3,2) = -\cos\psi_{B_{gd}} \sin\phi_{B_{gd}} + \sin\psi_{B_{gd}} \sin\theta_{B_{gd}} \cos\phi_{B_{gd}}$

$(1,3) = -\sin\theta_{B_{gd}}$

$(2,3) = \cos\theta_{B_{gd}} \sin\phi_{B_{gd}}$

TGD2BP $(3,3) = \cos\theta_{B_{gd}} \cos\phi_{B_{gd}}$

TRANSFORMATION GEOCENTRIC TO GEODETIC

TGC2GD $(1,1) = \cos\Delta\phi$

$(2,1) = 0.0$

$(3,1) = -\sin\Delta\phi$

$[T_{GD/GC}]$ $(1,2) = 0.0$

$(2,2) = 1.0$

$(3,2) = 0.0$

$(1,3) = \sin\Delta\phi$

$(2,3) = 0.0$

TGC2GD $(3,3) = \cos\Delta\phi$

INTERIAL SYSTEM DISTANCE: ORIGIN TO BODY

$$R_i = R_{gc} + h_{gc}$$

TRANSFORMATION GEOCENTRIC TO BODY (Geocentric Items)

TGC2BP $(1,1) = \cos\theta_{B_{gc}} \cos\psi_{B_{gc}}$

$(2,1) = \cos\theta_{B_{gc}} \sin\psi_{B_{gc}}$

$(3,1) = -\sin\theta_{B_{gc}}$

$[T_{B/GC}]$ $(1,2) = -\sin\psi_{B_{gc}} \cos\phi_{B_{gc}} + \cos\psi_{B_{gc}} \sin\theta_{B_{gc}} \sin\phi_{B_{gc}}$

$(2,2) = -\cos\psi_{B_{gc}} \cos\phi_{B_{gc}} + \sin\psi_{B_{gc}} \sin\theta_{B_{gc}} \sin\phi_{B_{gc}}$

$(3,2) = \cos\theta_{B_{gc}} \sin\phi_{B_{gc}}$

$(1,3) = \sin\psi_{B_{gc}} \sin\phi_{B_{gc}} + \cos\psi_{B_{gc}} \sin\theta_{B_{gc}} \cos\phi_{B_{gc}}$

$(2,3) = -\cos\psi_{B_{gc}} \sin\phi_{B_{gc}} + \cos\psi_{B_{gc}} \sin\theta_{B_{gc}} \cos\phi_{B_{gc}}$

TGC2BP $(3,3) = \cos\theta_{B_{gc}} \cos\phi_{B_{gc}}$

or: (Geodetic Items)

$$|T_{B/GC}| = |T_{B/GD}| \cdot |T_{GD/GC}|$$

TRANSFORMATION INERTIAL TO GEOCENTRIC

TI2GC $(1,1) = -\sin \phi_{gc} \cos B_i$

$(2,1) = \sin B_i$

$(3,1) = -\cos \phi_{gc} \cos B_i$

$(1,2) = -\sin \phi_{gc} \sin B_i$

$[T_{GC/I}]$ $(2,2) = -\cos B_i$

$(3,2) = -\cos \phi_{gc} \sin B_i$

$(1,3) = -\cos \phi_{gc}$

$(2,3) = 0.0$

TI2GC $(3,3) = \sin \phi_{gc}$

TRANSFORMATION INERTIAL TO BODY

$$|T_{B/I}| = |T_{B/GC}| \cdot |T_{GC/I}| \quad \text{(TI2BP)}$$

INERTIAL EULER ANGLES

θ_i $\phi_i = \sin^{-1}(-\text{TI2BP}(1,3))$

$\psi_i = \tan^{-1}(\text{TI2BP}(1,2)/\text{TI2BP}(1,1))$

$\phi_i = \tan^{-1}(\text{TI2BP}(2,3)/\text{TI2BP}(3,3))$

QUATERNION COMPONENTS

$$\lambda_0 = \cos(\psi_i/2)\cos(\theta_i/2)\cos(\phi_i/2) + \sin(\psi_i/2)\sin(\theta_i/2)\sin(\phi_i/2)$$

$$\lambda_1 = -\sin(\psi_i/2)\sin(\theta_i/2)\cos(\phi_i/2) + \cos(\psi_i/2)\cos(\theta_i/2)\sin(\phi_i/2)$$

$$\lambda_2 = \cos(\psi_i/2)\sin(\theta_i/2)\cos(\phi_i/2) + \sin(\psi_i/2)\cos(\theta_i/2)\sin(\phi_i/2)$$

$$\lambda_3 = \sin(\psi_i/2)\cos(\theta_i/2)\cos(\phi_i/2) - \cos(\psi_i/2)\sin(\theta_i/2)\sin(\phi_i/2)$$

BODY INERTIAL DISPLACEMENTS

$X_i = R_i \cos \phi_{gc} \cos B_i$

$Y_i = R_i \cos \phi_{gc} \sin B_i$

$Z_i = -R_i \sin \phi_{gc}$

INERTIAL VELOCITY COMPONENTS

$$\begin{vmatrix} \dot{X}_i \\ \dot{Y}_i \\ \dot{Z}_i \end{vmatrix} = \begin{vmatrix} T_{GC/I} \end{vmatrix}' \cdot \begin{vmatrix} \dot{X}_{gc} \\ \dot{Y}_{gc} \\ \dot{Z}_{gc} \end{vmatrix} + \begin{vmatrix} \omega_p Y_i \\ -\omega_p X_i \\ 0.0 \end{vmatrix}$$

BODY COMPONENTS OF INERTIAL VELOCITY

$$\begin{vmatrix} U \\ V \\ W \end{vmatrix} = \begin{vmatrix} T_{B/I} \end{vmatrix} \cdot \begin{vmatrix} \dot{X}_i \\ \dot{Y}_i \\ \dot{Z}_i \end{vmatrix}$$

INERTIAL ROLL RATES FROM EARTH REFERENCED RATES

$$\begin{vmatrix} p_i \\ q_i \\ r_i \end{vmatrix} = \begin{vmatrix} T_{B/GC} \end{vmatrix} \cdot \begin{vmatrix} \cos \phi_{gc}(\dot{Y}_{gc}/R_i \cos \phi_{gc}) + \omega_p \\ -\dot{X}_{gc}/R_i \\ -\sin \phi_{gc}(\dot{Y}_{gc}/R_i \cos \phi_{gc}) + \omega_p \end{vmatrix} + \begin{vmatrix} p_e \\ q_e \\ r_e \end{vmatrix}$$

COMPUTE AERO ANGLES:

$$\sin \phi_v = \cos \phi_{B_{gd}} \sin \theta_{B_{gd}} (\sin \psi_{B_{gd}} \cos \gamma_{gd} - \cos \psi_{B_{gd}} \sin \gamma_{gd})$$
$$- \sin \theta_{B_{gd}} (\cos \psi_{B_{gd}} \cos \gamma_{gd} + \sin \psi_{B_{gd}} \sin \sigma_{gd})$$

$$\cos \phi_v = \sin \phi_{B_{gd}} \sin \gamma_{gd} (\sin \psi_{B_{gd}} \cos \gamma_{gd} - \cos \psi_{B_{gd}} \sin \sigma_{gd})$$
$$+ \cos \phi_{B_{gd}} \sin \theta_{B_{gd}} \sin \gamma_{gd} (\cos \psi_{B_{gd}} \cos \gamma_{gd} + \sin \psi_{B_{gd}} \sin \sigma_{gd})$$
$$+ \cos \phi_{B_{gd}} \cos \theta_{B_{gd}} \cos \gamma_{gd}$$

$$\phi_v = \tan^{-1} (\sin \phi_v / \cos \phi_v)$$

$$\dot{X}_{wgc} = \dot{X}_{wgd} \cos \Delta\phi - \dot{Z}_{wgd} \sin \Delta\phi$$
$$\dot{Y}_{wgc} = \dot{Y}_{wgd}$$
$$\dot{Z}_{wgc} = \dot{X}_{wgd} \sin \Delta\phi + \dot{Z}_{wgd} \cos \Delta\phi$$

$$\begin{vmatrix} U_w \\ V_w \\ W_w \end{vmatrix} = \begin{vmatrix} T_{B/I} \end{vmatrix} \cdot \begin{vmatrix} T_{I/GC} \end{vmatrix} \cdot \begin{vmatrix} \dot{X}_{wgc} \\ \dot{Y}_{wgc} \\ \dot{Z}_{wgc} \end{vmatrix} + \begin{vmatrix} \omega_p Y_i \\ -\omega_p X_i \\ 0.0 \end{vmatrix}$$

$$\alpha_B = \tan^{-1} \left(\frac{W - W_w}{U - U_w} \right)$$

$$\beta_B = \tan^{-1} \left(\frac{V - V_w}{U - U_w} \right)$$

Intentionally Left Blank

LOGIC FLOW CHART

SDDERV:

The derivatives of the rigid body six-degree-of-freedom equations are evaluated from current state variable values for insertion into the integrator rate array to advance the problem solution one step. The program variable values that are directly related to the derivative evaluation are the only ones computed.

	CALL ORTHOG: to orthogonolize the quaternion (λ_0, λ_1, λ_2, λ_3)	
	Compute inertial to body transformation (TI2BP)	
	Compute inertial velocity components ($\dot{X}_i$, $\dot{Y}_i$, $\dot{Z}_i$)	
	Compute longitude change (B_i) and current longitude (θ_L)	
	Compute inertial distance to body (R_i)	
	Compute geocentric latitude (ϕ_{gc}), earth radius ($R\phi_{gc}$), altitude (h_{gc})	
	CALL GCZGD: to get geodetic latitude (ϕ_{gd}) and altitude (h_{gd})	
	Compute inertial to geocentric transformation (TI2GC)	
	Compute geocentric velocity components ($\dot{X}_{gc}$, $\dot{Y}_{gc}$, $\dot{Z}_{gc}$)	
	Compute geodetic velocity components ($\dot{X}_{gd}$, $\dot{Y}_{gd}$, $\dot{Z}_{gd}$)	
	CALL ATMS: to get atmosphere properties at current altitude ($P, T, \rho, V_s, \upsilon$)	
NO	Are gravity components required?	YES
	CALL GRAV: to get X and Z gravity components ($g_{x_{gc}}$, $g_{z_{gc}}$)	
	Compute gravity components in body system (g_{xB}, g_{yB}, g_{zB})	
NO	Are wind effects requested?	YES
	CALL WIND: to get geodetic wind components ($\dot{X}_{w_{gd}}$, $\dot{Y}_{w_{gd}}$, $\dot{Z}_{w_{gd}}$)	
	Convert winds from geodetic to geocentric ($\dot{X}_{w_{gc}}$, $\dot{Y}_{w_{gc}}$, $\dot{Z}_{w_{gc}}$)	
	Convert winds from geocentric to inertial ($\dot{X}_{w_i}$, $\dot{Y}_{w_i}$, $\dot{Z}_{w_i}$)	
	Compute and add: inertial earth spin components ($\omega_p Y_i$, $\omega_p X_i$)	
	Compute body components of inertial winds (U_w, V_w, W_w)	
	Compute: Velocity in Air (Va), dynamic pressure (q), Mach Number (Mn), Reynolds number (Rn)	
	Compute: Body angles (α, β), body angle rates ($\dot{\alpha}$, $\dot{\beta}$) Total angle-of-attack (α_T), aerodynamic roll angle (ϕ_A)	

SDDERV (cont):

NO	Is aerodynamic routine activated?	YES
	CALL AERO: get aerodynamic forces and moments	
NO	Is thrust routine activated?	YES
	CALL THRUST: get thrusting forces and moments	
NO	Is vehicle mass properties routine activated?	YES
	CALL VEHMSP: get moments of inertia (I_{xx}, I_{yy}, I_{zz}, I_{xy}, I_{xz}, I_{yz}), moment of inertial rate ($\dot{I}_{xx}$, $\dot{I}_{yy}$, $\dot{I}_{zz}$, $\dot{I}_{xy}$, $\dot{I}_{xz}$, $\dot{I}_{yz}$), center of mass location (X_{cm}, Y_{cm}, Z_{cm})	
	Sum the external forces and moments (F_{xT}, F_{yT}, F_{zT}, M_{xT}, M_{yT}, M_{zT})	
	CALL TRNMOM: get the transferred moments due to center-of-mass movement (L_{xx}, M_{yy},N_{zz})	
	CALL TRNACL: get the transferred accelerations due to center-of-mass movement (A_{xx}, A_{yy},A_{zz})	
	Solve the simultaneous equations for p, q, r rates ($\dot{p}$, $\dot{q}$, $\dot{r}$)	
	Compute the body components of inertial accelerations ($\dot{U}_i$, $\dot{V}_i$, $\dot{W}_i$)	
	Compute quaternion rates ($\dot{\lambda}_0$, $\dot{\lambda}_1$, $\dot{\lambda}_2$, $\dot{\lambda}_3$)	
NO	Are range calculations requested?	YES
	Compute the trajectory path length rate ($\dot{T}_{PATH}$)	

Equations Evaluated

SDDERV:

INERTIAL TO BODY TRANSFORMATION

TI2BP	(1,1)	$= 2.(\lambda_0^2 + \lambda_1^2)-1.$
	(2,1)	$= 2.(\lambda_1 \lambda_2 - \lambda_0 \lambda_3)$
	(3,1)	$= 2.(\lambda_1 \lambda_3 + \lambda_0 \lambda_2)$
$[T_{B/I}]$	(1,2)	$= 2.(\lambda_1 \lambda_2 + \lambda_0 \lambda_3)$
	(2,2)	$= 2.(\lambda_0^2 + \lambda^2)-1.$
	(3,2)	$= 2.(\lambda_2 \lambda_3 - \lambda_0 \lambda_1)$
	(1.3)	$= 2.(\lambda_1 \lambda_3 - \lambda_0 \lambda_2)$
	(2,3)	$= 2.(\lambda_2 \lambda_3 + \lambda_0 \lambda_1)$
TI2BP	(3,3)	$= 2.(\lambda_0^2 + \lambda^2)-1.$

INERTIAL VELOCITY COMPONENTS

$$\begin{vmatrix} \dot{X}_i \\ \dot{Y}_i \\ \dot{Z}_i \end{vmatrix} = \begin{vmatrix} T_{B/I} \end{vmatrix}' \cdot \begin{vmatrix} U \\ V \\ W \end{vmatrix}$$

LONGITUDE CHANGE AND CURRENT LONGITUDE

$$B_i = \tan^{-1} (Y_i/X_i)$$

$$\theta_L = \theta_{L_o} - B_i - \omega_p t$$

INERTIAL DISTANCE TO BODY

$$R_i = \sqrt{X_i^2 + Y_i^2 + Z_i^2}$$

GEOCENTRIC LATITUDE AND ALTITUDE

$$\phi_{gc} = \sin^{-1} (-Z_i/R_i)$$

$$h_{gc} = R_i - R_{\phi_{gc}}$$

INERTIAL TO GEOCENTRIC TRANSFORMATION

$$
\begin{array}{ll}
\text{TI2GC} & (1,1) = -\sin\phi_{gc}\cos B_i \\
 & (2,1) = \sin B_i \\
 & (3,1) = -\cos\phi_{gc}\sin B_i \\
 & (1,2) = -\sin\phi_{gc}\sin B_i \\
[T_{GC/I}] & (2,2) = -\cos B_i \\
 & (3,2) = -\cos\phi_{gc}\sin B_i \\
 & (1,3) = -\cos\phi_{gc} \\
 & (2,3) = 0.0 \\
\text{TI2GC} & (3,3) = \sin\phi_{gc}
\end{array}
$$

GEOCENTRIC VELOCITY COMPONENTS

$$
\begin{vmatrix} \dot{X}_{gc} \\ \dot{Y}_{gc} \\ \dot{Z}_{gc} \end{vmatrix} = \begin{vmatrix} T_{GC/I} \end{vmatrix} \cdot \begin{vmatrix} \dot{X}_i - \omega_p Y_i \\ \dot{Y}_i + \omega_p \dot{X}_i \\ \dot{Z}_i \end{vmatrix}
$$

GEODETIC VELOCITY COMPONENTS

$$
\begin{aligned}
\Delta\phi &= \phi_{gd} - \phi_{gc} \\
\dot{X}_{gd} &= \dot{X}_{gc}\cos\Delta\phi + \dot{Z}_{gc}\sin\Delta\phi \\
\dot{Y}_{gd} &= \dot{Y}_{gc} \\
\dot{Z}_{gd} &= -\dot{X}_{gc}\sin\Delta\phi + \dot{Z}_{gc}\cos\Delta\phi
\end{aligned}
$$

GRAVITY COMPONENTS IN BODY SYSTEM

$$
\begin{vmatrix} g_{X_B} \\ g_{Y_B} \\ g_{Z_B} \end{vmatrix} = \begin{vmatrix} T_{B/I} \end{vmatrix} \cdot \begin{vmatrix} T_{GC/I} \end{vmatrix}' \cdot \begin{vmatrix} g_{X_{gc}} \\ 0.0 \\ g_{Z_{gc}} \end{vmatrix}
$$

GEOCENTRIC WIND VELOCITY COMPONENTS

$$
\begin{aligned}
\dot{X}_{w_{gc}} &= \dot{X}_{w_{gd}}\cos\Delta\phi - \dot{Z}_{w_{gd}}\sin\Delta\phi \\
\dot{Y}_{w_{gc}} &= \dot{Y}_{w_{gd}} \\
\dot{Z}_{w_{gc}} &= \dot{X}_{w_{gd}}\sin\Delta\phi + \dot{Z}_{w_{gd}}\cos\Delta\phi
\end{aligned}
$$

INERTIAL VELOCITY COMPONENTS

WINDS

$$
\begin{vmatrix} \dot{X}_{w_i} \\ \dot{Y}_{w_i} \\ \dot{Z}_{w_i} \end{vmatrix} = \begin{vmatrix} T_{GC/I} \end{vmatrix}' \cdot \begin{vmatrix} \dot{X}_{w_{gc}} \\ \dot{Y}_{w_{gc}} \\ \dot{Z}_{w_{gc}} \end{vmatrix} + \begin{vmatrix} \omega_p Y_i \\ -\omega_p X_i \\ 0.0 \end{vmatrix}
$$

NO WINDS

$$
\begin{aligned}
\dot{X}_{w_i} &= \omega_p Y_i \\
\dot{Y}_{w_i} &= -\omega_p X_i \\
\dot{Z}_{w_i} &= 0.0
\end{aligned}
$$

BODY COMPONENTS OF INERTIAL WIND VELOCITIES

$$\begin{vmatrix} U_w \\ V_w \\ W_w \end{vmatrix} = \begin{vmatrix} T_{B/I} \end{vmatrix} \cdot \begin{vmatrix} \dot{X}_{w_i} \\ \dot{Y}_{w_i} \\ \dot{Z}_{w_i} \end{vmatrix}$$

TOTAL VELOCITY W.R.T. AIR

$$V_a = \sqrt{(U-U_w)^2 + (V-V_w)^2 + (W-W_w)^2}$$

DYNAMIC PRESSURE

$$q = 1/2\rho\ V_a^2$$

MACH NUMBER

$$M_N = V_a/V_S, \text{ if } V_S \neq 0.0$$

UNITS REYNOLDS NUMBER

$$R_N = V_a/\nu \text{ if } \nu \neq 0.0$$

BODY ANGLES AND BODY ANGLE RATES

$$\alpha = \tan^{-1}\left(\frac{W - W_w}{U - U_w}\right)$$

$$\beta = \tan^{-1}\left(\frac{V - V_w}{U - U_w}\right)$$

$$\dot{\alpha} = (\alpha_{i-1} - \alpha_i)/\Delta t$$

$$\dot{\beta} = (\beta_{i-1} - \beta_i)/\Delta t$$

TOTAL ANGLE OF ATTACK

$$\alpha_T = \tan^{-1}\left(\frac{(W - W_w)^2 - (V - V_w)^2}{(U - U_w)^2}\right)^{1/2}$$

AERODYNAMIC ROLL ANGLE

$$\phi_A = \tan^{-1}\left(\frac{V - V_w}{W - W_w}\right)$$

SUM OF FORCES AND MOMENTS

$$F_x = F_{x_a} + F_{xT}$$

$$F_y = F_{y_a} + F_{yT}$$

$$F_z = F_{z_a} + F_{zT}$$

$$M_x = M_{x_a} + M_{xT}$$

$$M_y = M_{y_a} + M_{yT}$$

$$M_z = M_{z_a} + M_{zT}$$

MOMENT TRANSFER FOR CENTER OF MASS MOVEMENT (TRNMOM)

ABOUT X-X:

$$\begin{aligned}
L = {} & M_x - [(I_{zz} - I_{yy}) - m(Y_{cg}^2 - Z_{cg}^2)]qr - [I_{xy} - mX_{cg}Y_{cg}]pr \\
& -[I_{xz} - mX_{cg}Y_{cg}]\ qp + [I_{yz} - mY_{cg}Z_{cg}]\ (r^2 - q^2) \\
& -\{[\dot{I}_{xx} - 2m(Y_{cg}\dot{Y}_{cg} + Z_{cg}\dot{Z}_{cg}) - \dot{m}(Y_{cg}^2 + Z_{cg}^2)] - \dot{m}[(Y_{cg} - Y_e)^2 + (Z_{cg} - Z_e)^2]\}\ p \\
& +\{[\dot{I}_{xy} - m(\dot{X}_{cg}Y_{cg} + X_{cg}\dot{Y}_{cg}) - \dot{m}X_{cg}Y_{cg}] - m(\dot{X}_{cg}Y_{cg} - X_{cg}\dot{Y}_{cg}) - \dot{m}[Y_{cg}(X_{cg} - X_e) - X_e(Y_{cg} - Y_e)]\}\ q \\
& +\{[\dot{I}_{xz} - m(\dot{X}_{cg}Z_{cg} + X_{cg}\dot{Z}_{cg}) - \dot{m}X_{cg}Z_{cg}] - m(\dot{X}_{cg}Z_{cg} - X_{cg}\dot{Z}_{cg}) - \dot{m}[Z_{cg}(X_{cg} - X_e) - X_e(Z_{cg} - Z_e)]\}\ r \\
& -\dot{m}(Z_e\dot{Y}_e - Y_e\dot{Z}_e) - Y_{cg}\ [F_z - m\ \ddot{Z}_{cg} - 2\dot{m}(\dot{Z}_{cg} - Z_e) - \ddot{m}\ Z_e] \\
& +Z_{cg}[F_y - m\ddot{Y}_{cg} - 2\dot{m}(\dot{Y}_{cg} - \dot{Y}_e) - \ddot{m}Y_e]
\end{aligned}$$

ABOUT Y-Y:

$$\begin{aligned}
M = {} & M_y - [(\dot{I}_{xx} - I_{zz}) - m(Z_{cg}^2 - X_{cg}^2)]pr + [I_{xy} - mX_{cg}Y_{cg}]qr \\
& -(I_{yz} - mY_{cg}Z_{cg})pq - [I_{xz} - mX_{cg}Z_{cg}](p^2 - r^2) \\
& +\{[\dot{I}_{xy} - m(X_{cg}\dot{Y}_{cg} + \dot{X}_{cg}Y_{cg}) - \dot{m}X_{cg}Y_{cg}] - m(X_{cg}\dot{Y}_{cg} - \dot{X}_{cg}Y_{cg}) - \dot{m}[X_{cg}(Y_{cg} - Y_e) - Y_e(X_{cg} - X_e)]\}p \\
& -\{[\dot{I}_{yy} - 2m(X_{cg}\dot{X}_{cg} + Z_{cg}\dot{Z}_{cg}) - \dot{m}(X_{cg}^2 + Z_{cg}^2)] - m[(X_{cg} - X_e) + (Z_{cg} - Z_e)^2]\}\ q \\
& +\{[\dot{I}_{yz} - m(\dot{Y}_{cg}Z_{cg} + Y_{cg}\dot{Z}_{cg}) - \dot{m}Y_{cg}Z_{cg}] - m(\dot{Y}_{cg}Z_{cg} - Y_{cg}\dot{Z}_{cg}) - \dot{m}[Z_{cg}(Y_{cg} - Y_e) - Y_e(Z_{cg} - Z_e)]\}\ r \\
& -\dot{m}(X_e\dot{Z}_e - \dot{X}_eZ_e) - Z_{cg}[F_x - m\ddot{X}_{cg} - 2\dot{m}(\dot{X}_{cg} - \dot{X}_e) - \ddot{m}X_e] \\
& +X_{cg}\ [F_z - m\ddot{Z}_{cg} - 2\dot{m}(\dot{Z}_{cg} - \dot{Z}_e) - \ddot{m}Z_e]
\end{aligned}$$

ABOUT Z-Z:

$$
\begin{aligned}
N = \; & M_z - [(I_{yy} - I_{xx}) - m(X_{cg}^2 - Y_{cg}^2)]pq - [I_{xy} - mX_{cg}Z_{cg}]qr \\
& +(I_{yz} - mY_{cg}Z_{cg})qr - [I_{xy} - mX_{cg}Y_{cg}](q^2 - p^2) \\
& +\{[\dot{I}_{xz} - m(X_{cg}\dot{Z}_{cg} + \dot{X}_{cg}Z_{cg}) - \dot{m}X_{cg}Z_{cg}] - m(X_{cg}\dot{Z}_{cg} - \dot{X}_{cg}Z_{cg}) - \dot{m}[X_{cg}(Z_{cg} - Z_e) - Z_e(X_{cg} - X_e)]\}\, p \\
& +\{[\dot{I}_{yz} - m(Y_{cg}\dot{Z}_{cg} + \dot{Y}_{cg}Z_{cg}) - \dot{m}Y_{cg}Z_{cg}] - m(Y_{cg}\dot{Z}_{cg} - \dot{Y}_{cg}Z_{cg}) - \dot{m}[Y_{cg}(Zc_g - Z_e) - Z_e(Y_{cg} - Y_e)]\}\, q \\
& -\{[\dot{I}_{zz} - 2m(X_{cg}\dot{X}_{cg} + Y_{cg}\dot{Y}_{cg}) - \dot{m}(X_{cg}^2 + Y_{cg}^2)] - \dot{m}[(X_{cg} - X_e)^2 + (Y_{cg} - Y_e)^2]\}\, r \\
& -\dot{m}(Y_e\dot{X}_e - X_e\dot{Y}_e) - X_{cg}[F_y - m\ddot{Y}_{cg} - 2\dot{m}(\dot{Y}_{cg} - Y_e) - \ddot{m}Y_e] \\
& +Y_{cg}[F_x - m\ddot{X}_{cg} - 2\dot{m}(\dot{X}_{cg} - \dot{X}_e) - \ddot{m}X_e]
\end{aligned}
$$

ACCELERATIONS DUE TO CENTER OF MASS MOVEMENT (TRNACL)

X-DIRECTION FORCE:

$$
\begin{aligned}
F_{xx} = \; & F_x + mg_{x_B} - m[qw - rv + \ddot{X}_{cg} + rpZ_{cg} + qpY_{cg} - (q^2 + r^2)X_{cg} + 2q\dot{Z}_{cg} - 2r\dot{Y}_{cg}] \\
& -2\dot{m}[(\dot{X}_{cg} - \dot{X}_e) + q(Z_{cg} - Z_e) - r(Y_{cg} - Y_e)] - \ddot{m}(X_{cg} - X_e)
\end{aligned}
$$

Y-DIRECTION FORCE:

$$
\begin{aligned}
F_{yy} = \; & F_y + mg_{y_B} - m[ru - pw + \ddot{Y}_{cg} + pqX_{cg} + rqZ_{cg} - (p^2 + r^2)Y_{cg} + 2r\dot{X}_{cg} - 2p\dot{Z}_{cg}] \\
& -2\dot{m}[(\dot{Y}_{cg} - \dot{Y}_e) + r(X_{cg} - X_e) - p(Z_{cg} - Z_e)] - \ddot{m}(Y_{cg} - Y_e)
\end{aligned}
$$

Z-DIRECTION FORCE:

$$
\begin{aligned}
F_{zz} = \; & F_z + mg_{z_B} - m[pv - qu + \ddot{Z}_{cg} + qrY_{cg} + prX_{cg} - (p^2 + q^2)Z_{cg} + 2p\dot{Y}_{cg} - 2q\dot{X}_{cg}] \\
& -2\dot{m}[(\dot{Z}_{cg} - \dot{Z}_e) + p(Y_{cg} - Y_e) - q(X_{cg} - X_e)] - \ddot{m}(Z_{cg} - Z_e)
\end{aligned}
$$

$A_{xx} = F_{xx}/m$
$A_{yy} = F_{yy}/m$
$A_{zz} = F_{zz}/m$

SOLVE THE THREE SIMULTANEOUS EQUATIONS FOR P-Q-R. RATES

$A_1 = I_{xx} - m(Y_{cg}^2 + Z_{cg}^2)$
$A_2 = -I_{xy} + mX_{cg}Y_{cg}$
$A_3 = -I_{xz} + mX_{cg}Z_{cg}$
$B_1 = A_2$
$B_2 = I_{yy} - m(X_{cg}^2 + Z_{cg}^2)$
$B_3 = -I_{yz} + mY_{cg}Z_{cg}$
$C_1 = A_3$
$C_2 = B_3$
$C_3 = I_{zz} - m(X_{cg}^2 + Y_{cg}^2)$

$$E_1 = B_2C_3 - B_3C_2$$
$$E_2 = A_3C_2 - A_2C_3$$
$$E_3 = A_2B_3 - A_3B_2$$
$$E_4 = C_2N - C_3M$$
$$E_5 = B_3M - B_2N$$
$$E_6 = A_2N - A_3M$$

$$D = A_1E_1 + B_1E_2 + C_1E_3$$

$$\dot{p} = (E_1L + B_1E_4 + C_1E_5)/D$$
$$\dot{q} = (-A_1E_4 + E_2L + C_1E_6)/D$$
$$\dot{r} = (-A_1E_5 - B_1E_6 + E_3L)/D$$

BODY COMPONENTS OF INERTIAL ACCELERATION

$$\dot{U}_i = A_{xx} + Y_{cg}\,\dot{r} - Z_{cg}\,\dot{q}$$
$$\dot{V}_i = A_{yy} + Z_{cg}\,\dot{p} - X_{cg}\,\dot{r}$$
$$\dot{W}_i = A_{zz} + X_{cg}\,\dot{q} - Y_{cg}\,\dot{p}$$

QUATERNION RATES

$$\dot{\lambda}_0 = -0.5\,(p\,\lambda_1 + q\,\lambda_2 + r\,\lambda_3)$$
$$\dot{\lambda}_1 = -0.5\,(-p\,\lambda_0 - r\,\lambda_2 + q\,\lambda_3)$$
$$\dot{\lambda}_2 = -0.5\,(-q\,\lambda_0 + r\,\lambda_1 - p\,\lambda_3)$$
$$\dot{\lambda}_3 = -0.5\,(-r\,\lambda_0 - q\,\lambda_1 + p\,\lambda_2)$$

TRAJECTORY PATH LENGTH RATE

$$\dot{T}_{PATH} = \sqrt{\dot{X}_i^2 + \dot{Y}_i^2 + \dot{Z}_i^2}$$

LOGIC FLOW CHART

TRNACL:

Computes the XYZ components of acceleration acting at the body reference point due to mass movement, center-of-mass movement, and offset center-of-mass.

	Zero-out temporary storage		
NO	Is there a rate of change of the mass flow ($\dot{m}$)?		YES
		Do for each thrust motor	
		Compute and sum the incremental force components due to the rate of change of the mass flow	
NO	Is there a mass flow rate ($\dot{m}$)?		YES
		Do for each thrust motor	
		Compute and sum the incremental force components due to the mass flow rate	
	Compute the force components due to coriolis and gravity acceleration, center-of-mass position, and rate of change of position.		
	Sum the force components		
	Add the external forces		
	Compute the body acceleration components		

LOGIC FLOW CHART

TRNMOM:

Computes the resultant moments about the body reference point due to mass movement, center-of-mass movement, and offset center-of-mass.

	Zero-out temporary storage		
NO	Is there a rate of change of the mass flow ($\dot{m}$)?		YES
		Do for each thrust motor	
		Compute and sum incremental forces due to the incremental rate of change of the mass flow	
NO	Is there a mass flow rate ($\dot{m}$)?		YES
	Compute the various force components due to mass flow rate that induce moments (l) about the x-x axis		
	Compute the various force components due to mass flow rate that induce moments (m) about the y-y axis		
	Compute the various force components due to mass flow rate that induce moments (n) about the z-z axis		
	Compute moment component due to x-axis offset		
	Compute moment component due to y-axis offset		
	Compute moment component due to z-axis offset		
	Compute moment components due to offset and moving center-of-mass		
	Compute moment components due to body angular rates		
	Sum the XYZ moment components		
	Add the external moments		

PTMASS:

The main program designator, PTMASS, calls into execution two routines, PMINLZ and PMDERV, that initialize and evaluate the motion of a point source over a rotating-oblate planet. Included in the equation of motion are the effects of a thrusting and swiveling propulsion system, varying mass, wind shear and gusts, aerodynamic forces, and forces induced by a control system.

The problem position, velocity, and angular **data** on usually given to the code relative to a geodetic axis system and then transformed to a geocentric system for continued computation. Forces are summed in a velocity axis (wind axis) system and transferred to an earth-centered system. Integration of the motion derivatives takes place in an earth-fixed earth-centered axis system. Transformation in the form of direction cosines are used throughout to relate the various axis systems.

LOGIC FLOW CHART

PMINLZ:

Problem input data, referenced to a geodetic or geocentric axis system, are transformed into earth-center referenced values for insertion into the integrator state array as initial values for code start-up.

<table>
<tr><td>NO</td><td colspan="2">Is this a point mass problem start?</td><td>YES</td></tr>
<tr><td rowspan="8"></td><td colspan="2">NO Is input data GEOCENTRIC referenced?</td><td>YES</td></tr>
<tr><td rowspan="3">CALL GD2GC: get geocentric latitude (ϕ_{gc}), altitude (h_{gc}) and earth-center distance to point (R_e)</td><td colspan="2">Compute geocentric velocity components ($\dot{X}_{gc}$, $\dot{Y}_{gc}$, $\dot{Z}_{gc}$)</td></tr>
<tr><td colspan="2" rowspan="2">CALL RADIUS: get earth radius at geocentric latitude (R_{gc})</td></tr>
<tr><td>Compute geodetic velocity components ($\dot{X}_{gd}$, $\dot{Y}_{gd}$, $\dot{Z}_{gd}$)</td></tr>
<tr><td>Compute geocentric velocity components ($\dot{X}_{gc}$, $\dot{Y}_{gc}$, $\dot{Z}_{gc}$)</td><td colspan="2">Compute earth-centered distance to point (R_e)</td></tr>
<tr><td colspan="3">Compute longitude change ($\Delta\theta_L$) from start position</td></tr>
<tr><td colspan="3">Compute earth center referenced displacement (X_e, Y_e, Z_e)</td></tr>
<tr><td colspan="3">Compute geocentric to earth-center transformation (TGC2E)</td></tr>
<tr><td colspan="3">Compute earth-center referenced velocities ($\dot{X}_e$, $\dot{Y}_e$, $\dot{Z}_e$)</td></tr>
</table>

EQUATIONS EVALUATED

PMINLZ:

GEODETIC VELOCITY COMPONENTS

$$\dot{X}_{gd} = V_{T_{gd}} \cos \gamma_{gd} \cos \sigma_{gd}$$
$$\dot{Y}_{gd} = V_{T_{gd}} \cos \gamma_{gd} \sin \sigma_{gd}$$
$$\dot{Z}_{gd} = -V_{T_{gd}} \sin \gamma_{gd}$$

GEOCENTRIC VELOCITY COMPONENTS

GEODETIC START

$$\Delta\phi = \phi_{gd} - \phi_{gc}$$
$$\dot{X}_{gc} = \dot{X}_{gd} \cos \Delta\phi - \dot{Z}_{gd} \sin \Delta\phi$$
$$\dot{Y}_{gc} = \dot{Y}_{gd}$$
$$\dot{Z}_{gc} = \dot{X}_{gd} \sin \Delta\phi + \dot{Z}_{gd} \cos \Delta\phi$$

GEOCENTRIC START

$$\dot{X}_{gc} = V_{T_{gc}} \cos \gamma_{gc} \cos \sigma_{gc}$$
$$\dot{Y}_{gc} = V_{T_{gc}} \cos \gamma_{gc} \sin \sigma_{gc}$$
$$\dot{Z}_{gc} = -V_{T_{gc}} \sin \gamma_{gc}$$

EARTH CENTERED DISTANCE TO POINT

$$R_e = R_{gc} + h_{gc}$$

EARTH CENTER REFERENCED DISPLACEMENTS

$$\Delta\theta_L = \theta_{L_o} - \theta_L$$
$$X_e = R_e \cos \phi_{gc} \cos \Delta\theta_L$$
$$Y_e = R_e \cos \phi_{gc} \sin \Delta\theta_L$$
$$Z_e = -R_e \sin \phi_{gc}$$

TRANSFORMATION: GEOCENTRIC TO EARTH CENTERED

$$
\begin{aligned}
\text{TGC2E } (1,1) &= -\sin \phi_{gc} \cos \Delta\theta_L \\
(1,2) &= \sin \Delta\theta_L \\
(1,3) &= -\cos \phi_{gc} \cos \Delta\theta_L \\
[T_{E/GC}] \ (2,1) &= -\sin \phi_{gc} \sin \Delta\theta_L \\
(2,2) &= -\cos \Delta\theta_L \\
(2,3) &= -\cos \phi_{gc} \sin \Delta\theta_L \\
(3,1) &= -\cos \phi_{gc} \\
(3,2) &= 0.0 \\
\text{TGC2E } (3,3) &= \sin \phi_{gc}
\end{aligned}
$$

$\Delta\theta_L = \theta_{L_0} - \theta_L$

θ_{L_0} : initial longitude

θ_L : current longitude

ϕ_{gc} : geocentric latitude.

EARTH CENTER REFERENCED VELOCITY COMPONENTS

$$
\begin{vmatrix} \dot{X}_e \\ \dot{Y}_e \\ \dot{Z}_e \end{vmatrix} = \begin{vmatrix} T_{E/GC} \end{vmatrix} \cdot \begin{vmatrix} \dot{X}_{gc} \\ \dot{Y}_{gc} \\ \dot{Z}_{gc} \end{vmatrix}
$$

LOGIC FLOW CHART

PMDERV:

The earth-center referenced accelerations of the point mass are computed from the current state variable values for insertion into the integrator rate array to advance the problem solution one time step.

	Compute earth-centered distance to point (R_e)	
	Compute geocentric latitude (ϕ_{gc})	
	Compute longitude change ($\Delta\theta_L$) and current longitude (θ_L)	
	CALL RADIUS: get earth radius at geocentric latitude	
	CALL GC2GD: get geodetic latitude (ϕ_{gd}) and altitude (h_{gd})	
	CALL ATMS: get atmospheric properties (P, T, ρ, V_s, ν) at altitude	
NO	Are gravity components required?	YES
	CALL GRAV: get X and Z gravity components ($g_{x_{gc}}$, $g_{z_{gc}}$)	
NO	Are wind effects required?	YES
	CALL WINDS: get geodetic wind components ($\dot{X}_{w_{gd}}$, $\dot{Y}_{w_{gd}}$, $\dot{Z}_{w_{gd}}$)	
	Compute geocentric to earth-center transformation (TGC2E)	
	Compute earth-center referenced gravity components (g_{x_e}, g_{y_e}, g_{z_e})	
	Compute geocentric velocity components ($\dot{X}_{gc}$, $\dot{Y}_{gc}$, $\dot{Z}_{gc}$)	
	Compute geocentric to geodetic transformation (TGC2GD)	
	Compute geodetic velocity components ($\dot{X}_{gd}$, $\dot{Y}_{gd}$, $\dot{Z}_{gd}$)	
	Compute point velocity components in geodetic system (U_{gd}, V_{gd}, W_{gd})	
	Compute velocity in air (V_a), aero flight path angles (γ_a, σ_a)	
	Compute Mach number (MN), Reynolds number (R_N), dynamic pressure (q)	
NO	Has an aerodynamic roll angle table been input?	YES
	CALL ROLANG: get aerodynamic roll angle (ϕ_a)	
	Compute velocity frame to geodetic transformation (TVF2GD)	

PMDERV (cont):

NO	Is thrust routine activated?	YES
	CALL THRUST: get thrust forces	
NO	Is point mass control activated?	YES
	CALL PTMSCO: get aero angles (α,β) or force coefficients (C_D, C_L, C_Y)	
	NO Are control forces to be computed from coefficient?	YES
	Compute body to velocity frame transformation (TB2VF)	Compute velocity frames forces ($F_{A_{vf_x}}$, $F_{A_{vf_y}}$, $F_{A_{vf_z}}$)
	Compute velocity frame referenced thurst force (T_{VF_x}, T_{VF_y}, T_{VF_z})	
NO	Are aerodynamic forces requested? YES	
	CALL AERO: get aerodynamic forces in velocity frame ($F_{A_{vf_x}}$, $F_{A_{vf_y}}$, ($F_{A_{vf_z}}$)	
	Sum the forces in the velocity frame ($F_{T_{vf_x}}$, $F_{T_{vf_y}}$, $F_{T_{vf_z}}$)	
	Compute velocity frame to earth-centered transformation (TVF2E)	
	Compute earth-center referenced forces (F_{x_e}, F_{y_e}, F_{z_e})	
	Compute earth center referenced accelerations ($\ddot{X}_e$, $\ddot{Y}_e$, $\ddot{Z}_E$)	

EQUATIONS EVALUATED

PMDERV:

EARTH CENTERED DISTANCE TO POINT:

$$R_e = \sqrt{X_e^2 + Y_e^2 + Z_e^2}$$

GEOCENTRIC LATITUDE:

$$\phi_{gc} = \sin^{-1}(-Z_e/R_e)$$

LONGITUDE CHANGE AND CURRENT LONGITUDE:

$$\Delta\theta_L = \tan^{-1}(Y_e/X_e)$$
$$\theta_L = \theta_{L_o} - \Delta\theta_L$$

TRANSFORMATION: GEOCENTRIC TO EARTH CENTERED:

$$\begin{aligned}
\text{TGC2E } (1,1) &= -\sin\phi_{gc}\cos\Delta\theta_L \\
(2,1) &= -\sin\phi_{gc}\sin\Delta\theta_L \\
(3,1) &= -\cos\phi_{gc} \\
(1,2) &= \sin\Delta\theta_L \\
[T_{E/GC}]\ (2,2) &= -\cos\Delta\theta_L \\
(3,2) &= 0.0 \\
(1,3) &= -\cos\phi_{gc}\cos\Delta\theta_L \\
(2,3) &= -\cos\phi_{gc}\sin\Delta\theta_L \\
\text{TGC2E } (3,3) &= \sin\phi_{gc}
\end{aligned}$$

EARTH CENTERED GRAVITY COMPONENTS:

$$\begin{vmatrix} g_{x_e} \\ g_{y_e} \\ g_{z_e} \end{vmatrix} = \begin{vmatrix} T_{E/GC} \end{vmatrix} \cdot \begin{vmatrix} g_{x_{gc}} \\ g_{y_{gc}} \\ g_{z_{gc}} \end{vmatrix}$$

GEOCENTRIC VELOCITY COMPONENTS:

$$\begin{vmatrix} \dot{X}_{gc} \\ \dot{Y}_{gc} \\ \dot{Z}_{gc} \end{vmatrix} = \begin{vmatrix} T_{E/GC} \end{vmatrix}' \cdot \begin{vmatrix} \dot{X}_e \\ \dot{Y}_e \\ \dot{Z}_e \end{vmatrix}$$

TRANSFORMATION: GEOCENTRIC TO GEODETIC

$$\Delta\phi = \phi_{gd} - \phi_{gc}$$

$$\begin{aligned}
\mathrm{TGC2GD}(1,1) &= \cos\Delta\phi \\
(2,1) &= 0.0 \\
(3,1) &= -\sin\Delta\phi \\
(1,2) &= 0.0 \\
[T_{GD/GC}]\ (2,2) &= 1.0 \\
(3,2) &= 0.0 \\
(1,3) &= \sin\Delta\phi \\
(2,3) &= 0.0 \\
\mathrm{TGC2GD}(3,3) &= \cos\Delta\phi
\end{aligned}$$

GEODETIC VELOCITY COMPONENTS:

$$\begin{vmatrix} \dot{X}_{gd} \\ \dot{Y}_{gd} \\ \dot{Z}_{gd} \end{vmatrix} = \begin{vmatrix} T_{GD/GC} \end{vmatrix} \cdot \begin{vmatrix} \dot{X}_{gc} \\ \dot{Y}_{gc} \\ \dot{Z}_{gc} \end{vmatrix}$$

POINT VELOCITY COMPONENTS - GEODETIC REFERENCE:

$$\begin{aligned}
U_{gd} &= \dot{X}_{gd} - \dot{X}_{w_{gd}} \\
V_{gd} &= \dot{Y}_{gd} - \dot{Y}_{w_{gd}} \\
W_{gd} &= \dot{Z}_{gd} - \dot{Z}_{w_{gd}}
\end{aligned}$$

VELOCITY IN AIR:

$$V_a = \sqrt{U_{gd}^2 + V_{gd}^2 + W_{gd}^2}$$

MACH NUMBER:

$$M_N = V_a/V_s$$

REYNOLDS NUMBER:

$$R_N = V_a/\nu$$

DYNAMIC PRESSURE:

$$q = \tfrac{1}{2}\,\rho\,V_a^2$$

FLIGHT PATH ANGLES: ELEVATION AND HORIZONTAL

$$\gamma_a = \sin^{-1}(-W_{gd}/V_a)$$
$$\sigma_a = \tan^{-1}(V_{gd}/U_{gd})$$

TRANSFORMATION: VELOCITY FRAME (WIND) TO GEODETIC

$$[T_{GD/VF}] \quad \begin{array}{ll} \text{TVF2GD}(1,1) & = \cos\gamma_a \cos\sigma_a \\ (2,1) & = \cos\gamma_a \sin\sigma_a \\ (3,1) & = -\sin\gamma_a \\ (1,2) & = \sin\gamma_a \cos\sigma_a \sin\phi_a - \sin\sigma_a \cos\phi_a \\ (2,2) & = \sin\gamma_a \sin\sigma_a \sin\phi_a + \cos\sigma_a \cos\phi_a \\ (3,2) & = \cos\gamma_a \sin\phi_a \\ (1,3) & = \sin\gamma_a \cos\sigma_a \cos\phi_a + \sin\sigma_a \sin\phi_a \\ (2,3) & = \sin\gamma_a \sin\sigma_a \cos\phi_a - \cos\sigma_a \cos\phi_a \\ \text{TVF2GD}(3,3) & = \cos\gamma_a \cos\phi_a \end{array}$$

VELOCITY FRAME AERO FORCES (POINT MASS CONTROL):

$$F_{A_{v_{f_x}}} = -C_D\, q\, S$$
$$F_{A_{v_{f_y}}} = C_y\, q\, S$$
$$F_{A_{v_{f_z}}} = -C_L\, q\, S$$

TRANSFORMATION: BODY TO VELOCITY FRAME (WIND):

$$[T_{VF/B}] \quad \begin{array}{ll} \text{TB2VF}(1,1) & = \cos\alpha \cos\beta \\ (2,1) & = -\cos\alpha \sin\beta \\ (3,1) & = -\sin\alpha \\ (1,2) & = \sin\beta \\ (2,2) & = \cos\beta \\ (3,2) & = 0.0 \\ (1,3) & = \sin\alpha \cos\beta \\ (2,3) & = -\sin\alpha \sin\beta \\ \text{TB2VF}(3,3) & = \cos\alpha \end{array}$$

VELOCITY FRAME THRUST COMPONENTS:

$$\begin{vmatrix} F_{T_{v_{f_x}}} \\ F_{T_{v_{f_y}}} \\ F_{T_{v_{f_z}}} \end{vmatrix} = \begin{vmatrix} T_{VF/B} \end{vmatrix} \cdot \begin{vmatrix} F_{T_{B_x}} \\ F_{T_{B_y}} \\ F_{T_{B_z}} \end{vmatrix}$$

SUMMATION OF FORCES:

$$F_{V_{f_x}} = F_{T_{vf_x}} + F_{A_{vf_x}}$$
$$F_{V_{f_y}} = F_{T_{vf_y}} + F_{A_{vf_y}}$$
$$F_{V_{f_z}} = F_{T_{vf_z}} + F_{A_{vf_z}}$$

TRANSFORMATION: VELOCITY FRAME TO EARTH CENTERED

$$\left|T_{E/VF}\right| = \left|T_{E/GC}\right| \cdot \left|T_{GD/GC}\right| \cdot \left|T_{GD/VF}\right|$$

EARTH-CENTERED FORCES:

$$\begin{vmatrix} F_{x_e} \\ F_{y_e} \\ F_{z_e} \end{vmatrix} = \left|T_{E/VF}\right| \cdot \begin{vmatrix} F_{V_{F_x}} \\ F_{V_{F_y}} \\ F_{V_{F_z}} \end{vmatrix}$$

EARTH CENTER REFERENCED ACCELERATIONS:

$$\ddot{X}_e = \frac{F_{x_e}}{m} + g_{x_e} - 2.0\,\omega_p\,\dot{Y}_e + \omega_p^2\,X_e$$
$$\ddot{Y}_e = \frac{F_{y_e}}{m} + g_{y_e} - 2.0\,\omega_p\,\dot{X}_e + \omega_p^2\,Y_e$$
$$\ddot{Z}_e = \frac{F_{z_e}}{m} + g_{z_e}$$

AERODYNAMICS:

The aerodynamic force and moment contribution to the main program (six-degree-of-freedom or point mass) force and moment equations are provided by the AERO routine. The total coefficients from which the aerodynamic forces and moments are computed result from a summation of incremental coefficients. The user supplies either a constant value for the incremental coefficient are a table of incremental coefficient values. These values are referenced to the body reference point (BRP) which is specified under the SIXDOF set of inputs.

Body or wind axis referenced coefficients are acceptable with body-axis coefficients preferred for the six-degree-of-freedom evaluation, and wind-axis coefficients preferred for the point mass evaluation. However, body-axis referenced force coefficients can be transformed to the wind-axis before the aerodynamic forces are computed for return to the main program.

During initialization, the routine AROINL determines which incremental coefficient tables are active (input) and passes this information - in flag form - to the AERO routine for faster execution.

LOGIC FLOW CHART

AROINL:

Determine the status of each aerodynamic coefficient table, and sets the table status flag if table has been input.

	Do for I = 1,7
	CALL ITABLE: set incremental C_x coefficient table flag if table C_{x_i} has been input
	Do for I = 1,8
	CALL ITABLE: set incremental C_y coefficient table flag if table C_{y_i} has been input
	Do for I = 1,8
	CALL ITABLE: set incremental C_z coefficient table flag if table C_{z_i} has been input
	Do for I = 1,7
	CALL ITABLE: set incremental C_l coefficient table flag if table C_{l_i} has been input
	Do for I = 1,8
	CALL ITABLE: set incremental C_m coefficient table flag if table C_{m_i} has been input
	Do for I = 1,8
	CALL ITABLE: set incremental C_n coefficient table flag if table C_{n_i} has been input

LOGIC FLOW CHART

AERO:

Evaluates the aerodynamic forces and moments for the main program by summation of incremental values.

NO	Is speed of sound (V_s) zero?			YES
		Do for I = 1,7		Set total force C_{X_T}, C_{Y_T}, C_{Z_T} coefficients to zero.
	NO	Is incremented C_{x_i} table flag set?	YES	
		CALL LOOKUP: get incremental C_{x_i} value		
		Do for I = 1,8		
	NO	Is incremented C_{y_i} table flag set?	YES	
		CALL LOOKUP: get incremental C_{y_i} value		
		Do for I = 1,8		
	NO	Is incremented C_{z_i} table flag set?	YES	
		CALL LOOKUP: get incremental C_{z_i} value		
NO	Is the main program point mass?			YES
		Do for I = 1,7		
	NO	Is incremented C_{l_i} table flag set?	YES	
		CALL LOOKUP: get incremental C_{l_i} value		
		Do for I = 1,8		
	NO	Is incremented C_{m_i} table flag set?	YES	
		CALL LOOKUP: get incremental C_{m_i} value		
		Do for I = 1,8		
	NO	Is incremented C_{n_i} table flag set?	YES	
		CALL LOOKUP: get incremental C_{n_i} value		
Compute total force coefficients: C_{X_T}, C_{Y_T}, C_{Z_T}				Set total moment C_{l_T}, C_{m_T}, C_{n_T} coefficients to zero.
NO	Is main program point mass?		YES	
Compute total moment coefficients: C_{l_T}, C_{m_T}, C_{n_T}				
NO	Are coefficients in WIND axis?			YES
	Convert force coefficients to wind axis			
	Compute aerodynamics forces (F_{X_A}, F_{Y_A}, F_{Z_A})			
NO	Is main program point mass?			YES
	Compute aerodynamics moments (M_{X_A}, M_{Y_A}, M_{Z_A})			

EQUATIONS EVALUATED

AERO:

$$C_{X_T} = C_{X_O} + C_{X_\alpha} |\alpha| + C_{X_{\alpha^2}} \alpha^2 + C_{X_\beta} |\beta| + C_{X_{\beta^2}} \beta^2 + C_{X_{\delta_q}} |\delta_q| + C_{X_{\delta_r}} |\delta_r|$$

$$C_{Y_T} = C_{Y_O} + C_{Y_\beta}\beta + C_{Y_{\beta^2}} \beta |\beta| + C_{Y_{\beta^3}} \beta^3 + C_{Y_{\delta_r}} \delta_r + C_{Y_p} \left(\frac{p\ell_{R_Y}}{2Va}\right) + C_{Y_r} \left(\frac{r\ell_{R_Y}}{2Va}\right) + C_{Y_{\dot\beta}} \left(\frac{\dot\beta\ell_{R_Y}}{2Va}\right)$$

$$C_{Z_T} = C_{Z_O} + C_{Z_\alpha}\alpha + C_{Z_{\alpha^2}} \alpha |\alpha| + C_{Z_{\alpha^3}} \alpha^3 + C_{Z_{\delta_q}} \delta_q + C_{Z_p} \left(\frac{p\ell_{R_Z}}{2Va}\right) + C_{Z_q} \left(\frac{q\ell_{R_X}}{2Va}\right) + C_{Z_{\dot\alpha}} \left(\frac{\dot\alpha\ell_{R_X}}{2Va}\right)$$

$$C_{\ell_T} = C_{\ell_o} + C_{\ell_\alpha}\alpha + C_{\ell_\beta}\beta + C_{\ell_{\delta_p}} \delta_p + C_{\ell_p} \left(\frac{p\ell_{R_Y}}{2Va}\right) + C_{\ell_r} \left(\frac{r\ell_{R_Y}}{2Va}\right) + C_{\ell_{\dot\beta}} \left(\frac{\dot\beta\ell_{R_Y}}{2Va}\right)$$

$$C_{m_T} = C_{m_o} + C_{m_\alpha}\alpha + C_{m_{\alpha^2}} \alpha |\alpha| + C_{m_{\alpha^3}} \alpha^3 + C_{m_{\delta_q}} \delta_q + C_{m_p} \left(\frac{p\ell_{R_Y}}{2Va}\right) + C_{m_q} \left(\frac{q\ell_{R_X}}{2Va}\right) + C_{m_{\dot\alpha}} \left(\frac{\dot\alpha\ell_{R_X}}{2Va}\right)$$

$$C_{n_T} = C_{n_o} + C_{n_\beta}\beta + C_{n_{\beta^2}} \beta |\beta| + C_{n_{\beta^3}} \beta^3 + C_{n_{\delta_r}} \delta_r + C_{n_p} \left(\frac{p\ell_{R_Y}}{2Va}\right) + C_{n_r} \left(\frac{r\ell_{R_Y}}{2Va}\right) + C_{n_{\dot\beta}} \left(\frac{\dot\beta\ell_{R_Y}}{2Va}\right)$$

COEFFICIENT CONVERSION: BODY TO WIND (VELOCITY FRAME)

$$\begin{vmatrix} C_{X_T} \\ C_{Y_T} \\ C_{Z_T} \end{vmatrix}_{VF} = \begin{vmatrix} T_{VF/B} \end{vmatrix} \cdot \begin{vmatrix} C_{X_T} \\ C_{Y_T} \\ C_{Z_T} \end{vmatrix}_{B}$$

AERODYNAMIC FORCES AND MOMENTS

$$F_{X_A} = C_{X_T} qS \qquad M_{X_A} = C_{\ell_T} qS\ \ell_{R_X}$$

$$F_{Y_A} = C_{Y_T} qS \qquad M_{Y_A} = C_{m_T} qS\ \ell_{R_Y}$$

$$F_{Z_A} = C_{Z_T} qS \qquad M_{Z_A} = C_{n_T} qS\ \ell_{R_Y}$$

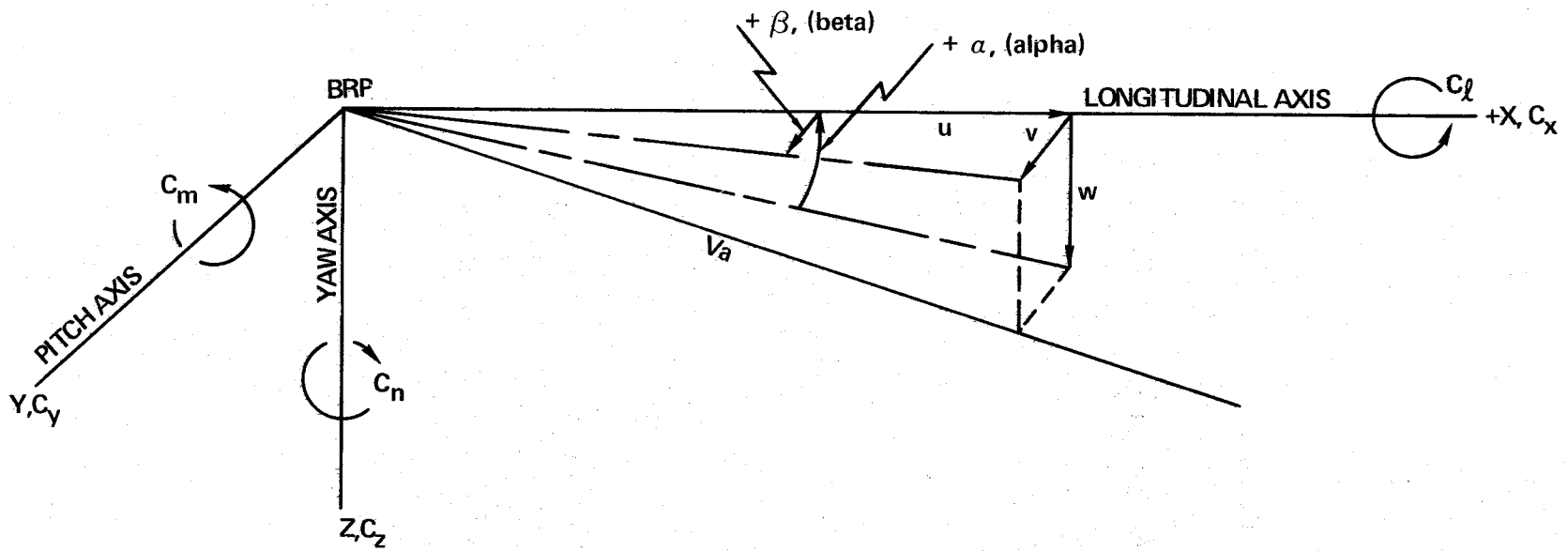


Figure B-1. Aerodynamic Parameters and Axis System

ATMOSPHERE:

Atmospheric properties, at altitude, are requested by the main program (SIXDOF and PTMASS) during code execution. The properties are determined by the ATMS routine from general arrays of atmospheric data. These data arrays are filled during initialization by the ATMSET routine.

ATMSET, acting upon a user request, selects the appropriate sets of predefined altitude, temperature, and molecular weights of air arrays and stores them into the corresponding general arrays. Atmospheric pressure values, at the base altitude, are computed by an exponential function and stored in a general array. Similarly, the temperature gradients and the molecular weights of air gradients are computed and stored.

LOGIC FLOW CHART

ATMSET:

Selects the user-requested set of atmosphere properties, computes the pressures and the property gradients, and stores all values in general arrays as a function of altitude layers.

		Do for I = 1,22	
	NO	Is the user requested name (IATMNM) in the available data list, ITMNMS?	YES
			Set indicator (INDATM) to I-1
	Set J to INDATM + 1		
	Store Hollerith name of requested atmosphere		
NO	Does indicator (INDATM) equal 1 to 19?		YES
	Store the maximum altitude (ZMAXM) for the user set		
	Compute gravity constant for data set latitude		
		Do for all altitude layers	
		Fill the general arrays with the selected altitude and temperature	
		Compute temperature - altitude gradient between altitude layers	
		Compute the ambient pressure at each altitude layer	
		Compute molecular weight of air-altitude gradient between layers	

EQUATIONS EVALUATED

ATMSET:

GRAVITY AT THE ATMOSPHERE SET LATITUDE:

$$g_{\phi_a} = 9.806160\ (1.0 - 0.0026373 \cos 2\phi_c + 0.0000059 \cos^2 2\phi_c)$$

EARTH RADIUS AT THE ATMOSPHERE SET LATITUDE:

$$R_{\phi_c} = 2g_{\phi_c}/(3.085462E{-}6 + 2.27E - 9 \cos 2\phi_c - 2.0E{-}12 \cos 4\phi_c)$$

MOLECULAR SCALE TEMPERATURE - ALTITUDE GRADIENT (BETWEEN LAYERS):

$$k_{1_i} = (T_{B_{i+1}} - T_{B_i})/(h_{B_{i+1}} - h_{B_i})$$

PRESSURE AT THE ATMOSPHERE SET ALTITUDES:

$$P_{B_{i=1}} = \text{S.L. pressure}$$

$$P_{B_{i+1}} = P_{B_i} \cdot \left(\frac{-1.179489E{-}3\ M_o\ (h_{B_{i+1}} - h_{B_i})}{T_{B_i}} \right) \quad \text{for } K_{1_i} = 0.0$$

$$P_{B_{i+1}} = P_{B_i}\ (T_{B_i}/(T_{B_{i+1}})^{(1.179489E{-}3\ Mo/K_{1_i})}) \text{ for } K_{1_i} \neq 0.0$$

MOLECULAR WEIGHT OF AIR - ALTITUDE GRADIENT (BETWEEN SET ALTITUDES):

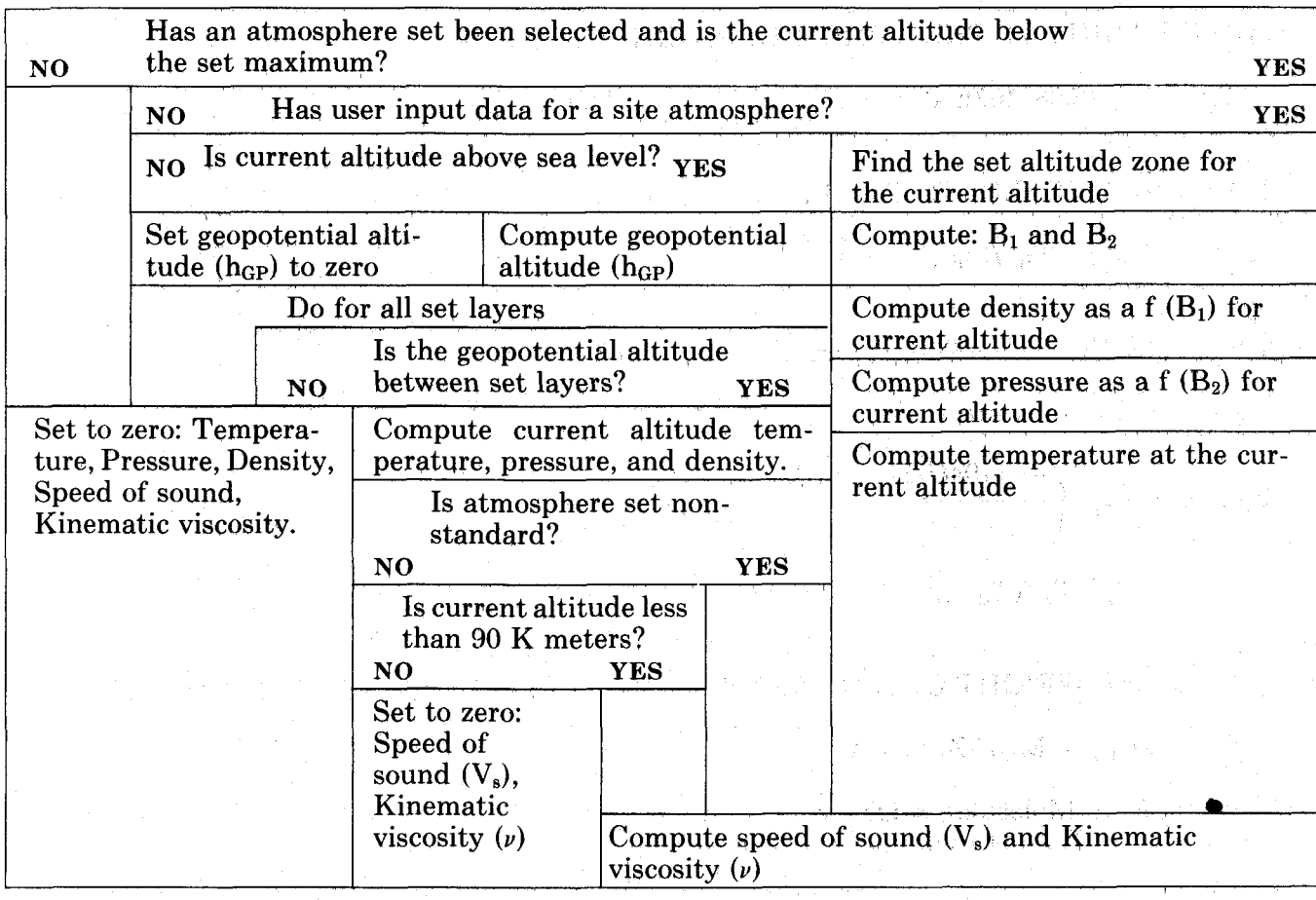
$$K_{2_i} = (M_{B_{i+i}} - M_{B_i})/(Z_{B_{i+1}} - Z_{B_i})$$

where: M = Molecular weight of air
B = Base of the atmospheric layer

LOGIC FLOW CHART

ATMS:

Computes the atmospheric properties: temperature, pressure, density, speed-of-sound, and kinematic viscosity at the current altitude, zero values are given if the altitude is above the tables with sea-level values returned for an altitude below sea level.



EQUATIONS EVALUATED

ATMS:

GEOPOTENTIAL ALTITUDE

$$h_{gp} = \left(\frac{g_{\phi_a}}{9.80665}\left(\frac{h_{gd}}{1 + h_{gd/R\phi_a}}\right)\right)$$

TEMPERATURE AT CURRENT ALTITUDE

$$M = M_{B_{i=1}} + K_{2_{i-1}}\,(h_{gp} - Z_{B_{i-1}})$$

$$T_{gp} = T_{B_{i-1}} + K_{1_{i-1}}\,(h_{gp} - h_{B_{i-1}})$$

$$T_a = \frac{M}{M_o}\,(T_{gp})$$

PRESSURE AT CURRENT ALTITUDE

$$P_a = P_{B_{i-1}} \cdot e^{\left(\frac{-1.179489E-3\ Mo\ (h_{gp} - h_{B_{i-1}})}{T_{B_{i-1}}}\right)} \text{ for } K_{1_{i-1}} = 0.0$$

$$P_a = P_{B_{i-1}}\left(\frac{T_{B_{i-1}}}{T_{gp}}\right)^{(1.179489\ M_o/K_{1_{i-1}})} \text{ for } K_{1_{i-1}} \neq 0.0$$

DENSITY AT CURRENT ALTITUDE

$$\rho_a = (M_o/P_a)/(8.31432E+3\ T_{gp})$$

SPEED OF SOUND

$$V_s = \sqrt{1.164005E+4\left(\frac{T_{gp}}{M_o}\right)}$$

KINEMATIC VISCOSITY

$$\nu = (1.458E-6\ T_{gp}^{1.5})/(\rho_a\,(T_{gp} + 110.4))$$

SITE EVALUATIONS

$$B_1 = \ln\left(\frac{\rho_{M_{i+1}}}{\rho_{M_i}}\right) / (Z_{B_{i+1}} - Z_{B_i})$$

$$B_2 = \ln\left(\frac{\rho_{M_{i+1}}}{\rho_M}\right) / \left(\frac{P_{B_{i+1}}}{P_{B_i}}\right)$$

DENSITY AT CURRENT SITE ALTITUDE

$$\rho_a = \rho_{M_{i-1}}\, e^{(B_1 (h_{gd} - Z_{B_{i-1}})}$$

PRESSURE AT CURRENT SITE ALTITUDE

$$P_a = P_{B_{i-1}}\, e^{(B_2 (h_{gd} - Z_{B_{i-1}})}$$

TEMPERATURE AT CURRENT SITE ALTITUDE

$$T_a = \left(\frac{28.9644}{8314.32}\right)\left(\frac{P_a}{\rho_a}\right)$$

SPEED OF SOUND AT CURRENT SITE ALTITUDE

$$V_s = \sqrt{1.164005E{+}4\ T_a/M_o}$$

KINEMATIC VISCOSITY

$$\nu = (1.458E{-}6\ T_a^{1.5})/(\rho_a\ (T_a + 110.4))$$

AUXILIARY CALCULATIONS

In keeping with the code philosophly in which only those calculations that have a direct bearing on the derivative evaluations are included in the main program calculations, a catch-all routine, AUXCLC, is provided and is called into execution immediately following a good integration time step. These computed values are available to the stage testing procedure for code control.

LOGIC FLOW CHART

AUXCLC:

Evaluates miscellaneous program variables for print internal output and stage testing.

	Compute geodetic ground referenced total velocity ($V_{T_{gd}}$)	
NO	Is total velocity zero?	YES
	Compute geodetic flight path angles (γ_{gd}, σ_{gd})	
	Compute geocentric ground referenced total velocity ($V_{T_{gc}}$)	
	Compute geocentric flight-path angles (γ_{gc}, σ_{gc})	
	CALL TQ2A: get inertial and geodetic referenced Euler angles (θ_i, ψ_i, ϕ_i, θ_{gd}, ψ_{gd}, ϕ_{gd})	
NO	Are Yawsonde sun angles calculations required?	YES
	Compute Yawsonde sun angle (σ_{SUN})	
NO	Are range calculations required?	YES
	Compute great circle range, down range, cross range (GC_R, X_R, Y_R)	
	Compute approximate surface path length (S_{PATH})	
NO	Are body axis g-load factors required?	YES
	Compute X, Y, Z body g-loads (X_{LOD}, Y_{LOD}, Z_{LOD})	
NO	Are critical frequency calculations required?	YES
	Compute damped natural pitch frequency (ω_p)	
NO	Are station/radar calculations required?	YES
	CALL RADAR: get radar information: Azimuth, elevation, and slant range	

EQUATIONS EVALUATED

AUXCLC:

GEODETIC | GEOCENTRIC

GROUND REFERENCED TOTAL VELOCITY

$$V_{T_{gd}} = \sqrt{\dot{X}_{gd}^2 + \dot{Y}_{gd}^2 + \dot{Z}_{gd}^2} \qquad V_{T_{gc}} = \sqrt{\dot{X}_{gc}^2 + \dot{Y}_{gc}^2 + \dot{Z}_{gc}^2}$$

FLIGHT PATH ANGLES

ELEVATION: $\gamma_{gd} = \sin^{-1}(-\dot{Z}_{gd}/V_{T_{gd}})$ $\qquad \gamma_{gc} = \sin^{-1}(-\dot{Z}_{gc}/V_{T_{gc}})$

AZIMUTH: $\sigma_{gd} = \tan^{-1}(\dot{Y}_{gd}/\dot{X}_{gd})$ $\qquad \sigma_{gc} = \tan^{-1}(\dot{Y}_{gc}/\dot{X}_{gc})$

YAWSONDE SUN ANGLE

$$TMP = (\cos\theta_B \cos\psi_B)(\cos\psi_{SUN}\cos\theta_{SUN}) + (\cos\theta_B \sin\psi_B)(\sin\psi_{SUN}\cos\theta_{SUN}) + (-\sin\theta_B)(-\sin\theta_{SUN})$$

$$\sigma_{SUN} = 90.0 - \cos^{-1}(TMP)$$

GREAT CIRCLE RANGE

$$R_{AVG} = \tfrac{1}{2}(R_{gc} + R_{\phi_{gc_o}})$$

$$\sphericalangle gc = \cos^{-1}(\sin\phi_{gc}\sin\phi_{gc_o} + \cos\phi_{gc}\cos\phi_{gc_o}\cos(\theta_L - \theta_{L_O}))$$

$$GC_R = R_{AVG}\cos\sphericalangle gc$$

DOWN RANGE:

$$\xi = \tan^{-1}\left(\frac{\sin(\theta_L - \theta_{L_O})\cos\phi_{gc}\cos\phi_{gc_o}}{\sin\phi_{gc} - \sin\phi_{gc_o}\cos\sphericalangle_{gc}}\right)$$

$$X_R = R_{AVG}\cos^{-1}(\cos\sphericalangle_{gc}/\cos(\sin^{-1}(\sin\sphericalangle_{gc}\sin\xi)))$$

CROSS RANGE

$$Y_R = R_{AVG}(\sin^{-1}(\sin\sphericalangle_{gc}\sin\xi))$$

APPROXIMATE SURFACE PATH LENGTH

$$S_{PATH} = S_{PATH_1} + (R_{gc}/R)\,V_{T_{gd}}\cos\gamma_{gd}\,\Delta t$$

BODY AXIS LOAD FACTORS

$$X_{LOD} = (Fx_{AERO} + Fx_{THRS}) / mg_{REF}$$
$$Y_{LOD} = (Fy_{AERO} + Fy_{THRS}) / mg_{REF}$$
$$Z_{LOD} = (Fz_{AERO} + Fz_{THRS}) / mg_{REF}$$

DAMPED NATURAL PITCH FREQUENCY

$$X_{cp} = (C_{m_\alpha}/-C_{z_\alpha})\ \ell_{r_x}$$
$$C_{m_{\alpha cg}} = C_{m_\alpha}\ ((X_{cp} - X_{cg})/X_{cp})$$
$$M_{\alpha cg} = C_{m_{\alpha cg}}\ q\ S\ \ell_{r_x}$$
$$X_{cq} = (Cm_q/-Cz_q)\ \ell_{r_x}$$
$$C_{m_{qcg}} = Cm_q\ ((X_{cq} - X_{cg})/X_{cq})$$
$$M_{qcg} = \left((C_{M_{q_{cg}}}\ q\ S\ \ell_{r_x^2})/2Va\right)$$
$$\omega_p = \sqrt{-(M_{\alpha_{cg}}/I_{yy}) - (M_{q_{cg}}/2I_{yy})}$$

LOGIC FLOW CHART

TQ2A:

Converts the quaternion information in the TI2B array (SDDERV) into inertial Euler angles, and through a sequence of transformation into geodetic Euler angles. The rotation sequence of conversion is user-specified.

	Do for 4 items			
	NO	Is requested rotation sequence in DATA list?		YES
		Set: ISEQ		
CALL ABORT; Write error message and return		Compute geodetic to geocentric transformation ($T_{gc/gd}$)		
		Compute geodetic to body transformation ($T_{B/gd}$)		
		ISEQ =		
		1	2	3	4
		YPR SEQUENCE	PRY SEQUENCE	PYR SEQUENCE	RPR SEQUENCE
		Compute inertial Euler angles (θ_i, ψ_i, ϕ_i)		
		Compute geodetic Euler angles (θ_{gd}, ψ_{gd}, ϕ_{gd})		

EQUATIONS EVALUATED

TQ2A:

GEODETIC TO GEOCENTRIC TRANSFORMATION

$$
[T_{gc/gd}] \quad
\begin{aligned}
TGD2GC(1,1) &= \cos \Delta\phi \qquad \Delta\phi = \phi_{gd} - \phi_{gc} \\
(2,1) &= 0.0 \\
(3,1) &= \sin \Delta\phi \\
(1,2) &= 0.0 \\
(2,2) &= 1.0 \\
(3,2) &= 0.0 \\
(1,3) &= -\sin \Delta\phi \\
(2,3) &= 0.0 \\
TGD2GC(3,3) &= \cos \Delta\phi
\end{aligned}
$$

GEODETIC TO BODY TRANSFORMATION (TGD2B)

$$
\left| T_{B/gd} \right| = \left| T_{B/I} \right| \cdot \left| T_{gc/I} \right| \cdot \left| T_{gc/gd} \right|
$$

EULER ANGLES

INERTIAL	GEODETIC
YAW-PITCH-ROLL SEQUENCE	
$\psi_i = \tan^{-1}$ (TI2B (1,2)/TI2B (1,1))	$\psi_{gd} = \tan^{-1}$ (TGD2B (1,2)/TGD2B (1,1))
$\theta_i = \sin^{-1}$ (–TI2B (1,3))	$\theta_{gd} = \sin^{-1}$ (–TGD2B (1,3))
$\phi_i = \tan^{-1}$ (TI2B (2,3)/TI2B (3,3))	$\phi_{gd} = \tan^{-1}$ (TGD2B (2,3)/TGD2B (3,3))
PITCH-ROLL-YAW SEQUENCE	
$\theta_i = \tan^{-1}$ (TI2B (3,1)/TI2B (3,3))	$\theta_{gd} = \tan^{-1}$ (TGD2B (3,1)/TGD2B (3,3))
$\phi_i = \sin^{-1}$ (–TI2B (3,2))	$\phi_{gd} = \sin^{-1}$ (–TGD2B (3,2))
$\psi_i = \tan^{-1}$ (TI2B (1,2)/TI2B (2,2))	$\psi_{gd} = \tan^{-1}$ (TGD2B (1,2)/TGD2B (2,2))
PITCH-YAW-ROLL SEQUENCE	
$\theta_i = \tan^{-1}$ (TI2B (1,3)/TI2B (1,1))	$\theta_{gd} = \tan^{-1}$ (TGD2B (1,3)/TGD2B (1,1))
$\psi_i = \sin^{-1}$ (–TI2B (2,2))	$\psi_{gd} = \sin^{-1}$ (–TGD2B (2,2))
$\phi_i = \tan^{-1}$ (TI2B (3,2)/TI2B (2,2))	$\phi_{gd} = \tan^{-1}$ (TGD2B (3,2)/TGD2B (2,2))
ROLL-PITCH-ROLL SEQUENCE	
$\theta_i = \cos^{-1}$ (TI2B (1,1))	$\theta_{gd} = \cos^{-1}$ (TGD2B (1,1))
$\phi_i = \tan^{-1}$ (TI2B (2,1)/TI2B (3,1))	$\phi_{gd} = \tan^{-1}$ (TGD2B (2,1))
$\psi_i = \phi_i$	$\psi_{gd} = \phi_{gd}$

CONTROL:

Attitude control of the flight vehicle being simulated by either the six-degree-of-freedom or point mass equations is accomplished by two routines: CONTRL - accessed by the SDDERV routine, and PTMSCO or ROLANG called by the PMDERV routine.

The CONTRL routine currently selects a value from tables of fin deflections (misalignments). This deflection influences a component of the total aerodynamic force and moment coefficients. The resulting forces and moments cause a deviation in the rigid-body vehicle flight path.

The PTMSCO routine controls the point source path through aerodynamic forces which are computed from force coefficients that were input in tabular form, or were obtained from the aerodynamics (AERO) routine as a result of computed aero angles (α, β). These angles are determined directly from input tables or indirectly, as a result of commanded attitude angles (θ_c, ψ_c), g-loadings, or a commanded elevation flight path angle (γ_c).

Two routines, SOALBT and SOLALP, which use Newton's method for solving nonlinear equations, are called to determine the aero angles. SOALBT computes the aero-angles resulting from commanded attitude angles; SOLALP computes the aero angles from the commanded normal and side g-loadings.

LOGIC FLOW CHART

CONTRL:

Performs a table look-up of fin deflections that induce angular rate effects into the aerodynamic moment evaluations.

Initialize table look up error flags (INVAL, ITFLAG)
CALL LOOKUP: get fin deflection angle (δp) to cause an angular rate about the X-X axis
CALL LOOKUP: get fin deflection angle (δq) to cause an angular rate about the Y-Y axis
CALL LOOKUP: get fin deflection angle (δr) to cause an angular rate about the Z-Z axis

LOGIC FLOW CHART

PTMSCO:

Lift, drag, and yaw force coefficients are determined by table look up. Aero angles are obtained by table look-up or are computed based upon tabular values of commanded attitude angles, load factors, or elevation flight-path angle.

Zero out commanded angles (ψ_c, θ_c, γ_c) and load factors (η_γ, η_σ)			
Zero out table look-up flags (INVAL, IFLAG)			
IXPTMC			
1	2	3	4,5,6
CALL LOOKUP: get drag coefficient (C_D)	CALL LOOKUP: get angle-of-attack (α)	CALL LOOKUP: get commanded yaw angle (ψ_c)	See following page
CALL LOOKUP: get lift coefficient (C_L)	CALL LOOKUP get angle-of-side slip (β)	CALL LOOKUP: get commanded pitch angle (θ_c)	
CALL LOOKUP: get yaw coefficient (C_y)		Convert angle to radius	
		CALL SOLABT: get α_{sx}, β_{sx}	
		$\alpha = \alpha_{sx}$, $\beta = \beta_{sx}$	

PTMSCO (cont):

IXPTMC		
4	5	6
CALL LOOKUP: get elevation load factor (η_γ)	CALL LOOKUP: get commanded elevation F.P.A. $(\gamma_c)_{i-1}$	User Supplied Controller
CALL LOOKUP: get horizontal load factor (η_σ)	Save the current altitude	
Set: A_2 = thrust force Compute: normal load force (A_3)	Compute incremental change of altitude	
Convert α, β degrees to radians	CALL LOOKUP: get commanded elevation F.P.A. $(\gamma_\gamma)_i$	
CALL AERO: get total aero coefficients	Restore current altitude	
Save normal force coefficient	Compute elevation F.P.A. for change in altitude	
Compute change in angle-of-attack	Compute time rate of change of elevation F.P.A. ($\dot{\gamma}$)	
CALL AERO: get total aero coefficients	Convert angles (α, β) to degrees	
Save the yaw coefficient	CALL AERO: get total lift coefficients and save	
Compute lift force change with angle-of-attack change (A_1)	Compute change in angle-of-attack ($\Delta\alpha$)	
CALL SOLALP: get angle-of-attack for load (α_{sx})	CALL AERO: get total lift coefficient for α change	
Compute change in angle-of sideslip	Compute change in lift coefficient for $\Delta\alpha$	
CALL AERO: get total aero coefficient	Compute lift force change	
Compute yaw force change with angle-of-sideslip change (A_1)	Compute time rate of change of elevation F.P.A. ($\dot{\gamma}$)	
Compute horizontal load force (A_3)	Compute angle-of-attack α_{sx} for change in horizontal F.P.A.	
CALL SOLALP: get angle-of-sideslip for load (β_{sx})		
Set: $\alpha = \alpha_{sx}$, $\beta = \beta_{sx}$		

EQUATIONS EVALUATED

PTMSCO:

IXPTMC = 4

NORMAL LOAD FORCE

$$A_3 = m\, g_{ref}\, \eta_\gamma$$

HORIZONTAL LOAD FORCE

$$A_3 = m\, g_{ref}\, \eta_\sigma$$

ANGLE-OF-ATTACK INCREMENTATION

$$\alpha = \alpha + \Delta\alpha_{PMC}$$

ANGLE-OF-SIDESLIP INCREMENTATION

$$\beta = \beta + \Delta\beta_{PMC}$$

LIFT FORCE CHANGE FOR ANGLE-OF-ATTACK INCREMENT

$$\frac{\Delta Cz_T}{\Delta\alpha_{PMC}} = \frac{(Cz_T - Cz_{T_s})}{\Delta\alpha_{PMC}}$$

$$A_1 = \frac{\Delta Cz_T}{\Delta\alpha_{PMC}}\, q\, S$$

YAW FORCE CHANGE FOR ANGLE-OF-ATTACK INCREMENT

$$\frac{\Delta Cy_T}{\Delta\beta_{PMC}} = \frac{(Cy_T - Cy_{T_s})}{\Delta\beta_{PMC}}$$

$$A_1 = \frac{\Delta Cy_T}{\Delta\beta_{PMC}}\, q\, S$$

NOTE: A_1 and A_3 are used as input to the aero-angle solving routine for computing either angle-of-attack or angle-of-sideslip.

IXPTMC = 5

ALTITUDE INCREMENTATION

$$h_{gd} = h_{gd} + \Delta h_{PMC}$$

ANGLE-OF-ATTACK INCREMENTATION

$$\alpha = \alpha + \Delta\alpha_{PMC}$$

ELEVATION F.P.A. CHANGE WITH ALTITUDE

$$\frac{\Delta\gamma_c}{\Delta h_{PMC}} = \frac{\gamma_{CN} - \gamma_{CS}}{\Delta h_{PMC}}$$

ELEVATION F.P.A. CHANGE WITH TIME

$$\frac{\Delta\gamma_c}{\Delta t} = \frac{\Delta\gamma_c}{\Delta h_{PMC}} \cdot V_{T_{gc}} \sin\gamma_{GC}$$

INCREMENTAL LIFT FORCE

$$\frac{\Delta C_{Z_T}}{\Delta\alpha_{PMC}} = \frac{C_{Z_T} - C_{Z_{T_s}}}{\Delta\alpha_{PMC}}$$

$$D_1 = \frac{\Delta C_{Z_T}}{\Delta\alpha_{PMC}} q S$$

GAIN FACTOR

$$\mathcal{Y} = mV_a/F_{X_T} \cos\alpha_{SX} + D_1$$

ELEVATION F.P.A. RATE

$$\dot{\gamma} = \left(\frac{F_{X_T} \sin\alpha_{SX} + D_1 \alpha_{SX}}{m - g_{ref} \cos\gamma_{GD}}\right)/Va$$

ANGLE-OF-ATTACK COMPUTED FOR γ CONTROL

$$\alpha_{SX} = \alpha_{SX} - \mathcal{Y}(\gamma_{SX} - \gamma_{GD}) + \mathcal{Y}_{PMC}\left(\frac{\Delta\gamma_c}{\Delta t} - \dot{\gamma}\right)$$

LOGIC FLOW CHART

SOLALP:

Uses Newton's method to solve for an aero-angle (α) that satisfies a commanded g-loading.

	Do for 20 iterations	
	Evaluate the equation H	
	Evaluate the derivative of H w.r.t. α: $\frac{\partial H}{\partial \alpha}$	
	Compute α increment: $\Delta\alpha$	
	Compute change in α	
	NO Is α increment less than ϵ_p?	**YES**
	Print nonconvergence message	
	CALL ABORT: complete abort procedure	

EQUATIONS EVALUATED

SOLALP:

EQUATION H:

$$H = A_1\alpha + A_2 \sin \alpha + A_3$$

EQUATION DERIVATIVE:

$$\frac{\partial H}{\partial \alpha} = A_1 + A_2 \cos \alpha$$

α INCREMENT:

$$\Delta\alpha = -\frac{H}{\frac{\partial H}{\partial \alpha}}$$

CHANGE IN ANGLE:

$$\alpha = \alpha + \Delta\alpha$$

LOGIC FLOW CHART

SOALBT:

Uses Newton's method to solve a system of two simultanous equations for the aero-angles (α, β) that satisfy the commanded attitude angles (θ_c, ψ_c) given the current flight path angles (γ, σ).

	Do for 20 iterations	
	Evaluate the equations H_1 and H_2	
	Evaluate the H_1 and H_2 derivatives w.r.t. α, β	
	Evaluate α, β increments: $\Delta\alpha$, $\Delta\beta$	
	Compute change in α and β	
	Compute resultant error in α, β	
	NO Is error less than ϵ_p?	YES
	Print nonconvergence message	
	CALL ABORT: complete abort procedure	

EQUATIONS EVALUATED

SOALBT:

SIMULTANEOUS EQUATIONS:

$$H_1 = \sin(\sigma - \psi_c)\cos\gamma - \left(\tan\beta/\sqrt{1.0 + \tan^2\alpha + \tan^2\beta}\right)$$

$$H_2 = \sin(\theta_c - \alpha)\sqrt{1.0 - \frac{\tan^2\beta}{(1.0 + \tan^2\alpha + \tan^2\beta)}} - \sin\gamma$$

EQUATION DERIVATIVES:

Set: $\overline{D2} = 1.0 + \tan^2\alpha + \tan^2\beta$

$$\overline{CBP} = \sqrt{1.0 - \frac{\tan^2\beta}{1.0 + \tan^2\alpha + \tan^2\beta}}$$

$$\frac{\partial H_1}{\partial\alpha} = ((\tan\beta\tan\alpha)/\cos^2\alpha)/(\overline{D2})^{3/2}$$

$$\frac{\partial H_1}{\partial\beta} = \left(-\frac{1.0}{(\overline{D2})^{1/2}} + \tan^2\beta/(\overline{D2})^{3/2}\right)/\cos^2\beta$$

$$\frac{\partial H_2}{\partial\alpha} = -\cos(\theta_c - \alpha)\,\overline{CBP} + \left(\sin(\theta_c - \alpha)\left(\frac{\tan\beta}{(\overline{D2})^{1/2}}\right)\frac{\partial H_1}{\partial\alpha}\right)/\overline{CBP}$$

$$\frac{\partial H_2}{\partial\beta} = \left(\sin(\theta_c - \alpha)\left(\frac{\tan\beta}{(\overline{D2})^{1/2}}\right)\frac{\partial H_1}{\partial\beta}\right)/\overline{CBP}$$

AERO-ANGLE INCREMENTS:

$$\Delta\alpha = \frac{-\left(\frac{\partial H_2}{\partial\beta}H_1 - \frac{\partial H_1}{\partial\beta}H_2\right)}{\frac{\partial H_1}{\partial\alpha}\frac{\partial H_2}{\partial\beta} - \frac{\partial H_1}{\partial\beta}\frac{\partial H_2}{\partial\alpha}}$$

$$\Delta\beta = \frac{-\left(-\frac{\partial H_2}{\partial\alpha}H_1 + \frac{\partial H_1}{\partial\alpha}H_2\right)}{\frac{\partial H_1}{\partial\alpha}\frac{\partial H_2}{\partial\beta} - \frac{\partial H_1}{\partial\beta}\frac{\partial H_2}{\partial\alpha}}$$

CHANGE IN AERO-ANGLES:

$$\alpha = \alpha + \Delta\alpha,\ \beta = \beta + \Delta\beta$$

RESULTANT ERROR:

$$\epsilon = \sqrt{\Delta\alpha^2 + \Delta\beta^2}$$

USRPMC:

This is currently a do-nothing routine and is included to satisfy the PTMSCO request. It is intended that coding for this routine be supplied by a user who has a specific requirement for controlling the point mass evaluations.

LOGIC FLOW CHART

ROLANG:

Determines an aerodynamic roll angle by table look-up for the point mass program.

Zero out table look-up flags (INVAL, ITFLAG)
Set roll angle (ϕ) to zero
CALL LOOKUP: get roll angle (ϕ)

GRAVITY:

The components of gravitational acceleration are required when evaluating the motion equation that simulate the path of an object near a planet. The GRAV routine computes the components in the geocentric coordinate system using the geocentric latitude and radial distance to the object. A fourth spherical harmonic (4th order) equation is used to compute the radial and tangential (toward north) acceleration components. The gravity component in the direction of increasing longitude is zero.

LOGIC FLOW CHART

GRAV:

Computes the fourth spherical harmonic components of gravitational acceleration-radial and tangential.

Compute the Legendre functions of geocentric latitude using the 2nd, 3rd, and 4th order terms (P_2, P_3, P_4, P_5, P_6, P_7)
Compute the geocentric components of the gravitational field (C_1, C_2, C_3)
Compute the radial direction gravitational acceleration component
Compute the tangential gravitational acceleration component

EQUATIONS EVALUATED

GRAV:

LEGENDRE FUNCTIONS:

$$P_2 = 1.0 - 3.0 \sin^2 \phi_{gc}$$
$$P_3 = \sin \phi_{gc} (3.0 - 5.0 \sin^2 \phi_{gc})$$
$$P_4 = 3.0 - 30.0 \sin^2 \phi_{gc} + 35.0 \sin^4 \phi_{gc}$$
$$P_5 = \sin \phi_{gc} \cos \phi_{gc}$$
$$P_6 = \cos \phi_{gc} (1.0 - 5.0 \sin^2 \phi_{gc})$$
$$P_7 = P_5 (7.0 \sin^2 \phi_{gc} - 3.0)$$

GEOCENTRIC COMPONENTS OF GRAVITATIONAL FIELD

$$C_1 = \mu/(R_i)^2$$ μ on p.25

$$C_2 = R_{eq}/R_i$$

$$C_3 = \left(\frac{R_{eq}}{R_i}\right)^2$$

where: Req = Equatorial radius
R_i = Radial distance: earth center to vehicle

RADIAL COMPONENT OF GRAVITY ACCELERATION

H_1, H_2, H_3 on p.25

$$g_z = C_1(1.0 + C_3(H_1P_2 + 0.8H_2C_2P_3 + (H_3/6.0)C_3P_4))$$

TANGENTIAL COMPONENT OF GRAVITY ACCELERATION

$$g_x = C_1C_3(-2.0\, H_1P_5 + 0.6H_2C_2P_6 + (2.0/3.0)H_3C_3P_7)$$

INITIALIZATION:

The process of initializing the code is divided into two parts: pre- and post-data read in. Pre-initialization assures that there are no residual (garbage) values associated with the code variables, and that each code variable has a nominal value compatible with code execution. Three routines: ZERCOM, ZERTAB/RINIT, and DEFALT accomplish this task. ZERCOM sets the contents of all labeled COMMON blocks to zero. The coding for this routine is generated by the PREPAR preprocessor as a by-product of building the labeled COMMON blocks. ZERTAB zeros out the table storage arrays by a call to RINIT which accesses the blank COMMON arrays. DEFALT provide the nominal code values to insure compatability during code execution. Since these routines are self-explanatory and possess no logic other than DO-loops and replacement statements, no logic flow charts will be given.

Post-initialization uses the preset nominal values or the current problem input values to compute a set of initial values for code startup either initially or after staging. These start up values are either code variable values or state variable values and are used for evaluating the differential equations associated with the user selected main program. The INITLZ routine controls the post-initialization process for both the problem start and restart after staging.

LOGIC FLOW CHART

INITLZ:

Initializes the variables associated with the differential equations of motion selected by the user for either problem start or restart after staging.

- Initialize major action flag (MAJSTG)
- Is this the first pass through routine? (NO / YES)
  - NO: Is thrust routine affected by staging? (NO / YES)
    - YES: Reinitialize THRUST routine variables
    - Set major action flag
  - NO: Is vehicle mass properties routine affected by staging? (NO / YES)
    - YES: Reinitialize VEHMSP routine variables
    - Set major action flag
  - NO: Is DEBUG routine affected by staging? (NO / YES)
    - YES: Reinitialize DEBUG routine variables
  - NO: Is aero routine SDAERO? (NO / YES)
    - NO: Is aero routine PMAERO? (NO / YES)
      - YES: Set aero angles (α, β, ϕ_a) to zero
      - Is AERO routine affected by staging? (NO / YES)
        - YES: Reinitialize AERO routine values
  - NO: Has major action flag been set? (NO / YES)
    - YES: Increment stage counter
    - Set equation counter to zero
  - YES: Set differential equations counter
  - Set program variables to read-in problem start values
  - NO: Is planet Earth and oblate? (NO / YES)
    - YES: Compute flattening factor (β_P) from planet radii
  - CALL GD2GC: get geocentric latitude, altitude, and radial distance
  - CALL ATMSET: select the atmosphere properties chosen by user
  - NO: Are Yawsonde sun angles required? (NO / YES)
    - YES: Compute transformation: sun to geodetic ($T_{GD/SN}$)

	NO	Is main program SIXDOF?		YES
		CALL SDINLZ: Initialize the six degree-of-freedom program variables		
	NO	Is main program PTMASS?		YES
		CALL PMINLZ: Initialize the point mass program variables		
	NO	Is AERO routine SDAERO or PMAERO?		YES
		NO	Is aero routine active?	YES
			CALL AROINL: Initialize active flags for aero tables	
	NO	Is THRUST routine active?		YES
		CALL THRINL: Initialize active flags for thrust tables		
	NO	Is vehicle mass properties routine active?		YES
		CALL VMPINL: Initialize active flags for VEHMSP routine		
	Compute center of mass location (X_{cg}, Y_{cg}, Z_{cg}) w.r.t. reference point			
	NO	Is STATION/RADAR routine active?		YES
		CALL STAINL: Initialize STATION routine variables		
	NO	Is DEBUG routine active?		YES
		CALL DBGINL: Initialize DEBUG routine variables		
	NO	Is file TAPE20 to be written?		YES
		CALL T20INL: Set up block of variables to be written		
	CALL BOPINL: Set up block of OUTPUT variables to be written at print intervals			
	Turn off major action flag			
	CALL INTARA: Set up STATE and RATE integration arrays			
CALL STGINL: compute staging variable locations				
CALL PUTNRA: place program values into integration STATE array				
Turn off first pass flag (IFIRST)				

EQUATIONS EVALUATED

INITLZ:

PLANET FLATTENING FACTOR

$$\beta_p = \frac{R_{EQ}^2}{R_{POL}^2} - 1.0$$

where: R_{EQ} = Equatorial radius
R_{POL} = Polar radius

TRANSFORMATION: SUN TO GEODETIC

$$\text{TSN2GD (1)} = \cos \psi_{SUN} \cos \theta_{SUN}$$
$$\text{TSN2GD (2)} = \sin \psi_{SUN} \cos \theta_{SUN}$$
$$\text{TSN2GD (3)} = -\sin \theta_{SUN}$$

where: ψ_{SUN}, θ_{SUN} = Sun azimuth and elevation

CENTER-OF-MASS w.r.t. BODY REFERENCE POINT

$$X_{cg} = X_{cm} - X_{BRP}$$
$$Y_{cg} = Y_{cm} - Y_{BRP}$$
$$Z_{cg} = Z_{cm} - Z_{BRP}$$

ZERCOM:

ZERTAB/RINIT:

DEFALT:

The coding for these routines is trivial and repetitious, and not suitable for flow charting. However, the interested user is referred to the coding for more specific details.

INTEGRATION:

Integration of the differential equations of motion associated with the main program is accomplished by a numerical integrator. The current integrator accessed by the code is the variable step RKF (Fehlberg fourth-fifth order Runga-Kutta) routine. It is a Sandia math library routine and, as such, can be readily replaced by other library integrators because the argument lists associated with the library routines have been standardized.

In support of the integrators and to allow interfacing between the integrator and the main program, five routines are provided (INTARA, PUTNRA, TAKFRA, INGRTR, and DERIV) to control the data flow.

After data read in and main program initialization, INITLZ accesses the routine INTARA to determine the number and sequence of the first order differential equations associated with the user requested main program, and to compute the absolute core location of the associated state and rate variable values. PUTNRA is called to fetch the current value of the differential equations state variables from COMMON storage and store these values into the integration array in the order determined by INTARA. This sequence of calls allow the integration arrays to be of arbitary length depending upon the problem solution requirements (i.e., the number of support routines requested).

During the computational phase (PHASE 2), the INGRTR routine controls the code problem solving activities between print intervals by repeated calls to the RKF routine. RKF accesses the DERIV routine and returns after one good integration time step. The derivative evaluator (SDDERV or PMDERV) called by the DERIV routine has access to the state variable values - updated by the integrator - via the TAKFRA routine. TAKFRA fetches the values from the integration array and returns them to COMMON storage - a reverse process of PUTNRA.

When the derivative evaluator has completed its task of evaluating the derivative rates, PUTNRA is again used to fetch the rate values from COMMON storage and store them into the integration rate array.

LOGIC FLOW CHART

INTARA:

Determines the number of first order differential equations associated with the current problem, and the absolute core location of the equation variable value.

	Compute and store the reference value core location			
NO	Is IMPROG = SIXDOF?			YES
	Increment the equation counter (NEQN)			
		Do for 13 variable names		
		CALL CWHERE: get COMMON block storage information for variable name		
		CALL CLOCAT: get absolute core location for variable name value		
		Fill arrays with subscripts associated with the state and rate variable names		
		NO	Are subscripts positive?	YES
			Increment subscripts by 1	
		Store the state variable name		
NO	Is IMPROG = PTMASS?			YES
	Increment the equation counter (NEQN)			
		Do for 6 variable names		
		CALL CWHERE: get COMMON block storage information for variable name		
		CALL CLOCAT: get absolute core location for variable name value		
		Fill arrays with subscripts associated with the state and rate variable names		
		NO	Are subscripts positive?	YES
			Increment subscripts by 1	
		Store the state variable name		
NO	Is THRUST routine activated?			YES
	Increment the equation counter (NEQN)			
	CALL CWHERE: get COMMON block storage information for thrust variable names			
	CALL CLOCAT: get absolute core location for variable name values			
	Fill arrays with subscript associated with the thrust state and rate variable names			
	NO	Is subscript positive?		YES
		Increment the subscript by 1		
	Store thrust state variable name			
NO	Is range calculation flag set?			YES
	Increment the equation counter (NEQN)			
	CALL CWHERE: get COMMON block storage information for range variable names			
	CALL CLOCAT: get absolute core location for variable name values			
	Fill arrays with subscript associated with the range state and rate variable names			
	NO	Is subscripts positive?		YES
		Increment the subscript by 1		
	Store range state variable name			

LOGIC FLOW CHART

PUTNRA:

Uses the reference location and subscripts of INTARA to store the current problem value of the differential equation variables into the state and rate integration arrays.

	Set the state or rate flag	
	Do for number of differential equations (NEQN)	
	Get variable value subscript from array	
	NO Is state flag set?	YES
	Fill the rate array	Fill the state array

LOGIC FLOW CHART

TAKFRA:

Takes the updated values from the integration state array and returns them to the COMMON blocks for code use.

	Do for number of differential equations (NEQN)
	Get variable value subscript from array
	Store value in absolute core location

LOGIC FLOW CHART

INGRTR:

Calls the current numerical integrators for one good time step, tests for staging, and continues the call and test sequence until print time pause.

	CALL RKF: accesses the DERIV routine to evaluate the problem differental equations for one time step. Set INTFLG.		
NO	Is/INTFLG/= 2?		YES
NO	Is INTFLG = 3?	YES	
NO	Is INTFLG = 6?	YES	
	Set INTFLG = -2		
	Return to RKF call		
	Write integration error message		
	CALL DMPSTP: get system core dump and stop.		
	Set current time and good time step		
	CALL AUXCLC: compute miscellaneous program values		
	Do for number of equations		
	Set current array of state variables		
	CALL PUTNRA: give current rate array values to the integration array		
	CALL STGTST: test current values for staging, set NOSTG		
NO	Is NOSTG ≠ 0?		YES
	Do for number of equations		Set: Integration start flag and exit time
	Save state and rate array		
	Save the current time		
NO	Is INTFLG = -2?	YES	
INTFLG = -2	Return to RKF call		

LOGIC FLOW CHART

DERIV:

Determines the derivative evaluator (SDDERV or PMDERV) for the problem and initiates the evaluation of the set of differential equations.

	Set program time to integrator time		
	CALL TAKFRA: fetch state variable values from the integration array and store in COMMON blocks for code use		
NO	Is IMPROG = SIXDOF?		YES
NO	Is IMPROG = PTMASS?	YES	CALL SDDERV: evaluate the six-degree-of-freedom set of differential equations
	CALL PMDERV: Evaluate the point mass set of differential equations		
	CALL PUTNRA: fetch the rate variable values from COMMON storage and store in the integration array		

LATITUDE CONVERSION

The current position of the vehicle at any time during its trajectory can be represented by latitude, longitude, and altitude - with latitude and altitude consistent with a geodetic or geocentric axis system.

Since it is possible to start problem execution in either axis systems, the starting position is usually referenced to a geodetic system. However, during code execution the geocentric axis system is used as an intermediate frame of reference.

Two routines (GD2GC and GC2GD) are used to convert latitude and altitude from one reference system to the other. GD2GC is a straight forward trigonometric solution for the geocentric latitude angle and altitude, whereas, GC2GD uses Newton's method to iterate to a solution in the geodetic frame.

The function routine RADIUS computes an earth radius relative to the geodetic or geocentric axis system origin.

LOGIC FLOW CHART

GD2GC:

Computes geocentric referenced latitude, altitude, and vehicle radial distance from geodetic referenced latitude and altitude.

NO — Is the current geodetic latitude at the poles or equator?	YES
Compute local geocentric latitude angle, ϕ_{Lgc}	Set: geocentric latitude and altitude to geodetic values
Compute local geocentric referenced vehicle distance, R_{Lgc}	
Compute geocentric referenced latitude angle, ϕ_{gc}	Compute geocentric vehicle distance using polar or equational radius
Compute geocentric vehicle distance, R	
CALL RADIUS: get geocentric earth radius, R_E	
Compute geocentric altitude, h_{gc}	

EQUATIONS EVALUATED

GD2GC

LOCAL GEOCENTRIC LATITUDE:

$$\phi_{L_{gc}} = \tan^{-1}\left(\tan \phi_{gd}\left(\frac{R^2_{POL}}{R^2_{EQ}}\right)\right)$$

LOCAL GEOCENTRIC VEHICLE DISTANCE:

$$R_{L_{gc}} = R_{\phi_{L_{gc}}} (\sin \phi_{L_{gc}}/\sin \phi_{gd})$$

GEOCENTRIC LATITUDE:

$$\phi_{gc} = \tan^{-1}\left(\frac{(h_{gd} + R_{L_{gc}}) \sin \phi_{gd}}{h_{gd} \cos \phi_{gd} + R_{L_{gc}} \cos \phi_{L_{gc}}}\right)$$

If ϕ_{gd} = A Pole or Equator

$$\phi_{gc} = \phi_{gd}$$

GEOCENTRIC VEHICLE DISTANCE:

$$R = (R_{L_{gc}} + h_{gc}) \sin \phi_{gd}/\sin \phi_{gc}$$

$R = R_{EQ} + h_{gc}$

or

$R = R_{POL} + h_{gc}$

where: R_{EQ}, R_{POL} = Equatorial and Polar radiis

GEOCENTRIC ALTITUDE:

$$h_{gc} = R - R_E$$

$$h_{gc} = h_{gd}$$

LOGIC FLOW CHART

GC2GD

Uses Newton's method to compute the geodetic latitude and altitude from geocentric coordinates.

	Evaluate the starting point: X_L		
	Compute constants B_1 and B_2		
	Do for 10 interations		
	Evaluate the function at X_L: $F(X_L)$		
	NO	Is the value less than 1.0E-13?	YES
	Evaluate the function derivative at X_L:$F'(X_L)$		
	Advance the solution 1 step: X_{L+1}		
Print nonconvergence message	Compute X and Z geodetic displacements		
	Compute geodetic latitude		
CALL ABORT:	Compute geodetic altitude		

EQUATIONS EVALUATED

GC2GD:

STARTING POINT:

$$X_L = \left(\frac{\sqrt{(R_E \cos \phi_{gc})^2 + (R_E \sin \phi_{gc})^2}}{R_{gc}} - 1.0 \right) R_{POL}^2$$

CONSTANTS:

$$B_1 = (R_E \cos \phi_{gc})^2 / R_{EQ}^2$$

$$B_2 = (R_E \sin \phi_{gc})^2 / R_{POL}^2$$

$$D_1^2 = (1.0 + X_L / R_{EQ}^2)^2$$

$$D_2^2 = (1.0 + X_L / R_{POL}^2)^2$$

FUNCTION EVALUATION:

$$F(X_L) = D_1^2 D_2^2 - B_1 D_2^2 - B_2 D_1^2$$

FUNCTION DERIVATIVE EVALUATION:

$$\begin{aligned} F'(X_L) \quad &= 2.0 \, [(1.0 + X_L/R_{EQ}^2) \, D_2^2/R_{EQ}^2 + (1.0 + X_L/R_{POL}^2) \, D_1^2/R_{POL}^2 \\ &- B_1 \, (1.0 + X_L/R_{POL}^2)/R_{POL}^2 - B_2 \, (1.0 + X_L/R_{EQ}^2)/R_{EQ}^2] \end{aligned}$$

SOLUTION ADVANCE:

$$X_{L+1} = X_L - \frac{F(X_L)}{F'(X_L)}$$

GEODETIC LATITUDE:

$$\phi_{gd} = \tan^{-1} (\sin \phi_{gc} \, (R_{EQ}^2 + X_L) / \cos \phi_{gc} \, (R_{POL}^2 + X_L))$$

GEODETIC ALTITUDE:

$$h_{GD} = \sqrt{\left[\left(1 - \frac{R_{EQ}^2}{(R_{EQ}^2 + X_L)} \right) R_E \cos \phi_{gc} \right]^2 + \left[\left(1 - \frac{R_{POL}^2}{(R_{POL}^2 + X_L)} \right) R_E \sin \phi_{gc} \right]^2}$$

LOGIC FLOW CHART

RADIUS:

A function routine to compute the planet geoid geocentric radius for a geocentric or geodetic latitude angle (ϕ_G), and β equals the planet flattening factor.

$$\text{Radius} = R_{EQ} / \sqrt{(1.0 + \beta_p \sin^2 \phi_G)}$$

R_{eq} = 6.3781660 E+6 meters p.25

β_f = .0067385 (use .006693423) p.25 (eccentricity)^2 or e^2

R_{pol} = 6.35678428E+6 meters p.25

$$\tan \phi_{gc} = (1-\beta_f) \tan \phi_{gd} \Rightarrow \tan \phi_{gd} = \frac{\tan \phi_{gc}}{(1-\beta_f)}$$

e.g. $\phi_{gc} = 35 \Rightarrow \phi_{gd} = 35.181°$ $\Delta = .181°$

or $\phi_{gd} = 35 \Rightarrow \phi_{gc} = 34.819°$ $\Delta = .181°$

r_{gc} = 6.371154996E+6 meters r_{gd} = 6.371091769E+6 m

Δ = 63.227 meters.

MATRIX OPERATION

Transformation of the various code items between the reference axis systems makes use of matrix arithmetic. Therefore, four routines (TRNPOS, MULT31, MULT33, and INVR3) are included with the code to provide this capability.

Matrix inversion is accomplished by the TRNPOS and INVR3 routines. TRNPOS computes the transpose or inverse of an orthogonal matrix while INVR3 computes the inverse of other 3x3 matrices.

Matrix multiplication is provided by the MULT31 and MULT33 routines, and, as their names imply, perform 3x1 and 3x3 matrix multiplication.

Since these routines consist of nested Do-loops with no logic, logic flow charts are not included here.

OUTPUT:

The listing of computed data or current core-stored data values takes place at various times throughout code execution. A summary page of data is presented immediately following code initialization or reinitialization after staging. At the user-specified print interval during the computation phase, two blocks of data are available for presentation. One is a block-formatted list of the current case values, and the other an array of current data values that can be dumped, in binary format, to the file TAPE20. After code execution to the stop criteria, an optional output phase (PHASE3) is available. During this phase, tabular listings of PHASE2 computed data may be presented in a series of code or user-specified formats. There is also a method provided to graphically present the data in the form of Y vs X plots.

The summary page generated by the WRTINL routine displays input data and data that reflect the first pass through the main program.

The time history block of data written to the output file at user-specified intervals is produced by the BOPWRT routine. BOPWRT loads an array with core values whose locations are defined as subscripts of a reference item. The subscripts which represent core locations were determined by the BOPINL routine during initialization. Similarly, the TAPE20 file block is produced by the T20WRT routine from subscripts provided by the T20INL routine.

The HEADER routine, called by the various output routines during execution, displays the AMEER logo and the stage, case, and page numbers.

PHASE3 causes the computed data of PHASE2 to be displayed in a tabular format through the use of the TABLOP routine and/or in graphic form through the use of the PLTMKR routine which uses the routines STRDAT, SKPFIL, and PLOTDT.

PLTMKR determines the types of plots requested and the dimensions for partitioning the three-dimensional data storage array DATA. STRDAT, using file TAPE20, determines, retrieves, and stores into the DATA array the required data items. SKPFIL advances the TAPE20 file through one EOF. PLOTDT uses the DISSPLA package routines to generate plots in the desired form.

LOGIC FLOW CHART

WRTINL:

Writes the report summary page of problem initial and first pass data.

	CALL HEADER: write the AMEER logo, stage, case, page number	
	Write current date and code version	
	Write problem title block	
	Write problem and planet specifications	
	Write subprogram status for problem	
	Write launch position and orientation specifications	
	Write vehicle characteristics for each body	
	Write aerodynamic routine specifications	
NO	Is thrusting required?	YES
	Write thrusting characteristics for each motor	
NO	Is station/radar calculations required?	YES
	Write station/radar orientation specifications	
	Write the current integration array names	
	Write the time-history output block names	
NO	Is station/radar calculation required?	YES
	Write station/radar output block names	
NO	Is thrusting required?	YES
	Write thrust routines block output names	
NO	Is aerodynamic routine active?	YES
	Write aerodynamic routine output block names	
NO	Is vehicle mass properties required?	YES
	Write vehicle mass properties output block names	
	CALL HEADER: write AMEER logo, stage, case, and page numbers	

LOGIC FLOW CHART

BOPINL:

Determines and stores the absolute core location for the time-history block variable values.

	Zero out main program and subprogram variable counters			
NO	Is main program SIXDOF?			YES
		Do until main program name list is blank		
			Increment the main program counter	
			CALL CWHERE: get common block data for variable name	
			CALL CLOCAT: get absolute core location for variable name value	
			Store core location and variable name into arrays	
	NO	Are load-factor values to be computed and printed?		YES
			Do for three items	
				Increment the main program counter
				CALL CWHERE: get common block data for name
				CALL CLOCAT: get absolute core location for name value
				Store core location and variable name into arrays
	NO	Is the critical frequency to be computed and printed?		YES
			Increment the main program counter	
			CALL CWHERE: get common block data for name	
			CALL CLOCAT: get absolute core location for name value	
			Store core location and variable name into array	
	NO	Are Yawsonde sun angles to be computed and printed?		YES
			Increment the main program counter	
			CALL CWHERE: get common block data for name	
			CALL CLOCAT: get absolute core location for name value	
			Store core location and variable name into arrays	
NO	Is main program PTMASS?			YES
		Do until main program name list is blank		
			Increment the main program counter	
			CALL CWHERE: get common block data for variable name	
			CALL CLOCAT: get absolute core location for name value	
			Store core location and variable name into arrays	

	NO	Is AERO subprogram active?		YES
		Set six-degree-of-freedom aero counter (NARO = 6)		
		NO	Has PMAERO been requested?	YES
			Set: NARO = 3	
			Do from 1 to NARO	
			Increment the aero counter	
			CALL CWHERE: get common block data for variable name	
			CALL CLOCAT: get absolute core location for name value	
			Store core location and variable name into array	
NO	Are range values to be computed and printed?			YES
		Do for six range items		
		Increment the main program counter		
		CALL CWHERE: get common block data for variable name		
		CALL CLOCAT: get absolute core location for name value		
		Store core location and variable name into arrays		
NO	Is station/radar computations active?			YES
		Do for 12 station/radar items		
		Increment the station/radar counter		
		CALL CWHERE: get common block data for variable name		
		CALL CLOCAT: get absolute core location for name value		
		Store core location and variable name into arrays		
NO	Has thrusting been requested?			YES
		Do until thrust name list is blank		
		Increment the thrust counter		
		CALL CWHERE: get common block data for thrust variable name		
		CALL CLOCAT: get absolute core location for thrust name value		
		Store core location and variable name into arrays		
NO	Is vehicle mass properties to be computed and printed?			YES
		Do for nine items		
		Increment the mass properties counter		
		CALL CWHERE: get common block data for mass properties variable name		
		CALL CLOCAT: get absolute core location for mass properties name value		
		Store core location and variable name into arrays		
	NO	Are moment of inertia rates to be used?		YES
			Do for six items	
			Increment the mass properties counter	
			CALL CWHERE: get common block data for M.I. rate variable name	
			CALL CLOCAT: get absolute core location for M.I. rate name value	
			Store core location and variable name into arrays	

LOGIC FLOW CHART

BOPWRT:

Determines the current values of the block print variables and writes the block of values to the output file.

NO	Have PHASE3 tabular prints been selected?		YES
NO	Are the number of lines required greater than 60?	YES	
	CALL HEADER: Write AMEER logo, stage, case, and page numbers		
	Sum up the line count		
	Set the reference location		
	Do for number of main program output values		
	Find the main program variable value location and store value in the write block array		
	Write the main program variable values to the output file		
NO	Are station/radar values to be written?	YES	
	Do from 1 to station/radar count		
	Find the station/radar variable value location and store value into the write array		
	Write the station/radar value to the output file		
NO	Are thrust values to be written?	YES	
	Do from 1 to thrust count		
	Find the thrust variable value location and store value into write block array		
	Write the thrust values to the output file		
NO	Are the vehicle mass properties to be written?	YES	
	Do from 1 to mass properties count		
	Find the mass properties value location and store value into write block array		
	Write the mass properties values to the output file		

HEADER:

Writes code logo, stage, case, and page number

Increment the page counter
Set line counter to zero
Set top-of-page carriage control, and write stage, case, and page number
Write AMEER logo
Increment line counter by four

LOGIC FLOW CHART

T20INL:

Writes the BCD name header block onto file TAPE20, and determines the reference items subscripts for the values associated with the header block names.

NO	Is this the first pass for this run?		YES
	Zero out the data block counter		
	Store the location of the reference item		
	Increment the case counter		
		Do until block name list is blank	
		Increment the data block counter	
		CALL CWHERE: get common block data items for variable name	
		NO Is sum of data items zero?	YES
		CALL CLOCAT: get absolute core location for name value	Write error message and current name
		Compute the reference item subscript	Decrement the data block counter
		Write current case, title block, and data counter value to file TAPE20	
		Write on end-of-file (E-O-F) onto file TAPE20	

T20WRT:

Writes a binary data block of code variable values to file TAPE20.

	BACKSPACE file TAPE20 over previous E-O-F	
	Do for number of data block variables	
		Fill output array with current core values using subscripts found by T20INL
	Write binary data block to file TAPE20	
	Write an end-of-file (E-O-F) onto file TAPE20	

LOGIC FLOW CHART

TABLOP:

Reads PHASE2 computed data stored on file, TAPE20, and tabulates the data according to a preassigned format or user-specified format.

- Do for all table requests
  - REWIND: TAPE20 data file
  - READ: file header record (ICASE, ITITLE, N20DAT)
  - Has an EOF been read? — NO / YES → Write no data error message
  - READ: BCD header block-data variable names
  - Is this a standard table request? — NO / YES
    - Are there any more standard table requests? — NO / YES → Write no data error message
    - Set: user table flag
    - Do for users table names
      - Is user table name in header block list? — NO / YES
        - Record the block subscript
    - Do for names of table
      - Is table name in the header block list? — NO / YES
        - Record the block subscript
    - Set: number of names counter
    - Set: number of names counter
    - Set: User table headings
    - Set: User table format
    - Set: current table headings
    - Set: current table format
  - CALL HEADER: Write code LOGO, case and page number
  - WRITE: Table headings
  - Do until all stored data is retrieved
    - READ: file (TAPE20) binary data block
    - Store data in print array according to subscripts
    - WRITE: Print data array
    - Increment the line counter
    - Is line counter greater than 45? — NO / YES
      - Return to header call

LOGIC FLOW CHART

PLTMKR:

Determines the type of plots requested, determines the dimensionality of the DATA array, requests STRDAT to store the required data, and sends the data to PLOTDT for plot generation.

- Initialize flag and counter variable
- Do for case requests
  - NO: Is case request blank? — YES
  - NO: Is this a stacked case request? — YES
    - Print: maximum number of cases exceeded
    - Set: Stacked case flag
- Set: max number of cases
- Set: number of cases requested
- Set up for plot start
- Do for plot requests
  - NO: Is plot request blank? — YES
  - NO: Is this a stacked plot request? — YES
    - Print: maximum number of plots exceeded
    - Set: Stacked plot flag
- Set: max number of plots
- Set: number of plots requested
- NO: Is there an error condition for plotting or stacking? — YES
  - Print: Error Message
- Compute the DATA array dimension
- Rewind data file (TAPE20)
- Do until case request is satisfied
  - READ file header record
  - NO: Are all TAPE20 files checked — YES
  - NO: Is file case the required case? — YES
    - Print: Error Message
  - CALL SKPFIL: skip to end-of-file
    - RETURN
- READ data file (TAPE20) data header block
- Save file case identifier
- NO: Is this a stacked case request? — YES
- Do for length of header block
  - NO: Is header block name the plot request name? — YES
    - Save the subscript of the block name
- Do for current plot request count
  - NO: Is there a subscript for the plot request? — YES
  - PRINT: error message
  - Recompute DATA storage dimensions
  - Reduce plot request count and array

	CALL STRDAT: Read the TAPE20 file and store the data		
NO	Is this a stacked case request?		YES
NO	Are all cases stored?	YES	
	Return to case request search		
	CALL PLOTDT: Plot the stored data		
NO	Have all plot requests been satisfied?		YES
	Return to: set up for plot start		
NO	Have all case requests been satisfied?		YES
	Increment the case counter		
	Intialize the plot counters		
	Return to: set up for plot start		

LOGIC FLOW CHART

STRDAT:

Retrieves data from the TAPE20 file, and stores it into the DATA array according to the dimensions computed by PLTMKR.

<table>
<tr><td rowspan="8"></td><td colspan="4">Do until EOF or storage limit</td></tr>
<tr><td colspan="4">Increment the storage counter</td></tr>
<tr><td>NO</td><td colspan="2">Is counter greater than storage limit</td><td>YES</td></tr>
<tr><td colspan="3">READ in a data value block record</td><td rowspan="2">PRINT: Storage Message</td></tr>
<tr><td>NO</td><td>Has an EOF been read?</td><td>YES</td></tr>
<tr><td rowspan="2"></td><td colspan="2">Do for all plot variables</td><td rowspan="2"></td><td rowspan="3">CALL SKPFIL: read TAPE20 file to EOF</td></tr>
<tr><td>Store the block data value into DATA array - using subscripts</td></tr>
<tr><td colspan="4">Decrement the storage counter</td></tr>
</table>

LOGIC FLOW CHART

PLOTDT:

Generates the user-requested plots using the data stored by STRDAT.

```
Blank out the title and axis label arrays
CALL DISSPLA routines: BGNPL, PAGE, HEIGHT, NOBRDR to initialize and set up plot
NO  Is this a stacked case plot?                                                  YES
    Generate title and axis labels for stacked cases
NO  Is this a stacked X-Y plot?                                                   YES
    Generate title and axis labels for stacked plots
    Determine X-axis min and max values for straight X-Y plots
    Do for plots requested
        Generate title and axis labels for plot
        Determine Y-axis min and max values
        CALL DISSPLA routines: TITLE, GRAF, GRID, CURVE, ENDPL
        to generate the requested plot
            Do for one stack plot request
            Determine X and Y axis max and min values
            CALL DISSPLA routines: TITLE, GRAF, GRID to set up the plot
                Do for number of stacked items
                    CALL CURVE: Stack the X-Y plots
            CALL ENDPL: finish plotting sequence
            Determine X and Y axis max and min values
            CALL DISSPLA routines: TITLE, GRAF, GRID to set up the plot
                Do for number of stacked cases
                    CALL CURVE: Stack the case plot
            CALL ENDPL: finish plotting sequences
```

STAGING

A definition: Staging is a method whereby the code calculations are interrupted at a predefined (user-selected) variable value, new data is introduced, and the current problem calculations continued.

The preparation for and the actual testing of program variables for staging requires the use of four routines: STGINL, STGTST, STGEQN, STGEQR, and one library routine ZEROIN.

During initialization, STGINL determines the absolute core location and the total number of user-requested variable values plus the default variable-altitude. These locations are passed to the compute phase for testing reference.

After each successful integration time step, the STGTST routine is called to determine whether the current calculated value of the stage variable has caused a sign change in the stage testing equation of STGEQN. If no sign change occur, integration is allowed to continue. However, when a sign change occurs, the routine ZEROIN is called. It uses the bisector-secant scheme to determine a best-guess root value. STGEQR is called to update the integration state array to reflect the ZEROIN root value by using a Hermite interpolating formula to predict the new values from current and past state and rate values. ZEROIN iterates to a specified error limit on the stage test variable value by using the derivative evaluation routine directly - bypassing the integrator.

LOGIC FLOW CHART

STGINL:

Determines the number of requested staging variables, the absolute core location of the variable values, and the reference location for subscript calculations.

	Determine core location of subscripting reference (LRFSTG)	
	Set variable counter to zero	
	Do for I = 1,4	
	NO Is staging variable list item blank?	YES
	Increment the variable counter	
	CALL CWHERE: get COMMON block, starting position, and dimension of stage variable name	
	CALL CLOCAT: get absolute core location of staging variable value	
	Increment the variable counter	
	CALL CWHERE: get COMMON block, starting position, and dimension of the altitude variable	
	CALL CLOCAT: get absolute core location of value stored for altitude	

LOGIC FLOW CHART

STGTST:

Tests for a sign change during the last "good" time step, and activates ZEROIN to interate to the desired value if sign change occurred.

- Initialize stage test flag (NOSTG = 0)
- Set testing error criteria
- Do until variable count is satisfied
  - Evaluate the staging equation
  - NO — Has staging equation changed sign? — YES
    - Save the stage equation value
    - NO — Is stage equation value zero? — YES
      - CALL ZEROIN: iterate to the requested staging variable value
      - NO — Has iteration procedure succeeded? — YES
        - Print error message
        - CALL DMPSTP: dump core
        - Save the root time
        - Do for number of equations
          - Set the state array to the root state values
          - Set the rate array to the last rate values
      - Set the stage test completion flag (NOSTG=1)

LOGIC FLOW CHART

STGEQN:

Evaluates the difference between the requested stage variable value and the current problem value of the stage variable .

	Determine the subscript of current value of the staging variable w.r.t. the reference location	
NO	Is subscript positive?	YES
	Increment the subscript	
	Set STGEQN to the difference of current and requested value of the staging variable	

LOGIC FLOW CHART

STGEQR:

Uses Hermite interpolation to determine integration state values consistent with the root value of ZEROIN, and calls the derivative evaluation routine to determine rate values.

	Determine current good step time increment	
NO	Is the increment zero?	**YES**
	Do for number of differential equations	
	Compute new state array values at root time - Hermite interpolation	
	CALL DERIV: update the derivative evaluation routine	
	CALL AUXPKG: update the auxillary calculations	
	Set STGEQR to current value of STGEQN	

STATION/RADAR:

Radar look information (azimuth, elevation, and slant range) is computed during the computational phase by the RADAR routine. The look values are computed relative to a station coordinate system whose origin is designated by a latitude, longitude, and altitude, and an axis orientation designated by an initial heading.

During initialization, the STAINL routine computes the earth-centered displacement components and the direction cosine transformation relating station coordinates to earth-centered coordinates.

LOGIC FLOW CHART

STAINL:

Computes the earth center to station XYZ displacements and the direction cosine transformation.

Compute earth center to station direction cosine transformation (TE2ST)
CALL GD2GC: compute geocentric latitude (ϕ_{ST}), altitude ($h_{GC_{ST}}$), and station distance to earth center (R_{ST})
Compute earth-centered station XYZ displacements ($X_{E_{ST}}$, $Y_{E_{ST}}$, $Z_{E_{ST}}$)
CALL RADAR: compute initial values of azimuth, elevation, and slant range

EQUATIONS EVALUATED

STAINL:

TRANSFORMATION EARTH CENTER TO STATION

TE2ST

$$(1,1) = \cos \Delta\phi_s \cos \phi_{gc_s} \cos \theta_{ST} - \sin \Delta\phi_s \sin \phi_{gc_s} \cos \theta_{ST}$$

$$(2,1) = -\cos \psi_{ST} \sin \theta_{ST} + \sin \psi_{ST} \sin \Delta\phi_s \cos \phi_{gc_s} \cos \theta_{ST} + \sin \psi_{ST} \cos \Delta\phi_s \sin \phi_{gc_s} \cos \theta_{ST}$$

$$(3,1) = \sin \psi_{ST} \sin \theta_{ST} + \cos \psi_{ST} \sin \Delta\phi_s \cos \phi_{gc_s} \cos \theta_{ST} + \cos \psi_{ST} \cos \Delta\phi_s \sin \phi_{gc_s} \cos \theta_{ST}$$

$$(1,2) = \cos \Delta\phi_s \cos \phi_{gc_s} \sin \theta_{ST} - \sin \Delta\phi_s \sin \phi_{gc_s} \sin \theta_{ST}$$

$[T_{ST/E}]$

$$(2,2) = \cos \psi_{ST} \cos \theta_{ST} + \sin \psi_{ST} \sin \Delta\phi_s \cos \phi_{gc_s} \sin \theta_{ST} + \sin \psi_{ST} \cos \Delta\phi_s \sin \phi_{gc_s} \sin \theta_{ST}$$

$$(3,2) = -\sin \psi_{ST} \cos \theta_{ST} + \cos \psi_{ST} \sin \Delta\phi_s \cos \phi_{gc_s} \sin \theta_{ST} + \cos \psi_{ST} \cos \Delta\phi_s \sin \phi_{gc_s} \sin \theta_{ST}$$

$$(1,3) = -\cos \Delta\phi_s \sin \phi_{gc_s} - \sin \Delta\phi_s \cos \phi_{gc_s}$$

$$(2,3) = -\sin \psi_{ST} \sin \Delta\phi_s \sin \phi_{gc_s} + \sin \psi_{ST} \cos \Delta\phi_s \cos \phi_{gc_s}$$

TE2ST

$$(3,3) = -\cos \psi_{ST} \sin \Delta\phi_s \sin \phi_{gc_s} + \cos \psi_{ST} \cos \Delta\phi_s \cos \phi_{gc_s}$$

where: $\Delta\phi_s = (\phi_{gd_s} - \phi_{gc_s})$

θ: longitude
ϕ = latitude
gd: geodetic
gc: geocentric

ψ_{ST} : station azimuth (what is it?)
station azimuth angle WRT north.
see definitions, p.186

STATION XYZ DISPLACEMENTS (from earth center)

$$Xe_{ST} = R_{ST} \cos \theta_{ST} \cos \phi_{gc_s}$$

$$Ye_{ST} = R_{ST} \sin \theta_{ST} \cos \phi_{gc_s}$$

$$Ze_{ST} = -R_{ST} \sin \phi_{gc_s}$$

LOGIC FLOW CHART

RADAR:

Computes azimuth, elevation, and slant range of a vehicle relative to a station.

<table>
<tr><td colspan="4">Compute earth-centered relative position of vehicle and station</td></tr>
<tr><td colspan="4">Transform displacements to station coordinates</td></tr>
<tr><td colspan="4">Compute slant range</td></tr>
<tr><td>NO</td><td colspan="2">Is there a component in the Y-Z plane?</td><td>YES</td></tr>
<tr><td colspan="2">Set: Azimuth to difference of heading angles.
Elevation to 90 degrees</td><td colspan="2" rowspan="3">Compute azimuth and elevation</td></tr>
<tr><td>NO</td><td>Is slant range zero?</td><td>YES</td></tr>
<tr><td></td><td colspan="2">Set: elevation to 0.0</td></tr>
</table>

EQUATIONS EVALUATED

RADAR

EARTH-CENTERED INCREMENTS

$$\Delta Xe = Xe - Xe_{ST}$$
$$\Delta Ye = Ye - Ye_{ST}$$
$$\Delta Ze = Ze - Ze_{ST}$$

STATION INCREMENTS

$$\begin{vmatrix} X_{ST} \\ Y_{ST} \\ Z_{ST} \end{vmatrix} = \begin{vmatrix} T_{ST/E} \end{vmatrix} \begin{vmatrix} \Delta Xe \\ \Delta Ye \\ \Delta Ze \end{vmatrix}$$

$$\begin{vmatrix} (1,1) & (1,2) & (1,3) \\ (2,1) & (2,2) & (2,3) \\ (3,1) & (3,2) & (3,3) \end{vmatrix} \begin{vmatrix} \Delta Xe \\ \Delta Ye \\ \Delta Ze \end{vmatrix}$$

SLANT RANGE

$$R_{ST} = \sqrt{X_{ST}^2 + Y_{ST}^2 + Z_{ST}^2}$$

AZIMUTH

$$AZ = \tan^{-1}(Y_{ST}/Z_{ST}) \qquad \text{if } \sqrt{Y_{ST}^2 + Z_{ST}^2} = 0.0,\ AZ = \sigma_o - \psi_{ST}$$

$$180 + |\tan^{-1} Y_{ST}/Z_{ST}|$$

ELEVATION

$$EL = - \sin^{-1}(X_{ST}/R_{ST}) \qquad \text{if } R_{ST} = 0.0,\ EL = 0.0$$

?

$$(+\sin^{-1} X_{ST}/\ell_{ST})$$

be careful to watch quadrants and signs

THRUST

The contributions of a propulsion system to the force and moment equations of the main program (SIXDOF or PTMASS) are determined by the THRUST routine.

THRUST determines the incremented mass flow ($\dot{m}$), the rate of change of mass flow ($\ddot{m}$), and the thrust force for each component (motor) of the system. The incremental forces are converted into force and moment components with respect to the vehicle body axis system and acting at the body reference point. The incremental-flow items and body components are summed to represent the total propulsion contribution before returning to the main program. The remaining mass is computed from the initial mass and the integrated mass flow.

During code initialization, the routine THRINL determines which of the thrusting, mass flow, or rate of change of mass flow tables are active (input), and passes the information in the form of a flag-array to the THRUST routine for use in requesting the table look-up routine during execution.

LOGIC FLOW CHART

THRINL:

Initializes the thrust vector alignment sines and cosines, computes the six-degree-of-freedom nozzle exit point transfer distance, and sets the active table array flags.

	Do for all motors (NMTR)		
	Compute the sines and cosines of thrust vector rotation and swivel angles (ϕ, ζ)		
NO	Is the main program SIXDOF?		YES
		Do for all motors (NMTR)	
		Compute X, Y, Z distances from reference point to the nozzle exit point	
	Do for all motors (NMTR)		
	CALL ITABLE: set table flag if TVAC table input		
	CALL ITABLE: set table flag if $\dot{m}$ table input		
	CALL ITABLE: set table flag if $\ddot{m}$ table input		

LOGIC FLOW CHART

THRUST:

Evaluates, in the vehicle body-axis system the force and moment components associated with a thrusting system, and computes the remaining mass of the vehicle.

Zero out table look-up flag (INVAL, ITFLAG)			
Zero out thrust force, moment, and mass loss components			
Do for all thrusters (NMTR)			
	NO	Is TVAC table flag set?	YES
		CALL LOOKUP: get vacuum thrust (T_{VAC_i})	
	NO	Is $\dot{m}$ table flag set?	YES
		CALL LOOKUP: get mass flow ($\dot{m}_i$)	
	NO	Is $\ddot{m}$ table flag set?	YES
		CALL LOOKUP: get rate of change of mass flow ($\ddot{m}_i$)	
	Sum the incremental mass rates ($\dot{m}_i$, $\ddot{m}_i$)		
	Compute the exit thrust (T_e)		
	NO	Is the exit thrust less than zero?	YES
		Set exit thrust to 0.0	
	Compute the body X, Y, Z thrust force components		
	Sum the body-force components		
	Compute the body X, Y, Z moment components		
	Sum the body-moment components		
Compute the remaining mass of the vehicle			

EQUATIONS EVALUATED

THRUST:

$$X_e = -(X_N - X_{BP})$$
$$Y_e = Y_N - Y_{BP}$$
$$Z_e = Z_N - Z_{BP}$$

THRUST FORCE COMPONENTS

$$T_X = T_e \cos \xi_T$$
$$T_Y = -T_e \sin \xi_T \cos \phi_T$$
$$T_Z = -T_e \sin \xi_T \sin \phi_T$$

THRUST MOMENT COMPONENTS

$$M_{X_{BP}} = T_Z Y_e - T_Y Z_e$$
$$M_{Y_{BP}} = T_X Z_e - T_Z X_e$$
$$M_{Y_{BP}} = T_Y X_e - T_X Y_e$$

REMAINING MASS

$$m = m_o + \xi_m + \int \dot{m}\, dt$$

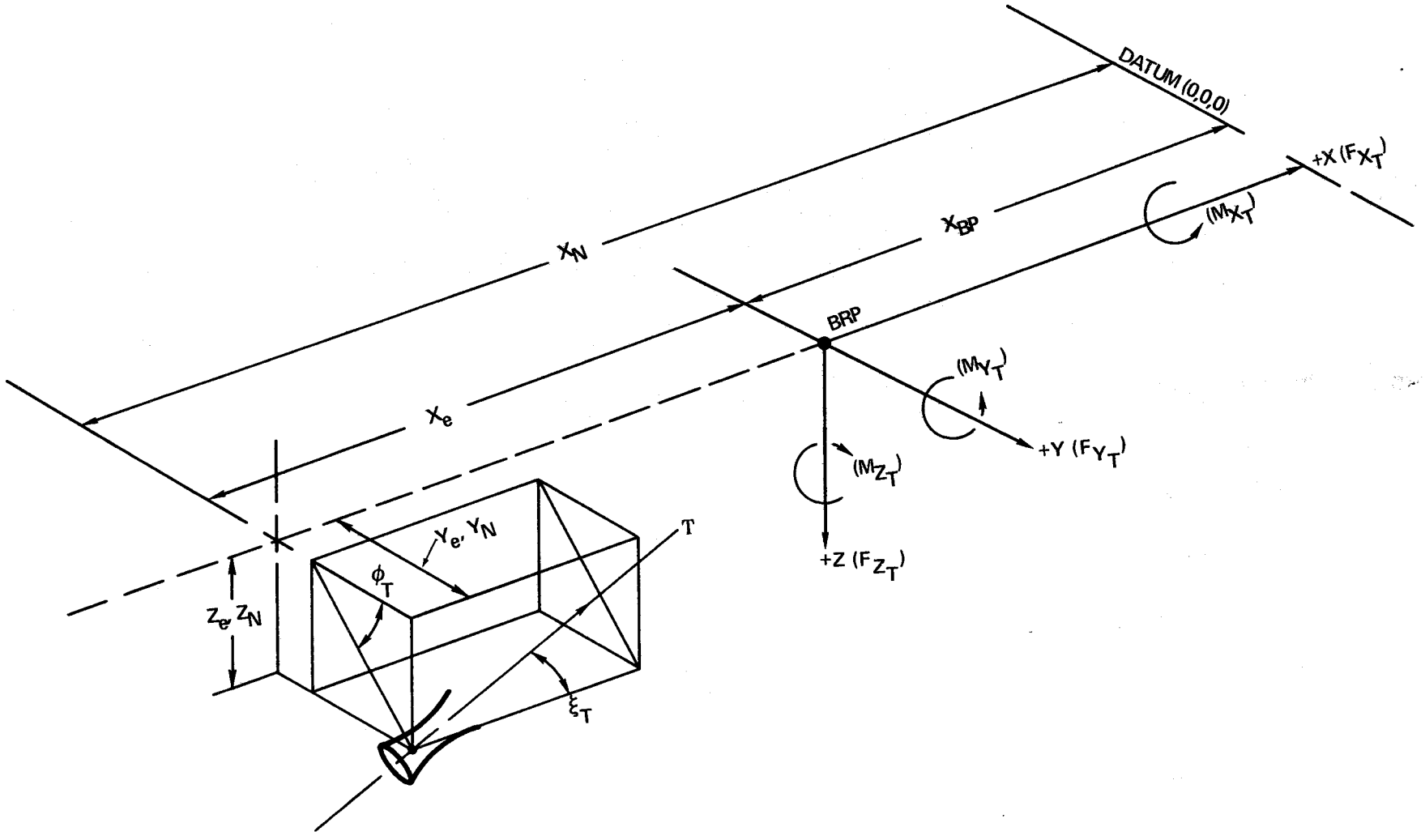


Figure B-2. Thrust Parameters and Axis System

VEHICLE MASS PROPERTIES:

The mass properties required by the rigid-body motion equations being evaluated by the code are described, during input, in tabular format. The routine (VMPINL) is used during initialization to determine whether a particular table is active (input) and, if so, sets a flag.

During execution of the derivative evaluator, the routine VEHMSP is called. It checks the table flag and, if set, calls the LOOKUP routine to determine the mass property value by linear interpolation of the values in the active table.

LOGIC FLOW CHART

VMPINL:

Determines the status of the mass properties tables, sets the table active flags, and calls LOOKUP to determine the mass property values for problem initialization.

	Zero-out table look-up flags (INVAL, ITFLAG)		
	Do for 15 tables		
	NO	CALL ITABLE: Is current table name active?	YES
		Set table active flag (IFLGMP)	
	Do for 3 center-of-mass positions		
	NO	Is table active flag set?	YES
		CALL LOOKUP: get center-of-mass location (X_{cm}, Y_{cm}, Z_{cm})	
	Do for 12 moment of inertia items		
	NO	Is table active flag set?	YES
		CALL LOOKUP: get moment of inertia item	

LOGIC FLOW CHART

VEHMSP:

Determines by linear table lookup the center-of-mass and the moments of inertia values associated with the current mass.

Zero-out the table look-up flags (INVAL, ITFLAG)			
Do for 3 center-of-mass tables			
	NO	Is table active flag set?	YES
		CALL LOOKUP: get center-of-mass location (X_{cm}, Y_{cm}, Z_{cm})	
Do for 12 moment-of-inertia tables			
	NO	Is table active flag set?	YES
		CALL LOOKUP: moment of inertia value (I_{xx}, I_{yy}, I_{zz}, I_{xy}, I_{xz}, I_{yz}, $\dot{I}_{xx}$, $\dot{I}_{yy}$, $\dot{I}_{zz}$, $\dot{I}_{xy}$, $\dot{I}_{xz}$, $\dot{I}_{yz}$)	
Compute distance of center-of-mass w.r.t. body-reference point (X_{cg}, Y_{cg}, Z_{cg})			

EQUATIONS EVALUATED

MASS PROPERTIES:

CENTER-OF-MASS LOCATION w.r.t. BODY REFERENCE POINT

$$X_{cg} = X_{cm} - X_{BRP}$$
$$Y_{cg} = Y_{cm} - Y_{BRP}$$
$$Z_{cg} = Z_{cm} - Z_{BRP}$$

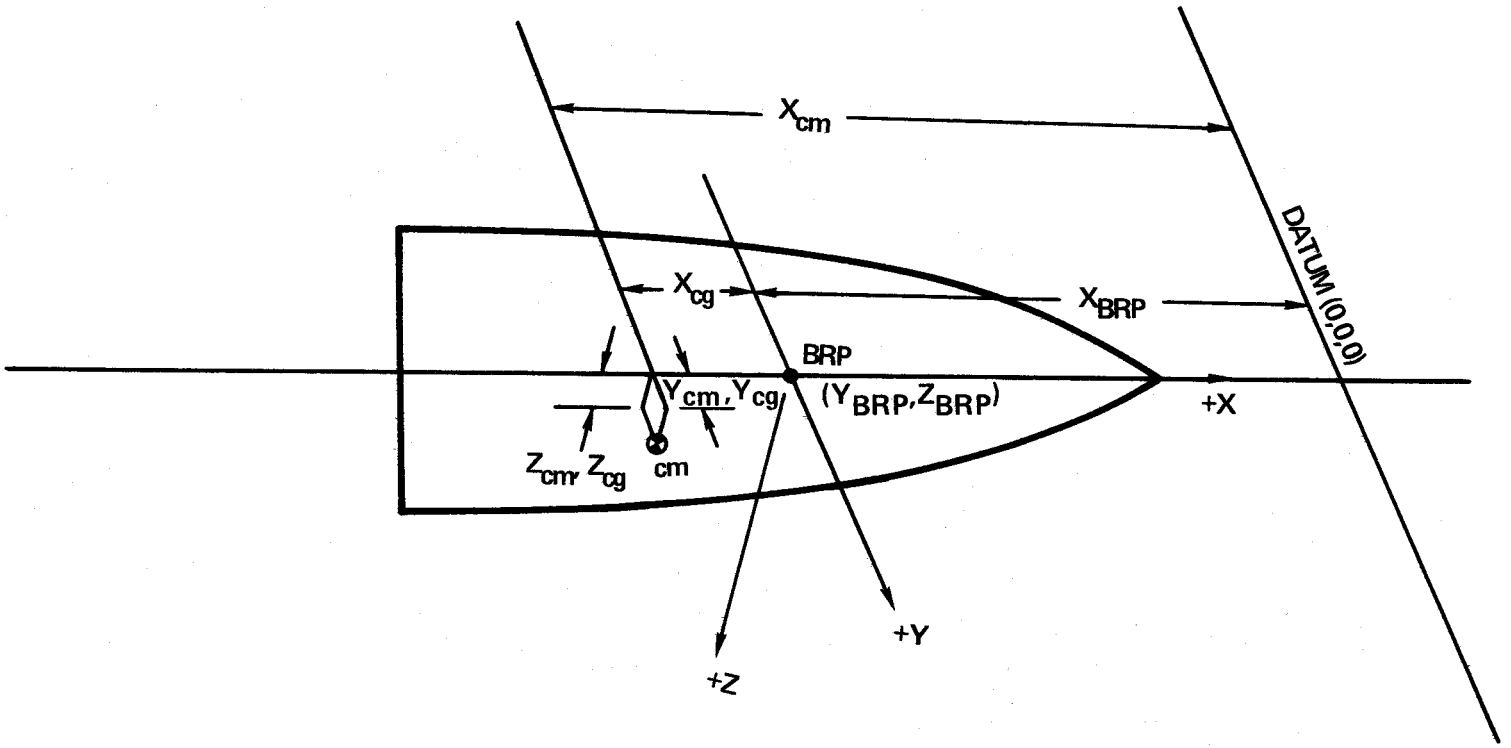


Figure B-3. Vehicle Mass Properties Parameters and Axis System

WIND:

The X Y Z - components of wind velocity referenced to the geodetic axis system are determined by table lookup, and furnished to the problem main program.

LOGIC FLOW CHART

WIND:

Determines the geodetic referenced wind-velocity components by table lookup.

Set the table look-up flags (INVAL, ITFLAG)
CALL LOOKUP: get the south-to-north wind velocity component ($\dot{X}_W$) from table TXWIND
CALL LOOKUP: get the west-to-east wind velocity component ($\dot{Y}_W$) from table TYWIND
CALL LOOKUP: get the up-to-down wind velocity component ($\dot{Z}_W$) from table TZWIND

APPENDIX C

Variables: Program, Output Block, Data File, and Tabular Lists

This Appendix contains a listing of the program variables, the output block variables, the data file (TAPE20) variables, and the variables of the tabular lists.

Contents

Alphabetical List of Program Variable Names With Definitions 183
List of Output Block Variable Names 190
List of TAPE20 File Variable Names 191
Tabular Lists With Variable Names 192

Alphabetical List of Program Variable Names With Definitions

ABSERR	Absolute error of integration
AENOZM	Propulsion motor nozzle exit area
ALPHAD	Angle-of-attack, α
ALPMAX	Maximum angle-of-attack for point mass control
ALPMIN	Minimum angle-of-attack for point mass control
ALPT8D	Total angle-of-attack, α_T
AZSTAD	Vehicle azimuth from station
BETAD	Angle-of-sideslip, β
BETER	Planet flattening factor
BETMAX	Maximum angle-of-sideslip for point mass control
BETMIN	Minimum angle-of-sideslip for point mass control
CLALPH	$\partial C_\ell/\partial\alpha$
CLBDOT	$\partial C_\ell/\partial\dot{\beta}$
CLBETA	$\partial C_\ell/\partial\beta$
CLDELP	$\partial C_\ell/\partial\delta_p$
CLP	$\partial C_\ell/\partial\ (p\ell y/2V_a)$
CLR	$\partial C_\ell/\partial\ (r\ell y/2V_a)$
CLT	Total roll moment coefficient
CLZERO	C_{ℓ_o}
CMADOT	$\partial C_m/\partial\dot{\alpha}$
CMALPH	$\partial C_m/\partial\alpha$
CMALP2	$\partial C_m/\partial\alpha^2$
CMALP3	$\partial C_m/\partial\alpha^3$
CMDELQ	$\partial C_m/\partial\delta_q$
CMP	$\partial C_m/\partial\ (p\ell y/2V_a)$
CMQ	$\partial C_m/\partial\ (q\ell x/2V_a)$
CMT	Total pitching-moment coefficient
CMZERO	C_{m_o}
CNBDOT	$\partial C_n/\partial\dot{\beta}$
CNBETA	$\partial C_n/\partial\beta$
CNBET2	$\partial C_n/\partial\beta^2$
CNBET3	$\partial C_n/\partial\beta^3$
CNDELR	$\partial C_n/\partial\delta_r$
CNR	$\partial C_n/\partial(r\ell y/2V_a)$
CNT	Total yawing moment coefficient
CNZERO	C_{n_o}
CXALH	$\partial C_x/\partial\alpha$
CXALP2	$\partial C_x/\partial\alpha^2$
CXBETA	$\partial C_x/\partial\beta$
CXBET2	$\partial C_x/\partial\beta^2$
CXDELQ	$\partial C_x/\partial\delta_q$
CXDELR	$\partial C_x/\partial\delta_r$
CXT	Total axial force coefficient
CXZERO	C_{x_o}
CYBDOT	$\partial C_y/\partial\dot{\beta}$
CYBETA	$\partial C_y/\partial\beta$
CYBET2	$\partial C_y/\partial\beta^2$
CYBET3	$\partial C_y/\partial\beta^3$

CYDELR	$\partial C_y/\partial \delta_r$
CYP	$\partial C_y/\partial\ (p\ell y/2V_a)$
CYR	$\partial C_y/\partial\ (r\ell y/2V_a)$
CYT	Total yaw force coefficient
CYZERO	C_{y_0}
CZADOT	$\partial C_z/\partial \dot{\alpha}$
CZALPH	$\partial C_z/\partial \alpha$
CZALP2	$\partial C_z/\partial \alpha^2$
CZALP3	$\partial C_z/\partial \alpha^3$
CZDELQ	$\partial C_z/\partial \delta_q$
CZP	$\partial C_z/\partial(pl_z/2V_a)$
CZQ	$\partial C_z/\partial\ (q\ell x/2V_a)$
CZT	Total normal force coefficient
CZZERO	C_{z_0}
DELPD	Roll control deflection (misalignment)
DELQD	Pitch control deflection (misalignment)
DELRD	Yaw control deflection (misalignment)
DELTS	Print time increment
DLAPMC	Point mass control alpha increment
DLBPMC	Point mass control beta increment
DLHPMC	Point mass control altitude increment
DREFXM	Aero coefficient reference length, x-direction
DREFYM	Aero coefficient reference length, y-direction
DREFZM	Aero coefficient reference length, z-direction
DYNP8K	Dynamic pressure
EAMASK	Initial mass adder
ELSTAD	Vehicle elevation from station
EPSPMC	Point mass control error tolerance
FIXXK1	Moment of inertia rate - about X-X
FIXXK	Moment of inertia - about X-X
FIXYK1	Product of inertia rate
FIXYK	Product of inertia
FIXZK1	Product of inertia rate
FIXZK	Product of inertia
FIYYK1	Moment of inertia rate - about Y-Y
FIYYK	Moment of inertia - about Y-Y
FIYZK1	Product of inertia rate
FIYZK	Product of inertia
FIZZK1	Moment of inertia rate - about Z-Z
FIZZK	Moment of inertia - about Z-Z
FK1	Molecular temperature/altitude gradient
FK2	Molecular weight/altitude gradient
FLAM01	Quaternion component 0 - rate
FLAM0	Quaternion component 0
FLAM11	Quaternion component 1 - rate
FLAM1	Quaternion component 1
FLAM21	Quaternion component 2 - rate
FLAM2	Quaternion component 2
FLAM31	Quaternion component 3 - rate
FLAM3	Quaternion component 3
FMACH	Mach number

FMASLK	Total mass loss
FMASL1	Total mass loss rate
FMASL2	Total rate of change of mass loss
FMASSK	Current vehicle mass
FMASZK	Initial vehicle mass
FMB	Mol-weight of air
FMO	S.L. value of Mol-weight
FMU	Geocentric gravity constant
GAMGCD	Geocentric elevation flight path angle
GAMGDD	Geodetic elevation flight path angle
GANPMC	Point mass control gain increment
GCRNGM	Great circle surface range
GPHI	Reference latitude for gravity calculations
GREF	Reference gravity
GXBPM	Gravity component, x-direction
GYBPM	Gravity component, y-direction
GZBPM	Gravity component, z-direction
HB	Geopotential altitude base
HGCSTM	Station altitude, geocentric reference
HGCZM	Initial altitude, geocentric reference
HGC8M	Current altitude, geocentric reference
HGDSTM	Station altitude, geodetic reference
HGDZM	Initial altitude, geodetic reference
HGD8M	Current altitude, geodetic reference
H1	2nd harmonic gravity coefficient
H2	3rd harmonic gravity coefficient
H3	4th harmonic gravity coefficient
IACLSQ	Accelerometer calculations sequence flag
IATMNM	Atmosphere data request
IBODSQ	Body Euler angle rotation sequence
ICASE	Case designator
ICNVRT	English-Metric conversion flag
IDANGL	Flight path/aero angle flag
IDATB	TAPE20 data file
IDATUM	Datum reference point name
IDLAT	Latitude location flag name
IDPQR	Geodetic/geocentric roll rate
IEARTH	Earth shape flag
IMERG	TAPE9 merge data file
IMPROG	Main program designator
INPUT	TAPE5 Input file
IPLOT	TAPE10 Plot file
IPNCH	TAPE7 Punch file
IPRNT	TAPE6 Output file
ISAVE	TAPE8 Save file - restart file
ISPROG	Support program flag name
ITABL	TAPE11 Table storage file
ITBLUF	User table format storage
ITBLUH	User table Header storage
ITBLUN	User table Name storage

ITITLE	Title information storage
IXACL	Accelerometer calculations execution flag
IXAERO	Aerodynamic calculations execution flag
IXAROB	Aerodynamic Body to velocity conversion flag
IXCRFQ	Damped critical frequency execution flag
IXDBUG	Diebug routine execution flag
IXGRAV	Gravity calculations execution flag
IXLODF	Load factor calculations execution flag
IXPTMC	Point mass control flag
IXRNGE	Range calculations flag
IXROLT	Point mass roll table execution flag
IXSTA	Station/RADAR execution flag
IXTBLS	Standard tables execution flag
IXTBLU	User table execution flag
IXTHRS	Propulsion system execution flag
IXTP20	TAPE20 make flag
IXVMP	Vehicle mass properties execution flag
IXWIND	Wind component calculation execution flag
IXYAWS	Yawsonde calculation execution flag
MLAYRS	Number of atmosphere geometric layers
NACLS	Number of accelerometers
NAMSTG	Stage variable name array
NBODY	Number of bodies for problem
NLAYRS	Number of atmosphere geopotential layers
NMTR	Number of thrusting motors
NTABLE	Tabular output table number array
NTSTEP	Number of good steps/print
OMGCYR	Critical frequency-pitch
PB	Atmosphere routine pressure base
PEBPR	Earth-reference roll rate
PHACLD	Accelerometer roll angle
PHBGCD	Roll Euler angle-geocentric reference
PHBGDD	Roll Euler angle-geodetic reference
PHGCZD	Initial geocentric latitude
PHGDSD	Station latitude, geodetic reference
PHGDZD	Initial geodetic latitude
PHIA8D	Aerodynamic roll angle
PHIGCD	Current latitude, geocentric reference
PHIGDD	Current latitude, geodetic reference
PHIID	Roll Euler angle, inertial reference
PHMTRD	Rotation angle of the swivel angle plane
PIBPR1	Angular roll acceleration, inertial reference
PIBPR	Angular roll rate, inertial reference
PRES8P	Altitude ambient pressure
PSACLD	Accelerometer yaw angle
PSBGCD	Yaw Euler angle, geocentric reference
PSBGDD	Yaw Euler angle, geodetic reference
PSIID	Yaw Euler angle, inertial reference
PSISTD	Station azimuth angle w.r.t. North
PSUNGD	Yawsonde sun azimuth angle from North
QEBPR	Earth reference pitch rate

QIBPR1	Angular pitch acceleration inertial reference
QIBPR	Angular pitch rate - inertial reference
REBPR	Earth referenced yaw rate
RELERR	Relative error of integration
REQ8M	Planet equatorial radius
REYNOM	Reynolds number parameter
RHO8K	Altitude ambient density
RIBPR1	Angular yaw acceleration, inertial reference
RNGSTM	Slant-range station to body
RPHI	Earth radius at atmosphere reference latitude
RPOL8M	Planet polar radius
SGSUND	Sun angle w.r.t. vehicle x-axis
SIGGCD	Horizontal flight-path angle, geocentric reference
SIGGDD	Horizontal flight-path angle, geocentric reference
SIGZD	Initial flight-path heading
SPATHM	Surface path length
SREFAM	Aerodynamic reference area
TEMP8K	Altitude ambient temperature
THACLD	Accelerometer pitch angle
THBGCD	Pitch Euler angle, geocentric reference
THBGDD	Pitch Euler angle, geodetic reference
THTID	Pitch Euler angle, inertial reference
THTLD	Current local longitude
THTLZD	Initial longitude
THTSTD	Station longitude
TIMEZS	Problem start time zero
TIME	Current program time
TIMSDX	Time increment to start of stage
TMAXS	Maximum program execution time
TMB	Molecular scale temperature
TPATHM	Trajectory path length
TPATH1	Trajectory path length rate of change
TSTAGE	Time within the stage
TSUNGD	Yawsonde sun elevation angle, geodetic reference
UIBPR1	Body x-component of inertial acceleration
UIBPR	Body x-component of inertial velocity
USTA8M	Vehicle velocity station x-direction
VALSTG	Stage testing values array
VA88M	Vehicle velocity w.r.t. air
VIBPM1	Body y-component of inertial acceleration
VIBPM	Body y-component of inertial velocity
VSTA8M	Vehicle velocity station y-direction
VS88M	Speed of sound at altitude
VTE8M	Total velocity - earth reference
VTGCM	Total velocity - geocentric reference
VTGDM	Total velocity - geodetic reference
WIBPM1	Body z-component of inertial acceleration
WIBPM	Body z-component of inertial velocity
WSTA8M	Vehicle velocity station z-direction
XACL8M	Accelerometer location x-direction

XBRP8M	Body reference point x-distance from datum
XCAS	X-axis sensitivity factor
XCG8M1	BRP/c.g. separation velocity, x-direction
XCG8M2	BRP/c.g. separation acceleration, x-direction
XCG8M	BRP/c.g. distance, x-direction
XCMRFM	Center-of-mass reference location, x-direction
XDTM8M	X-direction datum reference
XESTAM	Earth-referenced displacement, station x-direction
XETHM1	Nozzle axis, x-direction rate of change
XE88M1	Earth-referenced velocity, x-direction
XE88M2	Earth-referenced acceleration, x-direction
XE88M	Earth-referenced displacement, x-direction
XFARON	Aerodynamic force - x-direction
XFTHRN	Thrust force, x-direction
XGC8M1	Velocity component, geocentric x-direction
XGD8M1	Velocity component, geodetic x-direction
XIBPM1	Velocity component, inertial x-direction
XIBPM	Displacement, inertial x-direction
XIMTRD	Thrust swivel angle
XLODFG	Body axial load factor
XMARON	Aerodynamic moment, about x-x axis
XMTHRN	Thrust moment, about x-x axis
XNOZEM	Nozzle exit distance, x-direction
XRNGDM	Down range distance, geodetic reference
XWGDM1	Wind component, geodetic x-direction
YACL8M	Accelerometer location - y-direction
YBRP8M	Body reference point y direction from datum
YCAS	Y-axis sensitivity factor
YCG8M1	BRP/c.g. separation velocity, y-direction
YCG8M	BRP/c.g. distance, y-direction
YCMRFM	Center-of-mass reference location, y-direction
YDTM8M	Y-direction datum reference
YESTAM	Earth reference displacement, station y-direction
YETHM1	Nozzle exit, y-direction rate of change
YE88M1	Earth-referenced velocity, y-direction
YE88M2	Earth-referenced acceleration, y-direction
YE88M	Earth-referenced displacement, y-direction
YFARON	Aerodynamic force, y-direction
YFTHRN	Thrust force, y-direction
YCG8M1	Velocity component, geocentric y-direction
YGD8M1	Velocity component, geodetic y-direction
YIBPM1	Velocity component, inertial y-direction
YIBPM	Displacement, inertial y-direction
YLODFG	Body yaw load factor
YMARON	Aerodynamic moment, about y-y
YMTHRN	Thrust moment, about y-y
YNOZEM	Nozzle exit distance, y-direction
YRNGDM	Cross range distance, geodetic reference
YWGDM1	Wind component, geodetic x-direction
ZACL8M	Accelerometer location, z-direction

ZBRP8M	Body reference point z-direction from datum
ZB	Atmosphere geometric altitude base
ZCAS	Z-axis sensitivity factor
ZCG8M1	BRP/c.g. separation velocity, z-direction
ZCG8M2	BRP/c.g. separation acceleration, z-direction
ZCG8M	BRP/c.g. distance, z-direction
ZCMRFM	Center-of-mass reference location, z-direction
ZDTM8M	Z-direction datum reference
ZESTAM	Earth reference displacement, station z-direction
ZETHM1	Nozzle exit, z-direction rate of change
ZE88M1	Earth-referenced velocity, z-direction
ZE88M2	Earth-referenced acceleration, z-direction
ZE88M	Earth-referenced displacement, z-direction
ZFARON	Aerodynamic force, z-direction
ZFTHRN	Thrust force, z-direction
ZGC8M1	Velocity component, geocentric, z-direction
ZGD8M1	Velocity component, geodetic, z-direction
ZIBPM1	Velocity component, inertial, z-direction
ZIBPM	Displacement, inertial, z-direction
ZLODFG	Body normal load factor
ZMARON	Aerodynamic moment, about z-z
ZMAXM	Maximum altitude for atmosphere data
ZMTHRN	Thrust moment, about z-z
ZNOZEM	Nozzle exit distance, z-direction
ZWGDM1	Wind component, geodetic z-direction

The OUTPUT Block Variable Names

At each time history print interval during execution, blocks of data values computed by the current main program and the activated support routines are listed. The following lists are the variable name that correspond to these values.

A, Six-degree-of-freedom

1 TIME	11 ALPHAD	21 YGD8M1	31 UIBPR	41 PSIID
2 TSTAGE	12 BETAD	22 ZGD8M1	32 VIBPR	42 PHIID
3 HGD8M	13 ALPT8D	23 XGC8M1	33 WIBPR	43 blank
4 PHIGDD	14 FMACH	24 YGC8M1	34 XIBPM	44 blank
5 THTLD	15 PHIA8D	25 ZGC8M1	35 YIBPM	45 blank
6 PHBGDD	16 REYNOM	26 FMASSK	36 ZIBPM	46 blank
7 PSBGDD	17 VA88M	27 DYNP8K	37 XIBPM1	47 blank
8 THBGDD	18 VTGDM	28 PIBPR	38 YIBPM1	48 blank
9 GAMGDD	19 VS88M	29 QIBPR	39 ZIBPM1	49 blank
10 SIGGDD	20 XGD8M1	30 RIBPR	40 THTID	50 blank

B, Point Mass

1 TIME	11 BETAD	21 YGD8M1	31 XE88M1
2 TSTAGE	12 ALPT8D	22 ZGD8M1	32 YE88M1
3 HGD8M	13 FMACH	23 XGC8M1	33 ZE88M1
4 PHIGDD	14 PHIA8D	24 YGC8M1	34 blank
5 THTLD	15 REYNOM	25 ZGC8M1	35 blank
6 GAMGCD	16 VA88M	26 FMASSK	36 blank
7 SIGGCD	17 VTGDM	27 DYNP8K	37 blank
8 GAMGDD	18 VTGCM	28 XE88M	38 blank
9 SIGGDD	19 VS88M	29 YE88M	39 blank
10 ALPHAD	20 XGD8M1	30 ZE88M	40 blank

C, Auxillary Calculations Associated With The Main Program

1 Range items: XRNG8M, YRNG8M, GCRNGM, SPATHM, TPATHM
2 Load Factors: XLODFG, YLODFG, ZLODFG
3 Critical Frequency: OMGCYR
4 Yawsonde: SGSUND

D, Support Routines

1 Aerodynamics: CXT, CYT, CZT, CLT, CMT, CNT
2 Station/RADAR: AZSTAD, ELSTAD, RNGSTM, USTA8M, VSTA8M, WSTA8M
3 Thrust: FMASL1, FMASL2, XFTHRN, YFTHRN, ZFTHRN, XMTHRM, YMTHRM, ZMTHRM
4 Vehicle Mass Properties: XCG8M, YCG8M, ZCG8M, FIXXK, FIYYK, FIZZK, FIXYK, FIXZK, FIXZK, FIXXK1, FIYYK1, FIZZK1, FIXYK1, FIXZK1, FIYZK1

The TAPE20 File Block Variable Names

At each time-history print interval during execution, a block of data values are dumped to file TAPE20. The names associated with the values are given below in the order of storage.

1 TIME	21 XGC8M1	41 RHO8K	61 AZSTAD	81 XCG8M
2 HGD8M	22 YGC8M1	42 TEMP8K	62 ELSTAD	82 YCG8M
3 PHIGDD	23 ZGC8M1	43 VS88M	63 RNGSTM	83 ZCG8M
4 THTLD	24 FMASSK	44 DYNP8K	64 USTA8M	84 FIXXK
5 PHBGDD	25 PIBPR	45 XRNGDM	65 VSTA8M	85 FIYYK
6 PSBGDD	26 QIBPR	46 YRNGDM	66 WSTA8M	86 FIZZK
7 THTGDD	27 RIBPR	47 GCRNGM	67 FMASL1	87 FIXYK
8 GAMGDD	28 UIBPM	48 SPATHM	68 FMASL2	88 FIXZK
9 SIGGDD	29 VIBPM	49 TPATHM	69 XFTHRN	89 FIYZK
10 ALPHAD	30 WIBPM	50 XLODFG	70 YFTHRN	90 FIXXK1
11 BETAD	31 XIBPM	51 YLODFG	71 ZFTHRN	91 FIYYK1
12 ALPT8D	32 YIBPM	52 ZLODFG	72 XMTHRN	92 FIZZK1
13 FMACH	33 ZIBPM	53 GAMGCD	73 YMTHRN	93 FIXYK1
14 PHIA8D	34 XIBPM1	54 SIGGCD	74 ZMTHRN	94 FIXZK1
15 REYNOM	35 YIBPM1	55 XE88M	75 CXT	95 FIYZK1
16 VA88M	36 ZIBPM1	56 YE88M	76 CYT	96 blank
17 VTGDM	37 THTID	57 ZE88M	77 CZT	97 blank
18 XGD8M1	38 PSIID	58 XE88M1	78 CLT	98 blank
19 YGD8M1	39 PHIID	59 YE88M1	79 CMT	99 blank
20 ZGD8M1	40 PRES8P	60 ZE88M1	80 CNT	100 blank

Tabular Lists With Variable Names

TABLE NO. 1 - Trajectory Summary

Quantity	Units
Time (TIME)	s
Altitude (HGD8M)	m
Down range (XRNGDM)	m
Cross range (YRNGDM)	m
Total velocity (VTGDM)	m/s
Mach number (FMACH)	
Longitude (THTLD)	deg (°)
Latitude, geodetic (PHIGDD)	deg (°)
Dynamic pressure (DYNP8K)	kPa
Axial load (XLODFG)	g's
Elevation flight-path angle (GAMGDD)	deg (°)
Azimuth flight-path angle (SIGGDD)	deg (°)
Weight (mass), (FMASSK)	kg

TABLE NO. 2 - Stability

Quantity	Units
Time (TIME)	s
Roll rate (PIBPR)	rad/s
Pitch rate (QIBPR)	rad/s
Yaw rate (RIBPR)	rad/s
Critical pitch frequency (OMGCYR)	rad/s
Critical yaw frequency (OMGCZR)	rad/s
Static margin, y	m
Static margin, z	m
Body Euler angle, pitch (THBGDD)	deg (°)
Body Euler angle, yaw (PSBGDD)	deg (°)
Body Euler angle, roll (PHBGDD)	deg (°)
Angle-of-attack (ALPHAD)	deg (°)
Angle-of-sideslip (BETAD)	deg (°)
Total angle-of-attack (ALPT8D)	deg (°)

TABLE NO. 3 - Air Properties and Loads

Quantity	Units
Time (TIME)	s
Altitude (HGD8M)	m
Ambient Pressure (PRES8P)	Pa
Dynamic Pressure (DYNP8K)	kPa
Ambient Temperature (TEMP8K)	K
Velocity in Air (VA88M)	m/s
Mach number (FMACH)	
Unit Reynolds number (REYNOM)	m^{-1}
Thrust force (XFTHRN)	N
Axial force (XFARON)	N
Normal force (ZFARON)	N
Side force (YFARON)	N
Weight (mass) (FMASSK)	kg
Elevation flight-path angle (GAMGDD)	deg (°)
Total angle of attack (ALPT8D	deg (°)

TABLE NO. 4 - Mass Properties and Load Factors

Quantity	Units
Time (TIME)	s
Weight (mass) (FMASSK)	kg
X - c.m. position (XCMRFM)	m
Y - c.m. position (YCMRFM)	m
Z - c.m. position (ZCMRFM)	m
X - Moment of inertia (FIXXK)	kg m^2
Y - Moment of inertia (FIYYK)	kg m^2
Z - Moment of inertia (FIZZK)	kg m^2
Axial load factor (XLODFG)	g's
Side load factor (YLODFG)	g's
Normal load factor (ZLODFG)	g's

TABLE NO. 5 - Trajectory - Tangent Plane

Quantity	Units
Time (TIME)	s
X	m
Y	m
Z	m
X RATE	m/s
Y RATE	m/s
Z RATE	m/s
Velocity	m/s
Altitude	m
Mach (FMACH)	
Dynamic pressure (DYNP8K)	Pa
Axial load (XLODFG)	g's
Elevation flight-path angle (GAMGDD)	deg (°)

TABLE NO. 6 - Station/RADAR Look Values

Quantity	Units
Time (TIME)	s
Azmiuth, station (AZSTAD)	deg (°)
Elevation station (ELSTAD)	deg (°)
Slant range (RNGSTM)	m
Range rate (RNGST1)	m/s

TABLE NO. 7 - Accelerations and Accelerometer Values

Quantity	Units
Time (TIME)	s
X-direction accelerations (AXX)	m/s^2
Y-direction accelerations (AYY)	m/s^2
Z-direction accelerations (AZZ)	m/s^2
Accelerometer No 1 - X Value	m/s^2
Accelerometer No 1 - Y Value	m/s^2
Accelerometer No 1 - Z Value	m/s^2

DISTRIBUTION:

ACUREX Corporation
Aerotherm Group
485 Clyde Ave
Mountain View, CA 94042
Attn: C. Nardo

Aero Jet Services Co.
P.O. Box 13222
Sacramento, CA 95813
Attn: D. Q. Matejka
Dept 3530/Bldg 2006

The Aerospace Corporation
Aerophysics Laboratory
Laboratory Operations
El Segundo, CA 90245
Attn: D. H. Platus

Commander (2)
US Army Armament Res and Development Com
DRDAR-LCA-F
Dover, NJ 07801
Attn: A. Loeb
D. Mertz

Director
USA ARRADCOM
Ballistic Research Laboratory
Aberdeen Proving Ground, MD 21005
Attn: C. Murphy
DRDAR-BU

General Electric Company (2)
Reentry and Environmental Systems Division
3198 Chestnut Street
Philadelphia, PA 19101
Attn: D. Nestler
J. Pettus

KAMAN Sciences Corporation
1700 Garden of the Gods Road
Colorado Springs, CO 80907
Attn: F. Barbara

Lockheed Missiles and Space Center
P.O. Box 504
Sunnyvale, CA 94086
Attn: J. P. Reding

Martin Marietta Aerospace
Orlando Division
P.O. Box 5837
Orlando, FL 32805
Attn: R. C. Ausec

McDonnel Douglas Astronautics Co.
5301 Bolsa Avenue
Huntington Beach, CA 92647
Attn: J. S. Murphy

NASA Langley Research Center
M/S 355
Hampton, VA 23665
Attn: J. Chambers

NASA Ames Research Center
M/S 227-8
Moffett Field, CA 94035
Attn: G. N. Malcolm

Naval Surface Weapons Center
Dahlgren, VA 22448
Attn: M. T. Dowling, K11

Naval Weapons Center
China Lake, CA 93555
Attn: R. Van Aken, Code 3203

Naval Surface Weapons Center
White Oak Laboratory
Silver Springs, MD 20910
Attn: W. C. Lyons

TRW, Defense and Space Systems Group
Ballistic Missile Division
P.O. Box 1310
San Bernardino, CA 92402
Attn: B. Hibbard, 526/611

US Army Armament Res and Devel Command
Aerodynamics Research and Concepts
Assistance Section (ARCAS)
DRDAR-CLB-PA
Aberdeen Proving Ground, MD 21010
Attn: M. Miller

USAF Armament Laboratory (3)
Eglin AFB, FL 32542
Attn: J. Hooten AFATL/DLYW
R. Greene
W. Martin

USAF Flight Dynamics Laboratory (SFSC)
Wright Patterson AFB, OH 45433
Attn: K. F. Stetson

USAF Weapons Laboratory
Kirtland AFB, NM 87115
Attn: I. E. Baldwin, AFWC/NTSSW

DISTRIBUTION: (cont)

1170 S. A. Moore
Attn: R. J. Orzel, 1172
1250 R. G. Clem
Attn: G. W. Stone, 1252
1252 G. G. Wilson
1421 W. J. Denison
Attn: R. G. Illing
1500 W. A. Gardner
Attn: T. J. Hoban, 1520
G. L. West, 1522
R. L. Eno, 1525
1580 T. S. Church
Attn: W. V. Hereford, 1582
2330 E. H. Barsis
Attn: G. M. Heck, 2334
C. R. Blaine, 2336
2610 D. C. Jones
Attn: A. R. Iacoletti, 2614
R. E. Jones, 2614
2640 J. L. Tischhauser
Attn: M. R. Scott
H. A. Watts, 2646
4300 R. L. Peurifoy, Jr.
Attn: C. C. Burks, 4310
G. J. Hildebrandt, 4320
E. E. Ives, 4330
H. W. Schmitt, 4340
4360 J. A. Hood
Attn: D. W. Doak, 4363
4741 J. R. Kelsey
5000 J. K. Galt
5500 O. E. Jones
Attn: W. Herrmann, 5530
T. B. Lane, 5520
5510 D. B. Hayes
Attn: W. D. Sundberg, 5511
5600 D. B. Shuster
Attn: G. J. Simmons, 5640
5610 A. A. Lieber
Attn: M. K. Taylor, 5613
5620 M. M. Newsom
Attn: C. E. Dalton, 5621
J. R. Phelan, 5621
5622 R. D. Andreas
Attn: R. D. Tucker
5630 R. C. Maydew
5631 H. R. Vaughn
5631 D. W. Barnette
5631 M. D. Bennett
5631 G. R. Eisler
5631 A. E. Hodapp
5631 R. A. La Farge
5631 W. L. Oberkampf
5632 C. W. Peterson, Jr.
5632 B. K. Baca
5632 I. T. Holt
5632 D. W. Johnson
5632 W. B. Pepper
5633 S. McAlees, Jr.
5634 D. D. McBride
Attn: E. L. Clark
5635 W. R. Barton
5635 A. B. Cox
5635 L. R. Rollstin
5635 H. A. Wente
5635 W. E. Williamson
5636 J. K. Cole
5636 W. H. Curry
5636 R. E. French
5636 D. E. Larson
5636 A. E. McIntyre
5636 E. J. Meyer (15)
5636 H. R. Spahr
5650 D. J. Rigali
8100 D. M. Olson
Attn: J. L. Wirth, 8150
R. D. Cozine, 8160
W. E. Alzheimer, 8120
8110 J. F. Barham, Jr.
Attn: D. J. Bohrer, 8111
H. M. Witek, 8116
W. R. Bolton, 8116
8151 L. A. Hiles
8152 G. N. Beeler
Attn: R. N. Everett
8320 R. L. Rinne
Attn: G. E. Strandin, 8324
8330 G. W. Anderson
Attn: R. E. Huddleston, 8332
J. Mansfield, 8332
8411 D. Benton
Attn: R. E. Humphrey
8214 M. A. Pound
3141 L. J. Erickson (5)
3151 W. L. Garner (3)
For DOE/TIC (Unlimited Release)
DOE/TIC (25)
(C. H. Dalin, 3154-3)