NASA CONTRACTOR REPORT

NASA CR-2770

NASA CR-2770

(NASA-CR-2770) CAPABILITIES AND APPLICATIONS OF THE PROGRAM TO OPTIMIZE SIMULATED TRAJECTORIES (POST). PROGRAM SUMMARY DOCUMENT (Martin Marietta Corp.) 132 p HC A07/MF A01 CSCL 09B H1/61

N77-19776

Unclas 22460

CAPABILITIES AND APPLICATIONS OF THE PROGRAM TO OPTIMIZE SIMULATED TRAJECTORIES (POST)

Program Summary Document

G. L. Brauer, D. E. Cornick, and R. Stevenson

Prepared by
MARTIN MARIETTA CORPORATION
Denver, Colo. 80201
for Langley Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • FEBRUARY 1977

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-2770		

4. Title and Subtitle	5. Report Date
Capabilities and Applications of the Program to Optimize Simulated Trajectories (POST) Program Summary Document	February 1977
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
G. L. Brauer, D. E. Cornick, and R. Stevenson	
9. Performing Organization Name and Address	10. Work Unit No.
Martin Marietta Corporation P.O. Box 179 Denver, CO 80201	506-26-10-10 506-26-10-08
	11. Contract or Grant No.
	NAS1-13611
12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
National Aeronautics and Space Administration Washington, D.C. 20546	Contractor Report
	14. Sponsoring Agency Code

15. Supplementary Notes

Langley Technical Monitor: John J. Rehder
Final Report

16. Abstract

This report summarizes the capabilities and applications of the three-degree-of-freedom (3DOF) version and the six-degree-of-freedom (6DOF) version of the Program to Optimize Simulated Trajectories (POST). The document supplements the detailed program manuals by providing additional information that motivates and clarifies basic capabilities, input procedures, applications and computer requirements of these programs. The information will enable prospective users to evaluate the programs, and to determine if they are applicable to their problems. The report provides enough information to enable managerial personnel to evaluate the capabilities of the programs and describes the POST structure, formulation, input and output procedures, sample cases, and computer requirements. It also provides answers to basic questions concerning planet and vehicle modeling, simulation accuracy, optimization capabilities, and general input rules. Several sample cases are also presented. These sample cases contain enough detail to enable them to serve as guidelines for new users.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Trajectory Optimization Projected Gradient Star Category 61 Computer Programming and Software	Unclassified-Unlimited Subject Category 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	131	$5.75

* For sale by the National Technical Information Service, Springfield, Virginia 22161

CONTENTS

FIGURES

TABLES

CAPABILITIES AND APPLICATIONS OF THE PROGRAM TO OPTIMIZE SIMULATED TRAJECTORIES (POST) PROGRAM SUMMARY DOCUMENT

By G. L. Brauer, D. E. Cornick, and R. Stevenson

Martin Marietta Corporation
P. O. Box 179
Denver, Colorado 80201

SUMMARY

This report summarizes the capabilities and applications of the three-degree-of-freedom (3DOF) version and the six-degree-of-freedom (6DOF) version of the Program to Optimize Simulated Trajectories (POST). The document supplements the detailed program manuals (ref. 1, 2, and 3) by providing additional information that motivates and clarifies basic capabilities, input procedures, applications and computer requirements of these programs. The information will enable prospective users to evaluate the programs, and to determine if they are applicable to their problems.

The report is presented in five chapters with the first containing a summary of what is believed to be the important features of both programs. Chapter 1 is intended to provide enough information to enable managerial personnel to understand the capabilities of the programs. The remaining chapters are provided to describe the POST structure, formulation, input and output procedures, sample cases, and computer requirements. These chapters provide answers to basic questions concerning planet and vehicle modeling, simulation accuracy, optimization capabilities, and general input rules. Several sample cases are also presented. These sample cases contain enough detail to enable them to serve as guidelines for new users. Should more detailed questions arise, it is recommended that the POST Formulation Manual, Utilization Manual, and Programmer's Manual be consulted.

INTRODUCTION

The original 3D version of POST was developed in 1970 as a Space Shuttle Trajectory Optimization Program. Since that time, the program has been significantly improved with additional capabilities added in the areas of vehicle modeling, trajectory simulation, and targeting and optimization. The program is capable of simulating and optimizing trajectories for a wide variety of aerospace vehicles operating in the vicinity of a single planetary body.

The popularity of 3D POST led to the development of the 6DOF version. 6D POST is identical in design and use as its 3DOF counterpart. The principal feature of 6D POST, in comparison to other 6DOF programs, is it's easy to use input procedure. This capability was obtained by the development of a special

NAMELIST input processor that does not contain the namelist size limitation of the standard NAMELIST routine.

During the development process, considerable effort was made to ensure that both versions were generalized in capability, yet easy to learn and use. As a result, 3D and 6D POST can be readily used by trajectory engineers without specialized training in areas such as optimization theory. Ease of use and the ability to be used on almost any kind of near-Earth trajectory problem has resulted in considerable interest in these programs throughout the industry. This interest has resulted in numerous questions being asked by potential users concerning the general capabilities of both programs. Thus, the purpose of this summary report is to answer these kinds of questions without requiring reference to the detailed program manuals. As a result, this report contains a broad spectrum of information related to program features, structure and design, utilization, sample cases, and computer requirements. These data will provide the potential user with basic capability information, and the experienced user with a summary for quick reference purposes.

The instruction manuals and source tapes for both 3D and 6D POST are available from:

Computer Software Management & Information Center
112 Barrow Hall
University of Georgia
Athens, Georgia

PROGRAM FEATURES

The 3D POST and 6D POST are general purpose FORTRAN codes designed for flexible 3D and 6D simulation and optimization of trajectories for aerospace vehicles. A summary of the key program features is presented in Figure 1.

In reviewing the program features, it is important to realize that 3D POST and 6D POST are separate programs. However, the executive structure and I/O characteristics are identical. The only significant difference between these two programs is that the rotational equations of motion are included in 6D POST as depicted in Figure 2. It is important to note, however, that the 6D version requires an additional 44 131 octal storage locations over its 3D counterpart.

Simulation Capabilities

POST is best described as an N-phase trajectory simulation program. This means that the POST input processor and executive structure enables the user to simulate the trajectory by a logical sequence of trajectory phases. In each phase, physical and nonphysical aspects of the simulation can be modeled to any accuracy deemed appropriate by the trajectory analyst. In this manner, each phase of the trajectory can be simulated accurately and efficiently by appropriate user input and program option selection. Figure 3 illustrates the relationship between trajectory phases, events, and POST input data structure. In

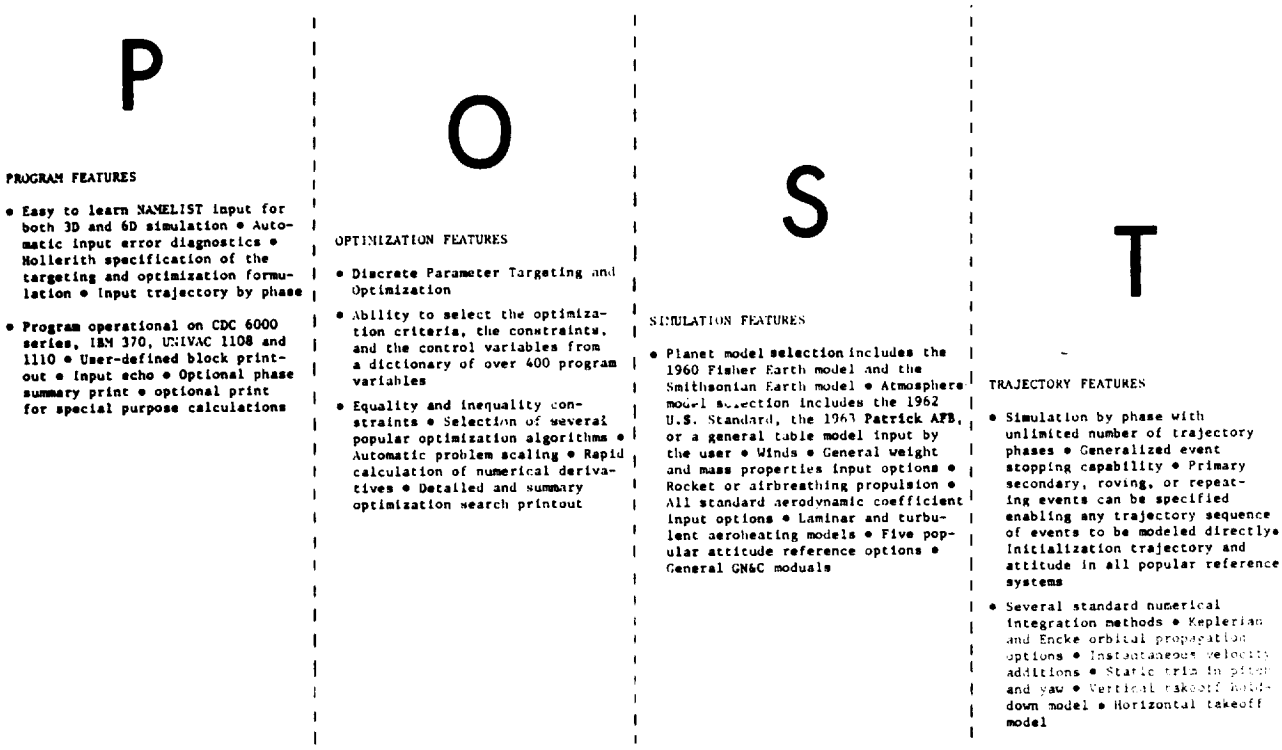


Figure 1.- Summary of Program Features

POST, the data structure is arranged according to the sequence of events defining the trajectory phases. It is noticed that every phase is specified by an event that defines the initiation of the phase. Thus, each event (other than the first and last events) serves two purposes: (1) it ends the previous phase, and (2) it starts the subsequent phase. The input data cards for each phase are located between the two events that define the phase. This basic input rule is also illustrated in Figure 3, where any or all of the simulated data can be changed in any phase. The data used in the simulation of a specific phase are the sum of the data input in all previous phases plus any new additional data and/or options. If the added input data correspond exactly to any previous input data, then the program will use the latter data in the trajectory simulation. There are no restrictions on the number of events in a given problem, and the event criteria can be selected as any variable computed in the program. The capability to define the problem by phases is an important feature of these programs because it enables complex problems to be formulated in a step-by-step fashion.

The simulation capabilities can be categorized in three types: (1) the planet model, (2) the vehicle model, (3) trajectory simulation options. Summaries of these capabilities are presented in Tables 1, 2, and 3, respectively. Generally each particular simulation model has several options available. However, if a particular model is not available automatically then general models can be used augmented by user input data. For example, two commonly used atmosphere models are available that require no user input. These are the 1962 U.S. Standard and the 1963 Patrick AFB atmosphere models. However, in many cases, such as a Mars entry study, the user can and must input his own model via generalized input of pressure, density, and temperature (or speed of sound). The detailed procedures for this type of input are described in reference 1.

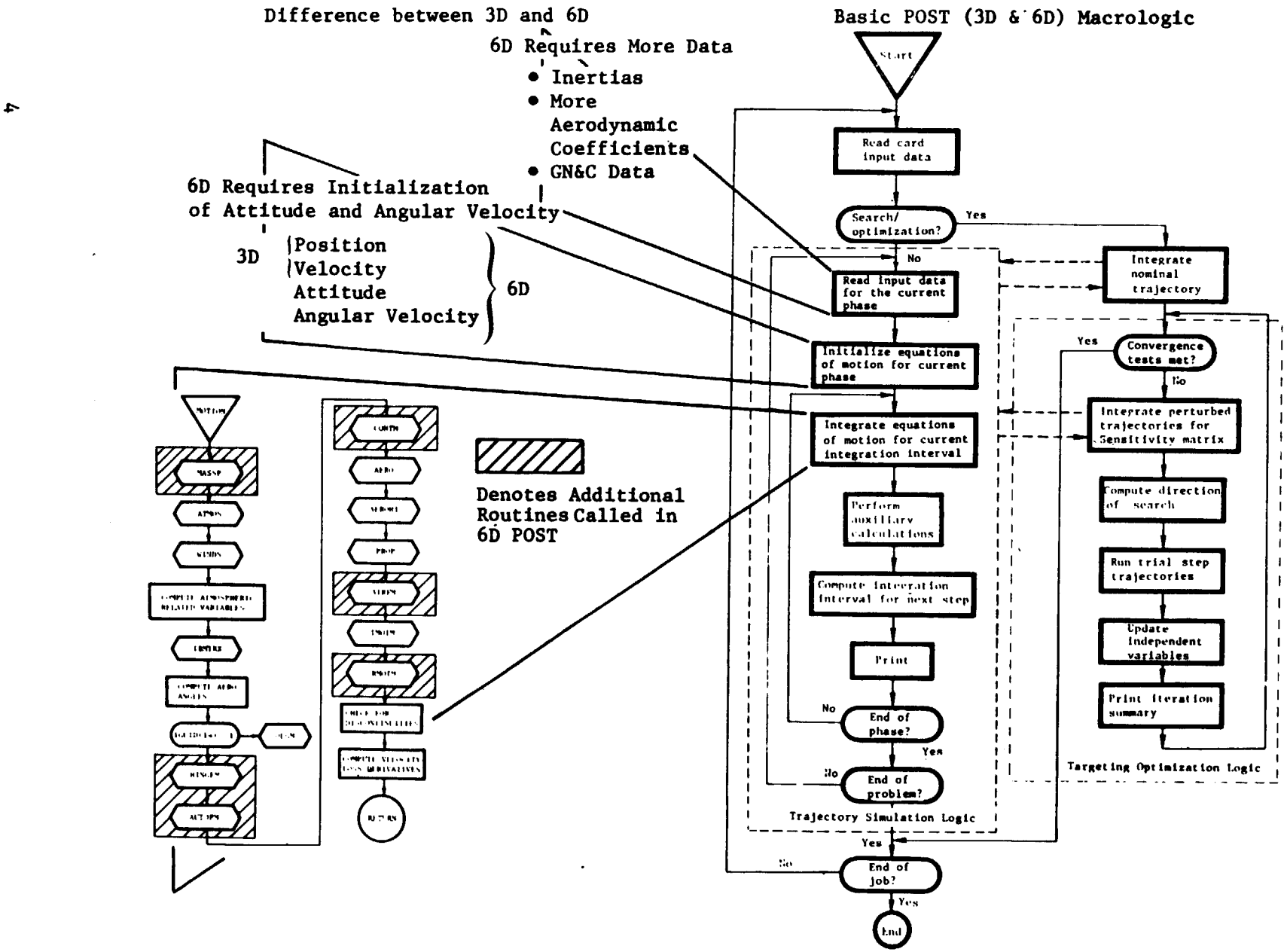
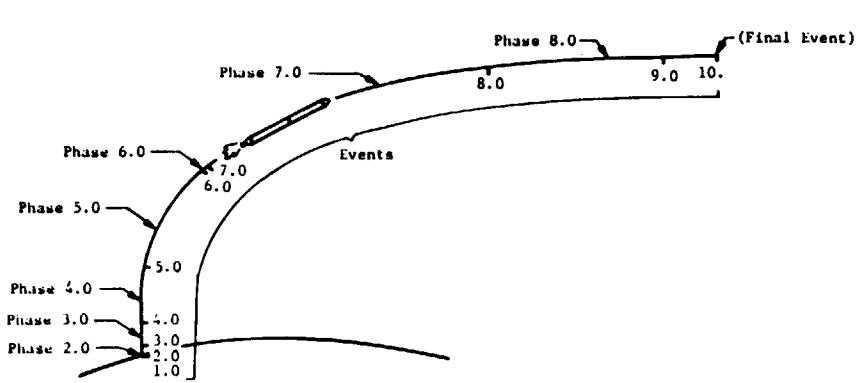


Figure 2.- Relationship Between 3D and 6D POST



Phase No.	Description
1.0	Hold for Thrust Buildup
2.0	Vertical Rise
3.0	Initial Pitchover Maneuver
4.0	Fly g-Turn to Minimize Loads
5.0	Fly Pitch Program After Maximum q
6.0	Delay for Tank Separation
7.0	Linear Sine Steering
8.0	Fly Accel. Limit Until Engine Shutdown
9.0	Coast before Payload Deployment

Event No.	Description
1.0	Ignite Engines @ t = -3s
2.0	Release Vehicle @ T/W = 1.25
3.0	Start Pitch Program @ V_R = 150 fps
4.0	Start Gravity Turn @ α = 0°
5.0	Start Pitch Rate @ q = 300 psf
6.0	Stage Droptank @ V_R = 6500 fps
7.0	Start Pitch Program II 5s After Staging
8.0	Throttle Main Engine @ a/g = 3.0
9.0	Shutoff Engines @ W_{Prop} = 0.0 lb
10.0	End Simulation 1 min. After Engine Shutdown

Phase 1.0
Phase 2.0
Phase 3.0
Phase 4.0
Phase 5.0
Phase 6.0
Phase 7.0
Phase 8.0
Phase 9.0

```
PSEARCH
C  INPUT IN THIS NAMELIST ALL DATA PERTAINING TO :
C
C                  1) OPTIMIZATION AND TARGETING FORMULATION
C                        OPTIMIZATION VARIABLE
C                        CONSTRAINTS AND TARGET CONDITIONS
C                        CONTROL VARIABLE
C                  2) ALGORITHM SELECTION AND CONTROL FLAGS
C
 $END
PSEGNDAT EVENT     = 1.
C  INPUT IN THIS NAMELIST ALL DATA AND PROGRAM CONTROL FLAGS PERTAINING
C  TO:
C
C                  1) PLANET MODEL  - OBLATE SPHEROID
C                                   - GRAVITY COEFFICIENTS
C                                   - ATMOSPHERE MODEL
C                  2) VEHICLE MODEL - MASS PROPERTIES AND REFERENCE
C                                     GEOMETRY
C                                   - ENGINE OPTION AND ENGINE LOCATIONS
C                                   - AERO OPTION
C                                   - GUIDANCE AND CONTROL
C                  3  TRAJECTORY    - INITIAL CONDITIONS
C                                   - INTEGRATION METHOD
C                                   - SPECIAL OPTION CONTROL FLAGS
C
 $END
PSTBLMLT
C  INPUT IN THIS NAMELIST ALL TABLE SCALE FACTORS FOR PHASE 1.0
C
C  INPUT THE THRUST TABLE ( FOR PHASE 1.0 )
C
C  INPUT THE NOZZLE EXIT AREA TABLE
C
C  INPUT THE CD OR THE CA TABLE FOR DROPTANK PLUS ORBITER
C
C  INPUT THE CL OR THE CN TABLE FOR DROPTANK PLUS ORBITER
C
 $ENDPHS    = 1.
C
PSEGNDAT EVENT     = 2.0, CRITR = 4MASMG, VALUE  = 1.25,
C  TURN OFF MOLDDOWN MODEL VIA DATA INPUT HERE.
 $ENDPHS    =1.
C
PSEGNDAT EVENT     = 3.0, CRITR = 4MVELR, VALUE  = 150.,
 $ENDPHS    =1.
C
PSEGNDAT EVENT     =4.0, CRITR = 5HALPHA, VALUE  = 0.0,
C  INPUT DATA REQUIRE TO USE GRAVITY TURN IN  THIS PHASE
 $ENDPHS    =1.
C
PSEGNDAT EVENT     =5.0, CRITR = 4MDYNP,  VALUE  = 300.0,
PSEGNDAT EVENT     =5.0, CRITR = 4MDYNP,  VALUE  = 300.0,
C  INPUT DATA TO CHANGE STEERING OPTION
 $ENDPHS    =1.
C
PSEGNDAT EVENT     =6.0, CRITR = 4MVELR,  VALUE  = 6500.0,
C  INPUT JETTISON WEIGHT
C
PSTBLMLT
C  INPUT ANY NEW OR CHANGED SCALE FACTORS
C
C  INPUT CD OR CA TABLES FOR ORBITER ( WITHOUT DROPTANK )
C
C  INPUT CL OR CN TABLES FOR ORBITER ( WITHOUT DROPTANK )
C
 $ENDPHS    =1.
C
PSEGNDAT EVENT     =7.0, CRITR = 5HTDVRP, VALUE  = 5.0,
C  INPUT DATA REQUESTING LINESIAN STEERING
 $ENDPHS    =1.
C
PSEGNDAT EVENT     =1.0, CRITR = 4MASMG,  VALUE  = 3.0,
C  INPUT DATA REQUESTING THROTTLING TO 3G LIMIT
 $ENDPHS    =1.
C
PSEGNDAT EVENT     =8.0, CRITR = 5HWPROP, VALUE  = 0.0,
C  INPUT DATA TO TURN OFF ENGINES
 $ENDPHS    =1.
C
PSEGNDAT EVENT     =10.0, CRITR = 5HTDVRP, VALUE  = 60.,
 $ENDPHS    =1.
 $ENDPOB    =1.
 $ENDJOB    =1.
```


Figure 3.- Relationship Between Trajectory Sequence of Events and POST Input Data Structure

TABLE 1.- SUMMARY OF PLANET SIMULATION OPTIONS

Planet Model Characteristic	Options	Description
Oblate Spheroid and Gravitation Model	1. Spherical 2. 1960 Fisher Earth 3. Smithsonian Earth	These models are defined by the equatorial radius, the polar radius, the rotation rate, the gravitational constant μ, and the second, third, and fourth gravitational harmonics, J_2, J_3, and J_4. For a spherical planet, J_2, J_3, and J_4 are all zero.
Atmosphere	1. General	1. Atmospheric pressure, temperature, density, and speed of sound are computed from user input tables as a function of altitude. Speed of sound and density tables can be omitted, in which case they are computed from input values of ratio of specific heats, molecular weight, and the Universal gas constant.
	2. 1962 U.S. Standard	2. The 1962 U.S. Standard Atmosphere model is given as a function of geopotential altitude. In POST, the molecular weight is assumed constant.
	3. 1963 Patrick AFB	3. In this model, pressure and temperature are calculated as a function of geometric altitude and a set of prestored polynomial coefficients.
Winds	1. Geographic	1. The wind velocity is input directly in the geographic frame by defining the Easterly, Northernly, and radial components of the wind velocity.
	2. Meteorologic	2. The wind velocity is computed from the total wind speed and meteorologic wind heading.

TABLE 2.- SUMMARY OF VEHICLE SIMULATION OPTIONS

Vehicle Characteristic	Options	Description
Propulsion	1. Rocket Engines	1. Vacuum thrust, nozzle exit area, and flowrate or I_{sp} can be input for up to 15 engines per phase.
	2. Jet Engines	2. The ratio of total thrust to the atmospheric pressure ratio and the ratio of specific fuel consumption to the square root of the atmospheric temperature ratio can be input for up to 15 engines per phase.
	3. Ramjet Engines	3. Thrust coefficient and specific fuel consumption can be input for up to 15 engines per phase.
Aerodynamic	1. Lift and Drag Model	1. Lift, drag, and sideforce coefficients (3D), and pitch, yaw, and roll aerodynamic coefficients (6D) are input as tables.
	2. Axial and Normal Force Model	2. Axial, normal, and sideforce coefficients (3D), and pitch, yaw, and roll aerodynamic coefficients (6D) are input as tables.
Aeroheating	1. Chapman's Equation	1. Heat rate is computed from Chapman's equation with the nose radius as an input.
	2. General Table Lookup	2. Heat rate is computed as a table lookup based on as many as three independent variables.
	3. Modified Chapman's Equation	3. Heat rate is computed as the product of Chapman's heatrate and a general table lookup.
	4. Turbulent Flow	4. Similar to modified Chapman's equations with different exponents.
	5. Maximum Centerline	5. Heat rate is computed based on correction for altitude and angle of attack.
Steering	1. Open Loop	1. 3D attitude is calculated from tables, polynomials, or linear feedback; 6D attitude commands are computed from tables.
	2. Closed Loop	2. All closed loop models must be coded specifically for each application.

TABLE 3.- SUMMARY OF TRAJECTORY SIMULATION OPTIONS

Simulation Features	Options	Descriptions
Integration Methods	1. Runge Kutta-4th Order	1. The standard 4th order Runge-Kutta single step method for a set of first order ordinary differential equations.
	2. Runge Kutta-8th Order	2. An eighth order single step method.
	3. Predictor-Corrector	3. A variable-step variable-order method developed by F. T. Krogh of the Jet Propulsion Laboratories.
Table Interpolation	1. Piecewise Constant	1. The function is assumed to be a step function based on the user's input.
	2. Piecewise Linear	2. Linear interpolation is used between data points. The tables can be monovariant, bivariant, or trivariant.
	3. Piecewise Cubic	3. Cubic interpolation is used between data points. The tables can be monovariant, bivariant, or trivariant.
Events	1. Primary	1. Events that must occur, and must occur in ascending order according to the event number.
	2. Secondary	2. Events that must occur in ascending order between their bounding primaries. The occurrence of a primary nullifies the previous secondaries.
	3. Roving	3. Events that can occur any time after the occurrence of all smaller primaries.
	4. Repeating	4. Events that can repeat at a specified increment or at specified values.
Orbital Propagators	1. Laplace's Method	1. A semianalytical method for propagating the position and velocity of a nonthrusting vehicle in vacuum flight over a spherical planet.
	2. Encke's Method	2. A rapid method for propagating orbits perturbed by planet oblateness and atmosphere.
Special Purpose	1. ΔV Additions	1. At any specified event an instantaneous velocity change can be added to the vehicle.
	2. Static Trim	2. The engine gimbal angles or flap deflections are computed to balance the pitch and yaw moments caused by thrust of other engines and aerodynamic forces.
Launch	1. Vertical Holddown	1. This model is used to simulate vehicle holddown by maintaining the position and velocity relative to the planet constant until launch.
	2. Horizontal Takeoff	2. This option is used to simplify horizontal takeoff by allowing the vehicle to accelerate only in the local horizontal plane

One additional degree of complexity usually arises in 6D trajectory work, that is, the guidance, navigation, and autopilot equations must generally be coded in FORTRAN and added to 6D POST in the appropriate subroutines. Procedures for accomplishing this effort are described in reference 2.

Targeting and Optimization Capabilities

POST has a complete discrete parameter nonlinear programming capability. This means that POST can minimize (or maximize) a user-selected performance function, subject to nonlinear target conditions and/or inequality constraints. The control variables can be any parameter that influences the performance criteria and/or the mission constraints. The performance function, the target conditions and constraints, and the control variables can be selected from a

dictionary of over 400 program variables. Some typical examples of these variables are shown in Figure 4. POST also contains several popular optimization algorithms that can be selected by the user. A brief summary of these algorithms is given in Table 4. As indicated, the accelerated projected gradient algorithm is normally used as the basic optimization technique. This algorithm is a combination of Rosen's projected gradient method for nonlinear programming and Davidson's variable metric method for unconstrained optimization. In the targeting mode, the minimum norm algorithm is used to satisfy the trajectory constraints. The cost and constraint gradients required by these algorithms are computed normally as first differences calculated from perturbed trajectories. In some particularly difficult cases, symmetric differences are used to more accurately approximate the derivatives. To reduce the costs of calculating numerical sensitivities, only that portion of the trajectory influenced by any particular independent variable is reintegrated on the perturbed runs. This feature saves a significant amount of computer time when targeting and optimization is performed.

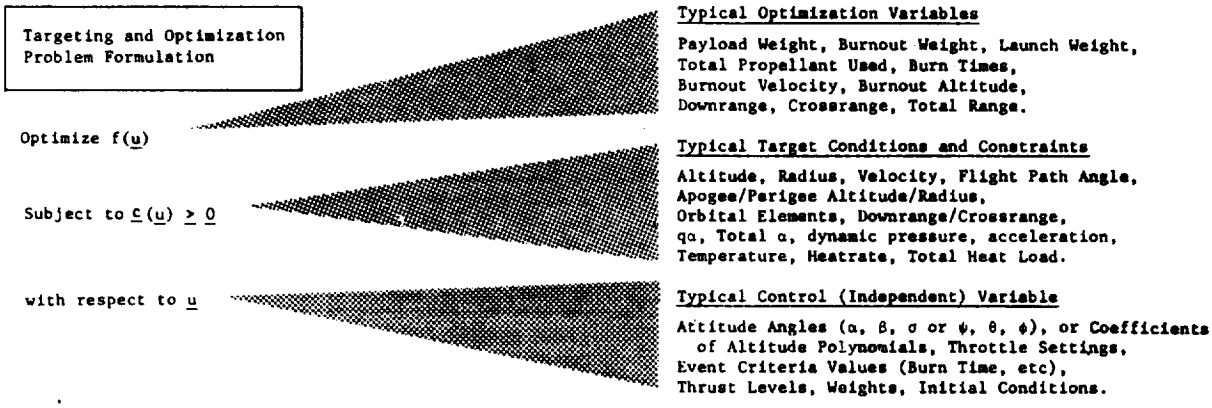


Figure 4.- Optimization Summary

TABLE 4.- TARGETING AND OPTIMIZATION PROBLEM SOLUTION METHODS

General Class of Problems	Examples	Available Algorithm in POST	Recommended Algorithm
Constrained Optimization with Inequalities Optimize: f (u) Subject to: c (u) ≥ 0	Ascent to Orbit Max: W_{PLD} (Payload Weight) Subject to: h_{BO} = 303 805 ft (Altitude) γ_{BO} = 0 (FPA) V_{BO} = 25 841 fps (Velocity) q ≤ 400 psf (Dynamic Pressure)	Accelerated Projected Gradient Method Single Penalty Functions Using: - Steepest Descent - Conjugate Gradients - Davidon's Method	Accelerated Projected Gradient Method
Unconstrained Optimization Optimize: f(u)	Sounding Rocket Max: $h_{\dot{h}=0}$ (Max Altitude)	Steepest Descent Conjugate Gradients Davidon's Method	Davidon's Method
Targeting: Determine u Such That c (u) = 0	Space Shuttle Entry Determine the Entry γ and Azimuth Such That Landing Site is Reached, i.e., Latitude φ = Specified Value Longitude θ = Specified Value	Steepest Descent Conjugate Gradients Davidon's Method Newton Raphson	Newton Raphson

All program inputs are made using an improved NAMELIST input processor. The key features of this extended NAMELIST capability are illustrated in Figure 5. The permissable sequence of NAMELISTs are also given in Figure 6. As indicated, $SEARCH and at least one $GENDAT are required per problem. $TBLMLT and $TAB are optional within $GENDAT depending on whether tabular data are required in that phase.

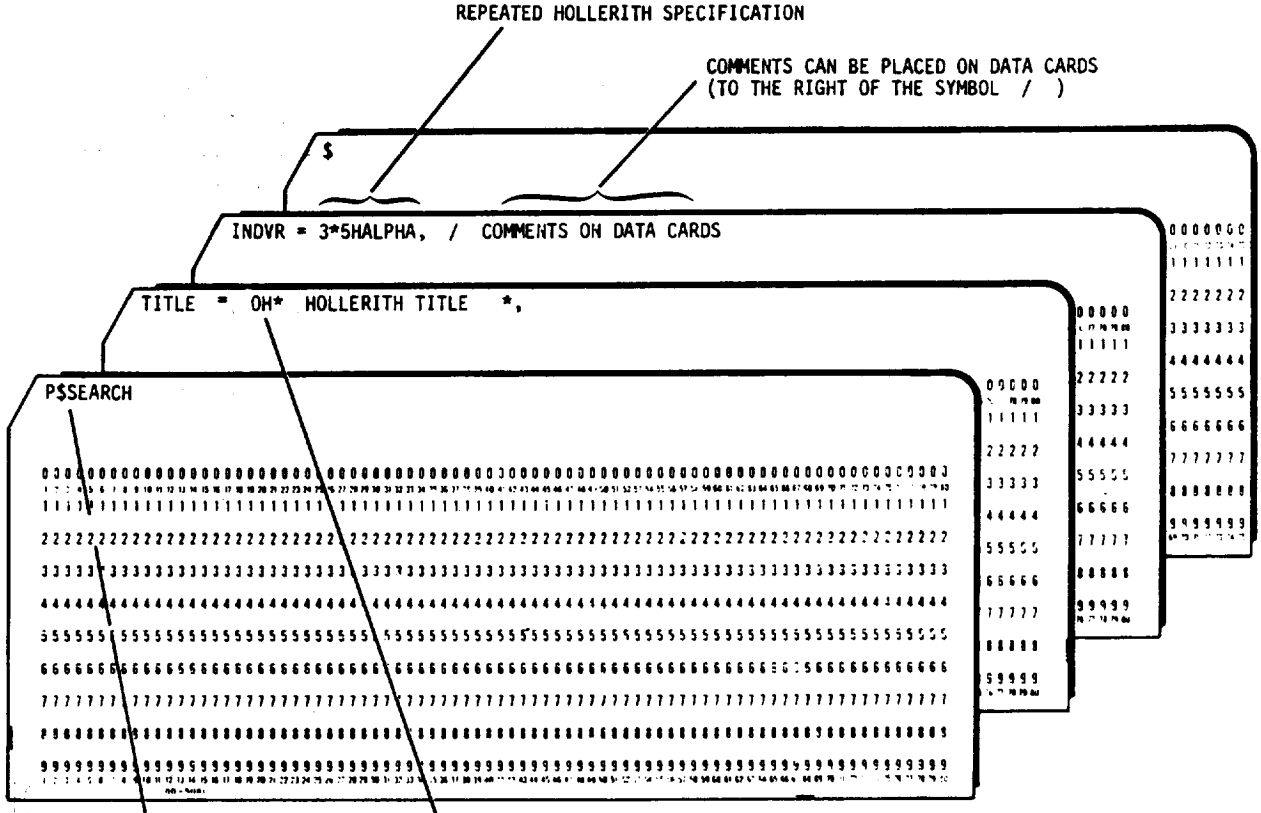


Figure 5.- Extended Namelist Capabilities

Targeting & Optimization Input

General Simulation Input for Phase 1

Table Data for Phase 1

General Data for Remaining Simulation Phases

```
PSSEARCH
C INPUT IN THIS NAMELIST ALL DATA PERTAINING TO :
C
C                  1) OPTIMIZATION AND TARGETING FORMULATION
C                        OPTIMIZATION VARIABLE
C                        CONSTRAINTS AND TARGET CONDITIONS
C                        CONTROL VARIABLE
C                  2) ALGORITHM SELECTION AND CONTROL FLAGS
 $
PSGENDAT EVENT     = 1,
C INPUT IN THIS NAMELIST ALL DATA AND PROGRAM CONTROL FLAGS PERTAINING
C TO:
C
C                  1) PLANET MODEL - CREATE SPHEROID
C                                  - GRAVITY COEFFICIENTS
C                                  - ATMOSPHERE MODEL
C                  2) VEHICLE MODEL- MASS PROPERTIES AND REFERENCE
C                                    GEOMETRY
C                                  - ENGINE OPTION AND ENGINE LOCATIONS
C                                  - AERO OPTION
C                                  - GUIDANCE AND CONTROL
C                  3  TRAJECTORY   - INITIAL CONDITIONS
C                                  - INTEGRATION METHOD
C                                  - SPECIAL OPTION CONTROL FLAGS
 $
PSTBLMLT
C INPUT IN THIS NAMELIST ALL TABLE SCALE FACTORS FOR PHASE 1.0
 $
C INPUT THE THRUST TABLE ( FOR PHASE 1.0 )
 $
C INPUT THE NOZZLE EXIT AREA TABLE
 $
C INPUT THE CD OR THE CA TABLE FOR DROPTANK PLUS ORBITER
 $
C INPUT THE CL OR THE CN TABLE FOR DROPTANK PLUS ORBITER
C
 ENDPHS     = 1,
 $
PSGENDAT EVENT     = 2.0, CRITP  = 4HASMG, VALUE  =0.25,
C TURN OFF HOLDDOWN MODEL VIA DATA INPUT HERE.
 ENDPHS     =1,
 $
PSGENDAT EVENT     = 3.0, CRITP  = 4HVELR, VALUE
 ENDPHS     =1,
 $
PSGENDAT EVENT     =4.0, CRITP = 5H
C INPUT DATA REQUIRE TO USE CF
 ENDPHS     =1,                                  VALUE  = 6500.0,
 $
PSGENDAT EVENT

                    CHANGED SCALE FACTORS

      CD OR CA TABLES FOR ORBITER ( WITHOUT DROPTANK )
 $
C INPUT CL OR CN TABLES FOR ORBITER ( WITHOUT DROPTANK )
C
 ENDPHS     =1,
 $
PSGENDAT EVENT     =7.0, CRITP = 5HTDVRP, VALUE  = 5.0,
C INPUT DATA REQUESTING LINESINE STEERING
 ENDPHS     =1,
 $
PSGENDAT EVENT     =8.0, CRITP = 4HASMG,  VALUE  = 3.0,
C INPUT DATA REQUESTING THROTTLING TO 3G LIMIT
 ENDPHS     =1,
 $
PSGENDAT EVENT     =9.0, CRITP = 5HWPROP, VALUE  = 0.0,
C INPUT DATA TO TURN OFF ENGINES
 ENDPHS     =1,
 $
PSGENDAT EVENT     =10.0, CRITP = 5HTDVRP, VALUE  = 60.,
 ENDPHS     =1,
 ENDPRB     =1,
 ENDJOB     =1,
 $
```

Figure 6.- POST Input Structure

The program also has the capability to accept input in either English or metric units. The only restriction being that the input units cannot be mixed. The output variables can also be printed in either English or metric units, but not both.

All English to metric conversion constants can be changed by input if desired. The stored values of these constants were obtained from reference 3.

Care must be taken when using metric system input and output. The input units must be of the same type as the English units. For example, values for weights must be input in units of force (newtons) rather than mass (kg). Variables output in nautical miles in English units must be kilometers in metric.

The program also has the capability to print a summary of the table data input. This feature is useful in that input errors in the table format are easily detected by reviewing this printout.

In addition to the normal input procedures described above, the program has the capability to process more than one input problem per pass at the computer. This feature, referred to as Multiple Runs, enables the user to change the basic input deck (which represents a single problem) by adding a subsequent set of data cards representing the changes (addition and/or deletions) immediately after the original set of data cards. Once the first problem is completed, the program will automatically modify the input file as defined by the additional cards and run the resulting data file. This capability is extremely useful in performing parametric studies where only a few input variables are changed from one run to the next.

There are two basic categories of POST output: (1) trajectory data, and (2) program control data. Trajectory data can be output in the form of a computer printout and/or as a profile tape. The typical computer printout consists of (1) an input echo, (2) an input summary, (3) a trajectory printout that optionally includes a phase data summary at the beginning of each event, (4) special trajectory printouts, such as orbital parameter, tracking data, etc, and (5) targeting and optimization iteration traces and summaries. Trajectory data can also be written on a profile tape for storage or auxiliary output purposes. Typical output obtained from the profile tape includes trajectory plots, orbital ground tracks superimposed on a world map, and punched cards. The computer routines required to generate these types of output from the profile tape are computer/system-dependent and not contained in POST, but are readily available at most modern computing centers. The second type of output, namely, program control data, is contained in the initial input summary and is updated in the phase data summaries output at the beginning of each event. Employing these summaries, the user can always determine exactly what options are being employed in the simulation during any phase of the trajectory simulation. The various forms of POST output are summarized in Figure 7.

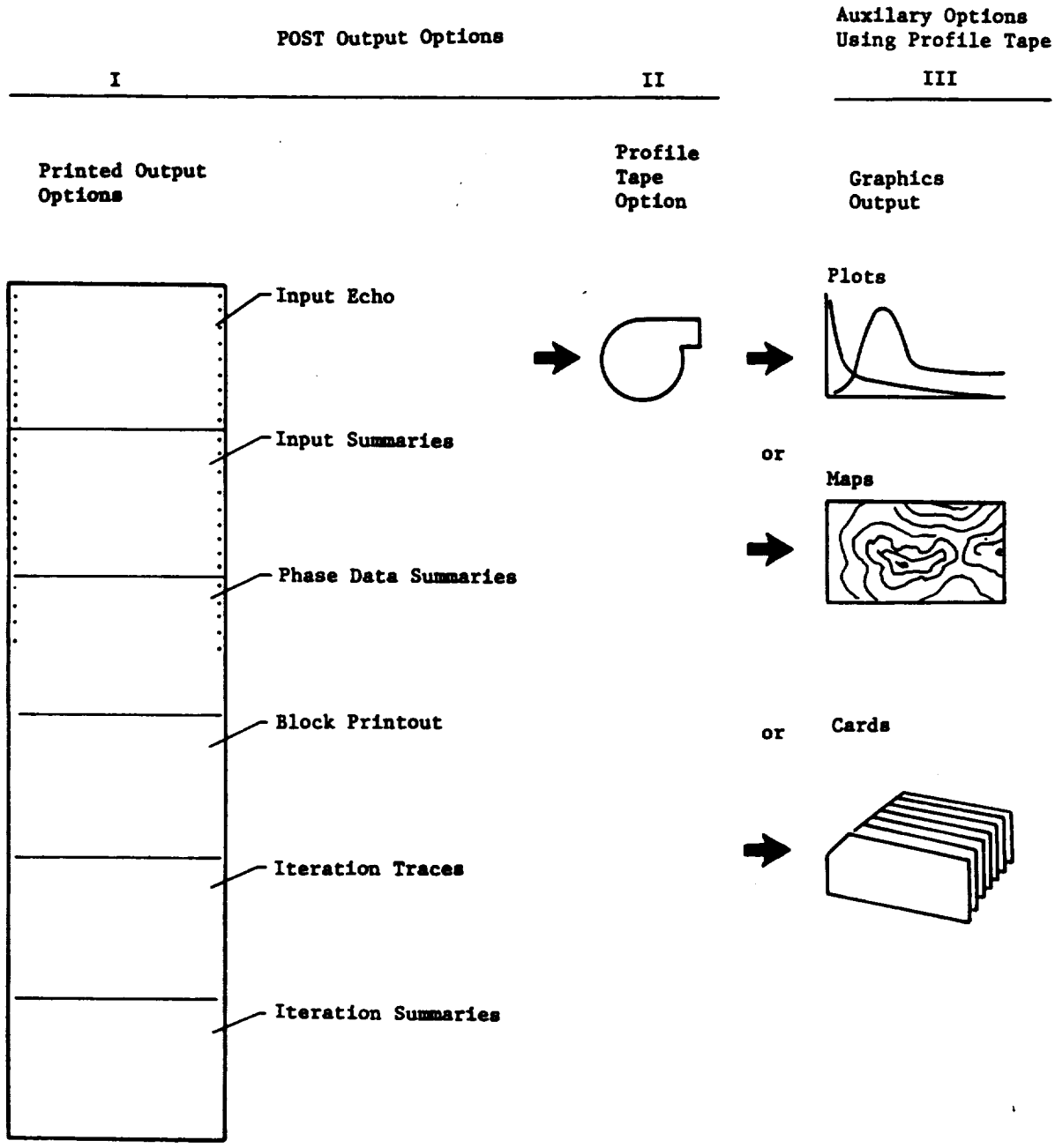


Figure 7.- POST Output Options

The basic trajectory printblock can be defined, in terms of size and content, by the user. Any common variable computed in the program can be output by including its hollerith name in the printblock definition. If the user does not wish to define a printblock, then a nominal default printblock can be used. Any printblock may be modified with additions and/or deletions according to the rules described in reference 1.

The targeting and optimization output has two levels of detail. The iteration summary is the output most frequently used and it is always displayed in the printout. This summary contains all the information required to understand the progress of the iteration process. Information, such as trajectory sensitivities, stepsize limits, univariant curvefit summaries, current control variables, current constraint errors, and current performance are included in the summary. When the iteration is judged to be improperly converging, a detailed iteration trace can be requested. This trace gives all information required to determine the exact iteration cycling calculations and prints the data in the exact chronological order that they were calculated internally. The data are usually only required when new problems are being formulated, and are extremely useful in identifying problem formulation and setup errors. It also gives key information that can be used by experienced POST users to speed up the convergence of the targeting and optimization algorithms.

Program Applications

During the past several years, POST has been used to solve hundreds of performance and mission analysis problems for atmospheric and orbital vehicles. A brief summary of the more standard applications of POST is contained in Table 5. The spectrum of problems shown in Table 5 gives some indication of the overall versatility of POST.

TABLE 5.- TYPICAL APPLICATIONS OF POST

Type of Mission	Type of Vehicle	Optimization Variables	Typical Constraints		CPU Time Required to Solve Problems, min
			Equality	Inequality	
Ascent to Near-Earth Orbit	Titan IIIC & D&E, Space Shuttle, Single Stage to Orbit (VTO and HTO)	Payload, Weight at Burnout Fuel, Burntime, Ideal Velocity, Initial Weight	Radius Flight Path Angle Velocity	Dynamic Pressure Accelerations	2 → 20
Ascent to Synchronous Equatorial Orbit	Titan IIIC, Shuttle/ Tug	Payload	Apogee Perigee Inclination	Dynamic Pressure Angle of Attack Pitch Rates	3 → 50
Ascent Abort	Space Shuttle	Abort Interval	Landing Site Longitude and Latitude	Acceleration Dynamic Pressure	2 → 5
ICBM Ballistic Missile	Titan II, Minuteman I & II, Safeguard	Payload Misc Distance	Latitude Longitude Crossrange Downrange	Flight Path Angle at Entry Acceleration during Entry	2 → 20
Reentry	Space Shuttle, X-24C, Single Stage to Orbit	Heat Rate Total Heat Crossrange	Latitude Longitude Crossrange Downrange	Heat Rate Acceleration	3 → 15
Orbital Maneuvers	Transtage, Space Tug, IUS, Solar Electrical Propulsion	Payload Fuel	Radius Velocity Flight Path Angle Argument of Perigee Period Apogee, Perigee	Attitude Angles Perigee Altitude	0.5 → 10
Aircraft Performance	X-24B and C, Space Shuttle Subscale, Subsonic Jet Cruise, Hypersonic Bombers and Interceptors	Mach Number Cruise Time Payload	Downrange Crossrange Dynamic Pressure Velocity and Mach Altitude	Dynamic Pressure Dynamic Pressure at Max Altitude	0.1 → 5

The generality of POST that makes it ideal for detailed vehicle performance work also gives it a unique capability for estimating performance for advanced vehicle concepts. A few applications of POST to advanced vehicle/mission concepts are (1) single-stage to orbit trajectory optimization, (2) hypersonic cruise vehicle optimization, (3) parachute simulations for solid rocket booster recovery concepts, (4) guidance algorithm development for maneuverable reentry vehicles, (5) ascent optimization for a Mars sample return mission, and (6) simulation of a Mars entry using advanced video guidance concepts.

PROGRAM STRUCTURE AND FORMULATION

Executive Design

POST executive logic was designed to provide (1) readily learned input and output procedures, (2) flexible trajectory simulation, (3) detailed vehicle modeling, and (4) generalized targeting and optimization. These goals are met by the POST executive structure, which is presented in Figure 8. As indicated, executive routines are used throughout. These routines control the program execution flow by calling subroutines containing the actual mathematical computations. This modular structure allows the program to be modified quickly and easily.

Both 3D POST and 6D POST are structured in three overlay levels, (0, 0), (1, 0), and (2, 0), respectively. The first overlay (0, 0) is the master executive overlay, which controls the overall program. This overlay controls the read-in of input data and determines which trajectory computations are to be performed. All utility routines are contained overlay in (0, 0).

Overlay (0, 0) first calls overlay (1, 0), which reads the namelist input data from cards and stores the processed data on discs for later use.

Overlay (2, 0) is called by (0, 0) after (1, 0) has completed the input processing tasks. The first decision in overlay (2, 0) concerns the type of simulation; i.e., single trajectory or search/optimization mode. If a single trajectory is to be run, the program calls overlays (2, 1) (2, 2), and (2, 3) sequentially, then returns to the master overlay (0, 0). If the search/optimization mode is to be used, the program control is turned over to subroutine MINMYS, which calls overlays (2, 1), (2, 2), (2, 3), (2, 5), and (2, 6) as required to perform the search/optimization function. When convergence has been achieved or the maximum number of iterations has been exceeded, control reverts back to the master overlay (0, 0) for the next problem.

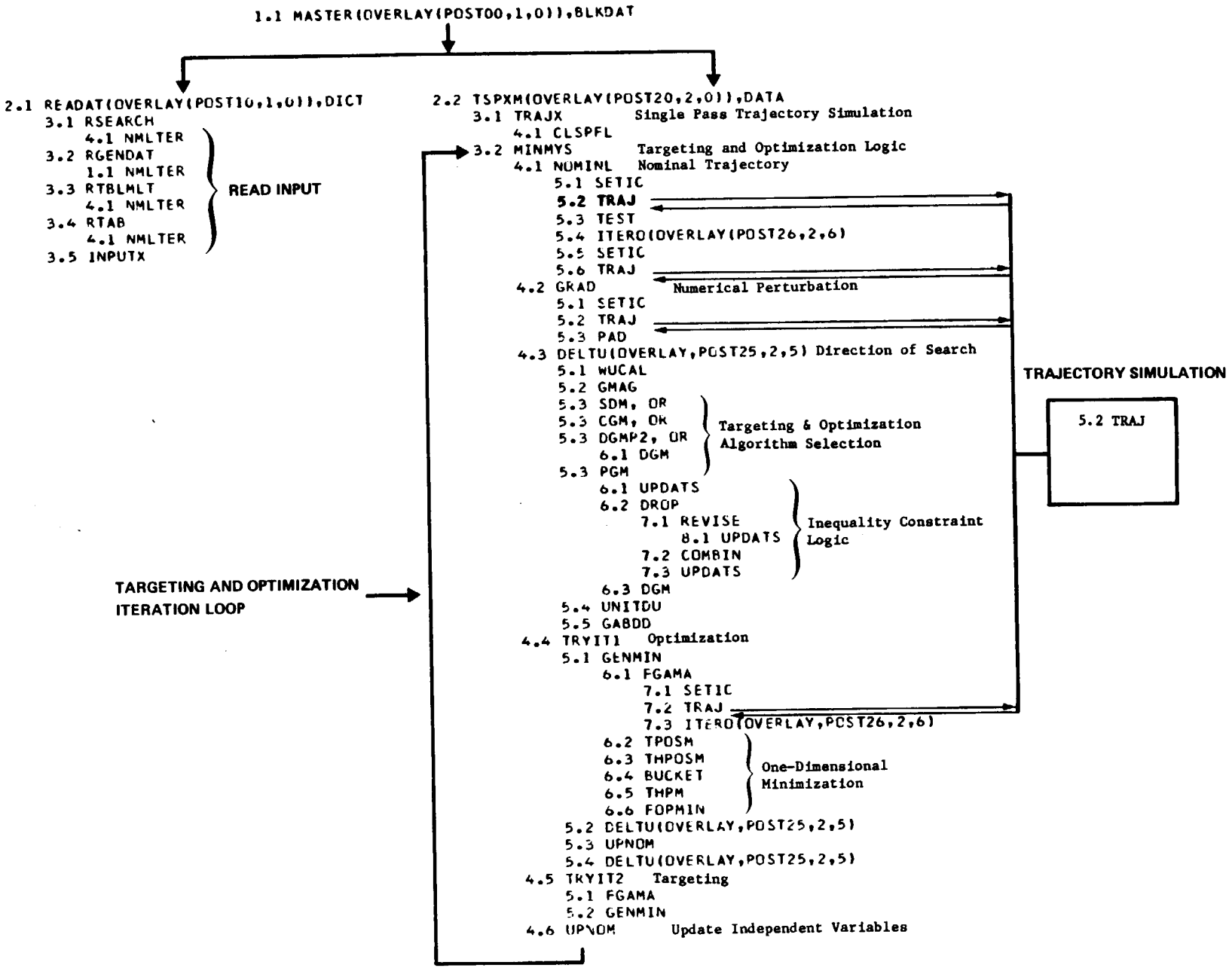


Figure 8.- POST Executive Structure

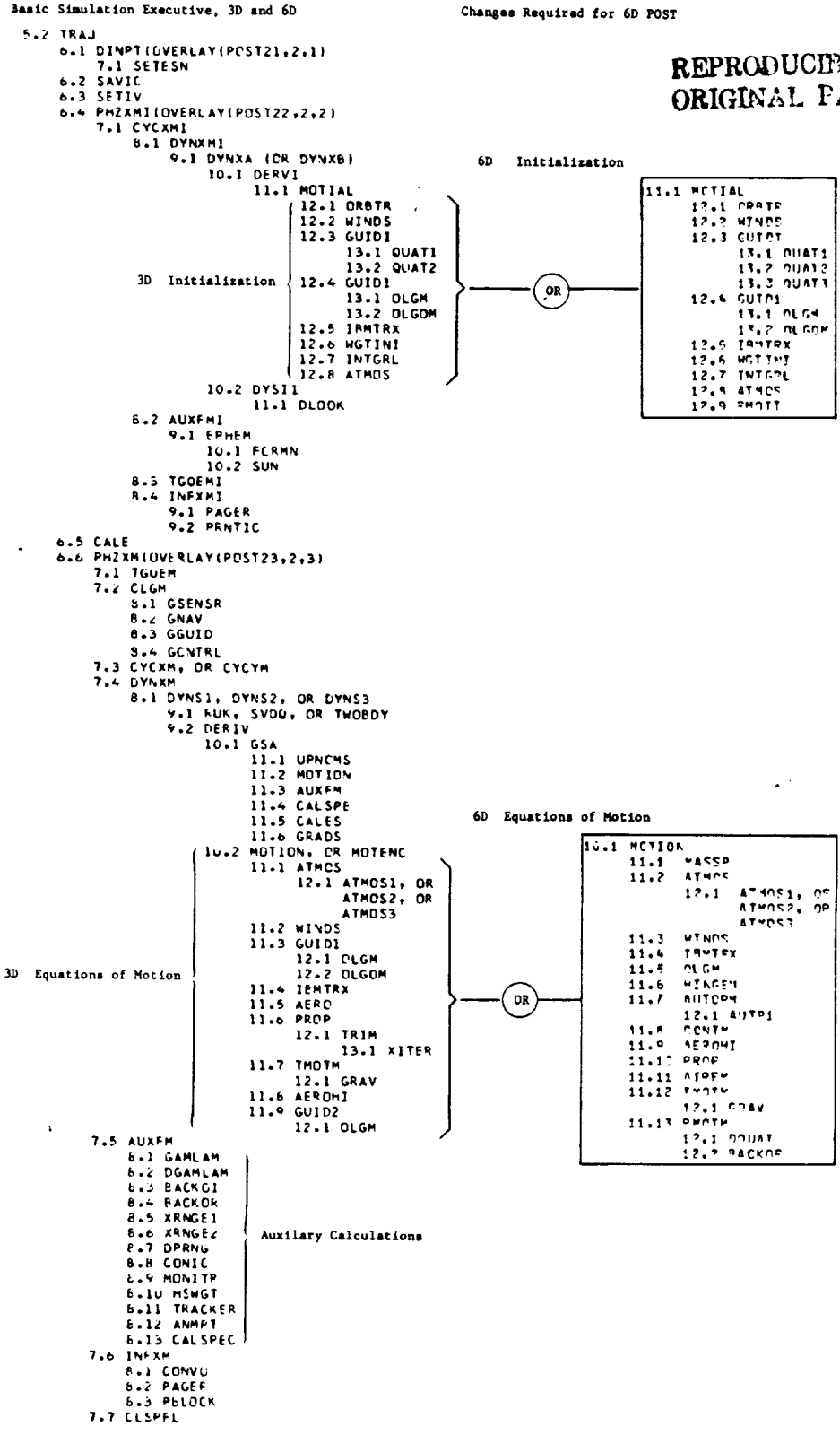


Figure 8.- Concluded

An outline of the approximate calling sequence for each routine is included in Figure 8, and shows which subroutines are called by a given routine, thereby allowing the detailed logic flow to be followed easily. The overall program logic described by the overlays is as follows:

1) Overlay (2, 1) reads the previously processed input data from tape, locates the data for the current phase (event), and initializes the program values based on this input;

2) Overlay (2, 2) initializes the equations of motion for the current phase;

3) Overlay (2, 3) integrates the equations of motion from time t_i to a specified stopping condition for the current phase;

4) Overlay (2, 5) calculates the control corrections based on the search/optimization algorithm being used, limits the control parameters that violate the control parameter constraints, and tests for convergence;

5) Overlay (2, 6) prints out an iteration summary at the end of each iteration. It also performs any other information output tasks required by search/optimization algorithm, such as printing trail step summaries.

The program dictionary (subroutine DICT) performs a one-to-one mapping of variables in common and the Hollerith names by which the user can select the variables for a variety of uses, including output, stopping conditions, control variables, and targeting variables.

POST uses a generalized table storing and lookup procedure whereby the size of tables is limited only by the total data storage allocation of 1500 cells. Each table has its own multiplier. This is accomplished by dimensioning the table by (2). The first location contains the address of the table and the second location contains the table function multiplier. The generalized table lookup (GENTAB) is set up to handle all allowable types of tables, namely, constant-value, monovariant, bivariant, and trivariant.

The interaction between the targeting/optimization logic and the trajectory simulation was designed so that the trajectory calculations represent an external evaluation of the objective function and the constraints. In this manner, any change to the trajectory simulation capability of the program automatically is available to the targeting and optimization logic.

Coordinate Systems

POST uses numerous coordinate systems to provide the necessary reference systems for calculating required and optional data. The key computational coordinate systems are illustrated in Figure 9 and 10, and the definitions of all coordinate systems used are summarized in Table 6. The equations of motion are integrated in the Earth-centered Inertial (ECI) frame. Thus, the external thrust and aerodynamic forces, which are computed in the body coordinate system,

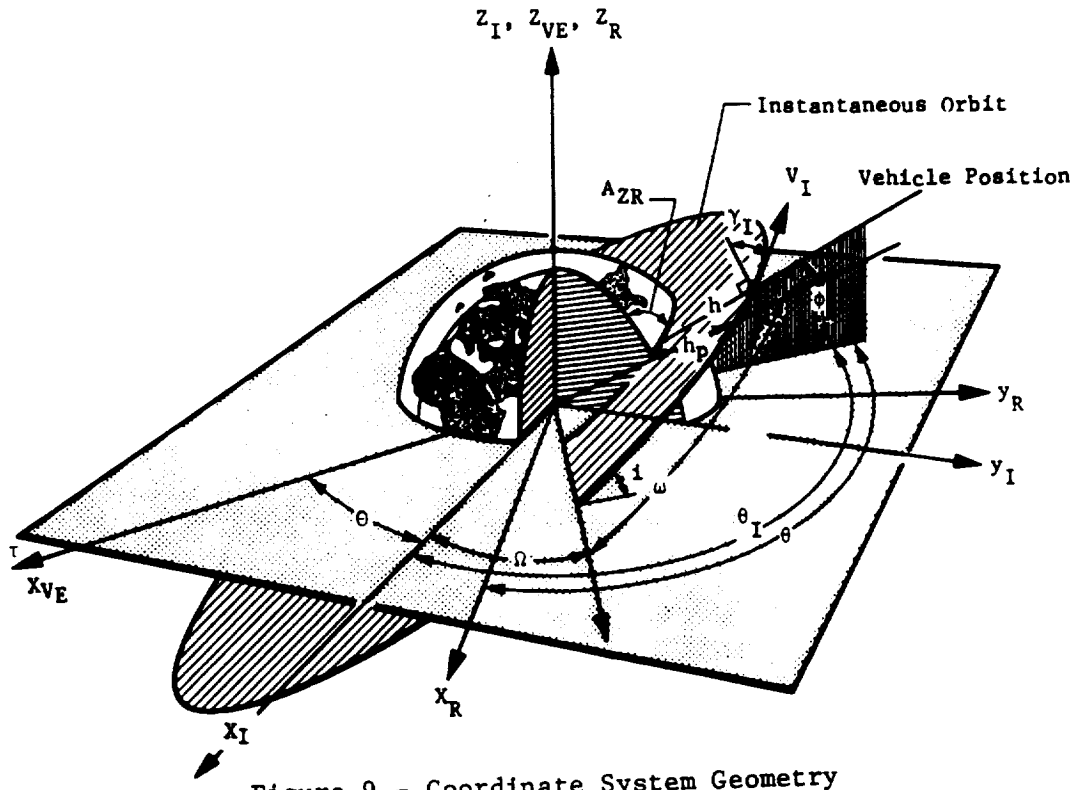
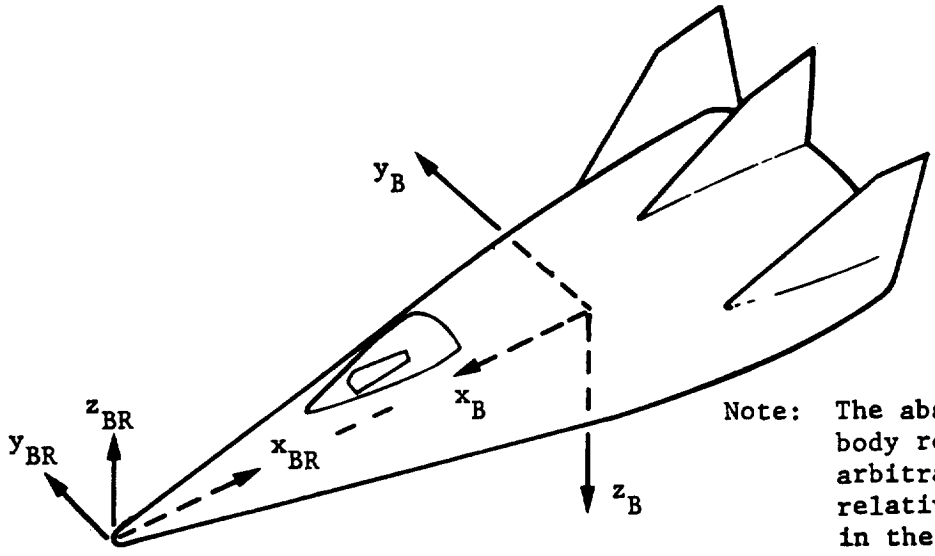


Figure 9.- Coordinate System Geometry



Note: The absolute location of the body reference system is arbitrary because only the relative distances are used in the Equations of Motion.

Figure 10.- Body Frame

TABLE 6.- POST COORDINATE SYSTEMS

Coordinate System	Axes	Definition
Earth-Centered Inertial (ECI)	(x_I, y_I, z_I)	Earth-centered Cartesian system with z_I coincident with the North Pole, x_I coincident with the Greenwich Meridian at time zero and and in the equatorial plane, and y_I completing a right-hand system. The translational equations of motion are solved in this system.
Earth-Centered Rotating (ECR)	(x_R, y_R, z_R)	Similar to the ECI system except that it rotates with the Earth so that x_R is always coincident with the Greenwich Meridian.
Earth Position Coordinates	(ϕ_g, θ, h)	These are the familiar latitude, longitude, and altitude designators. Latitude is positive in the Northern Hemisphere. Longitude is measured positive East of Greenwich. Altitude is measured positive above the surface of the planet.
Geographic (G) Axes	(x_G, y_G, z_G)	This system is located at the surface of the planet at the vehicle's current geocentric latitude and longitude. The x_G axis is in the local horizontal plane and points North, the y_G axis is in the local horizontal plane and points East, and z_G completes a right-hand system.
Inertial Launch (L) Axes	(x_L, y_L, z_L)	An inertial Cartesian system that is used as an inertial reference system from which the inertial attitude angles of the vehicle are measured. This coordinate system is automatically located at the geodetic ϕ and inertial longitude of the vehicle at the start of the simulation.
Body Reference (BR) Axes	(x_{BR}, y_{BR}, z_{BR})	Right-hand Cartesian system aligned with the body axes as follows. The x_{BR} axis is directed along the negative x_B axis, the y_{BR} axis is directed along the positive y_B axis, and the z_{BR} is directed along the negative z_B axis.
Orbital Elements	$(h_a, h_p, i, \omega, \theta, \Omega)$	A nonrectangular coordinate system used in describing orbital motion. The orbital elements are apogee altitude, perigee altitude, inclination, longitude of the ascending node, true anomaly, and argument of perigee.
Vernal Equinox (VE) Axes	(x_{VE}, y_{VE}, z_{VE})	A 1950 mean equator and equinox Earth centered inertial system. The x_{VE} axis is in the equatorial plane and is directed forward of the vernal equinox of 1950, the z_{VE} axis is directed along the north pole, and y_{VE} completes the right hand system.
Body (B) Axes	(x_B, y_B, z_B)	The body axes form a right-hand Cartesian system aligned with the axes of the vehicle and centered at the vehicle's center of gravity. The x_B axis is directed forward along the longitudinal axis of the vehicle, y_B points right (out the right wing), and z_B points downward, completing a right-hand system.

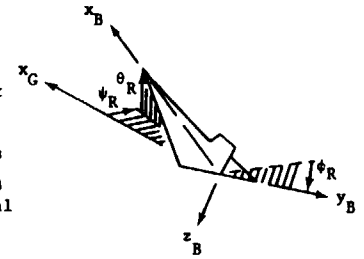
must be transformed to the ECI system. This is performed by the inverse of the transformation matrix, [IB]. The [IB] matrix is functionally dependent on the attitude of the vehicle, and is calculated based on the equations describing the attitude steering option selected by the user. POST contains four standard attitude reference systems as described in Figure 11. Any given trajectory problem may require use of one or more of these systems. For example, simulation of a complete ICBM trajectory typically involves the use of inertial Euler angles during first and second stage boost, relative Euler angles during third stage flight, inertial aerodynamic angles during the postboost maneuvers, and finally aerodynamic angles during reentry. The availability of all these options, while confusing to new users, is a valuable aid to the experienced trajectory analyst, and enables him to steer each phase of the trajectory in the most appropriate manner.

Relative Euler angles

ψ_R - Relative yaw angle. This is the azimuth angle of the x_B axis measured clockwise from the reference direction (first rotation),

θ_R - Relative pitch angle. This is the elevation angle of the x_B axis above the local horizontal plane (second rotation),

ϕ_R - Relative roll angle. This is the roll angle about the x_B axis (third rotation).

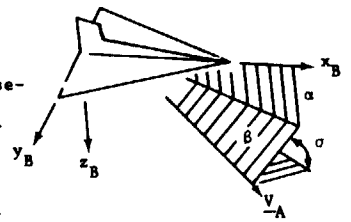


Aerodynamic angles

σ - Bank angle. Positive σ is a positive rotation about the atmosphere relative velocity vector (first rotation),

β - Sideslip. Positive β is a nose-left (negative) rotation when flying the vehicle upright (second rotation),

α - Angle of attack. Positive α is a nose-up (positive) rotation when flying the vehicle upright (third rotation).

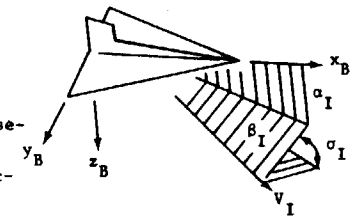


Inertial aerodynamic angles

σ_I - Bank angle. Positive σ_I is a positive rotation about the atmosphere inertial velocity vector (first rotation),

β_I - Sideslip. Positive β_I is a nose-left (negative) rotation when flying the vehicle upright (second rotation),

α_I - Angle of attack. Positive α_I is a nose-up (positive) rotation when flying the vehicle upright (third rotation).



Inertial Euler angles

ϕ_I - Inertial roll angle. The roll angle with respect to the L-frame (first rotation),

ψ_I - Inertial yaw angle. The yaw angle with respect to the L-frame (second rotation),

θ_I - Inertial pitch angle. The pitch angle with respect to the L-frame (third rotation).

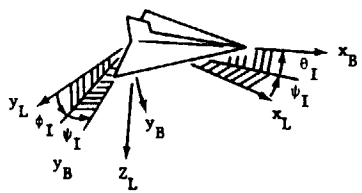


Figure 11.- Attitude Angles Used in POST

The matrix transformations between each pair of the coordinate systems are presented in Volume I. The basic relationships between the principal computational systems are depicted in Figure 12. The inverse transformations between these coordinate systems can be easily computed by merely transposing the matrix elements because of the orthonormality of these matrices.

Planet Model

The planet model consists of three basic categories of equations and input data: (1) oblate planet geometry and constants, (2) a gravitational model and its constants, and (3) an atmosphere model that includes winds. In each of these categories, the user can select prestored options to minimize the amount of required input data. On the other hand, if the desired option is not available, then the user can define his own model via input constants. The only inherent program limitations are defined by the equations representing the various models.

The oblate spheroid model is defined in terms of the equatorial radius R_E, the polar radius R_p, the rotation rate Ω_p, the gravitational constant μ, and the second, third, and fourth gravitational harmonics, J_2, J_3, and J_4, respectively. The 1960 Fisher Earth models are preloaded in POST. The geometry of this spheroid is illustrated in Figure 13.

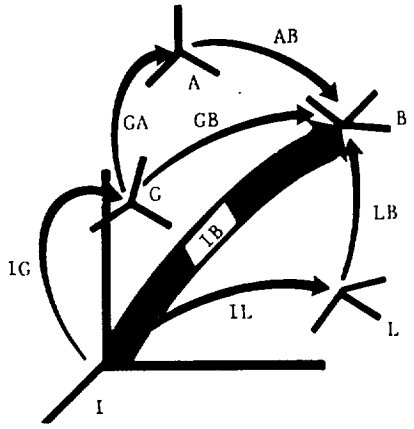


Figure 12.- Transformations between Coordinate Systems

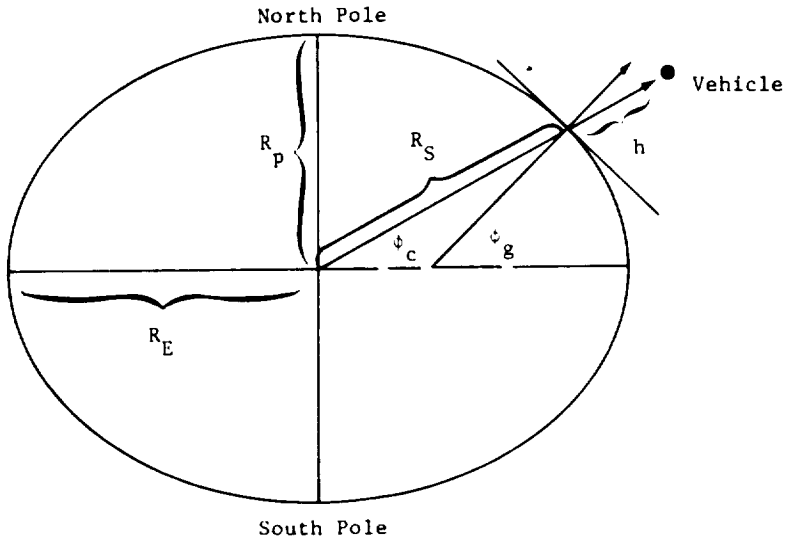


Figure 13.- Oblate Planet Geometry

The gravitational model includes optionally second, third, and fourth harmonic terms. This model is adequate for near-Earth ascent, on-orbit, and entry performance work. For extremely high altitude satellite maneuvers or ephemeris prediction, a more detailed model that includes lunar and solar perturbations and higher order harmonics is generally required. These terms are

not included in the standard POST program but are available in a special orbital version.

The gravitational accelerations calculated are given by the equations

$$G_{XI} = -\mu \frac{x}{r^3} P(z, r)$$

$$G_{XI} = -\frac{\mu}{r^3} P(z, r)$$

$$G_{ZI} = -\frac{\mu}{r^3}\left[\left(1 + JR^2\ (3 - 5Z^2)\right)\ z + H\frac{R^3}{r}\left(6z^2 - 7z^2\ Z^2 - \frac{3}{5}\ r^2\right) + DR^4\left(\frac{15}{7} - 10Z^2 + 9Z^4\right)\ z\right] \qquad (1)$$

where

$$P(z, r) = \left[1 + JR^2(1 - 5Z^2) + H\frac{R^3}{r}\ (3 - 7Z^2)\ z + DR^4\left(9Z^4 - 6Z^2 + \frac{3}{7}\right)\right]$$

and

$$x = x_I,\ y = y_I,\ z = z_I,\ r = r_I,$$

$$R = R_E/r_I,\ Z = z_I/r_I,\ J = \frac{3}{2}\ J_2,\ H = \frac{5}{2}\ J_3,\ D = -\frac{35}{8}\ J_4$$

POST has the optional capability of three atmospheric models--the general table lookup, the 1962 U.S. Standard atmosphere and the 1963 Patrick AFB atmosphere using polynomials. The general table lookup model gives the user the flexibility of inputting his own atmospheric model if none of the preloaded models is adequate. This is particularly useful in performing trajectory analysis for planets other than Earth.

The table lookup atmosphere model can be defined entirely by using tables that show pressure, temperature, speed of sound, and density as functions of altitude. The speed of sound and density tables can be omitted if desired; in this case, the speed of sound and density are computed as

$$C_s = \sqrt{K_1\ T}$$

$$\rho = K_2\ \frac{P}{T} \qquad (2)$$

where $K_1 = \frac{\gamma R^*}{M_0}$, $K_2 = \frac{M_0}{R^*}$, γ is the ratio of specific heats, M_0 is the molecular weight, and R* is the universal gas constant. The equations and constants defining the 1962 U.S. Standard and the 1963 Patrick AFB atmosphere models are given in reference 4.

The atmospheric wind velocity components can be input as tables using either meteorological or vector notation. If these tables, which are normally functions of oblate altitude, are not input, then the atmosphere is assumed to rotate uniformly with the planet.

The wind velocity components can be input directly in the geographic frame by defining u_W, v_W, and w_W, or by defining the wind speed $\left(V_W\right)$, the wind azimuth $\left(A_{ZW}\right)$, and the wind azimuth bias $\left(A_{ZWB}\right)$. The resulting wind velocity components in the G-frame are:

$$\underline{V}_{WG} = \begin{bmatrix} V_W(h) \cos \left(A_{ZW}(h) + A_{ZWB}\right) \\ V_W(h) \sin \left(A_{ZW}(h) + A_{ZWB}\right) \\ w_W(h) \end{bmatrix} \tag{3}$$

It is clear from the above equation that to input vector wind data A_{ZWB} must be input as zero, whereas for meteorologic data the preloaded value of 180 deg should be used.

The wind velocity in the ECI frame is then given by

$$\underline{V}_{WI} = [IG]^{-1} \underline{V}_{WG} \tag{4}$$

Thus, the atmospheric relative velocity vector in the ECI frame is

$$\underline{V}_{AI} = \underline{V}_I - \underline{\Omega}_p \times \underline{r}_I - \underline{V}_{WI}$$

Vehicle Model

The vehicle model consists of five general categories of equations and data: (1) mass properties, (2) propulsion, (3) aerodynamics, (4) aeroheating, and (5) guidance, navigation, and control (GN&C). The mass properties model includes the calculation of vehicle and propellant weights, moments and products of inertia, and the location of the vehicle center of gravity. The propulsion model computes engine thrust and flowrate for as many as 15 separate engines. Standard equations for modeling rocket, turbojet, and ramjet engines are available in the program. The aerodynamic model includes all standard ways used to describe aerodynamic force and moment coefficients for both 3DOF and 6DOF work. The aeroheating model includes all popular heatrate calculation methods, such as the standard Chapman's equation for laminar flow and a nonstandard maximum centerline technique developed for Space Shuttle. The GN&C model includes all equations used for 3DOF openloop steering, as well as general modules for specific 6DOF simulation of an actual flight control system. These modules include: (1) a sensor module, (2) a navigation module, (3) a guidance module, (4) an autopilot module, (5) a controls module, and (6) an airframe module.

Several specific models are available in each of these general modules; however, in most cases these modules must be either modified or replaced when a new system is to be simulated. Procedures for making these modifications or additions are described in reference 5.

Mass properties.- There are numerous options available for specifying the initial gross weight of the vehicle, calculating the time rate of change of the vehicle's gross weight, and computing the amount of weight to be jettisoned at specified events. The details of these options are presented in Volume II, and only the general principles are given in this document.

The basic approach employed to compute the time history of the vehicle's gross weight is to specify only the gross weight at the first event and then subtract weight losses due to propellant consumption, ablation, and staging. Using this approach, the gross weight of the vehicle at the beginning of the simulation is given by

$$W_G = W_{STG} + W_{PLD} \tag{5}$$

where W_{STG} is the gross weight without payload and W_{PLD} is the payload weight. The weight of the vehicle during any particular phase is given by

$$W_G(t) = W_G^+ - \int \dot{W}_G \, dt \tag{6}$$

where W_G^+ is the gross weight on the positive side of the event defining the beginning of the phase, and $\dot{W}_G$ is the total derivative of gross weight resulting from engine flowrates and/or thermal protection system ablation. For events other than the first, the change in gross weight across the events is computed as

$$W_G^+ = W_G^- - W_{jett} - W_{PR} \tag{7}$$

where W_{jett} is the jettison weight, and W_{PR} is the weight of propellant remaining in the previous stage. The propellant remaining can be computed for all engines or for a single selected engine, and is given by

$$W_{PR} = W_p^i - W_{PC} \tag{8}$$

where W_p^i is the initial weight of propellant and W_{PC} is the amount of propellant consumed. This latter term is computed as

$$W_{PC} = W_{PC}^i + \int \dot{W}_p \, dt \tag{9}$$

where $\dot{W}_p$ is total flowrate for the selected set of engines, and W_{PC}^i is the initial value used to carry this parameter over a selected set of phases. When W_{PC}^i is set to zero at the beginning of a phase, then W_{PC} represents only the amount of propellant used during that phase. The jettison weight, W_{jett}, can be computed as an input constant or determined from an input mass fraction table. When mass-fraction is used to determine jettison weight, then

$$W_{jett} = W_p^i \left(\frac{1}{\lambda} - 1 \right) \tag{10}$$

where λ is the stage mass fraction computed from a user's input table. The composite center of gravity and the inertia matrix are input in the vehicle reference system as defined in Figure 14. In 6D POST, the moments and produces of inertia are defined as the integrals

$$\begin{aligned} I_{xx} &= \int y^2 + z^2 \, dv \qquad I_{xy} = \int xy \, dv \\ I_{yy} &= \int x^2 + z^2 \, dv \qquad I_{xz} = \int yz \, dv \\ I_{zz} &= \int x^2 + y^2 \, dv \qquad I_{yz} = \int xz \, dv \end{aligned} \tag{11}$$

and the inertia matrix is given by

$$[I] = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} \tag{12}$$

Generally, the center of gravity coordinates and the elements of the inertia matrix are input as tables with gross weight as the independent variable.

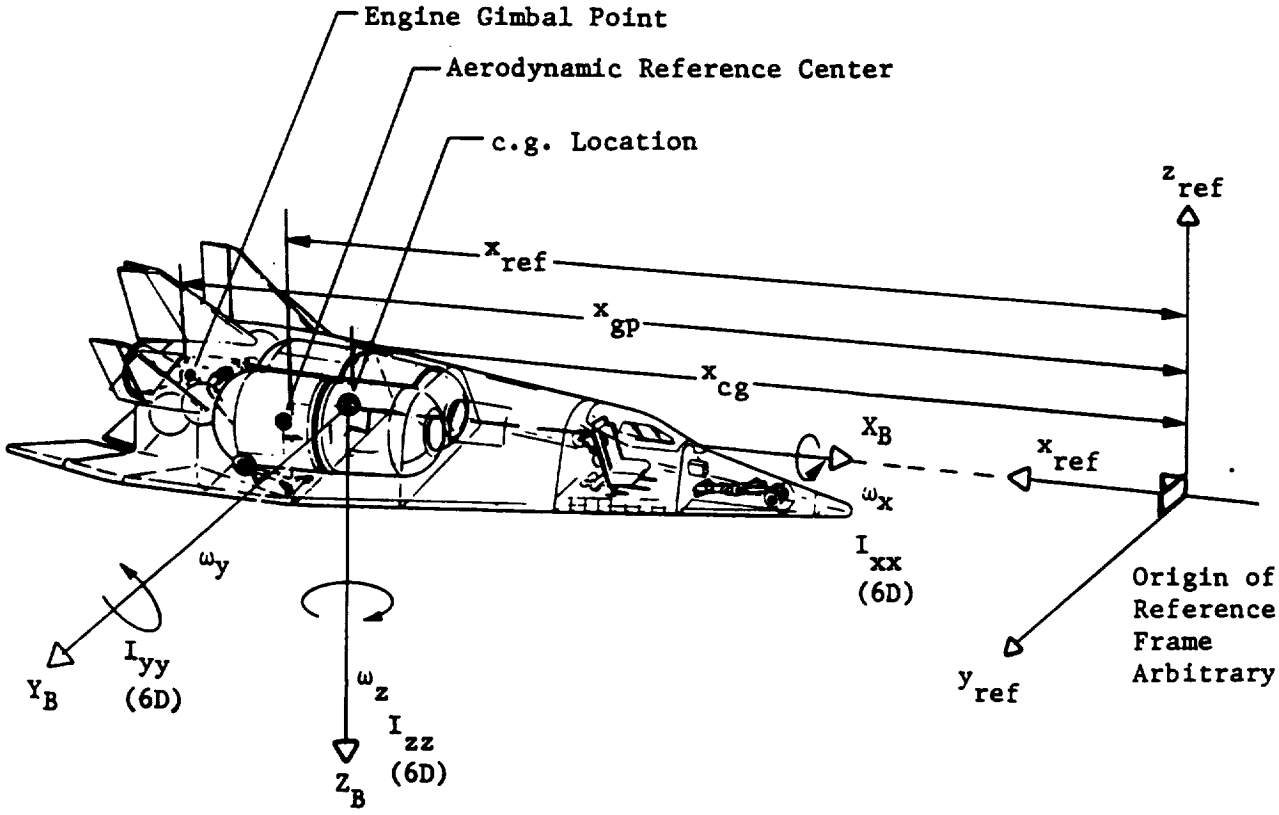
Propulsion.- POST can simulate both rocket and jet engines. As many as 15 engines can be used in either mode in any simulation phase. The equations used to calculate net thrust and flowrate per engine are summarized in Table 7. The thrust equation, computed in the Body frame, is given by

$$\underline{F}_{TB_i} = T_i \underline{u}_i \tag{13}$$

where T_i is the net thrust of the i-th engine, and $\underline{u}_i$ is a unit vector along the thrust direction. In the general case, the direction of the thrust is computed as

$$\underline{u} = \begin{bmatrix} \cos \delta e_p \cos \delta e_y \\ \sin \delta e_y \\ - \cos \delta e_y \sin \delta e_p \end{bmatrix} \tag{14}$$

where δe_p and δe_y are the pitch and yaw engine gimbal angles as defined in Figure 15. In most 3DOF simulation work, $\underline{u}$ is simply a unit vector in the direction of the x-body axis.



Mass Properties Input:

1) Moments and Products of Inertia in Body Frame (Tables)
2) Engine Gimbal Points in Body Reference Frame
3) Aerodynamic Reference Point in Body Reference Frame (Tables)
4) Center of Mass Location in Body Reference Frame (Tables)

Figure 14.- Mass Properties Input

TABLE 7.- BASIC THRUST AND FLOWRATE EQUATIONS

Rocket	Turbojet	Ramjet
Thrust $T_R = \eta T_{Vac} - A_E p(h)$ Exit Area Vacuum Thrust Throttle Setting	Thrust $T_J = \eta \frac{p}{P_{sl}} \left(T_{J/\delta}\right)$ δ = pressure ratio - $P(h)/P_{sl}$	Thrust $T_{RJ} = \eta C_T q A_E$ Thrust Coefficient
Flowrate 1) $\dot{W} = -\eta \dot{W}_P^{max}$ Maximum Propellant Flowrate 2) $\dot{W} = -\eta \frac{T_{vac}}{I_{sp_{vac}}}$ Vacuum I_{sp}	Flowrate $\dot{W} = \sqrt{\frac{T}{T_{sl}}} \frac{P}{P_{sl}} \left(\frac{sfc}{\sqrt{\theta}}\right)\left(\frac{Ti}{\delta}\right)$ Specific Fuel Consumption $\sqrt{\quad} = \sqrt{\frac{T}{T_{sl}}}$	Flowrate $\dot{W} = sfc\ T_{RJ}$

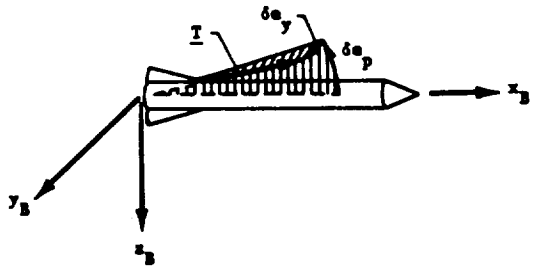


Figure 15.- Engine Gimbal Angles

Aerodynamics.- The aerodynamic force coefficients can be expressed in terms of the axial force, normal force, and side force, C_A, C_N, and C_Y, respectively. Here C_A and C_N produce forces that act in the $-x_B$ and $-z_B$ directions, and C_Y produces a force acting along y_B. The aerodynamic force coefficients can also be expressed in terms of the lift, drag, and side-force coefficients C_L, C_D, and C_Y (Figure 16), where C_L and C_D are directed normal

to, and along the velocity projection, in the x_B-z_B plane. Note that C_Y produces a side-force, $F_{A_{YB}}$, acting in the direction of y_B.

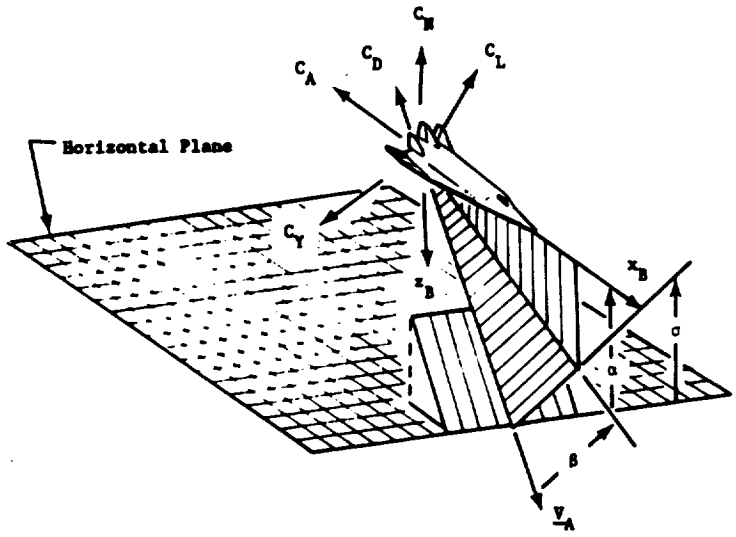


Figure 16.- Aerodynamic Angles

Lift and drag force coefficients are transformed internally to axial and normal force coefficients as follows:

$$\begin{bmatrix} C_A \\ C_N \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} C_D \\ C_L \end{bmatrix} \tag{15}$$

where α is the angle-of-attack.

The aerodynamic moments coefficients are defined as: C_ℓ - roll, C_m - pitch, and C_n - yaw. The pitch and yaw moment coefficients are used in the 3D POST static trim option, and the roll coefficient is added only in the 6 DOF version.

All aerodynamic coefficients can be input as constant, monovariant, bivariant, or trivariant tables. In general, there are four tables allocated to each coefficient in 3D POST, and eight tables per coefficient in 6D POST. Most of these tables can have arbitrary mnemonic multipliers. The mnemonic multiplier capability enables either the coefficient or its slope to be input directly into the program. The mnemonic multipliers can be input as the name of any computed variable in the output variable list. The coefficient for a given table is then the value of the table lookup multiplied by the value of the variable defined as the mnemonic multiplier. The values of the aerodynamic force coefficients are computed by summing individual contributions as defined

by user input. As an example, the axial force coefficient is computed from the general equation

$$C_A = C_{A_D} + C_A M_1 + \begin{cases} C_{A_\delta p}\ \delta p + C_{A_\delta y}\ \delta y \text{ -- 3D POST} \\ \\ C_{A_\delta a}\ \delta a + C_{A_\delta e}\ \delta a + C_{A_\delta r}\ \delta r + \sum_{i=1}^{3} C_{A_\delta f_i}\ \delta f_i \text{ -- 6D POST} \end{cases} \tag{16}$$

where each of the coefficients C_{A_0} through $C_{A_\delta f_3}$ can be defined as functions of as many as three variables, and the mnemonic multipliers M_1 through δf_3 can be defined by Hollerith input. Typical examples of C_A are:

$C_A = C_{A_0}(M) + C_A(M)\ \alpha$

C_{A_0}: monovariant tables; α: mnemonic multiplier

or

$C_A = C_A(\alpha, M)\ 1$

$C_A(\alpha, M)$: bivariant table; 1: no mnemonic multiplier

In 3D POST, δp and δy are generalized pitch and yaw control surface deflection angles. Similarly, in 6D POST δf_i, $i = 1,2,3$, are general deflection angles, and δa, δe, and δr are the aileron, elevator, and the rudder deflection angles, respectively. The detailed equations for all aerodynamic coefficients are given in the formulation manuals.

$$\underline{F}_{AB} = qS \begin{bmatrix} -C_A \\ C_Y \\ -C_N \end{bmatrix} \tag{17}$$

where the dynamic pressure q is given by

$$q = \frac{1}{2} \rho V_A^2$$

and S is the aerodynamic reference area.

Aeroheating.- POST provides for a wide variety of aeroheating calculations. Some of these options are specific in nature and apply only to particular vehicles, whereas others are quite general. The general option is based on trivariant table interpolation of a user-defined heat rate table and provides complete flexibility with regard to vehicle shape and heat transfer methods. An approximate maximum centerline heating rate option is also provided. This approximation is based on an analysis of heat rate data calculated by the MINIVER aerodynamic heating computer program. A least squares curvefit was used to obtain fifth order equations that describes the curves for reference heat rate and for the altitude-velocity and angle of attack corrections. The equations in this option were written for centerline locations aft of the nose of the vehicle. These equations are presented in the Formulation Manual. The other heat rate models are based on Chapman's equation

$$Q = K \frac{17600}{\sqrt{R_N}} \left(\frac{\rho}{\rho_{SL}}\right)^{\frac{1}{2}} \left(\frac{V_R}{V_c}\right)^{3.15}$$

where the scale factor, K, can be computed from a table look-up.

In addition to the basic heat rate calculation, POST contains several aerodynamic heating indicators that provide useful information associated with the heating environment. One such indicator for zero total angle of attack is

$$Q_I = \int_o^t qV_A \, dt,$$

which can be modified for various nonzero angle of attack situations.

These heat rate indicators can be used in conjunction with a ten panel vehicle heating model to incorporate the heating calculation in the vehicle weight calculations.

In this simplified model, the total heat for each panel is assumed to be a constant ratio of the total heat calculated by the selected option at a given location on the vehicle. The total thermal protection system (TPS) weight is then computed as the sum of the individual panel weights,

$$W_{TPS} = \sum_{i=1}^{10} W_{uA_i} A_i$$

where W_{uA_i} is the weight per unit area and A_i is the area of the i-th panel.

Guidance and Control.- POST can simulate both open loop and closed loop guidance. In 3D POST, the vehicle's actual attitude is computed directly from an attitude equation (polynomial, table, etc.) or from a user-programmed guidance equation. This approach simulates a *perfect* autopilot where the actual and the commanded attitudes are identical. In 6D POST, the complete flight control system can be simulated in substantial depth. However, this generally requires that the user code the detailed models associated with the particular sensor, navigation, guidance, autopilot, and control systems being simulated.

The sensor module computes information that describes the behavior of the sensing elements of the vehicle's navigation system. Thus, the primary functional responsibility of the module is that of simulating hardware characteristics of sensors. For example, the behavior of an inertial measurement unit (IMU) can be described by a mathematical model of the platform and the accelerometers. Frequently this module is used for error analysis purposes.

Sensor models called by this module are necessarily vehicle and subsystem dependent. As a consequence, the sensor model must be designed and implemented for each particular application.

There are many applications of the program that do not require a specific simulation of the sensors. Therefore, for convenience, a "perfect" sensor model is coded into this routine. This "perfect" sensor model sets the sensed program variables equal to their actual values as calculated in the simulation models.

The function of the navigation model is to estimate the state (position, velocity, etc.) of the vehicle based on the sensor outputs. Clearly, this module is also vehicle and subsystem dependent and must be designed and implemented for each specific application. The 6D POST contains no built-in navigation models. As a consequence, the estimated state is set equal to the actual state. This is also equivalent to simulation of perfect navigation.

The guidance module takes the output of the navigation model and computes a guidance command. Typically, the guidance command represents a desired change in the current attitude of the vehicle. This command is computed on the basis of meeting some specified trajectory condition, such as an inject condition or a landing condition. The autopilot is designed to remove the errors between the commanded values of the guidance variables and their actual (or sensed) values. This is accomplished by deflecting engines, control surfaces, and/or firing RCS jets.

The current version of 6D POST contains three preloaded guidance options: (1) an open-loop profile steering; (2) a closed loop v-h profile ascent algorithm; and (3) the constant drag Space Shuttle reentry scheme. If these methods are inadequate, the user may implement his own guidance algorithm into this module.

The function of the autopilot module is to generate a command that, when implemented through the deflection equations contained in the controls module, causes the vehicle to respond as prescribed by the guidance module. This functional responsibility is depicted in Figure 17.

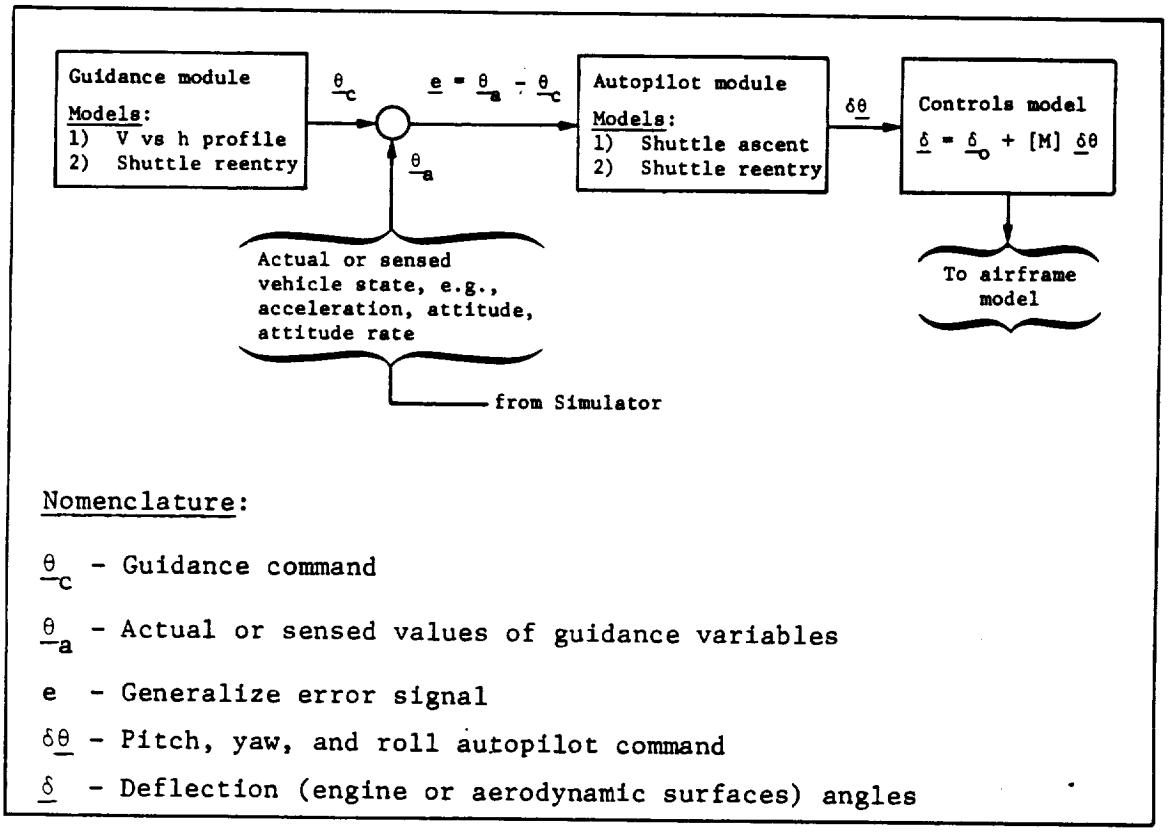


Figure 17.- Guidance and Control Block Diagram

The autopilot module calculates only autopilot commands based on the input guidance commands, and does not calculate engine or control surface deflections. The engine and control surface deflections are computed in the controls module as a linear function of the autopilot commands. The autopikot commands $\delta\theta$, $\delta\phi$, $\delta\psi$ represent changes in vehicle attitude. The mixing equations determine the engine and control surface deflections that create the control forces and moments.

Currently, there are two Space Shuttle autopilot models available in 6D POST. One autopilot is for ascent and the other for reentry. The ascent autopilot is somewhat standard and could be used on most ascent problems with little or no modification. The basic inputs to this model are: attitude commands from the guidance, inertial attitude angles, body rotational rates, translational accelerations, and preloaded engine deflection commands. The outputs are pitch, yaw, and roll autopilot commands, which are sent to the controls module to determine the engine deflection angles. The reentry autopilot is Space Shuttle-oriented and is probably not applicable to other vehicle

configurations. This model is intended to provide attitude control for Space Shuttle beginning at an altitude of approximately 400 000 ft and ending in the high subsonic flight regime. The control logic makes use of both aerodynamic control surfaces torques and reaction control jets.

The controls model converts pitch, yaw, and roll autopilot commands to aerodynamic control surface deflection angles and/or engine gimbal angles. The conversion of the autopilot commands into deflection angles is implemented through the matrix mixing logic given by the equation

$$\underline{\delta} = \underline{\delta}_o + [M]\ \delta\underline{\theta} \tag{18}$$

where $\underline{\delta}$ denotes a general deflection angle with a null position of $\underline{\delta}_o$, [M] the mixing gains, and $\delta\underline{\theta}$ the autopilot commands. The gains contained in the mixing matrix, [M], and the null deflections, $\underline{\delta}_o$, are specified by user input.

In 3D POST, there are five basic types of openloop guidance options for controlling the attitude of the vehicle during 3 DOF trajectory simulation. These options are as follows:

1) Body rates;

2) Aerodynamic angles;

3) Inertial aerodynamic angles;

4) Relative Euler angles;

5) Inertial Euler angles.

The body rate option is generally used to simulate strapdown systems with the body rates being computed from user-specified polynomials. The attitude of the vehicle is then determined by integration of the quaternary equation

$$\dot{\underline{e}} = \frac{1}{2}\,[E]\ \underline{\omega}. \tag{19}$$

When using this option the user must specify (1) the initial attitude of the vehicle, and (2) the coefficients and the Hollerith names of the arguments of the polynomials used to compute the body rates.

Atmospheric and inertial velocity relative aerodynamic angles are generally used to simulate reentry and orbital maneuvers, respectively. Similarly, relative and inertial Euler angles are typically used to simulate vehicles that employ local horizontal or inertial reference systems. In all of these applications, the attitude angles can be computed based on five basic techniques:

1) Polynomial steering: Under this option the steering angles are computed from a cubic polynomial

$$\theta(y) = \sum_{i=0}^{3} c_i y^i \tag{20}$$

where the user selects the coefficient c_i and the independent variable y. The highest-order coefficient that is input determines the degree of the polynomial. The argument can be selected as any internally computed variable; e.g., time, velocity, altitude, etc. The constant term of the polynomial, c_o, can be either input at the beginning of each phase or carried across as the value of the angle at the end of the previous phase. The polynomial coefficients are generally used as the independent variables for targeting/optimization.

2) Table steering: Under this option the steering angles are computed via table interpolation, which is denoted by

$$\theta(\underline{y}) = \theta_m \, T^n[f(\underline{y})]. \tag{21}$$

The user initially inputs the table multiplier θ_m, the order of interpolation n, and the table data $[\underline{y}, f(\underline{y})]$. The table multiplier or the dependent table values can be used as independent variables for targeting or optimization. The order of interpolation can either be linear or cubic. The tables can be monovariant, bivariant, or trivariant functions of the table arguments.

3) Piecewise linear steering: Under this option the steering angles are computed from a general piecewise linear function of the form

$$\theta(y) = c_1 + \left[\frac{c_2 - c_1}{y_2 - y_1}\right](y - y_1) \tag{22}$$

where c_1 is equal to θ at the beginning of the current phase, c_2 is the desired value of θ at a designated later event, y_1 is equal to the value of the designated event criterion at the beginning of the current phase, y_2 is the desired value of the designated event, and y is the current value of the designated event criterion.

This option is similar to the polynomial option except that the values of θ are specified directly rather than as θ_0, $\dot{\theta}$, $\ddot{\theta}$ and $\dddot{\theta}$. Clearly, θ is linear in time if y = t; otherwise θ is only linear in y. When the desired values of the steering angles are used as independent variables, the problem of cascaded steering effects is avoided and the targeting/optimization algorithm generally converges faster. This option also automatically computes the steering angle rates required

to change the attitude to the desired value at the designated event, which reduces the problems related to guessing accurate initial pitch rates.

4) Linear feedback steering: Under this option the steering angles are computed from the linear error-error rate feedback control law

$$\theta = c_1 + K_D\left(F_a - F_d\right) + K_R\left(\dot{F}_a - \dot{F}_d\right) \tag{23}$$

where c_1 is a nominal steering angle profile, K_D is the displacement error gain, K_R is the rate gain, $F_a - F_d$ is the error in the steering function error.

This option is, of course, the classic path control law, and enables the user to steer to a wide variety of trajectory profiles, such as velocity versus altitude profile, acceleration versus altitude profile, etc. This option is particularly useful for reentry trajectory shaping.

5) Generalized acceleration steering: Under this option the steering angles are computed by solving a set of user-specified equations. The dependent and independent variables in these equations must be selected from the dictionary of variables already computed in POST. The only restriction is that these equations must be explicitly a function of some derivative compound in the inner loop of the program.

In more precise terms, the steering variables are determined from the iterative solution of the problem:

For each instant of time, determine the values of the steering variables, $\underline{\theta}$, that satisfy the steering equations,

$$\underline{e}(\underline{\theta}) = \underline{y}(\underline{\theta}) - \underline{y}_d = \underline{0} \tag{24}$$

where $\underline{y}$ is a n-component vector of dependent variables, $\underline{y}_d$ is the desired value of these variables and $\underline{e}$ the error in dependent variables. A typical application of this option is control normal acceleration to one-g and axial acceleration to three-g by calculating the angle of attack and throttle setting that solves the equations

$$\begin{aligned} A_{xB}(\alpha, \eta) - 96.6 &= 0 \\ A_{zB}(\alpha, \eta) - 32.2 &= 9 \end{aligned} \tag{25}$$

Trajectory Simulation

The trajectory simulation features consist of the integration of both the translational and the rotational equations of motion, the event interrupt and sequencing logic, and the numerical (in some cases analytical integration) methods. The flexibility that the user has in selecting these three features is the key to the utility of the POST programs.

Translational equations.- The translational equations of motion are solved in the Earth-centered inertial coordinate system (ECI). These equations are

$$\dot{\underline{r}}_I = \underline{V}_I$$

$$\dot{\underline{V}}_I + [IB]^{-1} \left[\underline{A}_{TB} + \underline{A}_{AB}\right] + \underline{G}_I \qquad (26)$$

where $\underline{A}_{TB}$ is the thrust acceleration in the body frame, $\underline{A}_{AB}$ is the aerodynamic acceleration in the body frame, and $\underline{G}_I$ is the gravitational acceleration in the ECI frame. The external accelerations are first computed in the body frame and then transformed to the ECI frame. The external forces that cause these accelerations are illustrated in Figure 18. In 6D POST, the net translational force due to the reaction control system are also included in the total non-gravitational force acting on the vehicle.

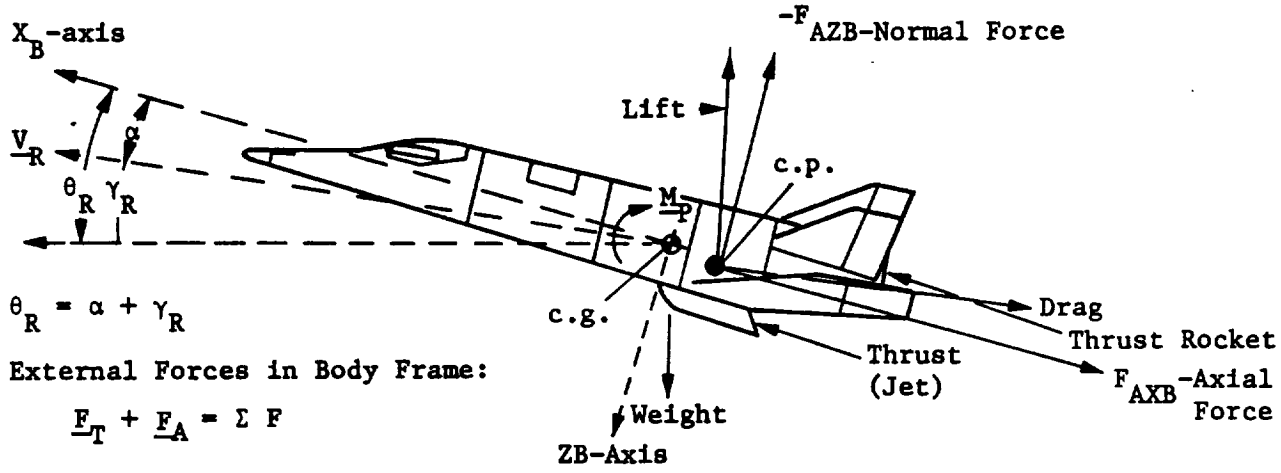


Figure 18.- Force System in Pitch Plane

There are a number of options for initializing the translational equations of motion. These options are summarized in Table 8. The most frequently used option is the planet relative spherical coordinates (ϕ, θ, h), and the local horizontal velocity components $\left(V_R, \gamma_R, A_{ZR}\right)$.

TABLE 8.- TRAJECTORY INITIALIZATION OPTIONS

Position	Velocity	Attitude	Angular Velocity
Inertial Rectangular $\left(x_I, y_I, z_I\right)$	Inertial Rectangular $\left(V_{xI}, V_{yI}, V_{zI}\right)$	Inertial Euler Angles $\left(\phi_I, \psi_I, \theta_I\right)$	Body Rates $\left(\omega_x, \omega_y, \omega_z\right)$
Earth - Relative Polar (r, ϕ, θ)	Inertial Local Horizontal $\left(V_I, \gamma_I, A_{ZI}\right)$	Relative Euler Angles $\left(\psi_R, \theta_R, \phi_R\right)$	Inertial Euler Rates $\left(\dot{\phi}_I, \dot{\alpha}_I, \dot{\theta}_I\right)$
Orbital Element (a, e, i, Ω, ω, θ)	Earth-Relative Local Horizontal $\left(V_R, \gamma_R, A_{ZR}\right)$	Aerodynamic Angles (α, β, σ)	Relative Euler Rates $\left(\dot{\psi}_R, \dot{\theta}_R, \dot{\phi}_R\right)$
	Atmospheric Relative Local Horizontal $\left(V_A, \gamma_A, A_{ZA}\right)$		
	Orbital Elements (a, e, i, Ω, ω, θ)		

Rotational equations.- In 6D POST, the rotational equations of motion are solved in the body-centered coordinate system. These equations are

$$\dot{\underline{e}} = \frac{1}{2} [E] \underline{\omega}_B$$

$$\dot{\underline{\omega}}_B = [I]^{-1} \left[\underline{M}_B - [\dot{I}] \underline{\omega}_B - \underline{\omega}_B \times [I] \underline{\omega}_B \right]$$

$$\underline{M}_B = \underline{M}_{TB} + \underline{M}_{AB} \tag{27}$$

where

$$\underline{M}_{TB} = -\sum_{i=1}^{N_{eng}} \underline{F}_{TB_i} \times \Delta \underline{R}_{BT_i}$$

and

$$\underline{M}_{AB} = qS\begin{bmatrix} d_R\ C_\ell \\ d_P\ C_m \\ d_Y\ C_n \end{bmatrix} - \underline{F}_{AB} \times \Delta\underline{R}_{AB}$$

$$\Delta\underline{R}_{AB} = \begin{bmatrix} -x_{ref} - x_{cg} \\ y_{ref} - y_{cg} \\ -z_{ref} - z_{cg} \end{bmatrix}$$

In the rotational equations, $\underline{e}$ is a four-dimensional vector of quaternary parameters, [E] is the quaternary matrix, $\underline{\omega}_B$ is the inertial angular velocity expressed in the body frame; $\underline{M}_B$ is the total external moment acting in the vehicle as a result of the thrust, the RCS, and the aerodynamic forces, and [I] and [E] matrices are given by

$$[I] = \begin{bmatrix} I_{XX} - I_{XY} - I_{XZ} \\ -I_{XY}\ I_{YY} - I_{YZ} \\ -I_{XZ} - I_{YZ}\ I_{ZZ} \end{bmatrix}$$

$$[E] = \begin{bmatrix} -e_1 & e_2 & e_3 \\ e_0 & e_2 & -e_3 \\ e_0 & -e_1 & e_3 \\ e_0 & e_1 & -e_3 \end{bmatrix} \qquad (28)$$

where the inertia matrix, I, is not necessarily assumed to be constant-valued. The rotational equations of motion are initialized by defining both the initial attitude angles and the initial attitude rates. There are three options for initialization as summarized in Table 7. The equations for each of these options are presented in reference 6.

Events and phases.- The concepts involved with the definition of the trajectory sequence of events, and the relationships between these events and simulation phases are fundamental to the use of POST programs. As a result, special emphasis should be placed on clearly understanding these simple yet key concepts.

In POST terminology, an event is defined in terms of three critical elements: (1) an event number, (2) event criteria, and (3) an event criteria value. These three elements combine to define a condition, which when it occurs, caused the trajectory simulation to be interrupted. The ability to interrupt the simulation based on any user-defined condition can be used for a variety of purposes. For example, an interrupt can be used to change vehicle data, such as aerodynamics or propulsion; or to change simulation control data, such as integration methods and so on.

Mathematically, the i-th event criteria defines a scalar-valued continuous function, $y_i(t)$, and the event numbers index and, in most cases, order these event criteria. The event criteria value, y_i^d, is used in conjunction with the event criteria, $y_i(t)$, to define the event interrupt equation

$$y_i(t) - y_i^d = 0 \tag{29}$$

The time at which the i-th event occurs is then computed by iteratively solving this equation for the value $t = t_i$, where $y_i(t_i) = y_i^d$. This concept is illustrated in Figure 19.

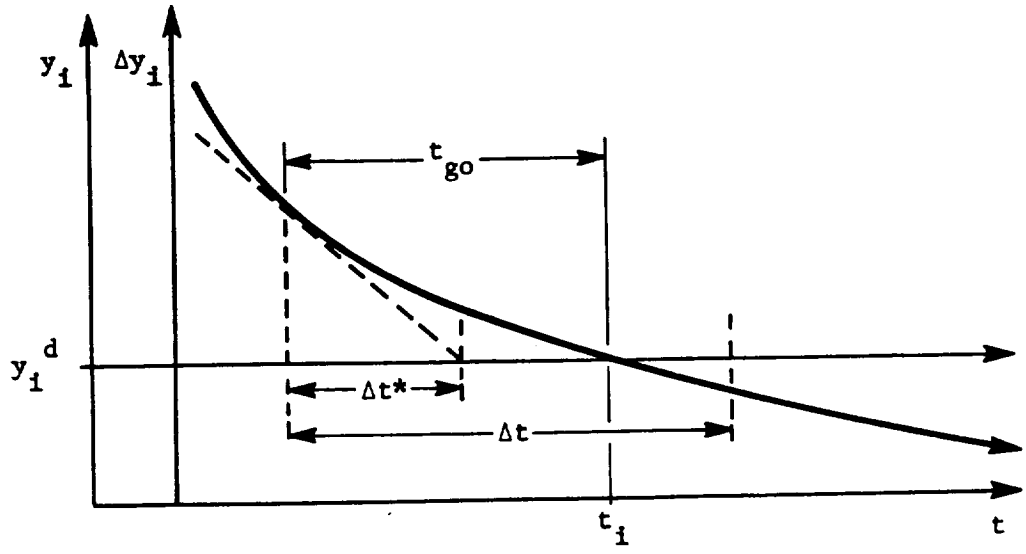


Figure 19.- Illustration of Time-to-Go Logic

Each event (other than the first event) serves to (1) terminate the previous phase, and (2) initiate the following phase. This can be explained as follows. Event i is said to occur at time $t = t_i$, and the plus (+) and minus (-) sides of the i-th event occur at the left and right limit times, t_i^+ and t_i^-, respectively. The trajectory data at time t_i^-, and in particular the last printblock in each phase, are computed using the simulation data defined in phase i-1. Similarly on the plus side of the event, at t_i^+, the trajectory data are recomputed using the new data (if any) defined by input for phase i. This subtle concept is very important in the simulations where mass is discontinuous due to staging, or accelerations are discontinuous because of changes in propulsion and configuration characteristics.

Four types of events have been defined to provide flexibility in setting up a given problem:

1) Primary events: These describe the main sequential events of the trajectory being simulated. These events must occur, and must occur in ascending order according to the event number. Most problems will usually be simulated by a series of primary events;

2) Secondary events: These are events that may or may not occur during the specified trajectory segment. Secondary events must occur in ascending order during the interval bounded by their primary events. The occurrence of a primary event will nullify the secondary events associated with the previous primary event if they have not already occurred;

3) Roving primary events: These events can occur any time after the occurrence of all primary events with smaller event numbers. They can be used to interrupt the trajectory on the specified criterion regardless of the state of the trajectory or vehicle.

4) Repeating roving events: These events are the same as primary roving events except the interrupt values are input differently. There are two options for criteria value specifications. Option 1: Input the initial value, the increment, and the number of times the event is to be repeated. Option 2: Input an array of event criteria values.

The cycling routine monitors as many as ten events at a time, depending on the types of events to determine which event is to occur next.

Multiple events are monitored in the following sequence:

1) The next primary event is monitored;

2) As many as nine primary roving events are then monitored, provided there are no secondary events. A roving primary event is added to the list of those being monitored as soon as the primary event immediately preceding that roving event has occurred;

3) Next, as many as nine secondary events are monitored, provided there are no primary roving events. (Note that caution must be exercised when using secondary events because of their nature. Because as many as nine secondary events are monitored at a time, any one of those nine will occur as soon as its criterion has been met. Because they are secondary events, the event that occurs will cancel all secondary events with smaller event numbers.);

4) Finally, a total of nine primary roving and secondary events are monitored.

Because the program can only monitor nine events (in addition to the next primary event), the sum of the primary roving events and the secondary events in a phase must be less than or equal to nine or a fatal error will result.

The time-to-go model iterative solves the event criteria equations and determines when the events occur during the trajectory simulation. Basically, the algorithm checks the values of the criteria being monitored at each integration step. If none of the criteria values has bracketed the desired cutoff value, then another integration step is taken. If a criterion variable is bracketed with the input step size, then a new stepsize is computed equal to the predicted time-to-go.

The predicted time-to-go for each event is computed from the basic secant equation

$$\Delta t^* = -\Delta y_i(t)\ \Delta t/[\Delta y_i(t + \Delta t) - \Delta y_i(t)] \tag{30}$$

where $\Delta y(t)$ is the difference between the actual and the desired value of the event criterion. If more than one event is bracketed, then the minimum predicted time-to-go is used as the integration stepsize. This process is repeated until the criterion value is within the specified tolerance of the desired value. Again see Figure 19.

In POST, the simulation data are input by phase, where each phase is defined by two events--the event initiating the phase and the event terminating the phase. As a result, the user is required to define via events the sequence of trajectory phases that specify from beginning to end the problem being simulated. Physically the data for each phase are located in the data deck between the Hollerith specification of the events defining the phase.

Specific event numbers, which are selected by the user, must satisfy certain conditions based on the type of events employed for a given problem. In general, the event numbers must be monotonic increasing, but they need not be consecutive.

Integration methods.- There are three general purpose numerical integration methods available as automatic program options: (1) standard fourth order Runge-Kutta, (2) an eighth order Runge-Kutta, and (3) a variable-step variable-order predictor-corrector. The vast majority of users employ the standard

Runge-Kutta integrator for atmospheric flight simulation. However, recent computational experience with the predictor-corrector method indicate that it is substantially more efficient on certain classes of problems, particularly those requiring orbital propagation. This method, developed by F. T. Krogh of the Jet Propulsion Laboratories, represents the state-of-the-art in numerical solution of systems of ordinary differential equations. It includes all of the following facilities:

1) A core integrator for advancing the solution from one uniform step to the next consisting of variable order Adams predictor-corrector equations requiring the storage of a difference table for only the highest ordered derivatives;

2) A method to start integration with first-order equations and to increase the order to as high a level as numerical stability permits;

3) Algorithms for changing the stepsize and updating accordingly the difference tables of the highest-order derivatives including appropriate smoothing to prevent numerical instability;

4) Algorithms for deciding when and by how much to change the stepsize based on the accuracy requested by the user;

5) Tests for numerical stability of the predictor-corrector order and stepsize tentatively chosen in the context of the current differential equation set;

6) Algorithms for the automatic selection of the core integrator to fit the characteristics of the set of differential equations at hand;

7) An interpolation algorithm for obtaining dependent variable values to the user-specified accuracy at values of the independent variable different from normal integration steps.

In addition to these general purpose numerical methods, 3D POST contains two special integration methods for propagating orbital trajectories: (1) Laplace's Method, and (2) Encke's Method.

Laplace's method is an iterative technique for propagating the position and velocity (in planet-centered coordinates) of a nonthrusting vehicle in vacuum during flight over a spherical planet. The technique is based on the analytical solution of the two-body equations, and yields the inertial state at time $t + \Delta t$ as a function of the state at time t.

The equations used in Laplace's method are:

$$\underline{r}_I(t + \Delta t) = f\underline{r}_I(t) + g\underline{V}_I(t)$$

$$\underline{V}_I(t + \Delta t) = \dot{f}\underline{r}_I(t) + \dot{g}\underline{V}_I(t) \qquad (31)$$

The coefficients f, g, $\dot{f}$, and $\dot{g}$ are computed analytically from the change in eccentric anomaly during the time period Δt. The change in eccentric anomaly is calculated by solving a special version of Kepler's equation via the Newton-Raphson algorithm.

The Encke method used in 3D POST is modified from the usual Encke technique in that it rectifies the reference conic at every integration step and does not use the standard Q-series expansion in calculating the gravitational increment.

The Encke method should be used for orbital problems where small perturbing accelerations, such as the oblateness of the planet, atmospheric drag, or solar electric propulsion, must be included in the simulation. Numerical results indicate that, for problems involving small perturbations from Keplerian motion, Encke's method is approximately four times faster than Cowell methods, which integrate the total acceleration.

The Encke method determines the total motion by summing the motion due to the two-body equations and the motion due to the perturbations to the two-body equations. The position and velocity in the inertial planet-centered system at time $t + \Delta t$ is given by

$$\underline{r}_I(t + \Delta t) = \underline{r}_2(t + \Delta t) + \Delta\underline{r}(t + \Delta t)$$

$$\underline{V}_I(t + \Delta t) = \underline{V}_2(t + \Delta t) + \Delta\underline{V}(t + \Delta t) \tag{32}$$

where $\underline{r}_2$, $\underline{V}_2$ denotes the Keplerian motion computed by Laplace's equations; that is,

$$\underline{r}_2(t + \Delta t) = f\underline{r}_2(t) + g\underline{V}_2(t)$$

$$\underline{V}_2(t + \ t) = \dot{f}\underline{r}_2(t) + \dot{g}\underline{V}_2(t)$$

and $(\Delta r, \Delta V)$ denotes the numerical solution of the differential equations

$$\Delta\dot{\underline{r}} = \Delta\underline{V}$$

$$\Delta\dot{\underline{V}} = \left[[IB]^{-1}\left[\underline{A}_{TB} + \underline{A}_{AB}\right] + \underline{G}_I\right] - \underline{g}_2\left(\underline{r}_2 + \Delta\underline{r}\right)$$

$$\Delta\underline{r}(t) = \Delta\underline{V}(t) = \underline{0},$$

where $\underline{g}_2\left(\underline{r}_2 + \Delta\underline{r}\right)$ is the two-body acceleration at $\underline{r}_2 + \Delta\underline{r}$.

Special Purpose Options

In 3D POST, there are numerous special purpose options that aid the user in modeling special situations that frequently arise in trajectory analysis. These options efficiently simulate these special situations by using specific equations that are derived on the basis of the assumptions employed. For example, many orbital maneuvers can be adequately modeled as a discontinuity in velocity. This special case does not require numerical solutions and is modeled using the classic rocket equation. This and other key special purpose options are summarized in Table 9.

Auxiliary Calculations

In addition to computing the basic variables, POST also computes numerous auxiliary variables that are related to (1) conic parameters, (2) range calculations, (3) tracking data, (4) analytic impact calculations, (5) velocity losses, and (6) velocity margins. Table 10 summarizes the auxiliary variables that can be optionally computed. All these auxiliary variables can be used as independent variables in the targeting and optimization, as event criteria in the simulation, or as output variables of general interest. Special print blocks can be requested for conic calculations, tracking station data, and velocity loss information. These print blocks are arranged in a convenient format and contain all of the auxiliary variables associated with the particular type of auxiliary output requested. In some cases, these complete special purpose print blocks are not required. If so, the particular output variable desired can be added to the normal trajectory print block by following the sample procedures as outlined in references 1 and 7.

Targeting and Optimization

The targeting and optimization capability in POST provides the trajectory analyst with a set of numerical tools that enable him to solve a broad spectrum of trajectory design problems. This capability is achieved by first providing the user with the option to specify a variety of problem types, such as full rang targeting, unconstrained optimization, and constrained optimization including equality and inequality constraints. Second, the trajectory analyst is provided the capability to define the details of the problem formulation using simple and direct input procedures. These procedures enable him to select the optimization variable, the constraints, and independent control variables by specifying only their Hollerith names, their event numbers, the constraint values and tolerances, and an initial guess for the control variables. Finally, the trajectory analyst is provided a library of numerical algorithms that can be used to iteratively solve any standard discrete parameter trajectory problem formulation. These algorithms include the popular accelerated gradient projection method for nonlinear programming formulations, Newton-Raphson for full rank targeting formulations, and the conjugate gradient or the Davidon methods for unconstrained optimization or constrained optimization employing penalty functions.

TABLE 9.- SPECIAL PURPOSE OPTIONS

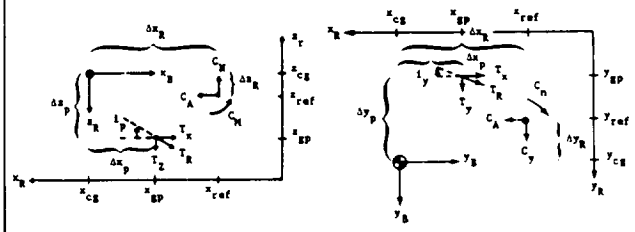
DESCRIPTION	BASIC EQUATIONS
INSTANTANEOUS VELOCITY ADDITION - At the beginning of each event an instantaneous velocity can be added. The direction of the impulse is along the thrust vector, which usually coincides with the logitudinal axis of the vehicle. The magnitude of the velocity addition can be input directly or calculated from the amount of propellant used. The direction of the impulse can be controlled via the attitude of the vehicle and/or the engine gimbal angles.	1) Calculate ΔV given the gross weight, W_G, and the amount of propellant consumed, W_{pc}. $\Delta V = g\, I_{sp}\, \ln\left(W_G / (W_G - W_{pc})\right)$ 2) Calculate W_{pc} given W_G and ΔV. $W_{pc} = W_G \left(1 - \exp\left(-\Delta V / g\, I_{sp}\right)\right)$ 3) Calculate the Inertial impulse vector given ΔV, the vehicle attitude, and the engine gimbal angles. $\underline{V}_I^+ = \underline{V}_I^- + \Delta V\, [IB]^{-1}(\underline{u})$
STATIC TRIM - The static trim option is used to calculate the engine gimbal angles or the flap deflection angles required to balance the pitch and yaw moments caused by thrust and aerodynamic forces. The static moment equation, in pitch and yaw, $M_P = A_{MP} + T_{M1P} + T_{M2P}$ $M_Y = A_{MY} + T_{M1Y} + T_{M2Y}$ is nonlinear and an iterative algorithm is used to evaluate the required solutions at every instant of time.	 Moments in Pitch Moments in Yaw
HOLDDOWN FOR VERTICAL TAKEOFF - This option is used to simulate vertical (rocket type) takeoff. When using this option, the relative position and velocity remains constant and the inertial position changes by the rotation of the Earth. The inertial velocity magnitude remains constant while its direction changes. This model simulates physical constraints that hold the rocket on the launch pad until the rocket is released.	The vector equation used to compute the accelerations that simulates the effects of the physical holddown forces is $\underline{A}_I = \underline{\Omega} \times \underline{V}_I$
HOLDDOWN FOR HORIZONTAL TAKEOFF - This option is used to simulate horizontal (aircraft type) takeoff. When using this option the vehicle accelerates in the local horizontal plane according to the forces described by the user input. The vertical component of acceleration is internally computed to produce the proper horizontal motion.	The accelerations used to simulate horizontal takeoff are: $\underline{A}_{XG} = [IG]\ \underline{A}_{SI}$, $\begin{bmatrix} A_V \\ A_H \\ A_C \end{bmatrix} = \begin{bmatrix} -A_{ZG} \\ \sqrt{A_{XG}^2 + A_{YG}^2} \\ \left(V_{XG}^2 + V_{YG}^2\right) / r_I \end{bmatrix}$, and $\underline{A}_I = [IG]^{-1}\ \underline{A}_{XG}$
ORBITAL PROPAGATION MODEL - The integration step, Δt, is generally specified in terms of an increment in time. However, this option enables the user to specify the integration step in terms of an input increment in true anomaly. This option is useful in orbital problems where the geometry is easily expressed as a function of true anomaly.	The following equations are then used to calculate Δt as a function of $\Delta\theta$. $\theta_2 = \theta_1 + \Delta\theta$ $E_2 = 2 \tan^{-1}\left\{\left(\frac{1-e}{1+e}\right)^{\frac{1}{2}} \tan\frac{\theta_2}{2}\right\}$ $r_2 = a\ 1 - e \cos E_2$ $\Delta E = E_2 - E_1$ $\Delta t = \sqrt{\frac{a^3}{\mu}}\ \Delta E - \sin\Delta E + \frac{r_2 r_1}{H} \sin\Delta\theta$: In these equations, subscripts 1 and 2 denote current and future values, respectively.

TABLE 10.- SUMMARY OF AUXILIARY CALCULATIONS

Position and Velocity

Geocentric Radius
Oblate Atitude
Inertia Velocity Magnitude
Planet Relative Velocity
Atmosphere Relative Velocity
Inertial Flight Path Angle
Planet Relative Flight Path Angle
Atmospheric Relative Flight Path Angle
Inertial Velocity in Geographic Frame
Relative Velocity in Geographic Frame
Atmospheric Velocity in Geographic Frame
Inertial Azimuth
Relative Azimuth
Geocentric Latitude
Geodetic Latitude
Inertial Longitude
Relative Longitude
Sensed Acceleration in Body Frame
Sensed Acceleration in ECI Frame

Conic Variables (Elliptic and Hyperbolic)

Orbital Energy
Apogee Radius and Altitude
Perigee Radius and Altitude
Semimajor Axis
Eccentricity
Inclination
Period
Argument of Perigee
True Anomaly
Longitude of Ascending Node
Angular Momentum
Perigee Latitude
Perigee Longitude
Argument of the Vehicle
Time to Perigee Passage
Time Since Perigee Passage
Eccentric Anomaly
Mean Anomaly
Perigee Velocity
Apogee Velocity
Velocity Required to Circularize
Circular Velocity
Longitude of the Vernal Equinox
Velocity Margin

Range Calculations

Dot Product Range
Crossrange Reference
  Inertial Reference
  Earth Relative Reference

Auxiliary Attitude Calculations

Aerodynamic Angles
Inertial Euler Angles
Relative Euler Angles
Inertial Aerodynamic Angles
Body Rates
Inertial Euler Angle Rates
Relative Euler Angle Rates

Tracking Data

Slant Range (Vector & Magnitude)
Elevation Angle
Azimuth Angle
Cone Angle
Clock Angle
Space Losses

Analytic Impact Calculations

Geodetic Latitude of Impact
Relative Longitude of Impact
Altitude of Impact
Position Vector at Impact
Velocity Vector at Impact

Velocity Losses

Drag Loss (Inertial & Relative)
Thrust Vector Loss (Inertial & Relative)
Atmospheric Pressure Loss (Inertial & Relative)
Gravity Loss (Inertial & Relative)
Ideal Velocity

Sun-Shadow Calculations

Position and Velocity in Vernal Equinox System
Greenwich Hour Angle
Cone and Clock Angles of Sun Vector
Shadow Function

Multiple Vehicles

Conic Variables
Position and Velocity Variables
Accelerations

Aerodynamic & Aeroheating

Aerodynamic Coefficients
Aerodynamic Forces & Moments
Dynamic Pressure
Total Angle of Attack
Dynamic Pressure & Angle of Attack
Mach Number
Reynolds Number
Heat Rate
Total Heat
Atmospheric Densigy Temperature, Pressure
  and Speed of Sound

Propulsion Calculations

Vacuum Thrust
Net Thrust
Total Thrust & Accelerations
Flowrate (Total)
Partial Flowrate
Propellant Consumed
Propellant Remaining
Throttle Setting
Gimbal Angles

All the targeting and optimization procedures used in POST are completely numerical. This means that all the function evaluations and derivatives used by the various algorithms, are computed numerically from only the trajectory simulation results. Thus, the user needs only to define a problem formulation that is consistent with the vehicle simulation, and the program automatically does the rest. This numerical approach, although somewhat slower in execution speed than some analytical methods, offers the advantages of "hands off" flexibility and a significantly lower program maintenance overhead.

Problem formulation.- Discrete parameter methods provide the basis of the targeting and optimization formulation and solution algorithms employed in POST. The use of the discrete parameter approach means that trajectory control variables are defined by a finite set of parameters, called independent variables, and given values for these variables the trajectory can be uniquely computed. A good example of this approach is the representation of the vehicle's pitch attitude by a piecewise linear function. In this parameterization, the nodes, θ_i, and the pitchrates, $\dot{\theta}_i$, are the independent variables that can be varied to steer the vehicle. A near optimal pitch program can be generated using these parameters to optimize a selected performance criterion, say payload, subject to both mission and vehicle constraints. In recent years, this discrete parameter approach to trajectory optimization has replaced the more complex calculus of variation techniques. This is because the discrete parameter methods are simpler to implement, understand, and use, and, as a result, are substantially more reliable. The reason for this is that discrete parameter methods are based on ordinary vector calculus, which is easier for most practicing trajectory analysts to understand than the complex procedures required in most variational trajectory programs.

Using discrete parameter concepts, a vast diversity of trajectory optimization and targeting problems can be formulated that have a common mathematical structure. First as described above, there is a vector $\underline{u}$ of control parameters which must be selected to define the trajectory

$$\underline{x}(t) = \Phi\ \underline{x},\underline{u},t:\underline{x}_0 \tag{33}$$

where the state variable $\underline{x}$ is typically composed of the components of the vehicle's position, velocity, and mass (or weight) and $\Phi[\cdot]$ denotes the numerical integration of the equations of motion. Second, for each trajectory problem there is a vector of constraints called dependent variables $\underline{y}(\underline{u})$, together with a vector of constraint limit values, $\underline{b}$. These constraints are calculated at a particular user-specified event. Thus, the dependent variables are computed as functions of the independent variables, $\underline{u}$, from the general relationship

$$\underline{y}(\underline{u}) = \underline{Y}(\underline{u},\bar{\underline{X}}(\underline{u}))$$

where $\bar{\underline{X}}(\underline{u})$ is the composite vector of state conditions at all events where constraints are defined. Thus,

$$\bar{\underline{X}}(\underline{u}) = \bigcup_i \phi_i\left[\underline{x}\left(t_i\right), \underline{u}, t_i : \underline{x}_0\right]$$

where the set $\{t_i\} = \left\{t : y_i(t) = y_i^d, \text{ for all } i\right\}$ denotes the times at which the dependent variable events occur. The number of constraints, n, can be less than, equal to, or greater than the number of control parameters, m. In constrained optimization, $n < m$, excess degrees of freedom exist in the formulation enabling some performance criteria to be optimized and the constraints to be satisfied. In full rank targeting, $n = m$, no excess degrees of freedom exist, and as a result it is possible to satisfy only the constraints. The final case, $n > m$, generally occurs only during the solution process when an excess of inequality constraints become active on some intermediate iteration. Finally, for each trajectory, there is an objective function $F(\underline{u})$, which is calculated in the same manner as the constraints. The object of the problem is then to determine the control parameters, $\underline{u}$, which are feasible in that all constraint parameters fall within their acceptable ranges and optimal in the sense that the objective is minimized (or maximized).

The general discrete parameter trajectory optimization problem is then the well known nonlinear programming problem. Symbolically, it is expressed as:

minimize: $F(\underline{u})$

subject to: $\underline{y}(\underline{u}) \ \underline{\alpha} \ \underline{b}$

where: $\underline{u}$ is the mxl column matrix of control parameters,

F is the scalar objective function of the vector of control parameters,

$\underline{y}$ is the nxl column matrix of dependent parameters,

$\underline{b}$ is the nxl column matrix of constraint parameter limits,

$\underline{\alpha}$ is the nxl column matrix of constraint parameter relations (each element is the appropriate relation of the triple $\leq$, $=$, or $\geq$).

Virtually all trajectory targeting and optimization problems can be cast in this structure. Specific examples of nonlinear programming formulations for ascent, reentry, and orbital maneuvers are presented in the Sample Case Section.

Algorithm macrologic.- POST uses an accelerated projected gradient algorithm (PGA) as the basic targeting/optimization technique. PGA is a combination of Rosen's projection method for nonlinear programming and Davidon's variable metric method for unconstrained optimization. The program also contains backup single-penalty function methods that use steepest descent, conjugate gradients, and/or the Davidon method.

The accelerated projected gradient algorithm is based on five intuitive working principles as follows:

One-Dimensional Search.- Using cost and constraint function gradient information, a direction of search is established. Then a one-dimensional minimization is performed in this direction on an appropriate function. In this manner, a difficult multidimensional optimization problem is replaced by a sequence of simple one dimensional minimizations.

Linearized Constraint Correction.- Assume that the current vector of control parameters is outside the feasible region. This correction scheme approximates the contours of constant constraints as uniformly spaced parallel hyperplanes based on their respective gradients and values for the current control parameter vector. Using this approximation the smallest correction to the control parameters that would satisfy all active constraints is computed; or that failing, minimize the sum of the squares of their violations. One-dimensional minimization of the sum of squares of the constraint errors is then performed along the direction of this correction to obtain the next iterate of control parameters.

Gradient Projection.- Once a feasible vector of control parameters is obtained, the negative gradient is resolved into two components; one parallel to and one normal to the hyperplane tangent to the boundary to the feasible region at the current point. A minimization is then performed along the direction of the parallel negative gradient component to obtain the next control parameter iterate. The function to be minimized in this one dimensional search is the fourth basic principle of the algorithm.

Estimated Net Cost Function.- Because the constraints are nonlinear, the tangent plane only coincides with the boundary of the feasible region at the point of tangency; hence, a search along the component of the gradient lying in the tangent plane will probably terminate at a point external to the feasible region. Therefore, the real object of the search should not merely be to find the minimum value of the cost function in the search direction. Rather it should be to find a unique point along the search ray that yields on correction back to the feasible region a new feasible point with the smallest value of the cost function. This point is approximately determined by minimizing along the parallel component of the gradient the cost function less an estimate of the deterioration of the cost function occasioned by correcting back to the feasible region. The estimate is based on linearized constraint-correction formulae.

Gradient Acceleration.- It is widely known that the convergence of unconstrained gradient algorithms can be drastically improved by using gradient information from several iterations to estimate the inverse of the Hessian matrix of a quadratic form approximating the cost function. In fact for a cost function of m control parameters, it can be shown that a Hessian-inverse estimating accelerated gradient scheme converges in m iterations and a conventional steepest descent algorithm converges only asymptotically. To similarly accelerate the projected gradient algorithm for constrained problems, it is assumed that the cost function is a quadratic in m-q variables over the constraint boundary. Here q is the number of active constraints defining the boundary.

Thus convergence should be accelerated to m-q iterations after the set of active constraints that determines the feasible region stabilizes.

For the widely known special cases of the general nonlinear programming problem, the accelerated projected gradient algorithm degenerates to the appropriate special purpose state-of-the-art programming procedures. For example, if no constraints are present, the algorithm degenerates to the Davidon-deflected gradient procedure. This procedure has long been considered the method of choice for solving unconstrained parameter minimization problems. At the other extreme, if the problem has more active constraints than controls, the algorithm reduces to the Gauss' least squares procedure for minimizing constraint violation. This technique is generally conceded to be the best available for solving over-determined systems of equations. Similarly, if the number of constraints is precisely equal to the number of controls, the algorithm becomes the widely known Newton-Raphson procedure for solving systems of nonlinear equations. This scheme is certainly the simplest of the efficient methods for solving fully determined systems of equations.

A summary of macrologic for the gradient projection algorithm is presented in Figure 20. As indicated, if the initial guess violates any constraints, then the algorithm attempts to satisfy all the constraints by taking constraint restoration steps. Once a feasible control parameter vector has been found, the algorithm generates a sequence of iteration pairs. Each pair consists of an optimization step followed by a constraint step. If the user's initial control-parameter estimate is not feasible, however, a steadily improving sequence of constraint-correction steps is undertaken until a feasible solution is found. Furthermore, the subsequent optimization step is omitted after any constraint-correction step which fails to yield a feasible control-parameter vector.

Finally the algorithm has two stopping conditions. First the search is stopped if both the change in the cost function and the length of the change in the control-parameter vector between two successive optimization steps fall below their respective input tolerances. Symbolically

$$\left| F\left(\underline{u}^{\nu+2}\right) - F\left(\underline{u}^{\nu}\right) \right| < \varepsilon_F,$$

and

$$\left\| \underline{u}^{\nu+2} - \underline{u}^{\nu} \right\| < \varepsilon_u$$

where $\underline{u}^{\nu}$ and $\underline{u}^{\nu+2}$ are the control-parameter vectors resulting from the optimization steps in two consecutive iteration pairs. Second, the procedure is terminated if the maximum permissible number of iterations specified by the user is exceeded.

The decomposition approach is based on partitioning of the total mission to an ordered sequence of self-contained mission segments such as ascent, reentry, and so on. The constraints in each mission segment are then used to

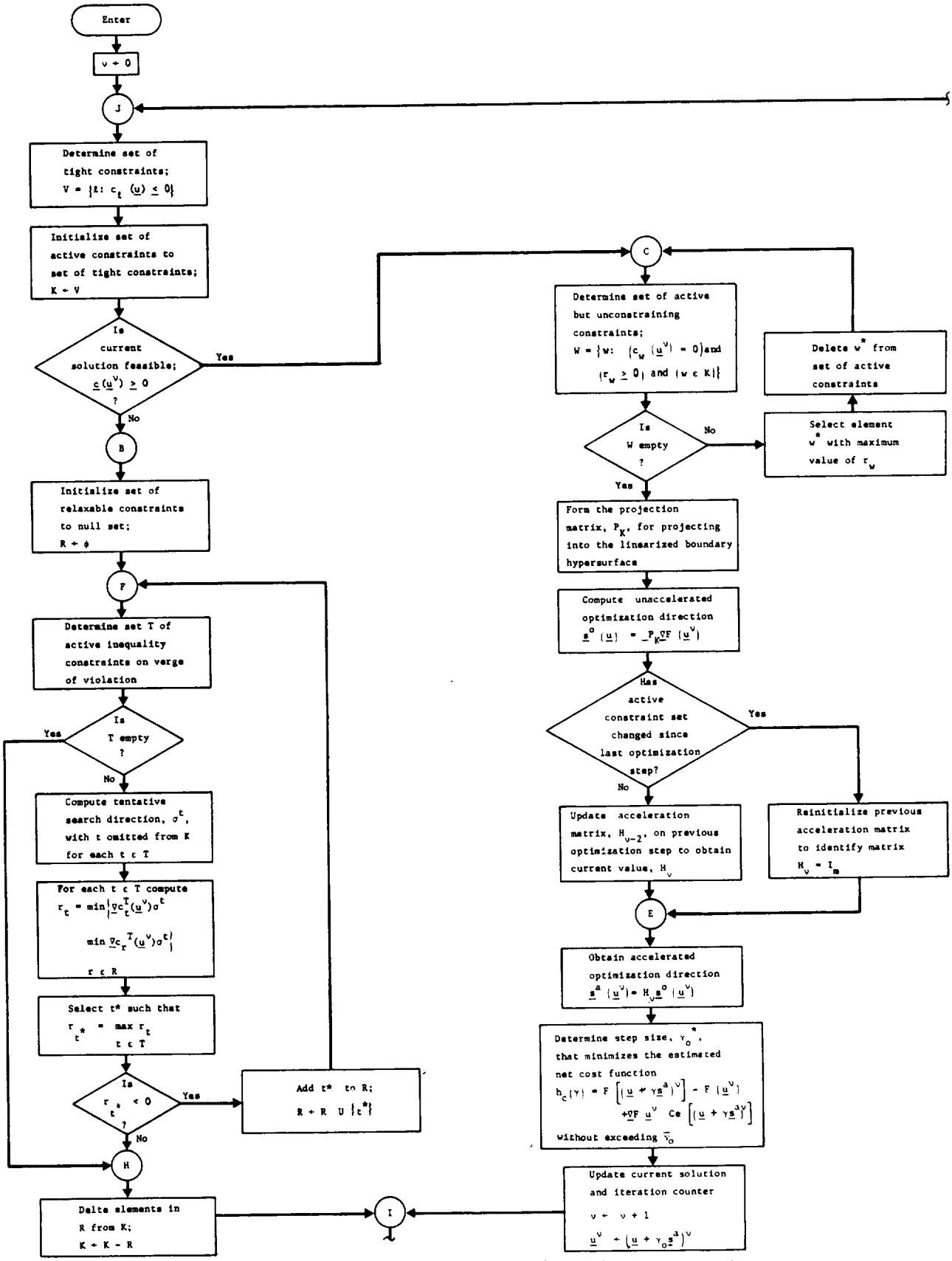
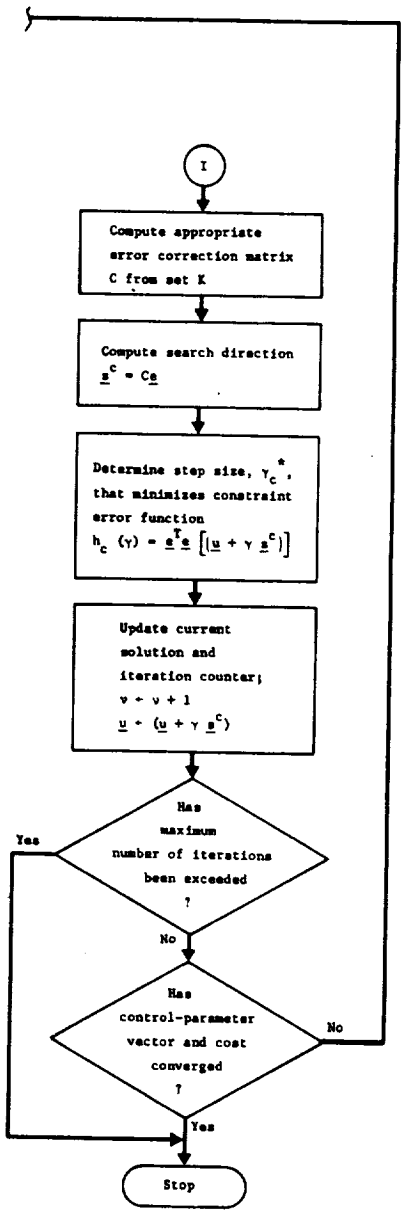


Figure 20.- Projected Gradient Macrologic



ν	-	Iteration Index
V	-	Set of Tight Constraints
t	-	Element of the Set of Tight Constraints
$\underline{c}$	-	Constraint Written in the Form $\underline{c} = \underline{y}(\underline{u}) - \underline{b}$
K	-	Set of Active Constraints
$\underline{u}$	-	Vector of Control Parameters
R	-	Set of Relaxed Constraints
ϕ	-	Null Set
T	-	Set of Active Inequality Constraints on Verge of Violatron
$\underline{c}^t$	-	Direction of Search with Constraint, t, Omitted from Set K
r^t	-	Coefficient of the Expansion of the Direction of Search in Terms of the Constraint Normals.
W	-	Set of Active of Nonbinding Inequality Constraints
w	-	Element of Set W
P_k	-	Projection Matrix for Set of Active Constraints
$\underline{s}^o$	-	Direction of Search for Optimization (Projected Gradient)
H_ν	-	Davidon Deflection Matrix
h_c	-	Estimated Net Cost Function
γ_o	-	Stepsize Parameter in the Optimization Direction
$\underline{s}^c$	-	Constraint Restoration Direction
$[C]$	-	Minimum Norm Matrix = $S'(SS')^{-1}$
h_e	-	Constraint Error Function

Figure 20.- Concluded

define an associated full rank targeting subproblem. Sequential solution of the subproblems ensures that the majority of the mission constraints are satisfied on each master level iteration. A master problem, which represents the complete problem, is then formulated in terms of the individual subproblem performance functionals, constraints, and control variables. Included in the master problem are intersegment constraints that cannot be satisfied at the subproblem level. (Total ΔV budgets or propellant limits are examples of master-level constraints that couple subproblems.) The master problem is optimized using a two-level procedure with the master-level algorithm optimally coordinating solution of the subproblems. The dual-level algorithm implemented employs the accelerated gradient-projection algorithm to solve the nonlinear program defined by the master problem, and the Newton-Raphson algorithm to solve the sets of nonlinear equations defined by the subproblems.

A unique consequence of using a two-level algorithm in conjunction with this decomposition approach is that intermediate target values, which are held fixed at the subproblem level, can be used as independent variables to be optimized as the master level.

PROGRAM USE

This section provides the user with the basic procedures required to use the program. At this point, it is assumed that the user has studied the basic program capabilities and has concluded that the program can be used to solve his problem. The question to be answered next is what does the user have to do to execute the program.

Required Preliminary Analysis

The user must first define the problem as a sequence of events. This can best be done by constructing a worksheet containing the event descriptions in columnar form. The user should then assign an integer to each event beginning with the first event, say 10, and incrementing by an arbitrary integer, say 10, for subsequent events. There must be at least two events for each problem, but there is no upper limit to the number of events.

The user must specify the condition at which each event is to occur. This should be done by writing the name of the condition and the desired value beside each event. The first event begins at the initial conditions specified by input and does not require this information.

The user should next identify the vehicle characteristics and specific program options required at each event. Any required tabular data such as aerodynamic coefficients, etc, should be included. This identification can be the users own terminology.

The user can now translate the worksheet to program input with a minimum of effort by proceeding according to the discussions that follow. A typical schedule associated with this translation is given in Figure 21.

POST Input Phases

An execution of the program consists of (1) processing the input data for all phases, (2) checking for input errors, (3) initializing the equations of motion, (4) propagating the trajectory until interrupted by the occurrence of the user-specified condition for the next event, (5) reinitializing the equations of motion with new user inputs for the event causing the interruption, (6) repeating steps 4 and 5 until the user-specified final event is reached, (7) terminating the problem, and (8) processing subsequent cases, if any. A phase is defined as the time increment from the occurrence of an event to the occurrence of the next event. As a result, the names phase and event are used interchangeably in the following discussions.

The first step is to construct a skeleton deck of namelists in the proper sequence. Namelist $SEARCH is required at the beginning of each problem. This is followed by one namelist $GENDAT for each event defined on the worksheet. If a given phase requires table input, one namelist $TBLMLT must follow the $GENDAT namelist for that phase. The $TBLMLT namelist is followed, in turn, by one namelist $TAB for each table to be input. This basic NAMELIST sequence, which must be followed, is illustrated in Figure 22. This completes the skeleton deck setup. The procedure now is to translate the data described in the worksheet to program input variables.

The first step in supply the inputs is to translate the event conditions to program inputs. Three namelist $GENDAT inputs are required to accomplish this task for each event. First, the input variable EVENT is set equal to the event number listed on the worksheet (e.g., EVENT = 1.0). Second the Hollerith input variable CRITR is required for all events except the first and is set equal to the program symbol for the desired condition (e.g., CRITR = 6HALTITØ, for altitude above the oblate planet). Third, the input variable VALUE is required for all events except the first and is set equal to the desired value at which the event is to occur (e.g., VALUE = 10 000, for a value of 10 000 ft or meters depending on whether English or metric units were selected in namelist $SEARCH by setting the variable IØFLAG to the desired value). On completion of these inputs, the user has only to specify the vehicle characteristics, the steering options and the desired program options before the job is ready to be executed. A summary of key POST input rules is shown in Figure 23.

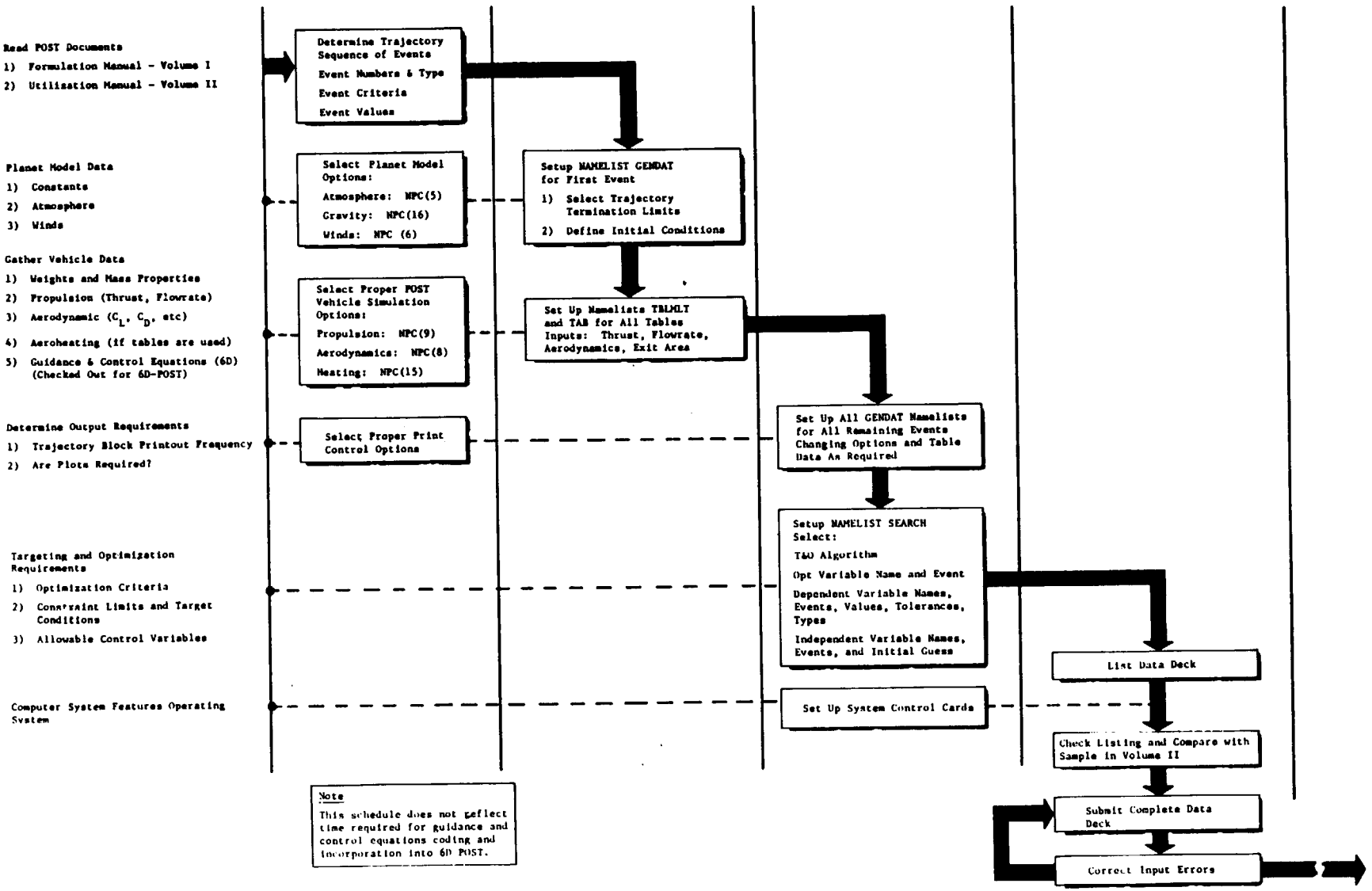


Figure 21.- POST Data Deck Setup Logic Flow Diagram

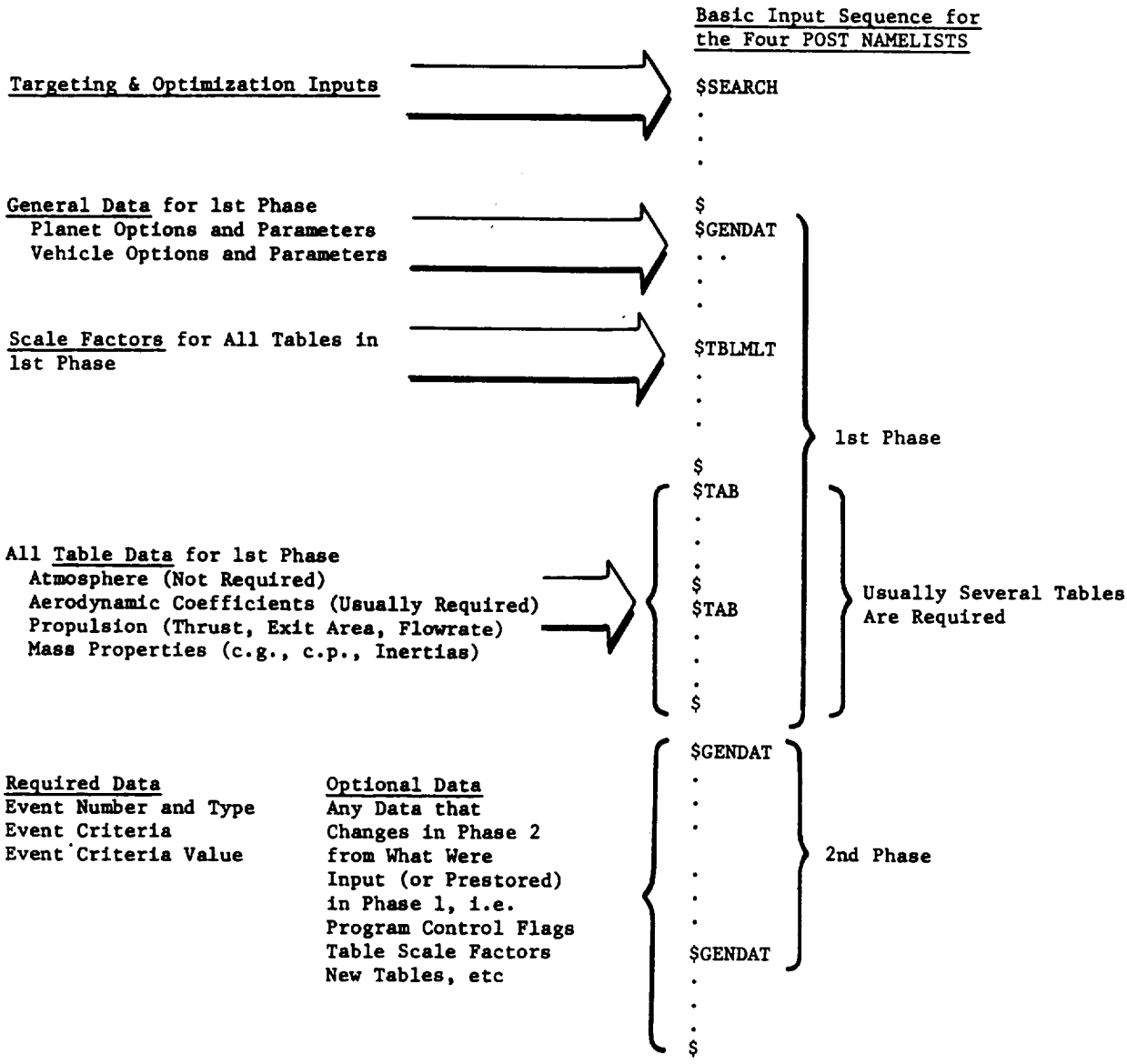


Figure 22.- POST Input Structure

Summary of Key POST

Input Rules

1. Every POST data deck must have SEARCH as its 1st NAMELIST. If targeting or optimization is not required then $SEARCH is empty

2. Every POST data deck must have at least two events, i.e., a beginning event and an ending event. The beginning event is based on TIME = XXX and does not, therefore, require criteria. The final event must be specified in the same manner as any other event with the addition of the end-problem card

   ENDPRB = 1, - Ends Problem

   and the end job card

   ENDJOB = 1, - Ends Job

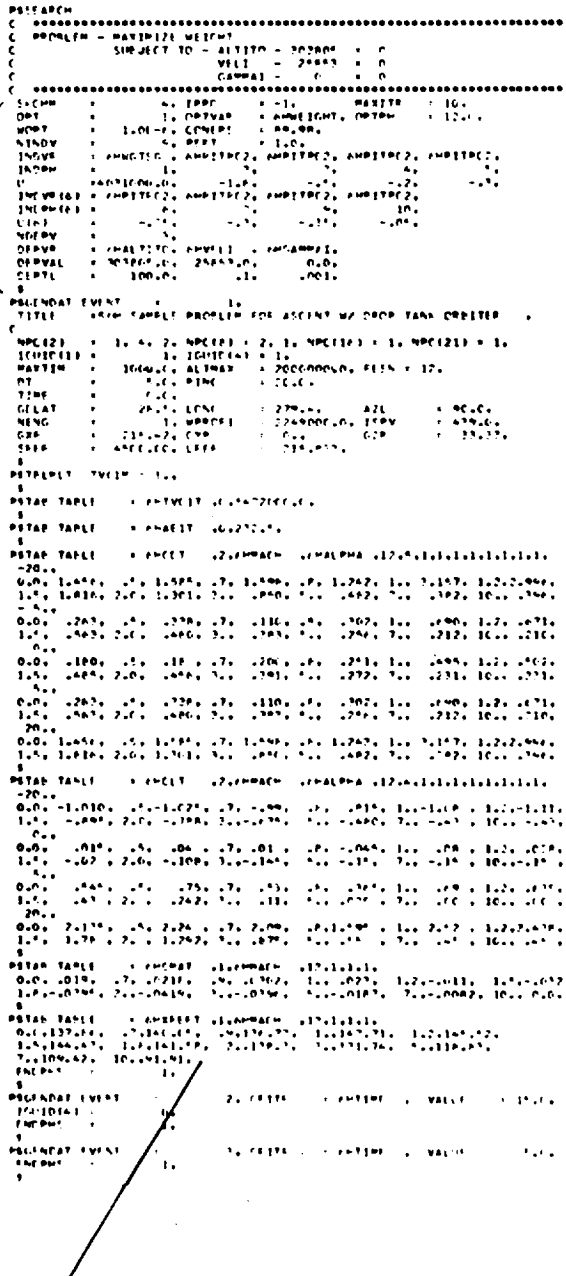
3. If tables are required in a phase then the sequence

   $GENDAT
   : - Begin Phase Data
   :
   $
   $TBLMLT
   :
   :
   $
   $TAB
   :
   :
   $
   $TAB
   :
   :
   ENDPHS = 1, - End Phase Data
   $

   must be maintained in that phase.

4. There is no program limit as to the number of Phases (Events) in any POST simulation

Typical POST Input Sequence



No $ after last Table entry in a phase

Figure 23.- Summary of Key POST Input Rules

Namelist Features

All program inputs are made using the namelist input feature.

The namelist input format allows card columns 2 through 80 to be used for input. Each variable input must be followed by a comma before the next variable is input. The variables are input as follows:

NAME1 = VALUE1, NAME2 = VALUE2, ETC.

Variables may be input one to a card or several to a card, depending on user preference.

Subscripted variables may be input as an array or as individual elements. The first element of the array is assumed if no subscript is present on a subscripted input variable. The following example shows various methods of input that are all equivalent:

EVENT = 1.0, 1.0,

or

EVENT(1) = 1.0, EVENT(2) = 1.0,

or

EVENT = 1.0, EVENT(2) = 1.0.

In general, decimal points should not be used with integer type variables. In any case, if no decimal point is input for a variable, the decimal is assumed to be after the last digit for that variable. As a result, it is best to omit decimal points for all variables unless required.

The namelist used in POST is an extension of the standard FORTRAN namelist and has the following added capabilities.

1) Special List Options.- The initial dollar ($) for each namelist input may be preceded in column 1 by any of three print control characters as follows: Blank will produce a card image print only for cards on which errors were detected; P will produce a print of each card encountered on the input file; and L will print each card encountered on the input file, but will also insert the card count after the card image.

2) Embedded Comment Cards.- C or slash in column 1 of any card will cause the card to be treated as a comment. Furthermore, when a slash is encountered in any other column, the slash and the remainder of the card to the right of the slash will be treated as a comment.

3) Special Hollerith Strings.- Hollerith strings may be input as OHXSTRINGX where STRING is the Hollerith character string and X is any character not contained in STRING. For example, to input the Hollerith characters ABCDEF111X in variable W, W = OH*ABCDEF111X*, would be acceptable. The symbol O for this option is the number zero.

4) Repeated Specification.- The following notation may be used to repeat a parameter value in any number of successive array locations: M*XXX, where XXX can be either REAL, INTEGER, HOLLERITH, LOGICAL, or OCTAL input parameter values and where M is the number of successive locations in any array where XXX is to be stored. For example, X = 10*1., would set X(1) through X(10) = 1.

5) Improved Error Diagnostics.- The extended namelist prints a diagnostic below any card in error with an arrow pointing to the erroneous column on the card.

General Data Input (GENDAT)

The general data inputs are the constant valued variables and arrays that can be input in any phase. These exclude table data and table multipliers that are described later. All general data inputs are made via namelist ($GENDAT).

The general data inputs include the following categories:

1) Aerodynamic Inputs

2) Aeroheating Calculation Inputs

3) Atmosphere Model/Winds Inputs

4) Event Criteria/Phase Definition Inputs

5) Gravitational Inputs

6) Methods of Guidance (Steering) Inputs

7) Initial Position and Velocity Inputs

8) Numerical Integration Method Inputs

9) Program Control (NPC) Inputs

10) Propulsion/Throttling Inputs

11) Vehicle/Propellant Weight Inputs

The selection of program options is accomplished by the user assigning particular values to the program control array, NPC(I). A summary of the program options controlled by this array is given in Table 11. The elements of this array are used internally to select the appropriate calculations for the

TABLE 11.- SUMMARY OF THE PROGRAM CONTROL FLAGS

Input Symbol	Units	Stored Value	Definition
NPC(1)	Integer	0	Conic Calculation Flag
NPC(2)	Integer	1	Integration Method Flag
NPC(3)	Integer	4	Velocity Vector Initialization Flag
NPC(4)	Integer	2	Position Vector Initialization Flag
NPC(5)	Integer	2	Atmosphere Model Flag
NPC(6)	Integer	0	Atmospheric Winds Flag
NPC(7)	Integer	0	Acceleration Limit Option Flag
NPC(8)	Integer	1	Aerodynamic Coefficient Type Flag
NPC(9)	Integer	0	Propulsion Type Flag
NPC(10)	Integer	0	Static Trim Option Flag
NPC(11)	Integer	0	Functional Inequality Constraints Option Flag
NPC(12)	Integer	0	Crossrange and Downrange Option Flag
NPC(13)	Integer	0	Propellant Jettison Option Flag
NPC(14)	Integer	0	Hold-down Option Flag
NPC(15)	Integer	0	Aeroheating Rate Option Flag
NPC(16)	Integer	0	Gravity Model Option Flag
NPC(17)	Integer	0	Weight Jettison Option Flag, Based FMASST
NPC(18)	Integer	0	Trajectory Termination Flag
NPC(19)	Integer	1	Flag to Control Printing of Input Conditions for Each Phase
NPC(20)	Integer	0	Flag to Specify the Type of Special Integration Step Size (DT) Prediction to be Used
NPC(21)	Integer	0	Flag to Indicate the Method by which Flowrate is to be Computed for Rocket Engines
NPC(22)	Integer	0	Throttling Parameter Input Option Flag
NPC(23)	Integer	0	Flag that Controls the Velocity Margin Calculations
NPC(24)	Integer	0	General Integration Variable Flag
NPC(25)	Integer	0	Velocity Loss Calculation Flag
NPC(26)	Integer	0	Special Aeroheating Calculations Flag
NPC(27)	Integer	0	Activation Flag for the Option to Integrate the Flowrate of Selected Engines
NPC(28)	Integer	0	Tracking Station Option Flag
NPC(29)	Integer	0	Analytical Vacuum Impact Point Calculation Flag
NPC(30)	Integer	0	Weight Calculation Option Flag
NPC(31)	Integer	0	A Flag to Activate the Vernal Equinox, Sun-Shadow, and Sun Angle Option
NPC(32)	Integer	0	The Parachute Drag Option Flag

options selected. Thus, although the program contains numerous options, any user who does not intend to use all program capabilities is not penalized in terms of run time. The inputs corresponding to the options selected via the NPC array must be specified as input in namelist GENDAT unless the stored values are to be used, in which case the variables need not be input. An example of a typical GENDAT setup is given in Figure 24.

The selection of the guidance and steering options is accomplished by the user assigning particular values to the guidance control array, IGUID(I). The general structure of this array is depicted in Figure 25, and Table 12 contains a summary of these guidance and steering flags. Admittedly, the hierarchical properties of the IGUID array makes it more difficult to learn than the NPC array. The reason for this additional complexity is that guidance flags are interrelated. This means that having selected a value for one IGUID implies that values must also be assigned to certain other specific IGUID elements. Over the years, other approaches have been evaluated, and this decision tree approach seems to be the most efficient way to provide flexibility with the minimum number of program control flags.

Tabular Data Input (TAB and TBLMLT)

The tabular data inputs are the tables and table multipliers that can be input in each phase.

Each table has its own numeric multiplier. All numeric table multipliers are input in namelist TBLMLT. The input symbol for a numeric multiplier is always formed by replacing the T at the end of the table name by an M which denotes multiplier. For example, the multiplier for table PREST is PRESM. All numeric multipliers are preset to 1.0 in the program unless overriden by input.

In addition to the numeric table multipliers, there are some special purpose multipliers for certain aerodynamic tables that are specified by Hollerith names. These are called MNEMONIC multipliers and can be any internally computed variable that is in the output variable list. These multipliers are used to multiply table lookup values of the corresponding aerodynamic coefficient tables. All MNEMONIC multipliers are input in namelist TBLMLT. The list of tables that have this feature and the corresponding MNEMONIC multipliers are as follows:

Table name	Mnemonic multiplier input symbol	Stored value	Probable value if constant valued or monovariant tables are input
CADPT	CADPNM	DFLP	DFLP
CADYT	CADYNM	DFLY	DFLY
CAT	CANM	ONE	ALPHA OR ALPTOT
CDDPT	CDDPNM	DFLP	DFLP
CDDYT	CDDYNM	DFLY	DFLY
CDT	CDNM	ONE	ALPHA
CLDPT	CLDPNM	DFLP	DFLP
CLT	CLNM	ONE	ALPHA
CMAT	CMANM	ONE	ALPHA
CMDPT	CMDPNM	DFLP	DFLP
CNAT	CNANM	ONE	ALPHA
CNDPT	DNCPNM	DFLP	DFLP
CWBT	CWBNM	ONE	BETA
CWDYT	CWDYNM	DFLY	DFLY
CYBT	CYBNM	ONE	BETA
CYDYT	CYDYNM	DFLY	DFLY

General Data Usually Required for a POST Simulation

I. First Event
- Event Number (Always Required)
- Criteria (Not Required for 1st Event)
- Value (Not Required for 1st Event)
- Final Event Sequence Number
- Trajectory Termination Parameters

II. Trajectory Integration and Printout
- Integrated Method Selection (NPC(2))
- Stepsize
- Print Increment
- Initial Conditions (Position, Velocity, Time, Weight)

III. Planet Model
- Gravity Model (Not Required When Available Options are Used)
- Atmosphere Model Selection

IV. Vehicle Model
- Propulsion (Not Required)
- Aerodynamics (Usually Required but Not in This Example)
- Steering (Always Required in at Least the 1st Phase)

V. Output
- Print Block (Not Required if Nominal Print Block is Satisfactory)
- Special Printout Options, e.g., Conic

VI. Subsequent Events
- Event Number
- Criteria
- Value

Sample GENDAT Input

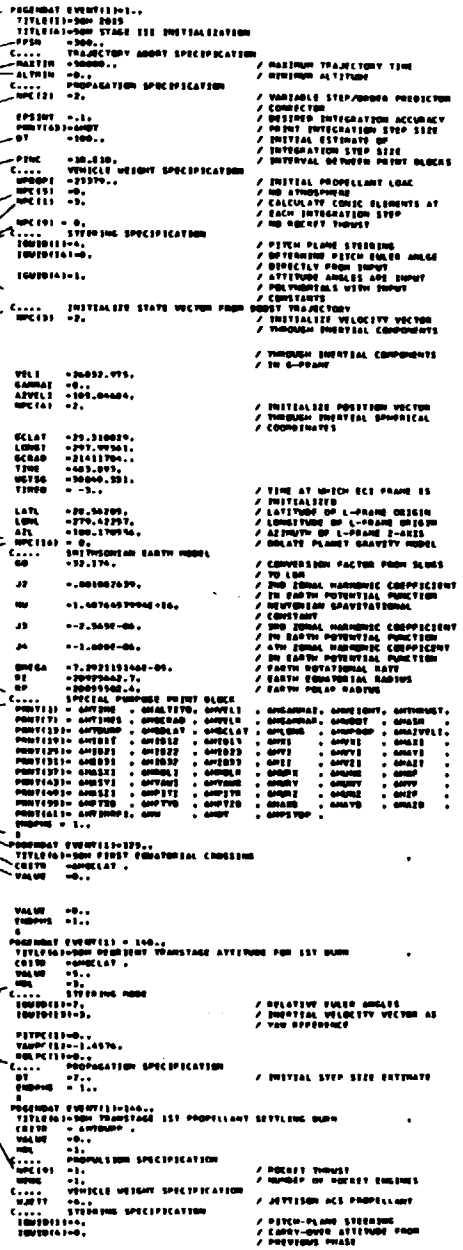


Figure 24.- Summary of General Data Inputs

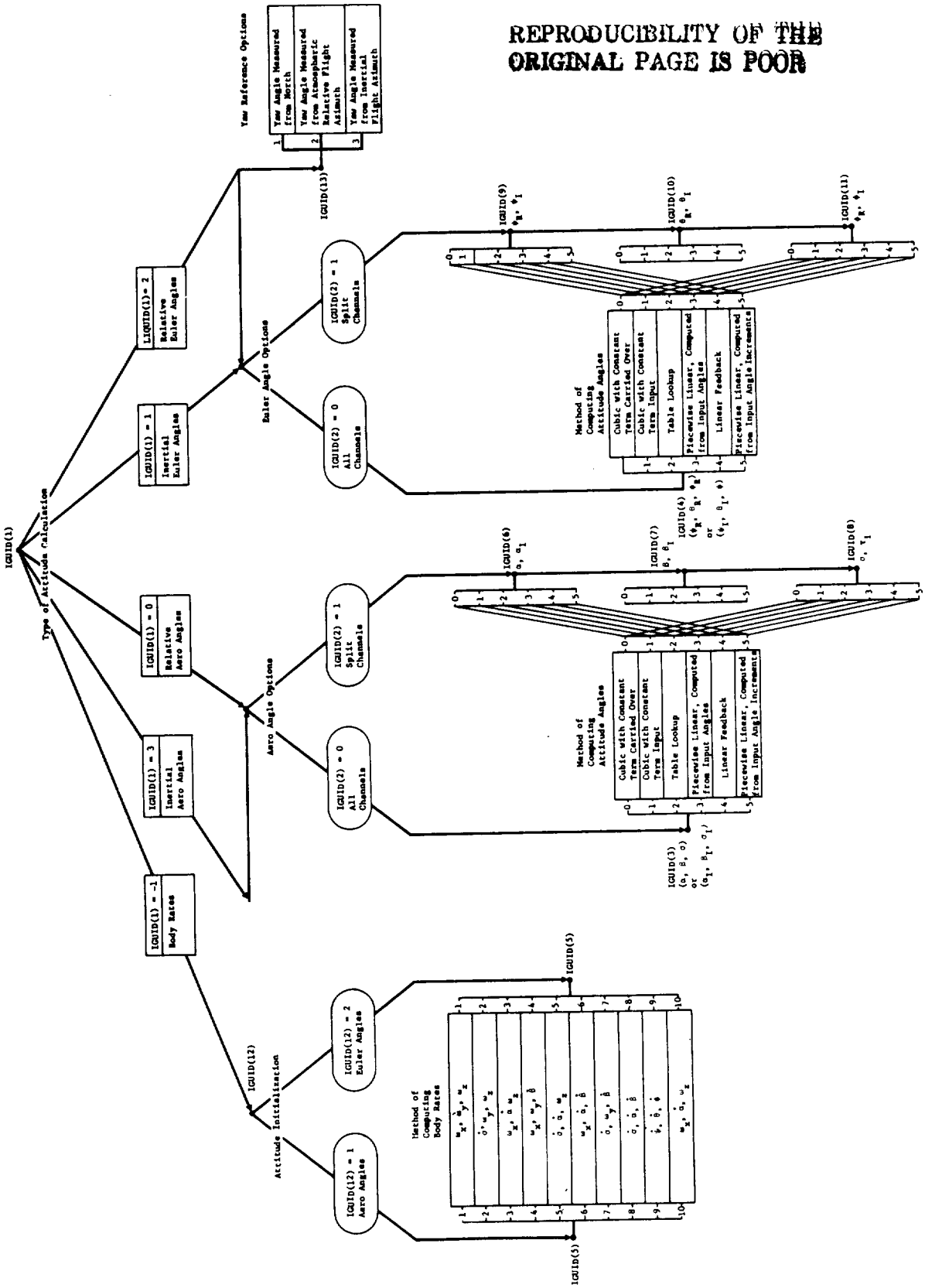


Figure 25.- IGUID Array Structure

TABLE 12.- SUMMARY OF THE GUIDANCE AND STEERING FLAGS

Input Symbol	Units	Stored Value	Definition
IGUID(1)	Integer	0	Type of Guidance (Steering) Desired, i.e., Body Rates, Aerodynamic Angles, or Euler Angles
IGUID(2)	Integer	0	Attitude Channel Selector.
IGUID(3)	Integer	0	Flag to Specify the Steering Option when Commanding All Channels Simultaneously Using Aerodynamic Angle of Attack, Sideslip, and Bank
IGUID(4)	Integer	0	Euler Engle Steering (Inertial or Relative)
IGUID(5)	Integer	1	Aerodynamic Angle Rate/Inertial Body Rate Combinations Flag
IGUID(6)	Integer	0	Separate Channel Options for Angle of Attack
IGUID(7)	Integer	0	Separate Channel Options for Side Slip Angle
IGUID(8)	Integer	0	Separate Channel Option for Bank Angle
IGUID(9)	Integer	0	Separate Channel Option for Yaw Angle
IGUID(10)	Integer	0	Separate Channel Option for Pitch Angle
IGUID(11)	Integer	0	Separate Channel Option for Roll Angle
IGUID(12)	Integer	2	Inertial Body Rate Initialization Flag
IGUID(13)	Integer	1	Yaw Reference Option Flag
IGUID(14)	Integer	0	General Open or Closed Loop Guidance Option
IGUID(15)	Integer	0	General Open Loop Guidance Override Option

Some examples of how these MNEMONIC table multipliers are used are as follows:

1) Assume that the normal force coefficient slope is given per degree ALPHA. In this case, CNAT would be input as the coefficient slope with the CNANM input as

   CNANM = 5HALPHA,

   the CNAT table lookup value would then be multiplied by the value of ALPHA to obtain the value of the normal force coefficient.

2) Assume that the normal force coefficient is given as a bivariant function of ALPHA and MACH. In this case, the MNEMONIC multiplier CNANM would be input as

   CNANM = 3HONE,

   the normal force coefficient would then be the actual table lookup value.

Each table is input in namelist TAB as the input array called TABLE. As a result, each table being input requires a separate input of namelist TAB. Termination of the table input data for a given phase is indicated by the presence of ENDPHS=1 in the last input of namelist TAB for that phase.

The table inputs for POST are generalized to include:

1) A preset allowable size for each table. The total size of all tables is limited only by the core storage allocated for tables. Both of these values can be changed by a simple program modification to satisfy user requirements.

2) Generalized argument specification. The argument to be used for each table is specified by input and can be any computer output variable.

3) Constant valued, monovariant, bivariant, or trivariant table types.

4) Linear or cubic interpolation capability.

An example of typical $TBLMLT and $TAB input is given in Figure 26.

Targeting and Optimization Input (SEARCH)

The targeting and optimization inputs are made via namelist $SEARCH.

POST has the capability of performing targeting with or without inequality constraints, unconstrained optimization, and constrained 9equality) optimization. The generality of POST enables the user to select the independent and dependent variables for the problem from a list of over 400 program variables.

Required Table Inputs

I. Table Scale Factors are input in NAMELIST TBLMLT. In this example there are no table multipliers. However, if one wanted to increase the thrust by 10% and reduce the drag by 5% this could readily be accomplished as follows:

```
P$TBLMLT
    TVC1M = 1.1,
    CDM = 0.95,
$
```

Insert Calls

II. Table Data

Examples of Constant Valued Tables

Examples of Monovariant Tables

Examples of Bivariant Tables

Sample Tab Input

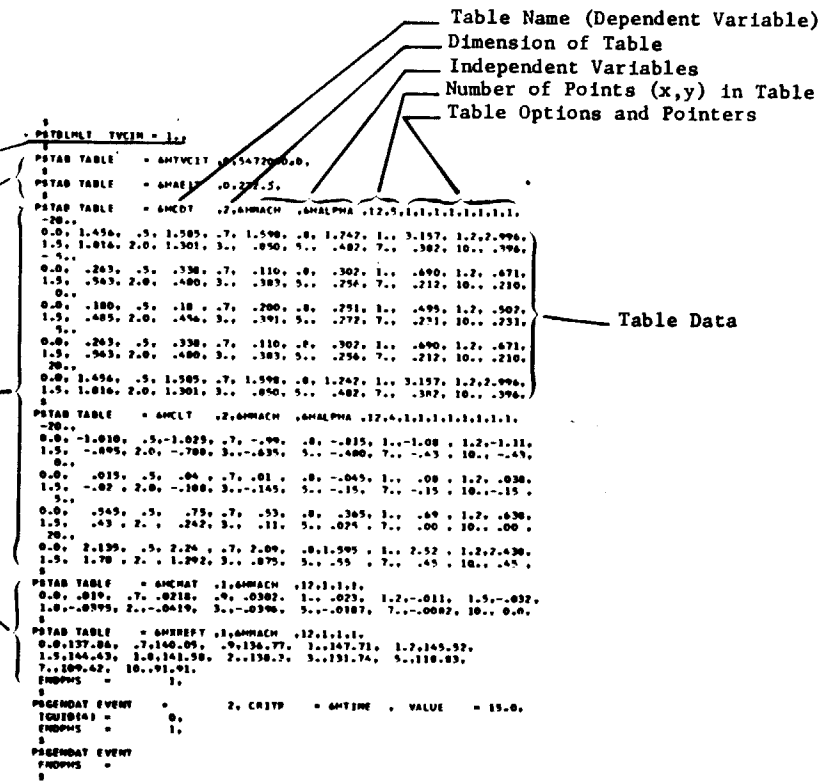


Figure 26.- Table Data Input

For an optimization problem, the optimization variable must be specified by input. The variables OPTVAR, OPTPH, and OPT are used for this purpose. For targeting (constrained) problems, the dependent variables must be specified by input. The variables NDEPV, DEPVR, DEPPH, IDEPVR, DEPVAL, and DEPTL are used for this purpose. Both sets of inputs are needed for a constrained optimization problem. In any case, the search mode (SRCHM), the number of iterations (MAXITR), and the independent variables (NINDV, INDVR, INDPH, and U) must be specified by input.

In addition to the above required namelist $SEARCH inputs, there are several others that can be used to further increase the rate of convergence on difficult optimization problems. For the most part, these inputs are related to problem scaling (MODEW and WOPT), search direction stepsize control (PCTCC), and convergence tolerances (CONEPS(I)).

Any type of event (primary, secondary, or roving) can be used in targeting/optimization. However, the user must ensure that the events selected will always occur. The association of an event number with the definitions of the targeting and optimization variables enables such things as intermediate targeting and optimization to be performed with the program. This correspondence also enables the program to remember the state variables at the beginning of the phases where the independent variables are introduced. Thus, when integrating the perturbed trajectories and the trial steps, only that segment of the trajectory affected by the control parameters being changed is integrated, thereby reducing the time required to generate the sensitivity matrix.

The program has the capability to either minimize or maximize a specified variable with or without satisfying specified constraints.

The following inputs are required to define the optimization variable. All variables associated with the optimization process are input in namelist $SEARCH.

1) The Hollerith name of the optimization variable. Input as the variable OPTVAR. Any of the output variables can be used as an optimization variable, provided it is sensitive to at least one of the independent variables.

2) Type of optimization desired - OPT. If no optimization is desired, input OPT as 0. If the variable defined by OPTVAR is to be minimized, input OPT equals -1. If OPTVAR is to be maximized, input OPT equals +1.

3) The phase number (EVENT) associated with the optimization variable - OPTPH. The variable specified by OPTVAR will be optimized at the plus side of the event specified by OPTPH.

4) The weighting for the optimization variable, WOPT. WOPT should be input as approximately 1 over the anticipated value of OPTVAR.

5) The decimal percentage change to be made in the control variables for the initial trial step on each iteration - PCTCC.

The dependent variables can be either equality constraints or inequality constraints based on user input. Any calculated output or independent variable can be used as a dependent variable and as many as 25 can be specified.

To constrain a control (independent) variable, declare it to be a dependent variable and set IDEPVR non-zero. For example, suppose ALPHA is to be a control variable in phase N, but must not exceed 45 degrees. Set INDRV(I) equals 5HALPHA, INDPH(I) equals N, DEPVR(J) equals 5HALPHA, DEPPH(J) equals N, DEPVAL(J) equals 45., and IDEPVR(J) equals 1.

The inequality constraints may be either functional or single-valued.

All inputs associated with the dependent variables are input in namelist $SEARCH. The required variables are as follows:

1) The Hollerith name of each dependent variable. Input in the array DEPVR(I), I=1, NDEPV.

2) The phase number (EVENT) associated with each dependent variable - DEPPH(I), I=1, NDEPV. The variable specified by DEPVR(I) will be satisfied at the plus side of the event specified by DEPPH(I). If DEPPH(I) is greater than FESN or not input, the last phase is assumed.

3) The desired value of each dependent variable - DEPVAL(I), I=1, NDEPV.

4) The desired accuracy tolerance in which DEPVR(I) is considered to be satisfied - DEPTL(I), I=1, NDEPV.

5) The number of dependent variables to be used - NDEPV. The first NDEPV variables in the array DEPVR(I) will be used as dependent variables.

6) The type of constraint for each DEPVR(I) - IDEPVR(I), I=1, NDEPV. The following values for IDEPVR(I) can be specified -

IDEPVR(I) = 0, if DEPVR(I) is to an equality constraint
= 1, if DEPVR(I) is to be an inequality constraint with an upper bound
= -1, if DEPVR(I) is to an inequality constraint with a lower bound.

The search/optimization option allows the user to specify as many as 25 control variables for each problem. The initial value of each control variable and the phase in which it occurs are also specified by input. If a control variable is a guidance (steering) variable, such as pitch rate or angle, the appropriate guidance option (IGUID) must be requested in namelist GENDAT for the corresponding phase.

The value of a given control variable as calculated by the targeting/ optimization algorithm will be carried over from one phase to the next until overriden by user input or by a new control variable. For example, if the linear term in the pitch angle polynomial (Hollerith input symbol PITPC2) is a control variable in phase 1.0, and is not a control variable in phase 2.0, the calculated value of the pitch rate in phase 1.0 will continue into phase 2.0 unless the coefficient PITPC(2) is input in namelist GENDAT for phase 2.0.

A control variable may be constrained by also defining it to be a dependent variable with an upper or lower bound.

The control parameters can be selected from any variables in the following categories:

1) Variables in namelist GENDAT such as initial vehicle position and velocity, initial vehicle orientation, vehicle attitude polynomial coefficients, etc. For example, suppose the initial velocity (VELI) is to be used as a control parameter. The inputs would then be as follows:

```
INDVR(1) = 4HVELI,

INDPH(1) = 1,

U(1) = AA.,
```

2) Constant valued table multipliers in namelist TBLMLT. For example, suppose the table multiplier for the thrust table (TVC1M) is to be used as a control parameter. The inputs would then be as follows -

```
INDVR(1) = 5HTVC1M,

INDPH(1) = XX.,

U(1) = AA.,
```

3) User-specified Y arguments from user-specified tables in namelist TAB. the tables and Y values to be used are specified in the variables TABL and TABLY which are input in namelist SEARCH. This feature is limited to monovariant tables. For example, suppose the first and fourth Y arguments of the monovariant PITT table are to be used as control parameters. The inputs would then be as follows -

```
TABL(1)  = 4HPITT, 4HPITT,

TABLY(1) = 1,4,

INDVR(1) = 6HTABL1, 6HTABL2,

INDPH(1) = XX., YY.,

U(1) = AA., BB.,
```

Each control parameter is uniquely specified by the following variables, which are input in namelist SEARCH, and are as follows:

1) The Hollerith name of each control variable. This is input as the array INDVR(I), I=1, NINDV.

2) The initial value of each control variable. Input as the array U(I), I=1, NINDV. If SRCHM is non-zero, U(I) overrides any value input for that variable in namelist GENDAT for phase INDPH(I).

3) The phase number (EVENT) at which each INDVR is initiated. Input as the array INDPH(I), I=1, NINDV. The variable whose Hollerith name appears in INDVR(I) is set equal to U(I) at the beginning of the phase specified by INDPH(I).

4) The perturbation for each control variable to be used to generate the sensitivities. Input as the array PERT(I), I=1, NINDV. The sensitivity DE(J)/DU(I) is determined by finite differencing U(I) by PERT(I) and calculating the change in each E(J). For variables whose nominal value is greater than 10.0, the value of PERT(I) should be input roughly six orders of magnitude less than the nominal value of the variable. The stored values for PERT(I) are 1.0E-4.

5) The number of control (independent) variables to be used. Input as the variable NINDV. The first NINDV variables in the array INDVR(I) will be used as control variables. The number of control variables must be greater than or equal to the number of target variables plus the optimization variable unless some of the target variables are inequality constraints. An example of a typical setup for namelist $SEARCH is presented in Figure 27.

$SEARCH Data Usually Required

I. Algorithm Input
- Algorithm Option
- Iteration Limit
- Iteration Printout Option

II. Independent Variables
- Number of Variables
- Hollerith Names of Variables
- Event Numbers of Variables
- Initial Guess of Variable for 1st Iteration
- Perturbations Used in Computing Sensitivities

III. Dependent Variables
- Number of Variables
- Type (equality, lower bound, upper bound)
- Hollerith Names of Variables
- Event Numbers for Variables
- Desired Values for Targets and Constraints
- Tolerances for Targets and Constraints

IV. Optimization Variable
- Maximize or Minimize
- Hollerith Name of Optimization Variable
- Event Number for Optimization Variable
- Maximum Stepsize for Optimization
- Monovariant Searches

Sample $ SEARCH Input

```
$SEARCH
C
C
 LISTIN =1.                        / INPUT NAMELIST AND TABLE
                                   / SUMMARY
 IDEB   =1.                        / PRINT TRIAL STEP SUMMARY
 MAXITR =3.                        / MAXIMUM PERMISSIBLE NUMBER OF
                                   / ITERATIONS
 SRCHM  =4..                       / USE ACCELERATED PROJECTED-
                                   / GRADIENT ALGORITHM
 IPRO   =0.                        / PRINT INITIAL AND FINAL
                                   / NOMINAL TRAJECTORIES
C.....  CONTROL VARIABLE SPECIFICATION
 NINDV  =9.                        / NUMBER OF CONTROL VARIABLES
 INDVR  =&HCRITR ,      &HPITPC1,      &HVAMPC1,
         &HPITPC2,      &HCRITR ,
         &HCRITP ,      &HPITPC1,      &HVAMPC1,
         &HCRITR ,
 INDPH  =140..          140..          140..
         149..          150..
         260..          260..          260..
         280..
 U      =6.90142019,    -3.67420779,   -7.93809841,
         -.0648845159,  321.0878947,
         180.1*6723.    1.69758620,    43.7987715,
         189.5226823.
 PERT   =.01.           .01.           .1.
         .001.          .01.
         .01.           .01.           .01.
         .01.
C.....  CONSTANTS CONTROLLING AUTOMATIC PERT SELECTION
 NPAD   =9.             5.             19.
 POLMAX =7.
 MODEM  =0.
 WU     =1..            .1.            1..
         10..           .1.
         1..            1..            .1.
         .1.
C.....  CONSTRAINT VARIABLE SPECIFICATION
 NDEPV  =9.
 DEPVR  =&MSERJAX,      &HECCEN ,      &HINC
         &HARGP  ,      &HLONG  ,
 NEPPH  =300..          300..          300..
         300..          300..
 DEPVAL =1.5P073648..   .07.           %..
         100..          50.00.
 DEPTL  =10000..        .0001.         .01.
         .01.           .01.
C.....  OBJECTIVE SPECIFICATION
 OPT    =1.                        / MAXIMIZE OBJECTIVE
 OPTVAR =&MWEIGHT,                 / OPTIMIZATION VARIABLE
 OPTPH  =300..                     / OPTIMIZATION PHASE
 PCTCC  =.1.                       / MAXIMUM RELATIVE CHANGE
                                   / ALLOWED IN MAGNITUDE OF
                                   / WEIGHTED CONTROL VECTOR
 STPMAX =1..                       / MAXIMUM ABSOLUTE CHANGE
                                   / ALLOWED IN MAGNITUDE OF
                                   / WEIGHTED CONTROL VECTOR
C.....  ITERATION CONVERGENCE TOLERANCES
 CONEPS =89.9.          1.0E-20.       1.0E-10.
         1.0E-04.       1.0E-20.
 P2MIN  =2..                       / VALUE OF SUM OF SQUARES
                                   / OF ERRORS BELOW WHICH
                                   / ITERATION IS CONSIDERED
                                   / TARGETED
C.....  CURVE FITTING TOLERANCES
 CONSER =1.0E-05.       1.0E-04.
 FITFRP =1.0E-06.       1.0E-04.
 $
```

Figure 27.- Optimization Inputs

SAMPLE CASES

This section presents six example problems that exemplify 3 DOF ascent, entry and orbital optimization problems; as well as, a 6 DOF simulation of the Space Shuttle entry. Each sample caae was selected to illustrate a key feature of the program. All examples were taken from actual trajectory problems and are not especially constructed sample cases. In fact, the data decks shown are the actual working data decks with the removal of redundant data cards being the only cosmetic clean-up. Hopefully, by studying these example cases, a new user will be able to correlate the details contained in references 2 and 4. In addition, the examples provide general guidelines that can be followed when setting up similar problems.

Example 1. Space Shuttle Ascent Optimization

An important ascent trajectory optimization problem during the conceptual phases of the Space Shuttle program was that of determining the maximum payload capability of various configuration concepts. One such Space Shuttle configuration is shown in Figure 28. The four key components of this configuration are the orbiter, two solid rocket boosters (SRBs), and the external tank (ET). This multibody nonsymmetrical configuration created special simulation requirements that motivated many of the features contained in POST. For example to accurately predict the performance capability of an unsymmetrical configuration, such as Space Shuttle, it is important to include the thrust vectoring losses encountered as the engines gimbal to balance the aerodynamic moments caused by the configurations nonsymmetrical shape. This fact led to the development of the static trim option employed in this sample case.

Trajectory Profile and Problem Formulation.- There are a number of ways to formulate the problem of maximizing payload for a given configuration. Each approach is based on (1) what is known about the configuration, and (2) what is known about the basic trajectory to be flown. In this first example, it is assumed that the user knows the dry weight and the propellant weight of each of the four major components of the vehicle. Assuming that all the propellant is consumed during the flight, which is ensured by terminating the simulation on the event criteria

$$W_{prop} = 0, \tag{35}$$

enables the payload weight to be computed from the equation

$$W_{PLD} = W_{BO} - W_{dry}, \tag{36}$$

Where W_{BO} is the total burnout weight (at the final event) and W_{dry} is the known weight of the remaining vehicle components. Because W_{dry} is constant for a given configuration, maximizing W_{BO} is equivalent to maximizing W_{PLD}. Thus,

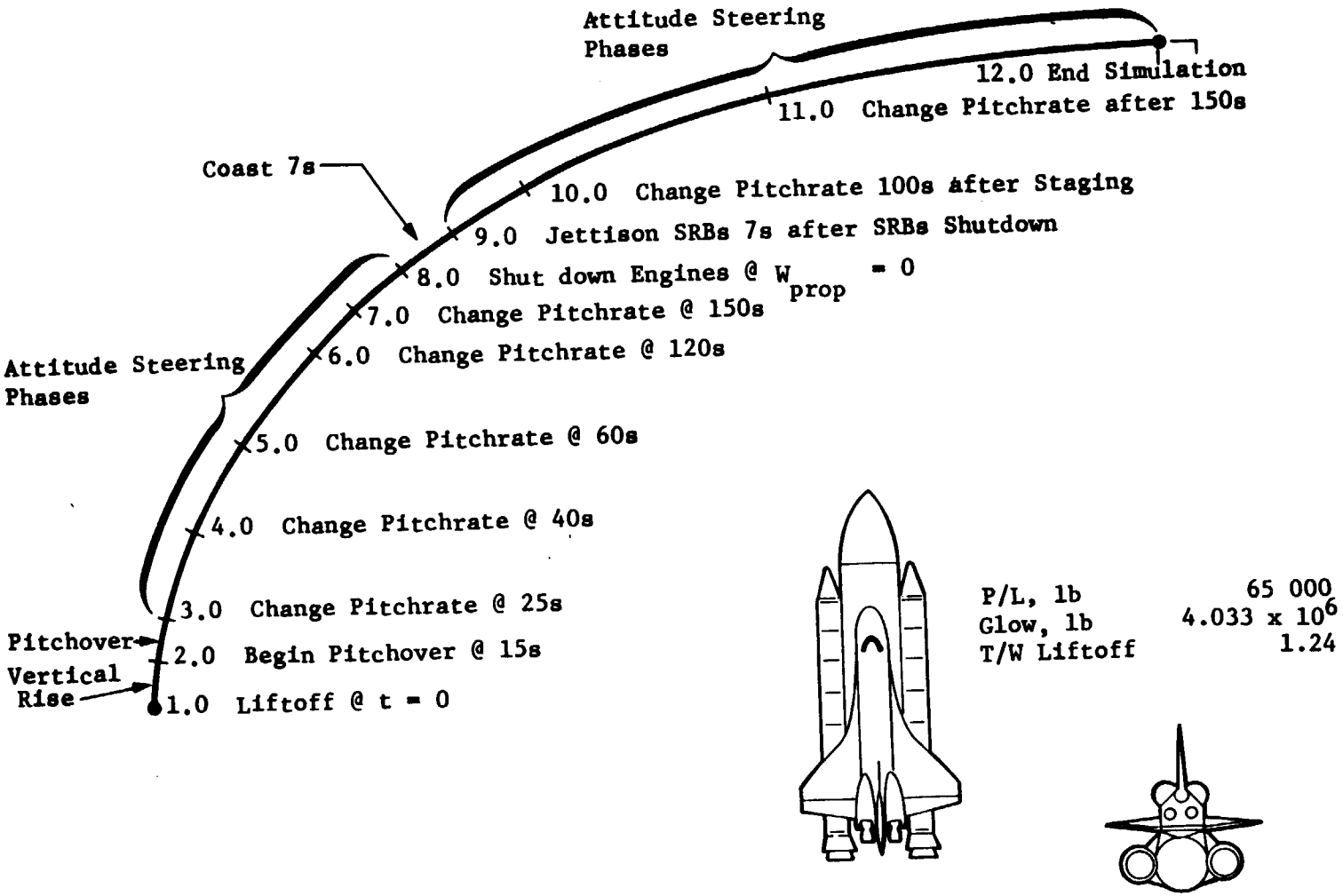


Figure 28.- Trajectory Profile and Vehicle Sketch for Space Shuttle Ascent Sample Case

in this example the optimization variable was selected to be W_{BO}, which is computed as the weight of the vehicle at the instant that the weight of propellant is zero.

As in any trajectory problem, there are a variety of ways in which to simulate this mission, and Figure 28 illustrates the approach taken in this example. As indicated, the simulation starts with a 15 second vertical rise, followed by a sequence of constant pitch rate steering segments. The static trim option is used during all early flight phases, and a three-g acceleration limit is enforced after 60 seconds of flight. Event 8.0 specifies burnout of the SRBs, which are jettisoned seven seconds later at Event 9. Notice also that in Event 9.0 new configuration data for propulsion and aerodynamics are input. These data represent the orbiter plus ET combination that are used for the remainder of the trajectory simulation. As mentioned earlier, the final event criteria is the weight of propellant. Because the last initialization of weight of propellant was in Event 9.0, the program variable W_{prop} represents the amount of propellant in the orbiter plus ET combination at any time. Thus, the final condition

$$W_{prop} = 0,$$

limits the amount of propellant that can be consumed in all flight phases after the occurrence of Event 9.0.

In this example, the mission requirements are the delivery of the payload to the perigee of a 50 x 100 n mi parking orbit. These requirements are mathematically equivalent to the three terminal equality constraints

$$h_f = 303\ 805.0 \text{ ft}$$

$$V_{If} = 25\ 853.0 \text{ fps}$$

$$\gamma_{If} = 0^\bullet.$$

where the subscript, f, denotes final burnout conditions. Extensive computational experience indicates that the $\left(h,\ V_I,\ \gamma_I\right)$ constraints are easier to satisfy than are their orbital counterparts $\left(h_p,\ h_a,\ \theta\right)$. The reason for this is probably related to the nonlinearities involved, with $\left(h,\ V_I,\ \gamma_I\right)$ appearing more linear in the independent variables.

Finally, the control parameters selected are eight inertial Eulerian pitchrates, and the initial gross weight of the vehicle at lift-off. Six of these pitchrates are used to steer the vehicle during the SRB boost phases, and two are used during the exoatmospheric phases. The motivation for using these particular control variables is computational experience, which shows that pitch angle steering is an efficient technique for optimizing ascent trajectories. The particular "breaktimes" in the pitch history were selected after a few single pass simulations. The initial gross weight of the vehicle, W_G, is

employed as an independent variable to maximize the payload because in this set-up there is a direct one-to-one correspondence between an increment in W_G and an increment in W_{PLD} because all vehicle dry weights and propellant weights are held constant during the optimization.

The previous discussion can be summarized by stating the precise mathematical formulation of the problem: Determine the control parameters

$$\underline{u} = (W_G, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5, \dot{\theta}_6, \dot{\theta}_7, \dot{\theta}_8), \tag{37}$$

that maximize: W_{BO}

Subject to: $h_f - 303\ 805\ \text{ft} = 0$

$V_{If} - 25\ 853\ \text{fps} = 0$

$\gamma_{If} - 0^\circ = 0$

The input data deck for this sample case is presented in Figure 29. It is important to understand the correspondence between the mathematical formulation presented and the specific input data contained in \$SEARCH in Figure 29. Table 13 is a summary of the key targeting and optimization variables as a function of the iteration number.

TABLE 13.- ITERATION SUMMARY FOR SPACE SHUTTLE SAMPLE CASE

Iteration	Optimization Variable, W_{BO} ∿ lb	Constraint Error, ∿ P_2	Optimization Indicator, ∿ CTHA
0	308 000	3.98×10^{7}	---
1	305 151	4.48×10^{3}	---
2	304 882	6.37×10^{-1}	89.903
3	308 320	7.02×10^{1}	---
4	308 369	4.17×10^{-3}	89.964
5	309 328	2.21×10^{-1}	89.976
6	310 278	6.09×10^{-1}	89.999
7	310 300	5.32×10^{-5}	---

```
P$SEARCH
C  ***************************************************************
C  PROBLEM - MAXIMIZE WEIGHT
C            SUBJECT TO - ALTITO - 303805  =  0
C                         VELI   -  25853  =  0
C                         GAMMAI -      0  =  0
C  ***************************************************************
 SRCHM     =         4, IPRO     = -1,        MAXITR    = 10,
 OPT       =         1, OPTVAR   = 6HWEIGHT, OPTPH     = 12.0,
 WOPT      =   1.0E-6, CONEPS    = 89.98,
 NINDV     =         9, PERT     = 1.0,
 INDVR     = 6HWGTSG , 6HPITPC2, 6HPITPC2, 6HPITPC2, 6HPITPC2,
 INDPH     =         1,        2,         3,         4,         5,
 U         =4031000.0,     -1.8,      -.5,       -.2,       -.3,
 INDVR(6)  = 6HPITPC2, 6HPITPC2, 6HPITPC2, 6HPITPC2,
 INDPH(6)  =         6,        7,         9,        10,
 U(6)      =     -.25,     -.3,     -.15,      -.05,
 NDEPV     =         3,
 DEPVR     = 6HALTITO, 6HVELI  , 6HGAMMAI,
 DEPVAL    = 303805.0,  25853.0,      0.0,
 DEPTL     =    100.0,       .1,     .001,
 $
P$GENDAT EVENT    =        1,
 TITLE     =50H SAMPLE PROBLEM FOR ASCENT W/ DROP TANK ORBITER    ,
C
 NPC(2)    =  1, 4, 2, NPC(8) = 2, 1, NPC(16) = 1, NPC(21) = 1,
 IGUID(1)  =        1, IGUID(4) = 1,
 MAXTIM    =   1000.0, ALTMAX    = 2000000.0, FESN = 12,
 DT        =      5.0, PINC      = 20.0,
 TIME      =      0.0,
 GDLAT     =     28.5, LONG      = 279.4,     AZL       = 90.0,
 NENG      =        1, WPROPI    = 2249000.0, ISPV      = 439.0,
 GXP       =   218.42, GYP       =  0.,       GZP       =  33.33,
 SREF      =  4500.00, LREF      =  218.833,
 $
P$TBLMLT  TVC1M = 1.,
 $
P$TAB TABLE     = 6HTVC1T ,0,5472000.0,
 $
P$TAB TABLE     = 6HAE1T  ,0,232.5,
 $
P$TAB TABLE     = 6HCDT   ,2,6HMACH  ,6HALPHA ,12,5,1,1,1,1,1,1,1,1,
-20.,
0.0, 1.456,  .5, 1.585, .7, 1.598, .8, 1.242, 1., 3.157, 1.2,2.996,
1.5, 1.816, 2.0, 1.301, 3.,  .650, 5.,  .482, 7.,  .382, 10., .396,
- 5.,
0.0,  .263,  .5,  .338, .7,  .110, .8,  .302, 1.,  .690, 1.2, .671,
1.5,  .563, 2.0,  .480, 3.,  .383, 5.,  .256, 7.,  .212, 10., .210,
  0.,
0.0,  .180,  .5,  .18 , .7,  .200, .8,  .251, 1.,  .495, 1.2, .502,
```

Figure 29.- Input Data Deck for Space Shuttle Ascent Sample

```
 1.5,  .485, 2.0,  .456, 3.,  .391, 5.,  .272, 7.,  .231, 10., .231,
   5.,
 0.0,  .263,  .5,  .338, .7,  .110, .8,  .302, 1.,  .690, 1.2, .671,
 1.5,  .563, 2.0,  .480, 3.,  .383, 5.,  .256, 7.,  .212, 10., .210,
  20.,
 0.0, 1.456,  .5, 1.585, .7, 1.598, .8, 1.242, 1., 3.157, 1.2,2.996,
 1.5, 1.816, 2.0, 1.301, 3.,  .850, 5.,  .482, 7.,  .382, 10., .396,
 $
P$TAB TABLE    = 6HCLT   ,2,6HMACH  ,6HALPHA ,12,4,1,1,1,1,1,1,1,1,
 -20.,
 0.0, -1.010,  .5,-1.025, .7, -.99,  .8, -.815, 1.,-1.08 , 1.2,-1.11,
 1.5,  -.895, 2.0, -.788, 3.,-.635,  5., -.480, 7., -.43 , 10., -.43,
   0.,
 0.0,   .015,  .5,  .04 , .7, .01 ,  .8, -.045, 1.,  .08 , 1.2, .038,
 1.5,  -.02 , 2.0, -.108, 3.,-.145,  5., -.15,  7., -.15 , 10.,-.15 ,
   5.,
 0.0,   .545,  .5,   .75, .7,  .53,  .8,  .365, 1.,  .69 , 1.2, .638,
 1.5,   .43 , 2. ,  .242, 3.,  .11,  5., .025 , 7.,  .00 , 10., .00 ,
  20.,
 0.0,  2.135,  .5, 2.24 , .7, 2.09,  .8,1.595 , 1., 2.52 , 1.2,2.438,
 1.5,  1.78 , 2. , 1.292, 3., .875,  5., .55  , 7.,  .45 , 10., .45 ,
 $
P$TAB TABLE    = 6HCMAT  ,1,6HMACH  ,12,1,1,1,
 0.0, .019,  .7, .0218,  .9, .0302,  1., .023,  1.2,-.011,  1.5,-.032,
 1.8,-.0395, 2.,-.0419,  3.,-.0396,  5.,-.0187,  7.,-.0082, 10., 0.0,
 $
P$TAB TABLE    = 6HXREFT ,1,6HMACH  ,12,1,1,1,
 0.0,137.86,  .7,140.05,  .9,136.77,  1.,147.71,  1.2,145.52,
 1.5,144.43,  1.8,141.58,  2.,138.3,  3.,131.74,  5.,118.83,
 7.,109.42,  10.,91.91,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =        2, CRITR    = 6HTIME  , VALUE    = 15.0,
 IGUID(4) =        0,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =        3, CRITR    = 6HTIME  , VALUE    = 25.0,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =        4, CRITR    = 6HTIME  , VALUE    = 40.0,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =        5, CRITR    = 6HTIME  , VALUE    = 60.0,
 NPC(7)   =        1,
 ASMAX    =      3.0,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =        6, CRITR    = 6HTIME  , VALUE    = 120.0,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =        7, CRITR    = 6HTIME  , VALUE    = 150.0,
```

Figure 29.- Continued

```
 DT        =      10.0,
 ENDPHS    =         1,
 $
P$GENDAT EVENT     =         8, CRITR     = 6HWPROP ,  VALUE     = 0.0,
 NPC(7)    =         0,
 NPC(9)    =         0, 0,
 WEICON    =       0.0,
 ENDPHS    =         1,
 $
P$GENDAT EVENT     =         9, CRITR     = 6HTDURP ,  VALUE     = 7.,
 DT        =      20.0,
 PINC      =      50.0,
 NPC(7)    =         1,
 NPC(9)    =         1,
 WJETT     =  665000.0,
 WPROPI    =  809000.0,
 ISPV      =     459.0,
 GXP       =     142.0,
 GYP       =       0.0,
 GZP       =      25.0,
 SREF      =    4840.0,
 LREF      =     135.0,
 $
P$TBLMLT
 $
P$TAB TABLE     = 6HTVC1T ,0,1431000.0,
 $
P$TAB TABLE     = 6HAE1T  ,0,154.54,
 $
P$TAB TABLE    = 6HCDT    ,2,6HMACH   ,6HALPHA ,12,7,1,1,1,1,1,1,1,1,
 -20.,
 0,.024, .2,.024, .6,.026, .8,.028, .9,.035, 1.3,.093, 1.5,.122,
 2,.116, 2.48,.1, 3,.092, 3.9,.082, 40,.03,
 - 4.,
 0,.024, .2,.024, .6,.026, .8,.028, .9,.035, 1.3,.093, 1.5,.122,
 2,.116, 2.48,.1, 3,.092, 3.9,.082, 40,.03,
   0.,
 0,.026, .2,.026, .6,.026, .8,.024, .9,.036, 1.3,.092, 1.5,.118,
 2,.106, 2.48,.091, 3,.082, 3.9,.074, 40,.022,
   5.,
 0,.042, .2,.042, .6,.04, .8,.042, .9,.076, 1.3,.124, 1.5,.142,
 2,.124, 2.48,.098, 3,.088, 3.9,.079, 40,.033,
  10.,
 0,.076, .2,.076, .6,.08, .8,.1, .9,.13, 1.3,.194, 1.5,.192,
 2,.165, 2.48,.127, 3,.114, 3.9,.095, 40,.057,
  20.,
 0,.36, .2,.36, .6,.362, .8,.44, .9,.41, 1.3,.39, 1.5,.36 , 2,.32,
 2.48,.242, 3,.224, 3.9,.216, 40,.238,
  30.,
 0,.36, .2,.36, .6,.36, .8,.44, .9,.41, 1.3,.39, 1.5,.36, 2,.32,
 2.48,.44, 3,.418, 3.9,.4, 40,.3,
```

Figure 29.- Continued

```
 $
P$TAB TABLE    = 6HCLT    ,2,6HMACH  ,6HALPHA ,12,7,1,1,1,1,1,1,1,1,
 -20.,
 0,-.07, .2,-.08, .6,-.12, .8,-.12, .9,-.12, 1.3,-.12, 1.5,-.12,
 2,-.13, 2.48,-.14, 3,-.12, 3.9,-.1, 40,-.14,
   4.,
 0,-.07, .2,-.08, .6,-.12, .8,-.12, .9,-.12, 1.3,-.12, 1.5,-.12,
 2,-.13, 2.48,-.14, 3,-.12, 3.9,-.1, 40,-.14,
   0.,
 0,.08, .2,.08, .6,.08, .8,.06, .9,.06, 1.3,.07, 1.5,.04, 2,0.0,
 2.48,-.02, 3,-.03, 3.9,-.04, 40,.03,
   5.,
 0,.29, .2,.29, .6,.29, .8,.28, .9,.28, 1.3,.3, 1.5,.24, 2,.17,
 2.48,.12, 3,.09, 3.9,.08, 40,.21,
  10.,
 0,.5, .2,.6, .6,.49, .8,.48, .9,.52, 1.3,.52, 1.5,.41, 2,.33,
 2.48,.25, 3,.2, 3.9,.15, 40,.4,
  20.,
 0,.94, .2,.94, .6,.92, .8,.9, .9,.94, 1.3,.89, 1.5,.75, 2,.63,
 2.48,.51, 3,.43, 3.9,.39, 40,.76,
  30.,
 0,.94, .2,.94, .6,.92, .8,.9, .9,.94, 1.3,.89, 1.5,.75, 2,.68,
 2.48,.67, 3,.65, 3.9,.62, 40,.76,
 $
P$TAB TABLE    = 6HCMAT   ,0,0.0,
 $
P$TAB TABLE    = 6HXCGT   ,1,6HWEICON,5,1,1,1,
 0,87.64, 202250,93.93, 404500,99.68, 606750,104.04, 809000,104.37,
 $
P$TAB TABLE    = 6HYCGT   ,0,0.0,
 $
P$TAB TABLE    = 6HZCGT   ,1,6HWEICON,5,1,1,1,
 0,31.33, 202250,31.5, 404500,31.75, 606750,32.42, 809000,33.83,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =       10, CRITR    = 6HTDURP ,  VALUE    = 100.0,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =       11, CRITR    = 6HTDURP ,  VALUE    = 150.0,
 NPC(1)   =        2,
 ENDPHS   =        1,
 $
P$GENDAT EVENT    =       12, CRITR    = 6HWPROP ,  VALUE    = 0.,
 ENDPHS   =        1,
 ENDPRB   =        1,
 ENDJOB   =        1,
 $
```

Figure 29.- Concluded

In this table, the function P_2 is a measure of the weighted error in the constraints, and is calculated from the equation

$$P_2 = \left[\frac{(h)_{BO} - 303\ 805}{100}\right]^2 + \left[\frac{(V_I)_{BO} - 25\ 853}{0.1}\right]^2 + \left[\frac{(\gamma_I)_{BO}}{0.001}\right]^2. \qquad (38)$$

Clearly, in the general case P_2 is the square of the norm of the weighted error vector $\underset{\sim}{e}$, and is computed as

$$P_2 = \underset{\sim}{e}' \underset{\sim}{e}$$

The condition $P_2 \leq 1$ is sufficient to ensure that all constraints are satisfied to within their specified tolerances.

The other important optimization output variable is the angle, CTHA. Mathematically, CTHA is defined as the angle between the unconstrained performance gradient vector and the projection of this vector to the plane that is a tangent constraint manifold. CTHA converges to 90° at the constrained optimum because the performance gradient becomes orthogonal to the constraint manifold at that point. This angle is computed from the equation

$$CTHA = \cos^{-1}\left(\frac{\underset{\sim}{g} \cdot P\underset{\sim}{g}}{\|\underset{\sim}{g}\| \|P\underset{\sim}{g}\|}\right) \qquad (39)$$

where $\underset{\sim}{g}$ is the unconstrained performance gradient and P is the projection matrix to the constraint tangent plane.

Typical output for this type of problem would be (1) the trajectory block printout for the nominal trajectory, (2) the sequence of iteration summaries, (3) the trajectory block printout of the final optimal trajectory, and (4) plots of all user-requested variables. These types of output are illustrated in Figure 30. The plot capability is not contained in POST but rather is generated from a profile type written by POST. The frequency and amount of trajectory printout can also be controlled by the user. In this example, the only printout is at the minus and plus sides of each event. These data give a brief but useful summary of the trajectory time histories.

Example 2. Single-Stage-to-Orbit (SSTO) Entry

This SSTO entry case, illustrated in Figure 31, features several advanced program options, that are particularly useful in entry trajectory analysis. For example, the trajectory control scheme used is significantly more complex than the simple open loop technique demonstrated in Example 1. In addition, several of the more advanced program simulation options are employed. Options include roving events, multiple time channels, feedback steering, monitor variables, print block changes, special variables, and multiple runs.

INITIAL CONDITION SUMMARY

```
SAMPLE PROBLEM FOR ASCENT W/ DROP TANK ORBITER

INPUT UNITS = ENGLISH, OUTPUT UNITS = ENGLISH

INITIAL CONDITIONS
      TIME   =  0.              TIMED  =  0.
      GCLAT  =  2.85000000E+01 GDLAT  =  2.85000000E+01 LONG   =  2.79400000E+02 LONG1  =  2.79400000E+02
      GCRAD  =  2.09257410E+07 ALTITD =  0.
      VELR   =  0.              GAMMAR =  0.              AZVELR =  0.

PROGRAM TERMINATION PARAMETERS
      FESN   =  12.000          MAXTIM =  1.00000000E+03 ALTMIN = -5.00000000E+03 ALTMAX =  2.00000000E+06

THE LAUNCH PAD INERTIAL (L) FRAME IS DEFINED BY
      LATL   =  2.85000000E+01 LONL   =  2.79400000E+02 AZL    =  9.00000000E+01

ATTRACTING PLANET MODEL
      RE     =  2.09257410E+07 RP     =  2.09257410E+07 OMEGA  =  7.29211000E-05 MU     =  1.407e539bE+16
      J2     =  0.              J3     =  0.              J4     =  0.
```

PHASE DATA SUMMARY

```
SAMPLE PROBLEM FOR ASCENT W/ DROP TANK ORBITER

BEGIN PHASE    1.000

1962 U.S. STANDARD ATMOSPHERE MODEL

AERODYNAMIC COEFFICIENTS SPECIFIED BY DRAG AND LIFT COEFFICIENTS
      SREF   =  4.50000000E+03 LREF   =  2.16833000E+02 LREFY  =  0.

PROPULSION CALCULATED FOR  1 ROCKET ENGINES

VEHICLE WEIGHT PARAMETERS
      WGTSG  =  4.03328112E+06 WPLD   =  0.              WPROP  =  2.24900000E+06 WJETT  =  0.
      GO     =  3.21740000E+01

NUMBER OF INTEGRALS FOR THIS PHASE =   6
INTEGRATION SCHEME = FOURTH ORDER RUNGE-KUTTA
      DT     =  5.00000000E+00 PINC   =  2.00000000E+01

USE INERTIAL ROLL, YAW, AND PITCH COMMANDS
      ROLPC  =  0.             , 0.             , 0.             , 0.             ,
      PITPC  =  0.             , 0.             , 0.             , 0.             ,
      YAWPC  =  0.             , 0.             , 0.             , 0.             ,
      YAWI   =  0.              YAWR   =  0.              DYAW   =  0.              DESN   =  0.
      PITI   =  0.              PITR   =  0.              DPITCH =  0.              DESN   =  0.
      ROLI   =  0.              ROLR   =  0.              DROLL  =  0.              DESN   =  0.

THE NEXT EVENT TO OCCUR WILL BE ONE OF THE FOLLOWING
      ESN    =  2.000           TYPE   = PRIMRY           CRITR  = TIME             VALUE  =  1.50000000E+01
      TOL    =  1.00000000E-06 MDL    =  1

TABLES AND MULTIPLIERS FOR THIS PHASE
      CDT    , 1.00000000E+00  CLT    , 1.00000000E+00  CMAT   , 1.00000000E+00  TVCIT  , 1.00000000E+00  AEIT   , 1.00000000E+00
      XREFT  , 1.00000000E+00

PROGRAM CONTROL FLAGS
      NPC    ( 1) =  0 NPC    ( 2) =  1 NPC    ( 3) =  4 NPC    ( 4) =  2 NPC    ( 5) =  4 NPC    ( 6) =  0 NPC    ( 7) =  0
      NPC    ( 8) =  2 NPC    ( 9) =  1 NPC    (10) =  0 NPC    (11) =  0 NPC    (12) =  0 NPC    (13) =  0 NPC    (14) =  0
      NPC    (15) =  0 NPC    (16) =  1 NPC    (17) =  0 NPC    (18) =  0 NPC    (19) =  1 NPC    (20) =  0 NPC    (21) =  1
      NPC    (22) =  0 NPC    (23) =  0 NPC    (24) =  0 NPC    (25) =  0 NPC    (26) =  0 NPC    (27) =  0 NPC    (28) =  0
      NPC    (29) =  0 NPC    (30) =  0 NPC    (31) =  0 NPC    (32) =  0 NPC    (33) =  0 NPC    (34) =  0 NPC    (35) =  0

GUIDANCE CONTROL FLAGS
      IGUID  ( 1) =  1 IGUID  ( 2) =  0 IGUID  ( 3) =  0 IGUID  ( 4) =  1 IGUID  ( 5) =  1 IGUID  ( 6) =  0 IGUID  ( 7) =  0
      IGUID  ( 8) =  0 IGUID  ( 9) =  0 IGUID  (10) =  0 IGUID  (11) =  0 IGUID  (12) =  2 IGUID  (13) =  1 IGUID  (14) =  0
      IGUID  (15) =  0 IGUID  (16) =  0 IGUID  (17) =  0 IGUID  (18) =  0 IGUID  (19) =  0 IGUID  (20) =  0 IGUID  (21) =  0
      IGUID  (22) =  0 IGUID  (23) =  0 IGUID  (24) =  0 IGUID  (25) =  0
```

TRAJECTORY BLOCK PRINTOUT

```
SAMPLE PROBLEM FOR ASCENT W/ DROP TANK ORBITER                                                                  PROBLEM NO.    1

                                                    *** PHASE    1.000 ***
TIME    0.              TIMES   0.              TDURP   0.              DENS    2.37690697E-03 PRES    2.11621660E+03 ATEM    5.18670000E+02
ALTITD-4.76837158E-07 GCRAD   2.09257410E+07 GDLAT   2.85000000E+01 GCLAT   2.85000000E+01 LONG    2.79400000E+02 LONG1   2.79400000E+02
VELI    1.34101168E+03 GAMMAI  0.              AZVELI  9.00000000E+01 XI      3.00354800E+06 VXI     1.32300480E+03 AXI     1.06798969E+00
VELR    0.              GAMMAR  0.              AZVELR  0.              YI     -1.61429627E+07 VYI     2.19022024E+02 AYI     -6.571e330E+00
VELA    0.              GAMMAA  9.00000000E+01 AZVELA  0.              ZI      9.98490063E+06 VZI     0.              AZI    3.61e57e23E+00
GAMAD   0.              AZVAD   0.              DWNRNG  0.              CRRNG   0.              DPRNG1  0.              DPRNG2  0.
THRUST  4.97997964E+06 WEIGHT  4.03326112E+06 WDOT    1.24646925E+04 WEICON  1.49011612E-08 WPROP   2.24900000E+06 ASMG    1.23e721e6E+00
ETA     1.00000000E+00 ETAL    1.00000000E+00 IPNULL   0.              IYNULL  0.              INCPCH  0.              INCYAW  0.
FTXB    4.97997964E+06 FAXB    -0.             AXB     3.97259353E+01 ALPHA   0.              ALPDOT  0.              ALPTOT  0.
FTYB    0.              FAYB    0.              AYB     0.              BETA    0.              BETDOT  0.              GALPHA  0.
FTZB    0.              FAZB    -0.             AZB     0.              BNKANG  9.00000000E+01 BNKDOT  0.              GALTOT  0.
CA      1.80000000E-01 CD      1.80000000E-01 DRAG    0.              ROLI    0.              YAWR    0.              ROLBD   0.
CN      1.50000000E-02 CL      1.50000000E-02 LIFT    0.              YAWI    0.              PITR    8.99999892E+01 PITBD   0.
CY      0.              HEATRT  0.              TLHEAT  0.              PITI    0.              ROLR    0.              YAWBD   0.
DYNP    0.              MACH    0.              REYNO   0.              ASXI    5.70200852E+00 ASYI    -3.44430413E+01 ASZI    1.69555760E+01

                                                    *** PHASE    1.000 ***
TIME    1.50000000E+01 TIMES   1.50000000E+01 TDURP   1.50000000E+01 DENS    2.31267089E-03 PRES    2.04551341E+03 ATEM    5.15341815E+02
ALTITD 9.33310129E+02 GCRAD   2.09266743E+07 GDLAT   2.84999856E+01 GCLAT   2.84999856E+01 LONG    2.79399993E+02 LONG1   2.79402684E+02
VELI    1.34683106E+03 GAMMAI  5.50859101E+00 AZVELI  9.00299007E+01 XI      3.02352433E+06 VXI     1.34110394E+03 AXI     1.32e46e09E+00
VELR    1.29291759E+02 GAMMAR  8.96267597E+01 AZVELR  2.13369064E+02 YI     -1.61404764E+07 VYI     1.07996292E+02 AYI     -6.25e04e04E+00
VELA    1.29291759E+02 GAMMAA  8.96267597E+01 AZVELA  2.13369064E+02 ZI      9.96534136E+06 VZI     8.10765341E+01 AZI     4.54117835E+00
GAMAD  -5.72311564E-03 AZVAD   2.00817421E+00 DWNRNG  0.              CRRNG   0.              DPRNG1  0.              DPRNG2  0.
THRUST  4.99634838E+06 WEIGHT  3.84631074E+06 WDOT    1.24646925E+04 WEICON  1.86970387E+05 WPROP   2.06202961E+06 ASMG    1.29485853E+00
ETA     1.00000000E+00 ETAL    1.00000000E+00 IPNULL   0.              IYNULL  0.              INCPCH  0.              INCYAW  0.
FTXB    4.99634838E+06 FAXB    -1.59202357E+04 AXB     -.16607773E+01 ALPHA  -1.48963081E-01 ALPDOT  0.              ALPTOT  3.44065788E-01
FTYB    0.              FAYB    0.              AYB     0.              BETA    3.10145300E-01 BETDOT  0.              GALPHA -2.87e41e42E+00
FTZB    0.              FAZB    -1.09857008E+03 AZB     -9.18942751E-03 BNKANG  1.23339716E+02 BNKDOT  0.              GALTOT  6.85e05e16E+00
CA      1.83025437E-01 CD      1.82991982E-01 DRAG    1.59173258E+04 ROLI    0.              YAWR    2.70029901E+02 ROLBD   0.
CN      1.26296037E-02 CL      1.31054076E-02 LIFT    1.13995728E+03 YAWI    0.              PITR    8.99649297E+01 PITBD   0.
CY      0.              HEATRT  0.              TLHEAT  0.              PITI    0.              ROLR    1.80000000E+02 YAWBD   0.
DYNP    1.93297185E+01 MACH    1.16179631E-01 REYNO   1.75964633E+08 ASXI    5.97065742E+00 ASYI    -3.6122UB22E+01 ASZI    1.98788049E+01

END OF PHASE    1.000
```

Figure 30.- Typical POST Output

TARGETING & OPTIMIZATION INPUT SUMMARY

```
SAMPLE PROBLEM FOR ASCENT W/ DROP TANK CRITER

*** INPUT SUMMARY FOR PROBLEM NO.   1 ***

SEARCH/OPTIMIZATION INPUTS
        NINDV  =   9             NDEPV  =   3             SRCHM   =   4             MODEW  =   1
        INDVR  =    WGTSG     ,     PITPL2     ,    PITPC2     ,    PITPL2     ,    PITPC2     ,    PITPL2     ,
                    PITPC2    ,     PITPC2     ,    PITPC2     ,
        INDPH  =    1.000,          2.000,          3.000,          4.000,          5.000,          6.000,
                    7.000,          9.000,          10.000,
        U      =  4.03328112E+06,-4.02959710E-01,-4.55853620E-01,-1.67889630E-01,-6.77243251E-01,-2.07672429E-01,
                 -6.65708451E-02,-1.30635729E-01,-1.16711775E-01,
        WVU    =  2.48077400E-07, 5.55555556E-01, 2.00000000E+00, 5.00000000E+00, 3.33333333E+00, 4.00000000E+00,
                  3.33333333E+00, 6.66666667E+00, 2.00000000E+01,
        PERT   =  2.48077400E-07, 5.55555556E-05, 2.00000000E-04, 5.00000000E-04, 3.33333333E-04, 4.00000000E-04,
                  3.33333333E-04, 6.66666667E-04, 2.00000000E-03,
        DEPVR  =    ALTITO      ,    VELI       ,    GAMMAI     ,
        DEPPH  =   900.000,        900.000,        900.000,
        DEPVAL =  3.03605000E+05, 2.56530000E+04, 0.            ,
        DEPTL  =  1.00000000E+02, 1.00000000E-01, 1.00000000E-03,
        IDEPVR =       0,              0,              0,
        IFDEG  =       0,              0,              0,
        OPTVAR =    WEIGHT      ,
        OPTPH  =   12.000,
        OPT    =   1              MAXITR =   10              IDEB    =   0              IRPO   =  -1
        PGEPS  =  1.00000000E+00 P2MIN  =  1.00000000E+00 STMNP1  =  1.00000000E-01 STMNP2 =  1.00000000E-01
        CONSX1 =  1.00000000E-06 CONSX2 =  1.00000000E-03 WCON    =  1.00000000E+02 FITER1 =  1.00000000E-06
        FITER2 =  1.00000000E-03 PCTCC  =  6.00000000E-01 WOPT    =  1.00000000E-06 CONEP1 =  8.99000000E+01
        CONEP2 =  1.00000000E-01 CONEP3 =  1.00000000E-01 CONEP4  =  1.00000000E-01 CONEP5 =  1.00000000E-01
        CONEP6 =  1.00000000E-01 GAMAX  =  1.00000000E+01
```

ITERATION SUMMARY

```
*** ITERATION NUMBER  1                                                                          PROBLEM NO.    1


CP/ITR 4.05468750E+00
PERT    1.00000000E+00       1.00000000E-04       1.00000000E-04       1.00000000E-04       1.00000000E-04       1.00000000E-04
        1.00000000E-04       1.00000000E-04       1.00000000E-04
PARTIALS OF ALTITO WITH RESPECT TO U(I)
SMAT   -2.38566020E+00       6.54143145E+05       9.62028831E+05       1.25603096E+06       3.61315005E+06       1.53065130E+06
        1.49398101E+06       2.67895543E+06       1.19850416E+06
PARTIALS OF VELI   WITH RESPECT TO U(I)
SMAT   -3.65042773E-02      -1.56383267E+03      -2.43463470E+03      -3.32878950E+03      -1.00727933E+04      -4.19535476E+03
       -3.91591001E+03      -7.40352320E+03      -4.92977078E+03
PARTIALS OF GAMMAI WITH RESPECT TO U(I)
SMAT   -3.63069205E-05       8.69061070E+00       1.28664238E+01       1.69211226E+01       4.98123988E+01       2.37016261E+01
        2.72735672E+01       6.23430600E+01       5.10367928E+01
G1(I)  -4.03100000E+00      -3.19744231E-11       1.77635684E-11       7.10542736E-12       1.59872116E-11       1.77635684E-11
        1.59872116E-11       0.                   8.88178420E-13
G1MAG   4.03100000E+00
G2(I)   1.49134225E+10       4.34262783E+06       1.63376689E+08       9.91744277E+07       4.46066795E+08       1.59127967E+06
        1.87427873E+08       1.85471988E+06       4.28943454E+07
G2MAG   1.49311047E+10
PGI(I)-1.17325747E-05       -4.27628347E-03      -5.40646816E-04       4.03532974E-04       5.00694880E-03       9.26137301E-04
       -9.69809896E-04      -6.32986975E-04       1.01904708E-03
PG1MAG 6.87707100E-05
WVEC    2.48077400E-07       5.55555556E-01       2.00000000E+00       5.00000000E+00       3.33333333E+00       4.00000000E+00
        3.33333333E+00       6.66666667E+00       2.00000000E+01
NAC                  3
IAC                  1                    2                    3
DP1DS   1.95399530E-02
DP2DS  -5.46177126E+08
STPMAX 1.00000000E+10
UMAG    3.11088422E+00
DUMAG   1.45612976E-01
GAMAST 0.                    1.45612976E-01       1.45796573E-01
P1TRY  -3.06000000E-01      -3.05150821E-01       0.                   0.                   0.                   0.
P2TRY   3.98196560E+07       4.47996228E+03       0.                   0.                   0.                   0.
YPRED   0.                   0.                   4.47946631E+03       0.                   0.                   0.
                                                                       0.                   0.                   0.
INDVR       WGTSG                PITPL2               PITPC2              PITPL2               PITPC2               PITPL2
            PITPC2               PITPC2               PITPC2
INDPH       1.000                2.000                3.000                4.000                5.000                6.000
            7.000                9.000                10.000
U(I)     4.02815082E+06      -1.84376756E+00      -5.05604901E-01      -2.01316794E-01      -3.11522280E-01      -2.57351877E-01
        -3.19577814E-01      -1.66303036E-01      -5.22747414E-02
WE(I)  -2.74660275E+01      -5.63949384E+01       2.33494442E+01
DEPVR       ALTITO               VELI                 GAMMAI
DEPPH     900.000              900.000              900.000
E(I)   -2.74660273E+03      -5.63949384E+00       2.33494442E-02
P1      -3.05150821E-01

OPTVAR      WEIGHT
OPTPH       12.000
OPTVAL 3.05150821E+05
P2        4.47996828E+03
```

Figure 30.- Concluded

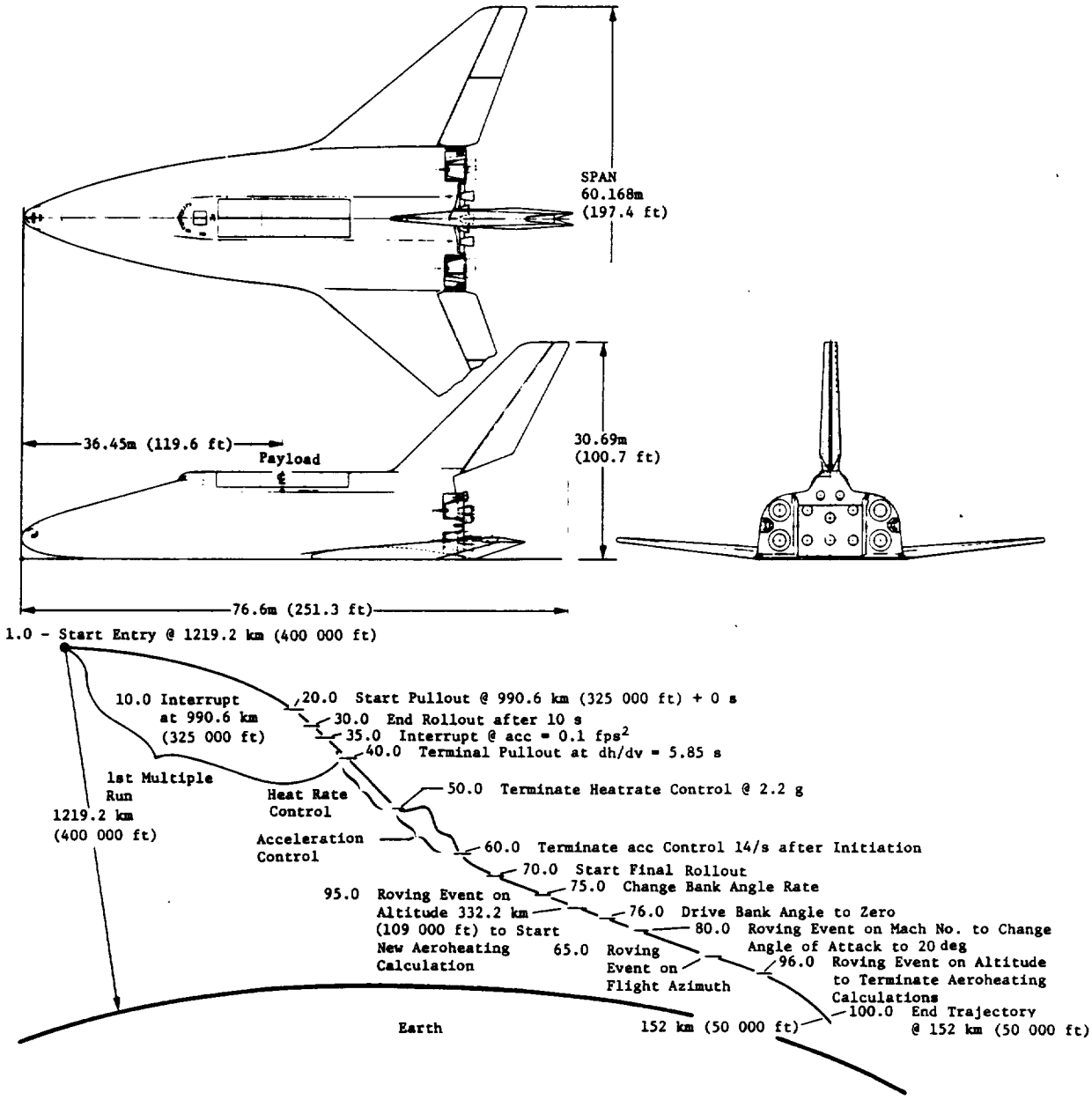


Figure 31.- Trajectory Profile and Sequence of Events for SSTO Entry Sample Case

The simulation is initiated and entry interface (400 000 ft), and terminated when the vehicle has descended to 50 000 ft. The trajectory control scheme attempts to minimize total heat by capitalizing on the ability of the TPS material to withstand a high heat rate for a short period of time. The basic optimization strategy is to attain the prescribed maximum heat rate as early as possible on entry into the atmosphere. This limit is then followed until the acceleration constraint is encountered. The bank-angle linear-feedback steering algorithm is then switched from controlling heat rate to controlling the acceleration profile. The acceleration limit is then followed until it is necessary to deviate from the limit to achieve the specified crossrange. This trajectory scheme is presented in Figure 31. Parametric results show that this simple approach provides the minimum total heat for this configuration by minimizing the entry flight time, subject to the heating rate, the acceleration, and the crossrange constraints.

One of the more difficult problems encountered in this example is that of determining the proper bank angle schedule to enable the vehicle to "pull out" of its initial entry plunge at the maximum heat rate that can be tolerated by the TPS. This "pull-out" maneuver is further constrained because it is important to minimize the heat rate "overshoot" when the linear feedback steering option is exercised. This is accomplished by using the POST iteration feature to solve the pull-out problem, which can be defined as:

determine the desired bank angle, ϕ_d, that satisfies the heat rate equation

$$\dot{Q}(\phi_d) = \dot{Q}_{max}, \tag{40}$$

when

$$\ddot{Q} = 0.$$

In the trajectory set-up, presented in Figure 31, the vehicle is initially banked ninety degrees ($\phi = 90°$) and ϕ_d is the desired bank angle when the vehicle reaches the 325 000 ft altitude. As such, ϕ_d really represents the amount of "un-bank" required to achieve the proper pull-out conditions.

After the proper pull-out conditions are satisfied, the vehicle descends using closed-loop steering to maintain heat rate and limit maximum deceleration. This leads to the second problem that must be solved to satisfy the crossrange requirements. If the vehicle remains on the maximum deceleration boundary too long, it will be unable to attain the required crossrange. If it departs too soon, it will be able to satisfy the crossrange, but it will increase the flight time, and, hence, the total heat. As a result, the following open loop optimization problem is formulated at the "bottom" of the trajectory: determine the acceleration boundary departure time, T_D, and the bank angle schedule characterized by the parameters $\{\phi_1 \quad \phi_2 \quad \phi_3 \quad \phi_4\}$

that maximize: T_D (i.e., stay on g-boundary as long as possible)

subject to: C_R 1100 n mi = 0

at the trajectory termination condition h = 50 000 ft.

Notice that in the formulation, which maximizes the time on the g-constraint, the optimization variable, T_D, is also one of the independent variables. This is not uncommon, but is, nevertheless, confusing to new users, who are not familiar with optimization theory.

As mentioned earlier, the complete SSTO entry problem is solved using the multiple run feature of POST. The first run is a one-dimensional iteration problem that determines the bank angle schedule required to achieve the proper pull-out conditions. The second run, initialized using the converged conditions from Run 1, simulates closed loop steering, first, along the heat rate boundary, and, second, along the acceleration boundary. When the time of departure is reached, the program uses the projected gradient algorithm to maximize T_D, subject to the crossrange constraint. Notice that only two runs and one pass at the computer are required to efficiently solve this problem. This is because the external iteration (Run 1) and optimization (Run 2) occur at the "top" and "bottom" of the trajectory. The majority of the flight time is simulated only once using closed-loop steering to satisfy both environmental constraints.

The data deck for this example is listed in Figure 32.

```
P$SEARCH                                  }
 SRCHM      = 4,                          |
 MAXITR = 3,                              |
 NINDV = 1,                               |
 INDVR = 5HDBANK,                         |
 INDPH = 20.,                             |
 U = -71.1220821,                         |  SEARCH DATA FOR FIRST PROBLEM
 PERT = .5,                               |
 NDEPV      = 1,                          |
 DEPVR      = 6HHEATRT,                   |
 DEPPH = 40.,                             |
 DEPTL = .1,                              |
 DEPVAL = 100.,                           |
 $                                        }
P$GENDAT EVENT(1)  = 1.0,
 TITLE = 50HMAX HEAT RATE =100.  GAMMAI = -0.80, PERIGEE = 0
 TITLE(7)   = 40HEAST   MISSION                                    ,
 FESN       = 100.,
 MAXTIM     = 4000.,
 ALTMIN     = -110000.,
 ALTMAX     = 450000.,
 NPC(1) = 3,
 NPC(2)     = 1,
 DT         = 20.,
 DTIMR(1)   = 1.,
 DTIMR(2)   = 1.,
 DTIMR(3)   = 1.,
 NPC(3)     = 2,
 VELI = 25600.,
 GAMMAI = -.8,
 AZVELI = 0.,
 AZVELI = 90.,
 NPC(4)     = 2,
 ALTITO     = 400000.,
 GCLAT      = 0.,
 LONG       = 0.,
 J2         = 1.0827E-3,
 MU         = 1.4076469E+16,
 OMEGA      = 7.292115E-5,
 RE         = 20925722.,
 RP         = 20855560.,
 NPC(5)     = 2, 0, 0, 2,
 LREF = 200.,
 RN        =1.0    ,
 NPC(9)     = 0,
 SREF = 12120.,
 WGTSG = 523076.,
 NPC(10)=0,
 NPC(11)    = 0,
 MONX(1)    = 6HDYNP   ,6HASMG   ,6HHEATRT,
```

Figure 32.- Input Data Deck for SSTO Sample Entry

```
 MONY(1)    = 6HTIME  ,6HTIME  ,6HTIME  ,
 NPC(12) = 0,
 IGUID      = 0, 0, 3,
 ALPHA      = 30.,
 DALPHA     = 30.,
 BNKANG = -90.,
 DBANK = -90.,
 NSPEC      = 1,
 SPECI      = 400000.,
 SPECI(6) = 100.,
 SPECI(7) = 310.,
 NPC(15)    = 1.,
 PINC = 50.,
 PRNT(91) =  6HTIMRF1,6HXMAX1 ,6HXMAX2 ,6HXMAX3 ,6HTLPWT ,6HUBAR  ,
             6HTIMRF2,6HYXMX1 ,6HYXMX2 ,6HYXMX3 ,6HTIMRF3, 6HSPECV1,
             6HVELAD ,6HW      ,
 $
P$TBLMLT
 $
 P$TAB TABLE = 6HCDT   ,1,6HMACH  ,            9,1,1,1,
   4.0, .430, 4.5, .420, 5.0, .410, 5.5, .400, 6.5, .390,
   8.0, .383, 10., .375, 30., .340, 100., .340,
 $
P$TAB TABLE = 6HCLT   ,1,6HMACH  ,            8,1,1,1,
   4.0, .622, 4.7, .600, 5.5, .585, 6.5, .570, 7.5, .560,
   8.5, .550, 10., .540, 30., .520,
 $
P$TAB TABLE      = 6HALPHAT,1,6HMACH  ,6,1,1,1,
             0.,10.0, 1.0,10.0, 2.0,20.0, 4.0,20., 5.0,30.0, 100.,30.0,
 ENDPHS  =1,
 $
P$GENDAT EVENT(1)  = 10.,
 CRITR      = 6HALTITO,    VALUE  = 325000.,
 NPC(1)     = 0,
 TIMRF(1)   = 0.,
 TIMRF(2)   = 0.,
 DT         = 10.,
      ENDPHS = 1,
 $
P$GENDAT EVENT(1)  = 20.,
 CRITR      = 6HTIMRF1,
 VALUE      = 0.0,
 TIMRF(1)   = 0.,
 MDL        = 1,
 DBANK = -71.1220821,
 DT    = 2.,
 ENDPHS     = 1,
 $
P$GENDAT EVENT(1)  = 30.,
 CRITR      = 6HTIMRF1,
 VALUE = 10.,
```

Figure 32.- Continued

```
 TIMRF(1)  = 0.,
 IGUID     = 0, 0, 0,
 MDL       = 1,
 ENDPHS    = 1,
 $
P$GENDAT EVENT     = 35.,
 CRITR     = 6HVELAD ,
 VALUE     = -.1,
 ENDPHS    = 1,
 $
P$GENDAT EVENT = 40., 1.,
 CRITR = 6HSPECV1,
 VALUE     = 5.85,
 MDL       = 1,
 DT        = 2.,
 TIMRF(1)  = 0.,
 IGUID(1)  = 0,
 IGUID(2)  =  1,
 IGUID(6)  = 0,
 IGUID(7)  = 0,
 IGUID(8)  = 4,
 KDG(3)    = +20.,
 KRG(3)    = 300.,
 IDGF(3)    = 1,
 DGF(3)    = 6HHEATRT,
 TIMRF(3)  = 0.,
 $
P$TBLMLT
 $
P$TAB TABLE = 6HGNOM3T,1,6HVELR   , 7,1,1,1,
        0., -70., 12000., -70., 13500., -20.,
     15500., -20., 18500., -45., 24000., -70., 30000., -70.,
 $
P$TAB TABLE = 6HGNMX3T,0,0.,
 $
P$TAB TABLE = 6HGNMN3T,0,-180.,
 $
P$TAB TABLE = 6HGDF3T ,0,100.,
 ENDPHS    = 1,
 $
P$GENDAT EVENT = 50., 0.,
 CRITR     = 6HASMG  ,
 VALUE = 1.4,
 MDL       = 1,
 TIMRF(1)  = 0.,
 TIMRF(3)  = 0.,
 KDG(3)    = +200.,
 KRG(3)    = 3000.,
 IDGF(3)    = 1,
 DGF(3)    = 6HASMG  ,
 $
```

Figure 32.- Continued

```
P$TBLMLT
 $
P$TAB TABLE = 6HGNOM3T,1,6HVELR   , 7,1,1,1,
        0., -70., 12000., -70., 13500., -20.,
     15500., -20., 18500., -45., 24000., -70., 30000., -70.,
 $
P$TAB TABLE = 6HGNMX3T,0,0.,
 $
P$TAB TABLE = 6HGNMN3T,0,-180.,
 $
P$TAB TABLE = 6HGDF3T ,1,6HTIMRF1,3,1,1,1,
    0., 1.4, 30., 1.4, 10.E10, 1.4,
 ENDPHS     = 1,
 $
P$GENDAT EVENT(1)  = 60.,
 CRITR      = 6HTIMRF3,
 VALUE = 141.,
 IGUID      = 0,0,3,
 DALPHA     = 30.,
 BNKANG     = -70.,
 DBANK = -26.7182816,
 TIMRF(3)   = 0.,
 MDL        = 1,
 DT         = 10.,
 ENDPHS     = 1,
 $
P$GENDAT EVENT = 65.,1.,
 VALUE = 90.0,
 CRITR = 6HAZVELR,
 CRITR = 1HU,
 VALUE = 0.,
 IGUID(1)   = 0.,
 IGUID(2)   = 1.,
 IGUID(6)   = 2.,
 IGUID(7)   = 0.,
 IGUID(8)   = 1.,
 BNKPC(1)   = 0.,
 ENDPHS = 1,
 $
P$GENDAT EVENT(1)  = 70.,
 CRITR      = 6HTIMRF3,
 VALUE      = 20.,
 MDL        = 1,
 TIMRF(3)   = 0.,
 DBANK = -23.6650460,
 ENDPHS     = 1,
 $
P$GENDAT EVENT(1)  = 75.,
 CRITR      = 6HTIMRF3,
 VALUE      = 20.,
 MDL        = 1,
```

Figure 32.- Continued

```
 TIMRF(3)  = 0.,
 DBANK = -11.1222061,
 ENDPHS    = 1,
 $
P$GENDAT EVENT = 76.,
 CRITR     = 6HTIMRF3,
 VALUE     = 20.,
 TIMRF(3)  = 0.,
 MDL       = 1,
 DBANK = 0.0,
 ENDPHS    = 1,
 $
P$GENDAT EVENT(1)  = 80., 1.,
 CRITR     = 6HMACH  ,
 VALUE     = 5.,
 MDL       = 1,
 IGUID(1)  = 0.,
 IGUID(2)  = 1.,
 IGUID(6)  = 2.,
 IGUID(7)  = 0.,
 IGUID(8)  = 1.,
 BNKPC(1)  = 0.,
 DT = 5.,
 $
P$TBLMLT
 $
 P$TAB TABLE = 6HCDT  ,2,6HMACH  ,6HALPHA , 15, 14,1,1,1,1,1,1,1,1,
  -90.,
    0.00, .030, 0.20, .030, 0.60, .030, 0.70, .033, 0.80, .039,
    0.90, .050, 1.10, .105, 1.20, .122, 1.30, .130, 1.40, .125,
    1.60, .112, 2.00, .100, 5.00, .069, 10.0, .050, 30.0, .028,
  0.00,
    0.00, .030, 0.20, .030, 0.60, .030, 0.70, .033, 0.80, .039,
    0.90, .050, 1.10, .105, 1.20, .122, 1.30, .130, 1.40, .125,
    1.60, .112, 2.00, .100, 5.00, .069, 10.0, .050, 30.0, .028,
  4.00,
    0.00, .042, 0.20, .042, 0.60, .042, 0.70, .045, 0.80, .050,
    0.90, .067, 1.10, .130, 1.20, .142, 1.30, .145, 1.40, .141,
    1.60, .128, 2.00, .110, 5.00, .070, 10.0, .050, 30.0, .028,
  8.00,
    0.00, .055, 0.20, .055, 0.50, .055, 0.60, .060, 0.75, .070,
    0.80, .080, 0.90, .112, 1.00, .150, 1.10, .168, 1.20, .170,
    1.40, .168, 1.80, .142, 3.00, .103, 5.00, .072, 30.0, .040,
  12.00,
    0.00, .075, 0.20, .075, 0.50, .093, 0.65, .112, 0.70, .123,
    0.80, .175, 0.90, .218, 1.05, .234, 1.30, .228, 1.50, .210,
    2.00, .171, 3.00, .182, 5.00, .100, 10.0, .083, 30.0, .060,
  16.00,
    0.00, .102, 0.20, .103, 0.50, .150, 0.65, .183, 0.75, .250,
    0.85, .305, 0.95, .325, 1.00, .330, 1.20, .325, 1.50, .295,
    2.50, .202, 3.50, .163, 5.00, .140, 10.0, .120, 30.0, .090,
```

Figure 32.- Continued

```
 20.00,
   0.00, .170, 0.20, .170, 0.40, .210, 0.60, .272, 0.70, .340,
   0.80, .397, 0.90, .427, 1.00, .440, 1.10, .441, 1.20, .432,
   1.50, .392, 2.00, .323, 3.00, .238, 5.00, .190, 30.0, .132,
 24.00,
   0.00, .260, 0.20, .260, 0.40, .320, 0.60, .400, 0.70, .466,
   0.80, .535, 0.90, .572, 1.00, .575, 1.30, .542, 1.60, .492,
   2.00, .421, 2.50, .347, 3.00, .312, 5.00, .270, 30.0, .190,
 28.00,
   0.00, .375, 0.20, .375, 0.40, .452, 0.60, .555, 0.70, .618,
   0.80, .682, 0.90, .722, 1.00, .735, 1.10, .730, 1.60, .638,
   2.00, .563, 2.50, .471, 3.00, .420, 5.00, .355, 30.0, .290,
 32.00,
   0.00, .510, 0.20, .510, 0.40, .596, 0.60, .713, 0.80, .841,
   0.90, .882, 1.00, .897, 1.10, .890, 1.50, .818, 2.00, .720,
   3.00, .544, 4.00, .490, 6.00, .453, 10.0, .431, 30.0, .388,
 36.00,
   0.00, .650, 0.20, .650, 0.40, .747, 0.60, .860, 0.80, .978,
   0.90,1.020, 1.00,1.038, 1.10,1.032, 1.50, .963, 2.00, .865,
   3.00, .682, 4.00, .618, 6.00, .580, 10.0, .550, 30.0, .490,
 40.00,
   0.00, .786, 0.20, .786, 0.40, .871, 0.60, .995, 0.80,1.101,
   0.90,1.136, 1.00,1.152, 1.10,1.150, 1.50,1.090, 2.00,1.010,
   3.00, .863, 4.00, .777, 6.00, .711, 10.0, .675, 30.0, .610,
 44.00,
   0.00, .911, 0.20, .911, 0.40,1.000, 0.60,1.115, 0.80,1.205,
   0.90,1.232, 1.00,1.248, 1.10,1.245, 1.50,1.195, 2.00,1.125,
   3.00,1.007, 4.00, .932, 6.00, .858, 10.0, .805, 30.0, .722,
 90.00,
   0.00, .911, 0.20, .911, 0.40,1.000, 0.60,1.115, 0.80,1.205,
   0.90,1.232, 1.00,1.248, 1.10,1.245, 1.50,1.195, 2.00,1.125,
   3.00,1.007, 4.00, .932, 6.00, .858, 10.0, .805, 30.0, .722,
$
P$TAB TABLE = 6HCLT   ,2,6HMACH  ,5HALPHA ,15,14,1,1,1,1,1,1,1,1,
-90.00,
   0.00, .065, 0.20, .065, 0.90, .052, 1.00, .037, 1.10, .020,
   1.20, .020, 1.60, .052, 1.70, .052, 2.00, .046, 2.50, .018,
   3.00, .000, 4.00,-.013, 6.00,-.021, 10.0,-.028, 30.0,-.028,
  0.00,
   0.00, .065, 0.20, .065, 0.90, .052, 1.00, .037, 1.10, .020,
   1.20, .020, 1.60, .052, 1.70, .052, 2.00, .046, 2.50, .018,
   3.00, .000, 4.00,-.013, 6.00,-.021, 10.0,-.028, 30.0,-.028,
  4.00,
   0.00, .220, 0.20, .220, 0.80, .220, 0.90, .217, 1.00, .200,
   1.20, .162, 1.25, .162, 1.45, .163, 1.60, .180, 1.90, .161,
   2.70, .094, 3.30, .070, 4.50, .050, 10.0, .027, 30.0, .005,
  8.00,
   0.00, .384, 0.20, .384, 0.80, .382, 0.90, .378, 1.00, .362,
   1.20, .305, 1.30, .294, 1.60, .280, 2.00, .250, 3.00, .160,
   4.00, .130, 4.50, .120, 7.50, .100, 10.0, .088, 30.0, .055,
 12.00,
```

Figure 32.- Continued

```
    0.00, .530, 0.20, .530, 0.80, .530, 0.90, .525, 1.00, .490,
    1.10, .463, 1.80, .372, 2.00, .350, 2.40, .300, 2.60, .278,
    3.10, .230, 3.70, .200, 4.80, .180, 10.0, .155, 30.0, .125,
  16.00,
    0.00, .660, 0.20, .660, 0.50, .650, 0.90, .628, 1.00, .608,
    1.20, .568, 1.60, .573, 1.80, .484, 2.00, .455, 2.80, .347,
    3.30, .310, 4.00, .285, 6.00, .255, 10.0, .233, 30.0, .188,
  20.00,
    0.00, .760, 0.20, .760, 0.60, .732, 0.80, .719, 1.00, .693,
    1.20, .667, 1.50, .622, 1.80, .579, 2.00, .547, 2.50, .478,
    3.00, .430, 4.00, .382, 6.00, .343, 10.0, .308, 30.0, .240,
  24.00,
    0.00, .838, 0.20, .835, 0.60, .808, 0.80, .790, 1.00, .770,
    1.20, .751, 1.50, .718, 1.80, .677, 2.00, .646, 2.50, .569,
    3.00, .528, 4.00, .483, 6.00, .441, 10.0, .400, 30.0, .313,
  28.00,
    0.00, .906, 0.20, .900, 0.60, .872, 0.80, .860, 1.00, .842,
    1.20, .825, 1.50, .797, 1.80, .760, 2.00, .735, 2.50, .675,
    3.00, .632, 4.00, .573, 6.00, .530, 10.0, .491, 30.0, .423,
  32.00,
    0.00, .970, 0.20, .961, 0.60, .930, 0.80, .917, 1.00, .900,
    1.20, .880, 1.50, .860, 1.80, .838, 2.00, .820, 2.50, .765,
    3.00, .720, 4.00, .670, 6.00, .626, 10.0, .584, 30.0, .534,
  36.00,
    0.00,1.020, 0.20,1.010, 0.60, .978, 0.80, .961, 1.00, .949,
    1.20, .933, 1.50, .910, 1.80, .890, 2.00, .877, 2.50, .840,
    3.00, .808, 4.00, .761, 6.00, .717, 10.0, .670, 30.0, .577,
  40.00,
    0.00,1.055, 0.20,1.042, 0.60,1.015, 0.80,1.001, 1.00, .990,
    1.20, .979, 1.50, .962, 1.80, .947, 2.00, .934, 2.50, .910,
    3.00, .882, 4.00, .842, 6.00, .794, 10.0, .752, 30.0, .663,
  44.00,
    0.00,1.070, 0.20,1.060, 0.60,1.037, 0.80,1.027, 1.00,1.018,
    1.20,1.010, 1.50, .998, 1.80, .987, 2.00, .980, 2.50, .961,
    3.00, .942, 4.00, .910, 6.00, .867, 10.0, .821, 30.0, .740,
  90.00,
    0.00,1.070, 0.20,1.060, 0.60,1.037, 0.80,1.027, 1.00,1.018,
    1.20,1.010, 1.50, .998, 1.80, .987, 2.00, .980, 2.50, .961,
    3.00, .942, 4.00, .910, 6.00, .867, 10.0, .821, 30.0, .740,
      ENDPHS = 1,
 $
P$GENDAT EVENT(1)  = 95., 1.,
 NPC(2) = 2,
 CRITR      = 6HALTITD,
 VALUE      = 102000.,
 TIMRF(1)   = 0.,
 MDL        = 3,
 NPC(26) = 0,
 ITAP       = 1,1,1,1,1,1,1,1,1,1,
 ARP        = 3288., 2955., 803., 1844., 297., 1408., 980., 240., 140., 0.,
 HRAT       = .06,   .17,   .43,  .78,   .97,  1.12,  1.62, 2.5,  3.5,  0.,
```

Figure 32.- Continued

```
 $
PSTBLMLT
 WUA1M      = 1.25,
 $
 PSTAB TABLE = 6HWUA1T   , 2 , 6HTHTPL , 6HTIME   , 6,3,1,1,1,1,1,1,1,1,
 1500.,
  5.991,.37, 7.313,.88, 8.517,1.36, 9.21,1.64, 9.903,1.96, 10.463,2.21,
 1800.,
  5.858,.50, 6.908,.91, 8.294,1.47, 9.21,1.85, 9.903,2.20, 10.463,2.48,
 2700.,
  5.737,.64, 6.908,1.23, 8.294,1.94, 9.21,2.45, 9.903,2.87, 10.404,3.20,
 ENDPHS=1,
 $
PSGENDAT EVENT(1)  = 98., 1.,
 CRITR      = 6HALTITO,
 TIMRF(1)   = 0.,
 VALUE      = 90000.,
 MDL        = 3,
 NPC(15)    = 0,
 NPC(26)    = 0,
 ENDPHS     = 1,
 $
PSGENDAT EVENT(1)  = 100.,
 CRITR      = 6HALTITO,
 VALUE      = 50000.,
 MDL        = 3,
 ENDPHS     =1,
 ENDPRB=1,

 $

PSSEARCH
 OPTVAR     = 6HTIMRF1,
 OPTPH      = 60.,
 PCTCC      = .1,
 NINDV      = 4,
 INDVR      = 6HDBANK ,6HDBANK ,6HDBANK ,6HCRITR ,
 INDPH      = 60., 70., 75., 60.,
 U          = -26.7182816,
              -23.6650460,
              -11.1222061,
              100.,
 PERT = 0.001,0.001,0.001,0.01,
 NDEPV      = 1,
 DEPVR      = 6HCRRNG ,
 DEPPH      = 100.,
 DEPVAL     = -1200.,
 DEPTL      = 5.,
 ENDJOB     = 1,
 $
```

SEARCH DATA FOR SECOND PROBLEM

Figure 32.- Concluded

Example 3. Orbital Maneuvers

This example illustrates the application of POST to a finite-burn orbit transfer problem. The basic problem is to determine the optimal location, duration, and attitude of two thrusting maneuvers that transfer an orbital vehicle (Transtage) from a near-Earth park-orbit to a geostationary orbit. Final weight is again used as the optimization variable. However, as demonstrated in Example 1, this is the same as maximizing the deliverable payload. A simplified version of this problem could be easily set up using the impulsive option. The more detailed simulation is presented, however, to illustrate a number of specific flight phases that naturally occur in orbital analysis problems. These phases are: (1) non-Keplerian coast segments using the Krogh integrator, (2) attitude reorientation maneuvers that simulate the kinematics (not dynamics) of the RCS, (3) propellant settling via thrusting maneuvers, and (4) long-duration finite-thrust orbital maneuvers.

The basic transfer geometry is illustrated in Figure 33. As indicated, the problem starts at booster burnout, which occurs 483.893 second into the flight. The first phase, Event 1.0 to Event 125.0, is a non-Keplerian coast to the first equatorial crossing. The occurrence of the equatorial crossing is demined by specifying "latitude equals zero" as an event criteria. This is accomplished via the $GENDAT input shown below.

```
P$GENDAT
 EVENT = 125.,
 CRITR = 5HGCLAT,
 VALUE = 0.0,
 ENDPHS = 1,
 $
```

represents the crossing condition

Event 125.0 is used to "trigger" a sequence of (1) reorientation, (2) propellant settling, and (3) main engine start-up transient phases. The primary main engine burn starts at Event 148.0 and ends some 325 seconds later at Event 150.0. Event 150.0 also initiates the coast-to-apogee defined by Event 260.0. At apogee the same basic sequence of maneuvers are used again to simulate the final circularization maneuver, which ends at Event 280.0. A long coast trajectory is then propagated until the vehicle reaches its fourth equatorial crossing at the final event, Event 300.0.

The POST input deck for this problem is shown in Figure 34. Comments are placed on each important data card to assist the reader in understanding the input set up. The ability to place comments on actual data cards is a feature unique to the POST NAMELIST processor, and is not available on most standard system NAMELIST routines.

The targeting and optimization formulation and input set up used in this case deserves some explanation. First, the optimization variable is WEIGHT at Event 300.0. This is also the WEIGHT at Event 280.0 because the variable WEIGHT does not change during the coast phase between Events 280.0 and 300.0. New users often think this practice to be inefficient. However, in this case it is not because the orbital conditions (constraints) are also defined at the fourth equatorial crossing, and so the trajectory must be propagated to Event 300.0 to calculate the dependent variables.

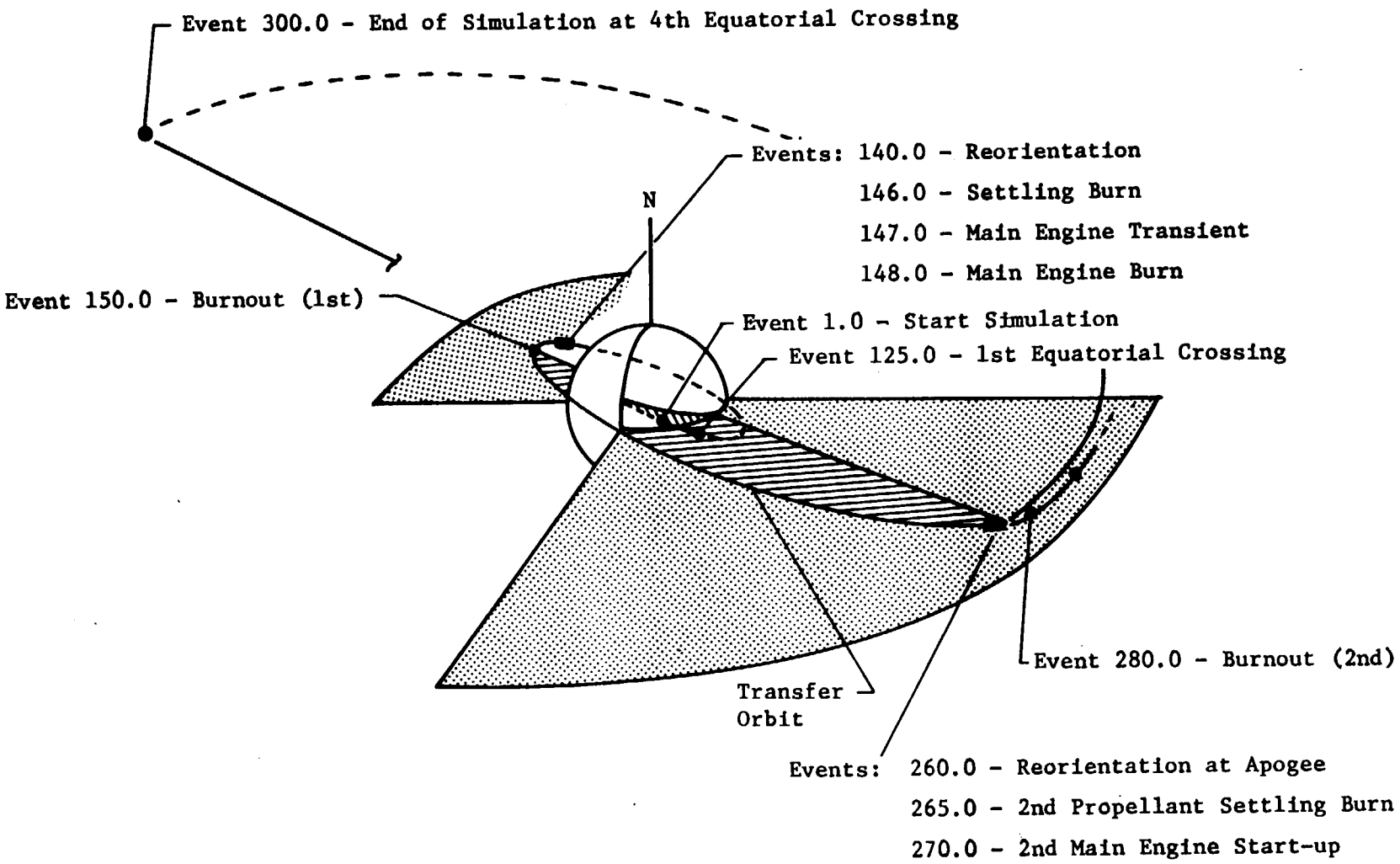


Figure 33.- Orbital Transfer Geometry

```
P$SEARCH
C
C
 LISTIN  =1,                                / INPUT NAMELIST AND TABLE
                                            / SUMMARY
 IDEB    =1,                                / PRINT TRIAL STEP SUMMARY
 MAXITR  =3,                                / MAXIMUM PERMISSIBLE NUMBER OF
                                            / ITERATIONS
 SRCHM   =4.,                               / USE ACCELERATED PROJECTED-
                                            / GRADIENT ALGORITHM
 IPRO    =0,                                / PRINT INITIAL AND FINAL
                                            / NOMINAL TRAJECTORIES
C....    CONTROL VARIABLE SPECIFICATION
 NINDV   =9,                                / NUMBER OF CONTROL VARIABLES
 INDVR   =6HCRITR ,          6HPITPC1,          6HYAWPC1,
          6HPITPC2,          6HCRITR ,
          6HCRITR ,          6HPITPC1,          6HYAWPC1,
          6HCRITR ,
 INDPH   =140.,              140.,              140.,
          148.,              150.,
          260.,              260.,              260.,
          280.,
 U       =6.90142019,        -3.67420779,       -7.93809841,
          -.0468945159,      321.887057,
          180.136723,        1.69258420,        43.7987715,
          109.526823,
 PERT    =.01,               .01,               .1,
          .001,              .01,
          .01,               .01,               .01,
          .01,
C....    CONSTANTS CONTROLLING AUTOMATIC PERT SELECTION
 NPAD    =9,                 5,                 13,
 PDLMAX  =2,
 MODEW   =0,
 WU      =1.,                .1,                1.,
          10.,               .1,
          1.,                1.,                .1,
          .1,
C....    CONSTRAINT VARIABLE SPECIFICATION
 NDEPV   =5,
 DEPVR   =6HSEMJAX,          6HECCEN ,          6HINC   ,
          6HARGP  ,          6HLONG  ,
 DEPPH   =300.,              300.,              300.,
          300.,              300.,
 DEPVAL  =138973648.,        .03,                5.,
          180.,              58.96,
 DEPTL   =10000.,            .0001,             .01,
          .01,               .01,
C....    OBJECTIVE SPECIFICATION
 OPT     =1.,                               / MAXIMIZE OBJECTIVE
```

Figure 34.- Input Data Deck for Orbital Maneuver Sample Case

```
 OPTVAR  =6HWEIGHT,                          / OPTIMIZATION VARIABLE
 OPTPH   =300.,                              / OPTIMIZATION PHASE
 PCTCC   =.1,                                / MAXIMUM RELATIVE CHANGE
                                             / ALLOWED IN MAGNITUDE OF
                                             / WEIGHTED CONTROL VECTOR
 STPMAX  =1.,                                / MAXIMUM ABSOLUTE CHANGE
                                             / ALLOWED IN MAGNITUDE OF
                                             / WEIGHTED CONTROL VECTOR
C....    ITERATION CONVERGENCE TOLERANCES
 CONEPS  =89.9,              1.0E-20,                1.0E-10,
          1.0E-05,           1.0E-20,
 P2MIN   =2.,                                / VALUE OF SUM OF SQUARES
                                             / OF ERRORS BELOW WHICH
                                             / ITERATION IS CONSIDERED
                                             / TARGETED
C....    CURVE FITTING TOLERANCES
 CONSEX  =1.0E-05,           1.0E-05,
 FITERR  =1.0E-06,           1.0E-03,
 $
P$GENDAT EVENT(1)=1.,
 TITLE(1)=50H 2015                                                      ,
 TITLE(6)=50H STAGE III INITIALIZATION                                  ,
 FESN    =300.,
C....    TRAJECTORY ABORT SPECIFICATION
 MAXTIM  =50000.,                            / MAXIMUM TRAJECTORY TIME
 ALTMIN  =0.,                                / MINIMUM ALTITUDE
C....    PROPAGATION SPECIFICATION
 NPC(2)  =2,                                 / VARIABLE STEP/ORDER PREDICTOR
                                             / CORRECTOR
 EPSINT  =.1,                                / DESIRED INTEGRATION ACCURACY
 PRNT(63)=6HDT     ,                         / PRINT INTEGRATION STEP SIZE
 DT      =100.,                              / INITIAL ESTIMATE OF
                                             / INTEGRATION STEP SIZE
 PINC    =10.E10,                            / INTERVAL BETWEEN PRINT BLOCKS
C....    VEHICLE WEIGHT SPECIFICATION
 WPROPI  =23379.,                            / INITIAL PROPELLANT LOAD
 NPC(5)  =0,                                 / NO ATMOSPHERE
 NPC(1)  =3,                                 / CALCULATE CONIC ELEMENTS AT
                                             / EACH INTEGRATION STEP
 NPC(9) = 0,                                 / NO ROCKET THRUST
C....    STEERING SPECIFICATION
 IGUID(1)=4,                                 / PITCH PLANE STEERING
 IGUID(16)=0,                                / DETERMINE PITCH EULER ANLGE
                                             / DIRECTLY FROM INPUT
 IGUID(4)=1,                                 / ATTITUDE ANGLES ARE INPUT
                                             / POLYNOMIALS WITH INPUT
                                             / CONSTANTS
C....    INITIALIZE STATE VECTOR FROM BOOST TRAJECTORY
 NPC(3)  =2,                                 / INITIALIZE VELOCITY VECTOR
                                             / THROUGH INERTIAL COMPONENTS
                                             / IN G-FRAME
```

Figure 34.- Continued

```
 VELI     =26032.975,
 GAMMAI   =0.,
 AZVELI   =105.04684,
 NPC(4)   =2,                              / INITIALIZE POSITION VECTOR
                                           / THROUGH INERTIAL SPHERICAL
                                           / COORDINATES
 GCLAT    =25.310029,
 LONGI    =297.99561,
 GCRAD    =21411784.,
 TIME     =483.893,
 WGTSG    =30840.331,
 TIMEO    = -3.,                           / TIME AT WHICH ECI FRAME IS
                                           / INITIALIZED
 LATL     =28.56205,                       / LATITUDE OF L-FRAME ORIGIN
 LONL     =279.42257,                      / LONGITUDE OF L-FRAME ORIGIN
 AZL      =100.170556,                     / AZIMUTH OF L-FRAME Z-AXIS
 NPC(16)  = 0,                             / OBLATE PLANET GRAVITY MODEL
C....     SMITHSONIAN EARTH MODEL
 GO       =32.174,                         / CONVERSION FACTOR FROM SLUGS
                                           / TO LBM
 J2       =.001082639,                     / 2ND ZONAL HARMONIC COEFFICIENT
                                           / IN EARTH POTENTIAL FUNCTION
 MU       =1.4076457994E+16,               / NEWTONIAN GRAVITATIONAL
                                           / CONSTANT
 J3       =-2.565E-06,                     / 3RD ZONAL HARMONIC COEFFICIENT
                                           / IN EARTH POTENTIAL FUNCTION
 J4       =-1.608E-06,                     / 4TH ZONAL HARMONIC COEFFICENT
                                           / IN EARTH POTENTIAL FUNCTION
 OMEGA    =7.292115146E-05,                / EARTH ROTATIONAL RATE
 RE       =20925662.7,                     / EARTH EQUATORIAL RADIUS
 RP       =20855502.4,                     / EARTH POLAR RADIUS
C....     SPECIAL PURPOSE PRINT BLOCK
 PRNT(1)  = 6HTIME  , 6HALTITO, 6HVELI  , 6HGAMMAI, 6HWEIGHT, 6HTHRUST,
 PRNT(7)  = 6HTIMES , 6HGCRAD , 6HVELR  , 6HGAMMAR, 6HWDOT  , 6HASM   ,
 PRNT(13)= 6HTDURP , 6HGDLAT , 6HGCLAT , 6HLONG  , 6HWPROP , 6HAZVELI,
 PRNT(19)= 6HIB11  , 6HIB12  , 6HIB13  , 6HXI    , 6HVXI   , 6HAXI   ,
 PRNT(25)= 6HIB21  , 6HIB22  , 6HIB23  , 6HYI    , 6HVYI   , 6HAYI   ,
 PRNT(31)= 6HIB31  , 6HIB32  , 6HIB33  , 6HZI    , 6HVZI   , 6HAZI   ,
 PRNT(37)= 6HASXI  , 6HROLI  , 6HROLR  , 6HURX   , 6HUNX   , 6HXF    ,
 PRNT(43)= 6HASYI  , 6HYAWI  , 6HYAWR  , 6HURY   , 6HUNY   , 6HYF    ,
 PRNT(49)= 6HASZI  , 6HPITI  , 6HPITR  , 6HURZ   , 6HUNZ   , 6HZF    ,
 PRNT(55)= 6HFTXB  , 6HFTYB  , 6HFTZB  , 6HAXB   , 6HAYB   , 6HAZB   ,
 PRNT(61)= 6HTIMRF1, 6HW     , 6HDT    , 6HPSTOP ,
 ENDPHS = 1.,
 $
PSGENDAT EVENT(1)=125.,
 TITLE(6)=50H FIRST EQUATORIAL CROSSING                              ,
 CRITR   =6HGCLAT ,
 VALUE   =0.,
 ENDPHS  =1.,
 $
```

Figure 34.- Continued

```
P$GENDAT EVENT(1) = 140.,
 TITLE(6)=50H REORIENT TRANSTAGE ATTITUDE FOR 1ST BURN          ,
 CRITR    =6HGCLAT ,
 VALUE    =5.,
 MDL      =3,
C....     STEERING MODE
 IGUID(1)=2,                                  / RELATIVE EULER ANGLES
 IGUID(13)=3,                                 / INERTIAL VELOCITY VECTOR AS
                                              / YAW REFERENCE
 PITPC(1)=0.,
 YAWPC(1)=-1.4576,
 ROLPC(1)=0.,
C....     PROPAGATION SPECIFICATION
 DT       =2.,                                / INITIAL STEP SIZE EXTIMATE
 ENDPHS   = 1.,
 $
P$GENDAT EVENT(1)=146.,
 TITLE(6)=50H TRANSTAGE 1ST PROPELLANT SETTLING BURN            ,
 CRITR    = 6HTDURP ,
 VALUE    =0.,
 MDL      =1,
C....     PROPULSION SPECIFICATION
 NPC(9)   =1,                                 / ROCKET THRUST
 NENG     =1,                                 / NUMBER OF ROCKET ENGINES
C....     VEHICLE WEIGHT SPECIFICATION
 WJETT    =6.,                                / JETTISON ACS PROPELLANT
C....     STEERING SPECIFICATION
 IGUID(1)=4,                                  / PITCH-PLANE STEERING
 IGUID(4)=0,                                  / CARRY-OVER ATTITUDE FROM
                                              / PREVIOUS PHASE
 $
P$TBLMLT
 $
C....     PROPELLANT SETTLING THRUST
P$TAB TABLE    =6HTVC1T ,          0.,                  206.83,
 $
C....     PROPELLANT SETTLING FLOW RATE
P$TAB TABLE    = 4HWD1T,           0.,                  0.93333333,
 ENDPHS = 1.,
 $
P$GENDAT EVENT(1)=147.,
 TITLE(6)=50H T/S FIRST BURN START-UP (1ST 3 SECONDS) 138FS1-1 ,
 CRITR    =6HTDURP ,
 VALUE    =15.,
 $
P$TBLMLT
 $
C.... TRANSTAGE FIRST BURN THRUST
P$TAB TABLE    =6HTVC1T ,          0.,                  16000.,
 $
C....     TRANSTAGE FIRST BURN FLOWRATE (FLOWRATE IS 16000 LB / 302 SEC
```

Figure 34.- Continued

```
C         PLUS 20 LB / 325 SEC FOR ABLATION
P$TAB TABLE    =6HWD1T  ,            0.,                    53.04167091,
 ENDPHS = 1.,
 $
P$GENDAT EVENT(1)=148.,
 TITLE(6)=50H TRANSTAGE FIRST BURN  (STEADY STATE)                    ,
 CRITR    =6HTDURP ,
 VALUE    =3.,
C....     STEERING SPECIFICATION
 PITPC(2)=-0.06746,                        / FIRST BURN PITCH RATE
 ENDPHS = 1.,
 $
P$GENDAT EVENT(1)=150.,
 TITLE(6)=50H FIRST BURN SHUTDOWN (138FS2-1)                          ,
 CRITR    =6HTDURP ,
 VALUE    =325.663005,
C....     PROPAGATION SPECIFICATION
 DT       =200.,                           / INITIAL STEP SIZE ESTIMATE
C....     STEERING SPECIFICATION
 PITPC(2)=0.,                              / ELIMINATE PITCH RATE
C....     PROPULSION SPECIFICATION
 NPC(9)   = 0.,                            / NO ROCKET THRUST
 ENDPHS   =1.,
 $
P$GENDAT EVENT    =260.,
 TITLE(6)=50H REORIENT TRANSTAGE ATTITUDE FOR 2ND BURN                ,
 CRITR    =6HTRUAN ,
 VALUE    =180.,
C....     PROPAGATION SPECIFICATION
 DT       =2.,                             / INITIAL STEP SIZE ESTIMATE
C....     STEERING SPECIFICATION
 IGUID(1)=2,                               / RELATIVE EULER ANGLES
 IGUID(13)=3,                              / INERTIAL VELOCITY VECTOR
 IGUID(4)=1,                               / ATTITUDE ANGLES ARE INPUT
                                           / POLYNOMIALS WITH INPUT
                                           / CONSTANTS
                                           / IS YAW REFERENCE
 PITPC(1)=2.9242,
 YAWPC(1)=50.639,
 ROLPC(1)=0.,
 ENDPHS   =1.,
 $
P$GENDAT EVENT    =265.,
 TITLE(6)=50H TRANSTAGE 2ND PROPELLANT SETTLING BURN                  ,
 CRITR    =6HTDURP ,
 VALUE    =0.,
C....     PROPULSION SPECIFICATION
 NPC(9)   =1,                              / ROCKET THRUST
 NENG     =1,                              / NUMBER OF ROCKET ENGINES
C....     VEHICLE WEIGHT SPECIFICATION
 WJETT    =43.,                            / JETTISON ACS PROPELLANT
```

Figure 34.- Continued

```
C....    STEERING SPECIFICATION
 IGUID(1)=4,                                  / PITCH PLANE STEERING
 IGUID(4)=0,                                  / CARRY OVER ATTITUDE FROM
                                              / FROM PREVIOUS PHASE
 $
PSTBLMLT
 $
C....    TRANSTAGE 2ND BURN PROPELLANT-SETTLING THRUST
PSTAB TABLE   =6HTVC1T ,          0.,                    187.23,
 $
C....    TRANSTAGE 2ND BURN PROPELLANT-SETTLING WEIGHT FLOW RATE
PSTAB TABLE   =6HWD1T  ,          0.,                    .83333333,
 ENDPHS  =1.,
 $
PSGENDAT EVENT   =270.,
 TITLE(6)=50H TRANSTAGE 2ND BURN CONSTANT ATTITUDE PHASE        ,
 CRITR   =6HTDURP ,
 VALUE   =6.,
 $
PSTBLMLT
 $
C....    TRANSTAGE 2ND BURN VACCUUM THRUST
PSTAB TABLE   =6HTVC1T ,          0.,                    16000.,
 $
C....    TRANSTAGE 2ND BURN WEIGHT FLOW RATE TABLE
PSTAB TABLE   =6HWD1T  ,          0.,                    53.05285972,
 ENDPHS  =1.,
 $
PSGENDAT EVENT   =280.,
 TITLE(6)=50H TRANSTAGE 2ND BURN SHUTDOWN (138FS2-2)            ,
 CRITR   =6HTDURP ,
 VALUE   =118.19,
C....    PROPAGATION SPECIFICATION
 DT      =1000.,                              / INITIAL STEP SIZE ESTIMATE
C....    PROPULSION SPECIFICATION
 NPC(9)  =0,                                  / NO ROCKET THRUST
 ENDPHS  =1.,
 $
PSGENDAT EVENT   =300.,
 TITLE(6)=50H 4TH EQUATORIAL CROSSING                           ,
 CRITR   =6HGCLAT ,
 VALUE   =0.,
 ENDPHS  =1.,
 ENDPRB  =1.,
 ENDJOB  =1.,
 $
```

Figure 34.- Concluded

Second, there are five orbital "target" conditions defined that describe the desired final orbit. The first four constraints are the standard Keplerian orbital elements specifying semimajor axis, eccentricity, inclination, and argument of perigee. The last constraint is longitude at Event 300.0. Now Event 300.0 defines the fourth equatorial crossing, which in this case is an ascending crossing. As a result, the final constraint is mathematically equal to the longitude of the ascending node. At first this seems a rather obtruse procedure for computing the longitude of node. However, this special technique is generally required to eliminate the problems encountered in targeting to very low inclination orbits, where as a result of small changes in the maneuver, the ascending node can discontinuously change by 180 deg. This is because during a long thrust maneuver a "seemingly" ascending node can be changed to a descending node with only minor changes in the yaw-turn steering parameters.

Third, there is a total of nine independent variables defined in this sample case. The first five of these variables are, because of their event numbers, associated with the first main engine apogee-raising burn. The first control variable in the event criteria at Event 140. Review of the data deck reveals that CRITR at Event 140.0 is latitude. Thus, this first control variable is the latitude at which the apogee raising maneuver is started. The second and third control variables are the initial relative pitch and yaw attitude angles, respectively. The fourth variable is the pitch rate during the burn, and the fifth variable can be shown to be the burn time of the main engine. The final four control variables are similarly defined, and apply to the final circularization maneuver. Notice in the final maneuver a constant relative attitude is maintained because the pitch rate coefficient PITPC2 has been omitted from the final set of control parameters.

The mathematical formulation of this orbital transfer problem may be summarized as follows:

Determine the values of the nine control parameters

$$\underline{u} = (\phi_1, \theta_1, \psi_1, \dot{\theta}_1, T_{B1}, \phi_2, \theta_2, \psi_2, T_{B2}) \tag{41}$$

that maximize: W_f,

subject to the final orbital conditions (constraints);

$$a(t_f) - 13\ 893\ 648 \text{ ft}$$

$$e(t_f) - 0.030 = 0,$$

$$i(t_f) - 5.0 \text{ deg} = 0,$$

$$\omega(t_f) - 180 \text{ deg} = 0,$$

$$\theta_R(t_f) - 58.96 \text{ deg} = 0,$$

where ϕ is geocentric latitude, θ is relative Euler pitch angle, ψ is relative Euler yaw angle measure from the azimuth of the inertial velocity vector, $\dot{\theta}$ is the time derivative of θ, T_B is the burn time of the main engine, a is semi-major axis, e is eccentricity, i is inclination, ω is argument of perigee, θ_R is the longitude of ascending node, and W is total spacecraft weight. The subscripts 1, 2, and f denote the first and second maneuvers and the final conditions, respectively. The relationship between this mathematical formulation and the required $SEARCH inputs can be distilled by studing Figure 34. There are a number of additional inputs in $SEARCH that are optimization algorithm control parameters, and are not related to the problem formulation. Brief descriptions of these parameters are given in the comments on the individual data cards. More complete definitions are contained in reference 1.

Example 4. Hypersonic Aircraft Point Performance

This example is presented to illustrate the application of the POST program to aircraft point performance problems. This is a unique application of POST in that the trajectory time history is not generated. Instead, the complete optimization takes place at a single fixed point in time--hence, the term point performance. See Figure 35.

The basic problem is to determine the maximum cruise velocity of a hypersonic aircraft such as the X-24C. The maximum velocity, of course, depends on the configuration and the particular propulsion system used. For a particular configuration and engine, the problem may be stated more precisely as follows:

Determine the cruise altitude, velocity, and angle of attack h, V_R, α to

maximize: V_R

subject to: $\dot{V}_R = 0$

$\dot{\gamma}_R = 0$

The constraints on the total derivative of relative velocity and flight path angle are used instead of the standard cruise conditions of "thrust equals drag" and "lift equals weight." This is due to two important accelerations that cannot be ignored in the hypersonic flight regime: (1) the thrust component in the lift direction, and (2) the centripetal acceleration.

The complete input deck for this example is presented in Figure 36. There are only two events in this case: EVENT = 1.0, the first event, and EVENT = 100.0, the final event. The final event criteria, CRITR = 5HTDURP, is the elapsed time between EVENT = 1.0 and EVENT = 100.0. Notice that the value of the final event criteria is zero, i.e., VALUE = 0.0. This means that zero time evolves between events 1.0 and 100.0. This type of event is referred to as a "zero-length event" in POST terminology.

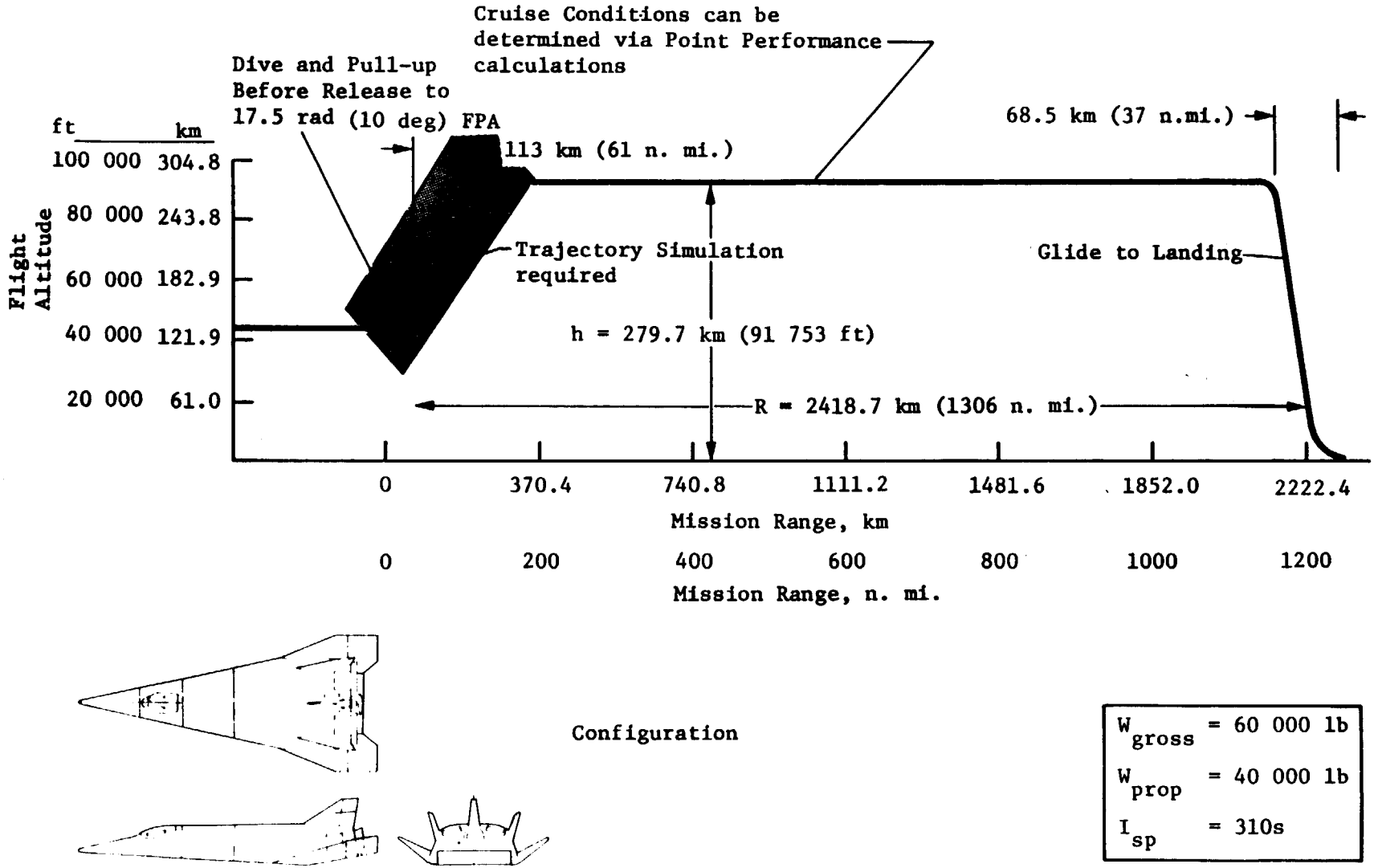


Figure 35.- Mission Profile and Vehicle Sketch for Hypersonic Point Performance

```
P$SEARCH
 CONEPS(1) = 89.,
 MAXITR    = 10,
 OPT       = 1,
 SRCHM     = 5,
 PERT      = .01, 1.E-4, 1.E-6,
 OPTVAR    = 6HVELR  ,
 OPTPH     = 100.,
 NDEPV     = 2,
 DEPVR     = 6HVELAD , 6HGAMAD ,
 IDEPVR    = 0,0,
 DEPPH     = 2*100.,
 DEPVAL    = 0.0,  0.0,
 DEPTL     = 0.001,  0.001,
 NINDV     = 3,
 INDVR     = 6HALTITO, 6HVELR  ,6HALPPC1,
 INDPH     = 3*1.,
 U         = 90000., 4500., 10.,
 PCTCC     = .02,
 $
P$GENDAT EVENT(1)  = 1.0,
 TITLE     = 0H*SET-UP FOR MAX. CRUISE RANGE --- POINT PERFORMANCE*,
 NSPEC(2)  = 1,
 FESN      = 100.0,
 WPROPI    = 9000.,
 WGTSG     = 27500.,
  SREF   = 569.2,
 MAXTIM    = 600.0,
C  *****  NPC(2)  SELECTS INTEGRATION SCHEME
 NPC(2)    = 1,
 PINC      = 10.,
 DT        = 5.,
C  *****  NPC(3)  SELECTS VELOCITY INITIALIZATION  4 = EARTH RELATIVE
C  **  COMPONENTS IN LOCAL HORIZONTAL FRAME
 NPC(3)    = 4,
 VELR      = 4500.,
 GAMMAR    = 0.,
 AZVELR    = 90.,
C  *****  NPC(4) SELECTS POSITION INITIALIZATION
C **  2 = SPHERICAL COORDINATES
 NPC(4)    = 2,
 TIME      = 0.,
 ALTITO    = 100000.,
 GDLAT     = 0.0,
 LONG      = 0.0,
 NPC(9)    = 1,
 NPC(9)    = 5,
 NENG      = 1,
C  *****  EARTH MODEL PER JPL REPORT 32-1306, 15 JULY 1968.
C  *****  NPC(5)  SELECTS ATMOSPHERE MODEL  ( 2 = 1962 STANDARD )
```

Figure 36.- Input Data Deck for Hypersonic Aircraft Sample Case

```
 NPC(5)     = 2,
C  *****  NPC(8) SELECTS AERODYNAMIC INPUT OPTION (1=INPUT CAT AND CNAT)
 NPC(8)     = 1,
 LREF       = 23.0,
 RN         = 1.,
C  *****  NPC(12) SELECTS RANGE CALCULATION (1 = GREAT CIRCLE RELATIVE)
 NPC(12)    = 1,
C  *****  NPC(22) DEFINES ENGINE THROTTLING    (  2  =  ETA CURVE )
C  *****  GUIDANCE COMMANDS
 IGUID      = 0,0,1,
 NPC(22)    = 1,
 ETAPC(1)   = 1.,
 PRNT(91)   = 6HVELAD , 6HSPECV3,
 $
PSTBLMLT
 AE1M       = 25.,
 $
C  **  CA TABLE X24C-7 65: CG 5/30/73
PSTAB TABLE      = 3HCAT, 2, 4HMACH, 5HALPHA, 12,  5,1,1,1,1,1,1,1,1,
  0.,
  0.0,.0467,.6,.0468,.8,.0483,1.0,.0856,1.2,.0944,1.5,.0735,2.0,.0643,
   3.0,.0514,4.0,.0461,5.0,.0414,6.0,.0403,100.,.0403,
  4.,
   0.,.0347,.6,.0348,.8,.0390,1.0,.0775,1.2,.0842,1.5,.0660,2.0,.0561,
   3.0,.0446,4.0,.0386,5.0,.0359,6.0,.0348,100.0,.0348,
  8.,
   0.0,.0257,.6,.0258,.8,.0325,1.0,.0743,1.2,.0798,1.5,.0622,2.0,.0503,
   3.0,.0400,4.0,.0343,5.0,.0319,6.0,.0311,100.0,.0311,
  12.,
   0.0,.0177,.6,.0178,.8,.0290,1.0,.0738,1.2,.0809,1.5,.0609,2.0,.0463,
   3.0,.0363,4.0,.0316,5.0,.0299,6.0,.0289,100.0,.0289,
  20.,
   0.0,.0077,.6,.0078,.8,.0310,1.0,.0794,1.2,.0912,1.5,.0628,2.0,.0417,
   3.0,.0313,4.0,.0311,5.0,.0334,6.0,.0312,100.,.0312,
 $
C  **  CN TABLE X24C-7 65: CG 5/30/73
PSTAB TABLE      = 4HCNAT,2, 4HMACH, 5HALPHA, 12,  5,1,1,1,1,1,1,1,1,
  0.,
  0.0,-.04, .6, -.04, .8, -.047, 1.0, -.033, 1.2, -.05, 1.5, -.032,
  2.0,-.032, 3.0,-.032, 4.0,-.032, 5.0,-.032, 6.0,-.032, 100.0,-.032,
  4.,
  0.0, .04, .6, .04, .8, .039, 1.0, .05, 1.2, .033, 1.5, .038,
  2.0, .029, 3.0, .025, 4.0, .025, 5.0, .019, 6.0, .016, 100.0, .016,
  8.,
  0.0, .125, .6, .125, .8, .125, 1.0, .137, 1.2, .12, 1.5, .12,
  2.0, .105, 3.0, .093, 4.0, .091, 5.0, .08, 6.0, .076, 100.0, .076,
  12.,
  0.0, .205, .6, .205, .8, .21, 1.0, .222, 1.2, .205, 1.5, .21,
  2.0, .191, 3.0, .172, 4.0, .161, 5.0, .148, 6.0, .141, 100.0, .141,
  20.,
  0.0, .366, .6, .366, .8, .383, 1.0, .395, 1.2, .377, 1.5, .409,
```

Figure 36.- Continued

```
 2.0, .39, 3.0, .35, 4.0, .317, 5.0, .297, 6.0, .287, 100.0, .287,
 $
P$TAB TABLE      = 6HWD1T  , 1, 6HMACH  , 6, 3,1,1,
 2.5,  6.390E-4,     3.0,  6.364E-4,
 3.5,  6.457E-4,     4.0,  6.344E-4,
 4.5,  6.311E-4,     4.65, 6.667E-4,
 $
P$TAB TABLE      = 6HTVC1T , 1, 6HMACH  , 9, 3,1,1,
 2.5, 1.33,    3.0, 1.32,    3.5, 1.27,
 4.0, 1.19,    4.25,1.13,    4.40,1.08,
 4.50,1.03,    4.60,0.95,    4.65,0.90,
 $
P$TAB TABLE      = 4HAE1T, 0,  1.0,
 ENDPHS     = 1,
 $
P$GENDAT EVENT      = 100.0,
 CRITR      = 6HTDURP ,
 VALUE      = 0.,
 ENDPHS     = 1,
 ENDPRB     = 1,
 ENDJOB     = 1,
 $
```

Figure 36.- Concluded

TABLE 14.- ITERATION SUMMARY FOR HYPERSONIC AIRCRAFT POINT PERFORMANCE SUMMARY

Iteration	Optimization Variable, Velocity	Constraint Error, $\sim P_2$	Optimization Indicator
0	4188	1.90×10^{3}	46.419
1	4284	1.60×10^{-2}	55.373
2	4401	8.21×10^{-1}	71.232
3	4455	1.23×10^{-1}	81.197
4	4507	8.56×10^{0}	---
5	4506	1.72×10^{-4}	85.659
6	4521	1.34×10^{-1}	87.019
7	4535	9.444×10^{0}	---
8	4534	2.11×10^{-3}	89.026
9	4536	1.05×10^{-1}	---

The iterative optimization begins by the user estimating an initial "guess" for the independent variables.

$$\underline{u} = \left(h,\ V_R,\ \alpha\right) = (90\ 000.,\ 4500.,\ 10.). \tag{42}$$

This initial guess, input in $SEARCH (see Figure 36) generally does not satisfy the constraints (cruise conditions). The projected gradient algorithm then determines the proper correction to this initial guess to satisfy the constraints. This correction is based upon numerical approximations to the partial derivatives of the constraints. This set of partials, called the sensitivity (or more precisely the Jacobian) matrix is given as

$$[S]_{2x3} = \frac{\partial\left(\dot{V}_R,\ \dot{\gamma}_R\right)}{\partial\left(h,\ V_R,\ \alpha\right)} = \begin{bmatrix} \frac{\partial \dot{V}_R}{\partial h} & \frac{\partial \dot{V}_R}{\partial V_R} & \frac{\partial \dot{V}_R}{\partial \alpha} \\ \frac{\partial \dot{\gamma}_R}{\partial h} & \frac{\partial \dot{\gamma}_R}{\partial V_R} & \frac{\partial \dot{\gamma}_R}{\partial \alpha} \end{bmatrix} \tag{43}$$

After a feasible solution is attained, that is, one that satisfies the constraints, the algorithm is designed to "move along" the constraint manifold and improve the cruise speed at each iteration. Conceptually, this is accomplished by projecting the gradient of cruise speed to the plane defined by the normals to the constraints. This mathematical operation determines the direction of "steepest ascent (descent)" in a plane that approximates the constraints. However, because of nonlinearities in the constraints, as the control parameters are varied along this direction of search the cruise condition becomes increasingly violated. The extent to which the control parameters are incremented is then determined by maximizing a composite function (called estimated net performance), which is the sum of the cruise speed and an estimate of the loss in cruise speed associated with this violation of the cruise conditions. After each such optimization step a constraint restoration step is taken to remove this induced constraint error. As a result, once a feasible solution is found, the optimization proceeds in a sequence of two steps: first, optimization; and second, constraint restoration. Significant, however, is the fact that in most normal restoration steps, the sensitivity matrix is not recomputed. This procedure substantially reduces the overall computer solution time required to optimize the problem. A brief summary of this iteration process is given in Table 14. Notice that after a feasible solution is achieved in iteration 1, all subsequent iterations satisfy the cruise equations.

This simple application of POST solved this "point" performance problem in a matter of only a few seconds on the CDC 6500. As a result, it eliminates the requirement of plotting families of thrust required and thrust available curves to determine graphically the cruise performance of a high performance aircraft configuration.

Example 5. Trajectory Decomposition

The optimization of an eastern hemisphere geostationary mission using the decomposition option is presented as Example 5. This mission, illustrated in Figure 37, has two basic flight regimes and is typical of the problems that can be efficiently optimized with the decomposition approach. It is important to note that the decomposition option is contained in a special version of 3D POST. However, the utilization procedures are available in reference 1. A general problem statement and the two formulations are given in Table 15. The standard nondecomposition formulation is a constrained discrete parameter optimization problem containing eight constraints and ten independent variables. Let this problem be denoted as P1(8x10). The data deck is presented in Figure 38. The inputs required by the decomposition appear first in the deck. The remainder of the data deck is the same as the standard nondecomposition input. As indicated, the complete mission was decomposed to three trajectory segments:

Ascent;

Apogee Raising Burn;

Final Circularization and Plane Change.

This problem could be decomposed differently; for example, the two orbital segments could easily be combined in a single subproblem. The master problem, as formulated, has only one constraint and three control variables; let the problem be denoted as MP1(1x3). It is important to note that in MP1(1x3) two of the three control variables are constraint values in the subproblems. This enables the master algorithm to optimize the burnout velocity of the ascent leg and the inclination of the transfer orbit. These are two important geostationary mission design variables. The constraint that zero propellant remains at final burnout is really the active inequality constraint $W_{pr} \geq 0$. This is to ensure that the program will not simulate consumption of more propellant than can be loaded in the tanks. This constraint is defined at the master level because it couples the subproblems due to the fact that Stage III is used in all three of the trajectory segments. The subproblems, represented as SP1(3x3), SP2(2x2), and SP3(4x4), are all relatively small targeting problems. Clearly, this sequence of subproblems is much easier to solve than P1(8x10). However, a computational tradeoff is involved because the sequence of subproblem $(SP_i : i = 1, 2, 3)$ must be solved on every trial step in univariate searches used to solve MP1(1x3). This fact emphasizes the importance of rapid subproblem solution.

Computational results for each formulation are presented in Table 16 and 17. The accelerated gradient projection option was used to solve the standard formulation. On iterations 3 and 7, it had to be restarted and did not converge in ten iterations. In comparison, the decomposition algorithm did not require manual restarting and converged in only six iterations. The increased reliability is probably more significant than the reduction in the number of iterations. This is because each iteration at the master level represents considerably more computational effort than does each iteration in the standard formulation.

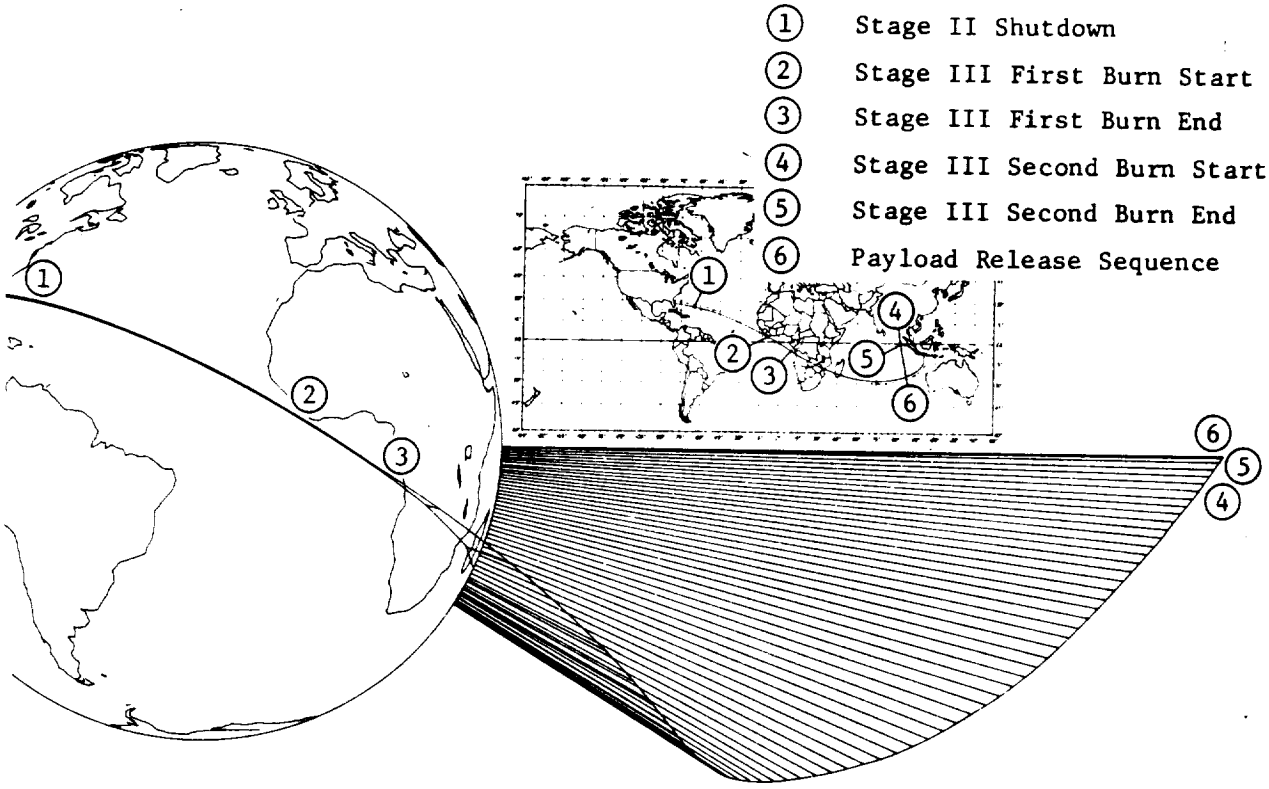


Figure 37.- Mission Profile for Decomposition Sample Case

Example 6. 6DOF Space Shuttle Entry Simulation

A typical data deck for a 6DOF closed-loop simulation of the Space Shuttle entry trajectory is presented. The changes, additions and modifications required to convert a "typical" 3D POST Space Shuttle entry data deck to this 6D POST data deck are highlighted in Figure 39. As indicated, the primary changes are related to the additional data cards required for (1) mass properties, (2) aerodynamics, and (3) the flight control system (FCS). More significant than these minor data additions are the vehicle-dependent subroutines that must be supplied by the user to model the FCS. Development, coding, and checkout of these subroutines represent the vast majority of the engineering effort in using 6D POST. However, this requirement is certainly not unique to 6D POST, but rather is required by all 6DOF simulation programs. The key issue then is really related to the ease in which these routines can be implemented into the program. These procedures, described in reference 5, have been made as simple as possible to reduce the "start-up" costs of using 6D POST. This simplistic simulation of Space Shuttle entry requires about 100 000 octal cells of computer memory, and executes in about three-times realtime on the CDC 6500.

... FORMULATION AND DECOMPOSITION FORMULATION OF THE GEOSTATIONARY MISSION

Determine the attitude steering history and burn times that maximize the payload a Titan IIIC class vehicle can deliver to an eastern hemisphere geostationary orbit. Assume that the elements for the optimum park orbit and transfer orbit are unknown and are to be determined.

Nondecomposition Formulation

Maximize: W_{pld} - Payload

Subject to the constraints:

$r^1 = 2.141197 \times 10^7$ ft - Ascent burnout radius

$\gamma^1 = 0.0$ deg - Ascent burnout flight path

$h_a^2 = 19\ 323$ n. mi. - Transfer apogee altitude

$V^3 = 10\ 087$ fps - SEO velocity

$\gamma^3 = 0.0$ deg - SEO flight path angle

$\Lambda^3 = 90.0$ deg - SEO flight azimuth

$\phi^3 = 0.0$ deg - SEO latitude

$W_{pr}^3 = 0.0$ lb - Propellant remaining

with the independent variables

W_{pld} - Payload

$\dot{\theta}_{11}$ - Initial Stage I pitchover rate

$\dot{\theta}_{12}$ - 2nd Stage I pitch rate

T_3^1 - Duration of Stage III burn to achieve park orbit

ψ^2 - Yaw angle during transfer burn

T_3^2 - Duration of transfer burn

ϕ^3 - Latitude to initialize circularization burn

ψ^3 - Yaw angle during circularization burn

θ^3 - Pitch angle during circularization burn

T_3^3 - Duration of circularization burn

Decomposition Formulation

Master Problem:

Maximize: W_{pld} - Payload

Subject to: $W_{pr}^3 = 0.0$ lb - Propellant remaining

with the independent variables

W_{pld} - Payload

V_d^1 - Ascent burnout velocity

i_d^2 - Inclination of the transfer orbit

Subproblem 1: Determine the values of $\dot{\theta}_{11}$, $\dot{\theta}_{12}$, and T_3^1 that solve the equations

$r^1 = 2.141197 \times 10^7$ ft - Fixed

$V^1 = V_d^1$ fps - Supplied by master algorithm

$\gamma^1 = 0.0$ deg - Fixed

Subproblem 2: Determine the values of ψ^2, T_3^2 that solve the equations

$h_a^2 = 19\ 323$ n. mi. - Fixed

$i^2 = i_d^2$ deg - Supplied by master algorithm

Subproblem 3: Determine the values of ϕ^3, ψ^3, θ^3, and T_3^3 that solve the equations

$V^3 = 10\ 087$ fps - Fixed

$\gamma^3 = 0.0$ deg - Fixed

$\Lambda^3 = 90.0$ deg - Fixed

$\phi^3 = 0.0$ deg - Fixed

```
P$SEARCH
C ------------------- MASTER PROBLEM ----------------
C
 SGFESN     =   100.          ,   250.          ,   400.
 SRCHMM=1,
 MAXITM=1,
 OPTVRM     =6HWPLD    ,
 OPTPHM     =10.0,
 OPVEIM=.02,
 NDEPVM     = 1               ,
 DEPVRM     = 6HWPROP         ,
 DEPPHM     =   400.          ,
 DEPVLM     =    0.0          ,
 DEPTLM     = .02,
C
 NINDVM     = 8               ,
C      IF MODEWM = 0, USE INPUT WUM AS WEIGHTING
 MODEWM     = 0,
C
 NOPTVM     = 1               ,
 IPRO=-1,
 IPRO=-2,
 NPAD=20,0,
 CONEPM(1)=+89.9999,
 CONEPM(2)=  1.0E-4, 1.0E-4, 1.0E-4, 1.0E-6,
 CONSXM(1)=  1.0E-5,
 CONSXM(2)=  1.0E-5,
 FITERM(1)=  1.0E-5,
 FITERM(2)=  1.0E-5,
 STMP1M = .01,
 STMP2M = .01,
 STPMXM = 10.0,
  IDEBM=1,
 NINDVM = 3,
 INDVRM=6HDEPVL1,6HWPLD   ,
        6HDEPVL1,
 INDPHM= 100.0   ,  1.0   ,
         250.0   ,
 UM     =+2.5678997933142E+4,+2.70000000000000E+3,
        +2.8500000000001E+1,
 WUM=1.0,.01,1.0,
 PERTM = 0.0               ,.02               ,
        0.0                ,
C      TRAJECTORY GENERATOR
 SRCHM      = 4,
 MAXITR     = 3               ,
 GAMAX=20.,
 NDEPV      = 8               ,
 DEPVR      = 6HVELI          , 6HGCRAD         , 6HGAMMAI
              6HINC           , 6HALTA          ,
```

Figure 38.- Input Deck for Trajectory D Composition

```
             6HVELI          , 6HGAMMAI        , 6HAZVELI        ,
 DEPPH     =   100.          ,   100.          ,   100.          ,
               250.          ,   250.          ,
               400.          ,   400.          ,   400.          ,
 DEPVAL    = 2.56786659E+4   , 2.14119741E+7   ,-6.04829803E-4   ,
               2.64999998E+1   , 1.93229930E+4   ,
               1.00874510E+4   , 0.0             , 90.0            ,
 DEPTL    =.02               ,20.              ,1.E-06           ,
              2.6E-5          , .02             ,
              .01             ,1.E-5            ,1.0E-4,
C
 NINDV     = 8               ,
 INDVR     = 6HPITPC2        , 6HCRITR         , 6HPITPC2        ,
             6HBETPC1        , 6HDVIMAG        ,
             6HBETPC1        , 6HALPPC1        , 6HDVIMAG        ,
 INDPH     =   10.           ,   100.          ,   20.           ,
               200.          ,   200.          ,
               300.          ,   300.          ,   300.          ,
 U         =-6.48967861E-1   , 6.37783102E+1   ,-4.30642829E-1   ,
               10.           , 7800.           ,
              -40.           , 1.0             ,  4800.          ,
 PERT = -.00001,-.001,-.00001,  -.00001, -.001,   -.00001,-.001,-.001,
C      IF MODEW = 0, USE INPUT WU WEIGHTING
 WU        = 20.,      .5,       20.,
 1.,  .01,
 .2,  10.,  .01,
 MODEW     = 0               ,
 $
P$GENDAT EVENT     =1.0,                          /LIFT-OFF STG 0 /
 TITLE    = 0H*SIMPLIFIED T-IIIC FLT/PLN-VII -- SUBPROBLEM I -- ASCENT *,
 MAXTIM    = 30000.,                 /200 -300 APOGEE RAISING AND TRANSFER
 ALTMIN    =-1.E6  ,                       /300 - 400 CIRCULARIZATION BURN
 NPC(2)    = 1,                            /RUNGE-KUTTA INTEGRATION/
 DT        =10,                            /INTEGRATION STEPSIZE/
 PINC      =1.E10,                         /PRINT ONLY ATP$GENDAT EVENTS/
 FESN      = 450.,
 NPC(3)    = 4,                            /VELOCITY INITIALIZATION = ZERO/
 NPC(4)    = 2,                            /POSITION INITIALIZATION/
 GDLAT     = 28.56,                        /GEODETIC LATITUDE/
 LONG      = 279.4,                        /RELATIVE LONGITUDE/
 LATL      = 28.56,                        /      ORIENTATION      /
 LONL      = 279.4,                        /           OF          /
 AZL       = 93.,                          /    LAUNCH   FRAME     /
 NPC(5)    = 2   ,                         /SELECTS ATMOSPHERE MODEL/
 NPC(8)    = 1   ,                       /SELECTS CA,CN,CY AERO COEF. INPUT/
 SREF      = 78.5,                         /AERODYNAMIC REF.AREA /
 NPC(9)    = 1   ,                         /SELECTS ONE ROCKET ENGINE/
 WGTSG     = 1400000,                      /INITIAL WEIGHT OF VEHICLE/
 WPLD      = 2700,                         /PAYLOAD WEIGHT /
 WPROPI    = 891210.,                      /INITAIL PROPELLANT WEIGHT /
C
```

Figure 38.- Continued

```
 NPC(16)    = 1 ,                          /SELECTS SPHERICAL EARTH/
 IGUID(1)   = 1 ,                        /SELECTS INERTIAL EULER ANGLE STEERIN
 IGUID(4)   = 1 ,                          /CONSTANT VALUES ARE INPUT /
 $
PSTBLMLT
 $
C                                                  / TABLE DATA DECRIBING STG 0 /
PSTAB TABLE      =6HTVC1T ,0, 1899300.,              / THRUST TABLE /
 $
PSTAB TABLE      =6HWD1T  ,0,7305.,                  / FLOWRATE TABLE/
 $
PSTAB TABLE      =6HAE1T  ,0,124.,                   / EXIT AREA TABLE /
 $
PSTAB TABLE      =6HCAT   ,0,2.,                     / AXIAL AERO COEFF./
 ENDPHS     = 1,
 $
PSGENDAT EVENT      = 10.,                         / END VERTICAL RISE AND STARTS/
 CRITR      = 6HTDURP ,                            / PITCH OVER MANEUVER          /
 VALUE      = 10,                                  /END VERTICAL FLIGHT AT 10 SEC /
 IGUID(4)   = 0 ,                                  / CARRY ATTITUDE OVER FROM     /
 PITPC(2)   = -1.4,                                / PREVIOUS PHASE /
 ENDPHS     =1 ,
 $
PSGENDAT EVENT      = 20.,
 VALUE      = 10.,
 CRITR      = 6HTDURP ,                            /DURATION OF PITCH RATE /
 VALUE      = 20,                                  /EQUALS 20 SEC /
 PITPC(2)   = -.3,
 ENDPHS     = 1,
 $
PSGENDAT EVENT      = 30,                                  / STGO BURNOUT  /
 CRITR      = 6HTIME  ,                            / STG I IGNITION /
 VALUE      = 122.,
 NPC(5)     = 0,                                   / TURNS OFF ATMOSPHERE /
 NPC(8)     = 0,              / TURNS OFF THE AERODYNAMIC CALCULATIONS
 PITPC(2)=-2.9298587316579E-01,       / 30.0 /
 DT         = 20.,
 WJETT      = 145400.,
 WPROPI     = 238380.,
 WEICON     = 0.0,
 $
PSTBLMLT
 $
PSTAB TABLE      = 6HTVC1T ,0,518500.,          / STG I THRUST
 $
PSTAB TABLE      = 6HWD1T  ,0,1740.,
 $
PSTAB TABLE      = 6HAE1T  ,0,0.0,
 ENDPHS     = 1,
 $
PSGENDAT EVENT      = 40.,
```

Figure 38.- Continued

```
 CRITR      = 6HTDURP ,
 VALUE      = 137.,
 PITPC(2)=-1.2280994008965E-02,        / 40.0 /
 WJETT      = 16500.,                   / STG I JETTISON WEIGHT /
 WPROPI     = 71595.,
 WEICON     = 0.0,
 $
PSTBLMLT
 $
PSTAB TABLE      =6HTVC1T ,0, 103230.,        / APPROXIMATE STGII THRUST /
 $
PSTAB TABLE      =6HWD1T  ,0, 333. ,
 ENDPHS     = 1,
 $
PSGENDAT EVENT      = 50.,
 CRITR      = 6HTDURP ,
 VALUE      = 215.,
 WJETT      = 8140.,
 WPROPI     = 24975.,
 WEICON     = 0.0,
 NPC(1)     = 3,                        / TURNS ON CONIC PRINT /
 IGUID      = 3,0,1,                    / STEER ON INERTIAL AERO ANGLES /
 $
PSTBLMLT
 $
PSTAB TABLE      =6HTVC1T ,0,16000.,          / STG III = TRANSTAGE /
 $
PSTAB TABLE      =6HWD1T  ,0,53.   ,
 ENDPHS     =1,
 $
C
C ------------------------------ END OF SUBPROBLEM I ----------------
C ------------------------------ BEGIN SUBPROBLEM II  ----------------
C
PSGENDAT EVENT      = 100,
 TITLE    = 0H*SIMPLIFIED T-IIIC FLT/PLN-VII -- SUBPROBLEM II- TRANSFER*,
 CRITR      =6HTDURP ,                  / 1ST TRANSTG SHUTDOWN/
 VALUE      = 15.,
 NPC(9)     = 0.,                       / TURNS OFF ENGINE /
 NPC(2)     = 3.,
 DT         = 600.,
 ENDPHS     = 1,
 $
PSGENDAT EVENT      =200,
 CRITR      = 6HGCLAT ,
 VALUE     =  0.0,
  NPC(2)    = 3,
 DT         = 5000.,
 PINC       = 1.E10,
 NPC(9)     = 3,
 ISPV       = 310.,
```

Figure 38.- Continued

```
 NPC(16)  = 1,
 NPC(25)  =        3,
 IGUID    = 3,0,1,
 IGUID(13)= 3,
 ALPPC(1)=0.0,                          /200.0 /
 ENDPHS    =1,
 $
C ------------------------------ END OF SUBPROBLEM II ----------------
C ------------------------------ BEGIN SUBPROBLEM III ----------------
C
P$GENDAT EVENT     =300.,               / START OF CIRCULARIZATION BURN/
 TITLE=OH*SIMPLIFIED T-IIIC FLT/PLN-VII -- SUBPROBLEM III-CIRCULARIZE *,
 CRITR      = 6HGCLAT ,
 VALUE      = 0.0,
 NPC(9)     = 3,
 ALPPC(1)=0.0,                          /300.0 /
 ENDPHS     =1,
 $
P$GENDAT EVENT      = 400.,
 CRITR      = 6HTDURP ,
 VALUE     =  0.0,
 WJETT      = 3800.,
 ENDPHS     = 1,
 $
C
C ------------------------------ END OF SUBPROBLEM III ----------------
C
P$GENDAT EVENT      = 450.,
 CRITR      = 6HTDURP ,
 VALUE     =  0.0,
 ENDPHS     = 1,
 ENDPRB     = 1,                        /     THE END OF PROBLEM  /
 ENDJOB     = 1,
 $
```

Figure 38.-Concluded

TABLE 16.- NONDECOMPOSITION ITERATION SUMMARY

	Key Optimization Parameters			Key Trajectory Variable	
Iteration	Optimization Variable, Payload, lb	Constraint Error P_2	Optimization Indicator, CTHA, deg	Park Orbit h_p x h_a n.mi. x n.mi.	Transfer Inclination, deg
0	2500.00	2.79E-5	--	--	--
1	2507.20	1.03E-5	--	--	--
2	2507.30	1.20E-3	--	--	--
3	2507.33	1.33E-3	2.12	--	--
4	2757.82	3.28E-3	--	--	--
5	2757.55	6.23E-1	--	-99 x 80.02	26.928
6	2751.48	4.63E-2	8.18	--	--
7	3030.51	1.09E-6	--	--	--
8	3029.80	1.89E-5	--	--	--
9	3029.43	1.49E-5	--	--	--
10	3028.56	1.19E-5	--	80.02 x 110	26.14

TABLE 17.- DECOMPOSITION ITERATION SUMMARY

	Key Optimization Parameters			Key Trajectory Variable		
Master Level Iteration	Optimization Variable, Payload, lb	Constraint Error P_2	Optimization Indicator, CTHA, deg	Park Orbit h_p x h_a n.mi. x n.mi.	Transfer Incl, deg	Subproblem Iterations
0	2500.00	1.15E-8	--	80.0 x 101.5	28.6	2, 3, 3
1	2674.80	2.89E-2	54.44	--	29.37	4, 6, 5
2	2773.91	4.33E-3	79.72	80.0 x 90.7	25.79	10, 17, 14
3	2781.03	1.08E-2	87.60	--	26.58	6, 10, 8
4	2782.67	1.04E-3	86.19	65.6 x 80.0	26.28	10, 9, 9
5	2783.70	3.85E-5	88.01	--	26.58	7, 9, 7
6	2784.71	2.09E-4	--	44.5 x 80.0	26.494	13, 10, 11
7	2784.40	6.20E-5	89.15	--	26.59	6, 9, 7
8	2784.51	1.20E-7	89.20	48.1 x 80.0	26.55	9, 7, 7
9	2784.62	5.78E-8	89.33	--	26.68	7, 9, 7
10	2784.66	3.15E-8	--	44.5 x 80.0	26.53	7, 9, 7

```
 P$SEARCH
  $ NO TARGETING/OPTIMIZATION, SIMULATION ONLY
 P$GENDAT EVENT      = 1.,
  TITLE(1)   = 50H6D POST REENTRY CHECK CASE
  FESN       = 10.,
  MAXTIM     = 2000.,
  DT         = .0625,
  DTG        = .0625,
  TIME       = 1500.,
  TIMRF(1)   = 1500.,
  DTIMR(1)   = 1.,
  DTIMR(2)   = 1.,
  NPC(3)     = 4,
  NPC(4)     = 2,
  ALTITO     = 152488.89,
  GCLAT      = 39.500882,
  LONG       = -121.21400,
  VELR       = 8734.5313,
  GAMMAR     = -.52896275,
  AZVELR     = 153.94194,
  AZL        = 153.94194,
 C      ATMOSPHERE
  NPC(5)     = 2,
  NPC(8) = 2,
  SREF       = 2690.,
  DREFR      = 78.05666667,
  DREFP      = 39.5666667,
  DREFY      = 78.05666667,
  LREF       = 110.7,
  WGTSG      = 182986.,
 C      PROPULSION
  NPC(9)     = 0,
 C      RANGE CALCULATION
  NPC(12)    = 1,
  LATREF     = 34.55577617,
  LONREF     = -120.5338,
  AZREF      = 154.,
 C      GRAVITY
  NPC(16)    = 1,
  MU         = 1.407662686E16,
  OMEGA      = 7.29211515E-5,
  RE         = 20902910.,
 C      AUTOPILOT
  NAUTOP     = 2,
 C      GUIDANCE
 C      INITIALIZE QUATERNIONS
  IGUID(12) = 1,
  IGUID(12) = 3,
  YAWR = 178.88212,   PITR = 17.370592,          ROLR = 56.46,
  ROLBD = 0.,         PITBD = 0.,                YAWBD = 0.,
```

Additional 6D POST Input

Figure 39.- Required Additions and Modifications to Convert 3D POST Input to 6D POST Input

```
C     USE CLOSED LOOP REENTRY GUIDANCE
 IGUID(14) = 2,
 IGUID(17) = 2,
C     TLATCD = TARGET GEOCENTRIC LATITUDE
 TLATCD    = 34.55577617,
C     TLONGD  = TARGET LONGITUDE
 TLONGD    = -120.5338,
 ITRANS = 1,
C     CONTROLS
 KRDA      = 1.,     KRDE      = 0.,     KRDR      = 0.,
 KPDA      = 0.,     KPDE      = 1.,     KPDR      = 0.,
 KYDA      = 0.,     KYDE      = 0.,     KYDR      = 1.,
 DAMIN     = -15.,   DAMAX     = 15.,
 DEMIN     = -40.,   DEMAX     = 15.,
 DRMIN     = -22.8,  DRMAX     = 22.8,
 DELECI    = -5.,    DELE      = -5.,
 EREFL     = 19047.,
C     PRINTOUT
 PINC = 1.0,
 PRNT(115) =
            6HTIMRF1,6HTIMRF2,6HGTIME ,6HATIME ,6HDELT  ,6H*     ,
            6HDELAZ ,6HLOD1  ,6HRDTREF,6HRK2ROL,6HTRANGE,6HDRAGGC,
            6HSLECT ,6HGASM  ,6HGLOD  ,6HGRDOT ,6HDRAGRP,6HALDREF,
            6HALPGC ,6HALPERR,6HROLRGC,6HROLRER,6HALPERI,6HDELECI,
            6HTROLN ,6HTROLP ,6HTPITN ,6HTPITP ,6HTYAWN ,6HTYAWP ,
            6HTONT  ,6HWDOTJ ,6HWPCONJ,6HRCSMXB,6HRCSMYB,6HRCSMZB,
            6HDELAC ,6HDELEC ,6HDELRC ,6HDELAD ,6HDELED ,6HDELRD ,
            6HDELA  ,6HDELE  ,6HDELR  ,6HDELF1 ,6HDELF2 ,6HUDCTR ,
            6HCKA   ,6HCKE   ,6HCKR   ,6HHMA   ,6HHME   ,6HHMR   ,
            6HCLA   ,6HCDA   ,6HCMA   ,6HCYB   ,6HCWB   ,6HCLLB  ,
            6HCYDR  ,6HCLLDR ,6HCWDR  ,6HCLDE  ,6HCDDE  ,6HCMDE  ,
            6HCYDA  ,6HCLLDA ,6HCWDA  ,6HCLF1  ,6HCDF1  ,6HCMF1  ,
            6HCYF1  ,6HCWF1  ,6HCLLF1 ,6HCHE   ,6HCMQ   ,6HCWR   ,
            6HCLLR  ,6HCWP   ,6HCLLP  ,6HCLF2  ,6HCDF2  ,6HCMF2  ,
 $
P$TBLMLT
 CLLBNM    = 6HBETA  ,     CLLDRN    = 6HDELR  ,
 CLLDAN    = 6HDELA  ,
 CMDENM    = 6HDELE  ,
 CWBNM     = 6HBETA  ,     CWDRNM    = 6HDELR  ,
 CWDANM    = 6HDELA  ,
 CYBNM     = 6HBETA  ,     CYDRNM    = 6HDELR  ,
 CYDANM    = 6HDELA  ,
 CDDENM    = 6HDELE  ,     CLDENM    = 6HDELE  ,
 CLF1N     = 6HONE   ,
 CDF1N     = 6HONE   ,
 CMF1N     = 6HONE   ,
 CYF1N     = 6HBETA  ,
 CWF1N     = 6HBETA  ,
 CLLF1N    = 6HBETA  ,
 CLF2N     = 6HONE   ,
```

6D POST Input

6D POST Input

Figure 39.- Continued

```
 CDF2N      = 6HONE    ,
 CMF2N      = 6HONE    ,
 $
P$TAB TABLE = 6HGENV1T,3,4HMACH,4HDYNP,4HTIME,2,2,2, 1,24*1,
         1700.,     0.,5.,1.,6.,10.,
                    5.,5.,2.,6.,20.,
         1800.,     0.,5.,3.,6.,30.,
                    5.,5.,4.,6.,40.,
 $
P$TAB TABLE      = 6HCLT    ,0,.672662268,
 $
P$TAB TABLE      = 6HCDT    ,0,.456788181,
 $
P$TAB TABLE      = 6HCMAT   ,0,-.0304227747,
 $
P$TAB TABLE      = 6HCYBT   ,0,-.0054,
 $
P$TAB TABLE      = 6HCWBT   ,0,-.00184,
 $
P$TAB TABLE      = 6HCLLBT  ,0,-.00162,
 $
P$TAB TABLE      = 6HCYDRT  ,0,.0004,
 $
P$TAB TABLE      = 6HCLLDRT,0,.00015,
 $
P$TAB TABLE      = 6HCWDRT  ,0,-.00028,
 $
P$TAB TABLE      = 6HCLDET  ,0,2.366278E-3,
 $
P$TAB TABLE      = 6HCDDET  ,0,1.58043E-3,
 $
P$TAB TABLE      = 6HCMDET  ,0,-2.84346E-3,
 $
P$TAB TABLE      = 6HCYDAT  ,0,1.323394E-4,
 $
P$TAB TABLE      = 6HCLLDAT,0,1.07534E-3,
 $
P$TAB TABLE      = 6HCWDAT  ,0,-3.5084E-4,
 $
P$TAB TABLE      = 6HCLF1T  ,0,0.,
 $
P$TAB TABLE      = 6HCDF1T  ,0,0.,
 $
P$TAB TABLE      = 6HCMF1T  ,0,0.,
 $
P$TAB TABLE      = 6HCYF1T  ,0,0.,
 $
P$TAB TABLE      = 6HCWF1T  ,0,0.,
 $
P$TAB TABLE      = 6HCLLF1T,0,0.,
 $
```

6D POST Input

Figure 39.- Continued

```
PSTAB TABLE      = 6HCHET  ,0,.046,
 $
PSTAB TABLE      = 6HCMQT  ,0,-2.8,
 $
PSTAB TABLE      = 6HCWRT  ,0,-.05,
 $
PSTAB TABLE      = 6HCLLRT ,0,.05,
 $
PSTAB TABLE      = 6HCWPT  ,0,-.022,
 $
PSTAB TABLE      = 6HCLLPT ,0,-.251,
 $
PSTAB TABLE      = 6HCLF2T ,0,-1.0195E-2,
 $
PSTAB TABLE      = 6HCDF2T ,0,-6.312E-3,
 $
PSTAB TABLE      = 6HCMF2T ,0,1.6234E-2,
 $
C      DELF1 = SPEED BRAKE DEFLECTION
PSTAB TABLE      = 6HDELF1T,1,6HMACH  ,5,1,1,1,
      0.,55., 4.,55., 5.,85., 8.,55., 10.,55.,
 $
C      DELF2 = BODY FLAP DEFLECTION
C      BODY FLAP DEFLECTION FOR FWD CG CONDITION
PSTAB TABLE      = 6HDELF2T,0,-11.7,
 $
PSTAB TABLE      = 6HRMPJT ,0,9952.,
 $
PSTAB TABLE      = 6HPMPJT ,0,33622.,
 $
PSTAB TABLE      = 6HYMPJT ,0,34072.,
 $
PSTAB TABLE      = 6HPNPJIT,1,6HDYNP  ,6,1,1,1,
      0.,.78, 5.,.62, 10.,.53, 15.,.50, 20.,.48, 200.,.48,
 $
PSTAB TABLE      = 6HPYJIT ,1,6HDYNP  ,6,1,1,1,
      0.,0.,  5.,.1,  10.,.16, 15.,.18, 20.,.19, 200.,.19,
 $
PSTAB TABLE      = 6HPRJIT ,1,6HDYNP  ,6,1,1,1,
      0.,.14, 5.,.23, 10.,.26, 15.,.27, 20.,.28, 200.,.28,
 $
PSTAB TABLE      = 6HRRJIT ,1,6HDYNP  ,6,1,1,1,
      0.,.77, 5.,.59, 10.,.50, 15.,.45, 20.,.42, 200.,.42,
 $
PSTAB TABLE      = 6HRYJIT ,1,6HDYNP  ,6,1,1,1,
      0.,-.70, 5.,-.32, 10.,-.32, 15.,-.30, 20.,-.28,200.,-.28,
 $
PSTAB TABLE      = 6HYRJIT ,1,6HDYNP  ,6,1,1,1,
      0.,-.11, 5.,-.12, 10.,-.12, 15.,-.13, 20.,-.13,200.,-.13,
 $
PSTAB TABLE      = 6HYYJIT ,0,1.0,
```

6D POST Input

Figure 39.- Continued

```
 $
P$TAB TABLE      = 6HPPPJIT,0,1.0,
 $
P$TAB TABLE      = 6HCKET  ,1,6HMACH  ,4,1,1,1,
               0.,6., 10.,6., 12.,2., 100.,2.,
 $
P$TAB TABLE      = 6HCKRT  ,1,6HMACH  ,4,1,1,1,
               0.,2., 2.,2., 2.01,4., 30.,4.,
 $
P$TAB TABLE      = 6HCKAT  ,1,6HDYNP  ,4,1,1,1,
               0.,10., 10.,10., 30.,2.5, 200.,2.5,
 $
P$TAB TABLE      = 6HYCGT  ,0,0.125,
 $
P$TAB TABLE      = 6HIXXT  ,0,759000.,
 $
P$TAB TABLE      = 6HIYYT  ,0,5765000.,
 $
P$TAB TABLE      = 6HIZZT  ,0,5912000.,
 $
P$TAB TABLE      = 6HIXZT  ,0,131000.,
 ENDPHS     = 1,
 $
P$GENDAT EVENT      = 10.,
 CRITR      = 6HTIMRF1,
 VALUE = 1510.,
 MDL        = 1,
 ENDPHS     = 1,
 ENDPRB     = 1,
 ENDJOB     = 1,
 $
```

6D POST Input

Figure 39.- Concluded

COMPUTER REQUIREMENTS

Machine Configurations

POST was originally developed for the CDC 6000 series computers. 3D POST has since been converted to the IBM 370/165/OS and the UNIVAC 1108/EXEC 8 computers. Both 3D POST and 6D POST are coded exclusively in FORTRAN IV and are compatible with the FORTRAN IV EXTENDED compiler. The 6D POST is not available on any computer/system other than the CDC 6600/NOS. The minimal computer requirements for 3D and 6D POST are given in Table V-1.

TABLE 18.- MINIMAL POST COMPUTER REQUIREMENTS

Computer	Operating System	Precision	3D Storage, Words	6D Storage, Words
CDC 6400 6500 6600	NOS, SCOPE 3.4	Single	$140\ 000_8$*	$140\ 000_8$
IBM 370/165	OS	Double	50 000	70 000
UNIVAC	EXEC 8	Double	55 000	65 000
*Nonoverlayed version.				

As with any larger computer program, special operating system-dependent techniques can be used to reduce core requirements or increase execution speed. For example, the overlay structure can be tailored to any given computer/system to reduce core size and execution time. The specific compiler used can also significantly impact execution speed. For example, the CDC 6600 POST executes about two times faster using the FTN compiler than it does using the RUN compiler. These system-dependent program tradeoffs and modifications can easily be made once the standard version is operating correctly on any particular computer system.

Computer Precision

There are two important numerical techniques used in POST that require considerable computational precision to function properly. These techniques are (1) numerical differentiation, and (2) numerical integration. The most difficult computational problem encountered in POST combines both numerical techniques, that is, the approximation of trajectory partial derivatives by numerical differentiation of numerically integrated trajectories. This numerical process dictates the computer precision requirements for POST. Computational experience indicates that a 64-bit word is usually adequate for the numerical differentiation of various trajectory variables. This means that double

precision is required to run POST on any computer that has a 32-bit single precision word. As a result, the program must be executed in double precision on IBM and UNIVAC computers. Single precision is adequate on all CDC 6000 series computers because CDC has a 60-bit single precision word.

To numerically differentiate the dependent variables (target conditions, inequality constraints, and the optimization variable) with respect to the independent variables, the user must input the perturbation to be used for each independent variable. The user must also select the integration method and the initial integration step size. Generally, the numerical truncation error associated with the numerical integration process tends to cancel itself in the difference quotient; however, propagation of local round-off error does not cancel. Fortunately in most trajectory problems, the number of integration steps is sufficiently small so round-off error is not a major problem. As a result, the selection of the integration method and step size, although important in terms of run time and accuracy, is not critical to the numerical differencing techniques employed. Critical, however, is the proper selection of the control variable perturbations that are used in the divided difference formula. If the perturbations are excessively large, accuracy is lost because of truncation error in the first and second difference formulas. If the perturbations are too small, accuracy is lost due to subtraction of the nominal value from the perturbed value on a finite word length computer. Problems associated with the word length can, and have for the most part, been eliminated by using either CDC computers or by using double precision arithmetic on computers with smaller single precision word length. However, control of the truncation error in the difference formula is difficult and numerical experience on a given problem/computer/operating system is sometimes required. To alleviate this problem an automatic perturbation step size controller is incorporated in the latest versions of the programs. This routine requires an initial guess for the perturbations in each independent variable. The initial guess is used unless the resulting divided difference quotient is judged to be sufficiently inaccurate as to require modification. Modification is accomplished by first rerunning the perturbation with its sign changed to obtain a symmetric difference for those variables with incorrect perturbations. The symmetric differences are then used on the present iteration. On the next iteration, the perturbations are adjusted to ensure the proper amount of variation in the dependent variables to secure adequate accuracy in the sensitivities.

Runtime

The runtime, as measured by the Central Processor Unit (CPU) clock, is a key input to all computer charging algorithms. As a result, it is important to understand what factors contribute to the CPU time requirements when running POST. Understanding will enable users to make reasonable estimates of computer budgets associated with using POST programs.

The CPU time for a POST run depends on numerous factors. The most important are as follows:

Computer hardware and system software characteristics;

Type of trajectory and vehicle being simulated;

Accuracy requirements;

Number of degrees of freedom in the targeting/optimization formulation.

The computer and the operating system used clearly play a primary role in determining the runtime of any program. In addition, the CPU time is not the only variable used in determining the cost-effectiveness of a given program/computer system interface. For example, the CPU time required to run a POST input deck on the CDC 6600/NOS/FTN may be only a fraction of that required to run the same deck on the CDC 6400/SCOPE 3.4/RUN; however, it may be less expensive to use the older and slower computer because of the charging algorithms involved and the demand on each system. The point being made is that the only practical way to estimate computer costs is through computational experimentation with POST on your computer system/problem. Generally, minor changes can be made to the program executive structure (overlays) to eliminate any severe incompatibilities between POST and the host computer system.

A second key factor impacting runtime is the type of trajectory and vehicle being simulated in any given problem. For example, exoatmospheric trajectories can be integrated more rapidly than atmospheric ascent of entry trajectories. Also problems that require large amounts of tabular data to describe the mass properties, the aerodynamics, and the propulsion system will require longer corresponding runtimes. The total flight time of the trajectory being simulated is a prime driver in the computational costs. Experience indicates that long duration flights generally require more computer time than shorter flights because of the direct increase in the number of required integration steps. The amount and frequency of printout data requested are also an important consideration that can always be controlled by the user. In extreme cases, runtime can double or even triple when the full 198 variable printblock is requested at every integration step. Thus, to reduce runtime, only the minimal amount of data required to accurately model the vehicle should be input and only the minimal amount of data required to interpret the results should be output.

Accuracy, which is directly related to the selection of the integration method and the integration stepsize, is an important consideration in every problem. Generally speaking, the standard fourth order Runge-Kutta algorithm with the appropriate stepsize is the most cost effective numerical integration method; however, under special circumstances higher order methods may be faster. For example, Keplerian orbits can be integrated ten times faster with the Krogh method than with Runge-Kutta. The best approach to determining the impact stepsize on accuracy and CPU time is to run several single pass trajectories. A compromise stepsize can then be determined from plots of integration error and CPU time versus stepsize.

The final factor influencing runtime is the number of degrees of freedom in the targeting and optimization formulation. The number is computed as the difference between the number of independent variables, m, and the average number of active constraints, $\bar{n}_a$. The number of degrees of freedom can be used to establish an estimate of the number of iterations, N, required by the accelerated projected gradient algorithm to achieve the optimum trajectory. This estimate is given by the formula

$$N = m - \bar{n}_a + K \tag{1}$$

where K is the number of iterations required to achieve a feasible, targeted trajectory. For example, if there are ten independent variables, three target conditions, and two inequalities, one of which is active, then

$$N = (10 - (3 + 1)) + K = 6 + K \tag{2}$$

is generally a reasonable estimate of the total number of iterations required to achieve an optimum trajectory. The number of iterations required to achieve a feasible trajectory, K, varies as a function of the initial guess. For a reasonably accurate initial guess, K can be estimated as the integer part of $\bar{n}_a/2$. In the case where there are no inequality constraints, then $\bar{n}_a$ is equal to the number of target conditions defined by the user.

The total CPU time can then be approximated as

$$CPU = N(T/I) \tag{3}$$

where T/I is the average time required to make a single iteration. T/I can be estimated in terms of the single pass CPU time, τ, as

$$T/I = (1 + m + 6)\ \tau$$

where 1 + m trajectories are required to obtain the sensitivity matrix, and six trajectories are required (on the average) to perform the univariant searches (minimum is two and the maximum is 10). This estimate for CPU per iteration is conservative due to the fact that POST does not integrate the complete trajectory on each of the m perturbed runs. As a result, it is generally more accurate to determine T/I directly from actual computations. For this reason, the CPU time used per iteration is computed and included in the iteration summary printout as CP/ITR. The calculation of CP/ITR requires a machine-dependent subroutine that usually must be modified for each computer/system.

REFERENCES

1. G. L. Brauer, D. E. Cornick, R. Stevenson: *Program to Optimize Simulated Trajectories (POST).* NASA CR-132691. National Aeronautics and Space Administration, Final Report, Vol II, April 1975.

2. G. L. Brauer, D. E. Cornick, R. Stevenson: *Program to Optimize Simulated Trajectories (POST).* NASA CR-132691. National Aeronautics and Space Administration, Final Report, Vol III, April 1975.

3. E. A. Mechtly: *The International System of Units, Physical Constants and Conversion Factors.* NASA SP-7012. National Aeronautics and Space Administration, 1973.

4. G. L. Brauer, D. E. Cornick, R. Stevenson: *Program to Optimize Simulated Trajectories (POST).* NASA CR-132691. National Aeronautics and Space Administration, Final Report, Vol I, April 1975.

5. G. L. Brauer, D. E. Cornick, R. Stevenson: *Six-Degree-of-Freedom Program to Optimize Simulated Trajectories (6D POST).* NAS1-13300-MCR-74-423. National Aeronautics and Space Administration, Final Report, Vol I, November 1974.

6. G. L. Brauer, D. E. Cornick, R. Stevenson: *Six-Degree-of-Freedom Program to Optimize Simulated Trajectories (6D POST).* NAS1-13300-MCR-74-423. National Aeronautics and Space Administration, Final Report, Vol I, November 1974.

7. G. L. Brauer, D. E. Cornick, R. Stevenson: *Six-Degree-of-Freedom Program to Optimize Simulated Trajectories (6D POST).* NAS1-13300-MCR-74-423. National Aeronautics and Space Administration, Final Report, Vol II, November 1974.

*U.S. GOVERNMENT PRINTING OFFICE: 1977 - 735-004/29