Final Report
Contract NAS 8-28425

Targetting and Guidance Program
Documentation

by

E. F. Harrold and J. F. Neyhard

Submitted July 15, 1974 to
National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center
Huntsville, Alabama 35812

by

IBM Federal Systems Division
18100 Frederick Pike
Gaithersburg, Maryland 20760

Work performed at
Advanced Systems Design Department
3 New England Executive Park
Burlington, Mass. 01803

Unclas
<table>
<thead>
<tr>
<th>TABLE OF CONTENTS</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Program Overview</td>
<td>3</td>
</tr>
<tr>
<td>User's Guide</td>
<td>4</td>
</tr>
<tr>
<td>Input</td>
<td>4</td>
</tr>
<tr>
<td>Output</td>
<td>10</td>
</tr>
<tr>
<td>Interdependence of Subroutines</td>
<td>11</td>
</tr>
<tr>
<td>Subroutine MAIN</td>
<td></td>
</tr>
<tr>
<td>Subroutine AUXOUT</td>
<td></td>
</tr>
<tr>
<td>Subroutine BCBCB</td>
<td></td>
</tr>
<tr>
<td>Subroutine BVAL5</td>
<td></td>
</tr>
<tr>
<td>Subroutine CBCB</td>
<td></td>
</tr>
<tr>
<td>Subroutine CKSET</td>
<td></td>
</tr>
<tr>
<td>Subroutine ELMNTS</td>
<td></td>
</tr>
<tr>
<td>Subroutine FORWARD</td>
<td></td>
</tr>
<tr>
<td>Subroutine GLMNTS</td>
<td></td>
</tr>
<tr>
<td>Subroutine NAVOUT</td>
<td></td>
</tr>
<tr>
<td>Subroutine PHASE</td>
<td></td>
</tr>
<tr>
<td>Subroutine ROTATE</td>
<td></td>
</tr>
<tr>
<td>Subroutine STATIS</td>
<td></td>
</tr>
<tr>
<td>Subroutine UCOAST</td>
<td></td>
</tr>
<tr>
<td>Subroutine USTAT</td>
<td></td>
</tr>
</tbody>
</table>
Introduction

A FORTRAN computer program has been developed which automatically targets two and three burn rendezvous missions and performs feedback guidance using the previously developed GUIDE algorithm. The program was designed to accept a large class of orbit specifications and automatically chooses a two or three burn mission depending upon the time alignment of the vehicle and target. The orbits may be specified as any combination of circular and elliptical orbits and may be coplanar or inclined, but must be aligned coaxially (i.e. line of intersection of orbital planes and orbital major axes coincident) with their perigees in the same direction. The program accomplishes the required targeting by repeatedly converging successively more complex missions. It solves the coplanar impulsive version of the mission, then the finite burn coplanar mission and finally the full plane change mission. The GUIDE algorithm is exercised in a feedback guidance mode by taking the targeted solution and moving the vehicle state step by step ahead in time adding acceleration and navigational errors and reconverging from the perturbed states at fixed guidance update intervals.

The targeting and guidance algorithm converges all two burn missions easily and exhibits good guidance behavior for these missions. Three burn missions were much more sensitive and required special loops to insure convergence. The outbound three burn mission had to be converged backwards in time and plane change was most readily incorporated by eliminating the third burn and solving the appropriate two burn mission, reintroducing the third burn at the end. In a targeting mode these techniques cause no particular problem and insure convergence. In guidance mode the convergence problems are more difficult to compensate for and may limit real time use. The program as it now stands attempts to optimize over all three burns and although it has maintained convergence for all missions attempted, the guidance corrections have been larger than desired. In the future it may be necessary to solve the guidance problem over the first burn as a rendezvous with the desired phasing or transfer orbit and to only introduce the third burn after completion of the first one.

Another study that needs to be undertaken is to optimize the soft constraint weights using the Monte Carlo capability built into the program. By altering the weights and noting the tradeoffs made between burn time and orbital injection error, a better estimate of optimal soft constraint weights can be obtained.
The remainder of this document describes the targeting and guidance program in detail, giving an overview of the program control and organization, a summary of program inputs and outputs and a detailed description of each of the subparts of the program. Also included in the document is a description of the GUIDE subroutine BVAL5, which was altered to incorporate the soft constraint formulation, and is fully documented. The other GUIDE subroutines are essentially the same as the ones described in the GUIDE 71/6 document\(^1\) and are not described here.

Program Overview

The program is controlled by routine MAIN, which oversees the impulsive targeting, the convergence of the orbital transfer, and the feedback guidance. The impulsive targeting is accomplished by first determining the elements of both orbits, then defining the transfer orbit and phasing orbit (3 burn only) and determining the velocities at apogee and perigee of each orbit. Next the delta v's are calculated and the burn and coast times calculated. The transfer orbit is chosen to be tangent at both end points to the principal orbits, and the mission is classified as inbound or outbound depending on whether apogee of the final (target) orbit is less than or greater than apogee of the initial orbit. The phasing orbit is chosen to lie as close as possible to the one which results from splitting the burn at perigee into two equal halves. A closed-form solution is used for initial costate.

The converged finite-burn solution is arrived at by repeatedly converging successively more complex missions, starting with a planar mission and gradually adding in the plane change required (10° steps). To maintain convergence for outbound 3-burn missions, it was necessary to rearrange each mission and converge it in a backwards fashion, from the target orbit to the vehicle (initial) orbit. The plane change mentioned above was facilitated by changing the 3-burn mission to a 2-burn mission where the planar-converged phasing orbit was substituted for the closer orbit. After converging the 2-burn mission with the total plane change, the 3-burn mission was reinstated and converged. Finally, the 3-burn outbound mission is turned around to its normal mode and reconverged.

After targeting has been done, the guidance portion of the program is run in a feedback mode, in which it is made to respond to simulated perturbations. The routine MAIN calls BCBCB or CBCB to propagate the vehicle along each arc of the mission, and Monte Carlo statistics are collected at appropriate points and summarized at the end.

Further details of the operation of the program, as well as the routines employed, are described in the pages which follow.
User's Guide

The program is set up using NAMELIST input for ease of operation. This allows default parameter values to be specified and reduces the amount of input necessary for program execution. Typical space tug vehicle parameters are hard coded as default values and tug missions can be performed by simply specifying the desired initial and final orbits. The basic program philosophy is to use the orbital definitions to define whether the mission will be two or three burns. If the mission is circular to circular coplanar, or if the orbital elements are defined with no positions along the orbits given, or if the positions of the vehicle and target allow a two burn rendezvous, a two burn orbital transfer will be defined. Under all other conditions three burn transfers will be used. The integer NOTARG is used to control which portions of the program are executed. If NOTARG=-1 only targeting is performed. If NOTARG=1 a converged solution for the orbital transfer is read in using NAMELIST NAMSL2 and only the feedback guidance part of the program will be executed. If NOTARG is any other value both the targeting and guidance will be performed. The inputs and outputs and individual subroutines will be described in detail in the sections which follow.

Program Inputs

The program inputs are broken into three basic groups: those which define the vehicle's capabilities, those used to specify the initial and final orbits, and those used to define the Monte Carlo and perturbation parameters needed for feedback guidance evaluation.

A. Vehicle Constants

The following parameters are used to specify the vehicle, and must be in metric units. If specific impulse is inputted it is used to calculate mass rate. The default values for the parameters are typical of a space tug configuration.

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM0</td>
<td>m0</td>
<td>Initial vehicle mass in kg</td>
<td>28803.1155 kg (63500 lbs)</td>
</tr>
<tr>
<td>THRUST</td>
<td>T</td>
<td>Thrust in kilo-Newton</td>
<td>66.7233 kn (15000 lbs)</td>
</tr>
<tr>
<td>Name</td>
<td>Symbol</td>
<td>Definition</td>
<td>Default Value</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>-----------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>SPFIMP</td>
<td>I&lt;sub&gt;sp&lt;/sub&gt;</td>
<td>Specific impulse in seconds</td>
<td>440 sec</td>
</tr>
<tr>
<td>AMDOT</td>
<td>m</td>
<td>Mass rate in kg/sec</td>
<td>15.4634 kg/sec</td>
</tr>
</tbody>
</table>

### B. Orbit Specifications

The vehicle and target orbits may be specified in four separate ways listed as sets 1-4 below. (It is assumed that both will be specified in the same fashion.) For all of the orbital definitions the perigee directions must be equal and coincident with the line of intersection of the orbital planes. If sets 2, 3 or 4 are used to specify the orbits, these conditions are satisfied automatically due to the way the orbital positions and coordinate systems are defined. If position and velocity vectors and times (set 1) are specified, the program will test to see that the conditions are satisfied and will stop if the proper perigee and line of nodes alignment is not found. When set 1 is used to specify the data the relative inclination between orbits is measured from vehicle to target orbit at perigee. In all other cases relative inclination is set by the input data. If sets 2, 3 or 4 are used to specify the orbits and the true anomalies (TANOM<sub>O</sub> and TANOM<sub>T</sub>) are greater than or equal to zero, they will be used to specify the orbital positions. If true anomalies are not specified and T<sub>0</sub> and T<sub>T</sub> are greater than or equal to zero they will be assumed to be mean anomalies and used to specify the orbital positions. If neither of the anomalies are specified, the orbital positions will be arbitrarily chosen to allow a two burn rendezvous. If no complete set of vehicle and target orbital data is available, the program will print the existing data and stop.

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RΩ(3), VΩ(3), TΩ</td>
<td>r&lt;sub&gt;0&lt;/sub&gt;, v&lt;sub&gt;0&lt;/sub&gt;, t&lt;sub&gt;0&lt;/sub&gt;</td>
<td>Position (km) and velocity (km/sec) vectors at t&lt;sub&gt;0&lt;/sub&gt; for vehicle orbit</td>
<td>T&lt;sub&gt;0&lt;/sub&gt;=-1</td>
</tr>
<tr>
<td>RΤ(3), VΤ(3), TT</td>
<td>r&lt;sub&gt;T&lt;/sub&gt;, v&lt;sub&gt;T&lt;/sub&gt;, t&lt;sub&gt;T&lt;/sub&gt;</td>
<td>Position (km) and velocity (km/sec) vectors at t&lt;sub&gt;T&lt;/sub&gt; for target orbit</td>
<td>TT=-1</td>
</tr>
<tr>
<td>Name</td>
<td>Symbol</td>
<td>Definition</td>
<td>Default Value</td>
</tr>
<tr>
<td>------------</td>
<td>--------</td>
<td>------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>A₀, E₀</td>
<td>a₀, e₀</td>
<td>Semi-major axis (km) and eccentricity for vehicle orbit</td>
<td>0, -1</td>
</tr>
<tr>
<td>AT, ET</td>
<td>aₜ, eₜ</td>
<td>Semi-major axis (km) and eccentricity for target orbit</td>
<td>0, -1</td>
</tr>
<tr>
<td>RELINC</td>
<td>i</td>
<td>Signed relative inclination (deg) as measured from vehicle to target orbit</td>
<td>0</td>
</tr>
<tr>
<td>TANOM₀,TANOMΤ*</td>
<td>f</td>
<td>True anomalies (not required) (deg)</td>
<td>-1</td>
</tr>
<tr>
<td>T₀, TT *</td>
<td>M</td>
<td>Mean anomalies if true anomalies not specified (not required) (sec)</td>
<td>-1</td>
</tr>
</tbody>
</table>

* described more fully in text above

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>HAP₀, HPG₀</td>
<td>-</td>
<td>Height at apogee and perigee for vehicle orbit (km)</td>
<td>None</td>
</tr>
<tr>
<td>HAPT, HPGT</td>
<td>-</td>
<td>Height at apogee and perigee for target orbit (km)</td>
<td>None</td>
</tr>
<tr>
<td>RELINC</td>
<td>i</td>
<td>Same as set 2</td>
<td>0</td>
</tr>
<tr>
<td>T₀, TT, TANOM₀, TANOMΤ</td>
<td>Same as set 2</td>
<td></td>
<td>-1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R₀MAG, V₀MAG,</td>
<td>R₀₀</td>
<td>Magnitude of position and velocity vectors (km) and flight angle between them for vehicle orbit</td>
<td>ROMAG: -1</td>
</tr>
<tr>
<td>FLT₀</td>
<td>α₀</td>
<td></td>
<td>FLT₀: -1</td>
</tr>
<tr>
<td>RₜMAG, VₜMAG,</td>
<td>Rₜₜ</td>
<td>Same for target orbit</td>
<td>FLTT: -1</td>
</tr>
<tr>
<td>FLTT</td>
<td>αₜ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RELINC</td>
<td></td>
<td>Same as set 2</td>
<td>0</td>
</tr>
<tr>
<td>T₀, TT, TANOM₀, TANOMΤ</td>
<td>Same as set 2</td>
<td></td>
<td>-1</td>
</tr>
</tbody>
</table>
C. Feedback Guidance Parameters

In order to exercise the feedback guidance portion of the program and collect statistics on performance, the magnitude of the navigation update errors at the start of the first coast, at the start of the second coast and in the middle of the last burn need to be specified. The time between guidance updates on coast and burn arcs needs to be specified and the number of separate Monte Carlo runs and time between statistical samples defined. The acceleration noise added at each guidance cycle is set at five percent of the thrust during burns and about 1/2 of the worst case gravity errors during coasts and can be changed if desired.

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELS(1)</td>
<td>$\Delta t_b$</td>
<td>Time between guidance updates during burns (sec)</td>
<td>20 sec</td>
</tr>
<tr>
<td>DELS(2)</td>
<td>$\Delta t_c$</td>
<td>Time between guidance updates during coasts (sec)</td>
<td>100 sec</td>
</tr>
<tr>
<td>NOISON</td>
<td>-</td>
<td>0 - no noise&lt;br&gt;1 - navigation and acceleration perturbations</td>
<td>0</td>
</tr>
<tr>
<td>SIGMAR(1), SIGMAV(1)</td>
<td>$\delta R$, $\delta V$</td>
<td>Standard deviation of position and velocity navigation errors (km/sec²) at end of second from last burn (only used during 3-burn mission)</td>
<td>0</td>
</tr>
<tr>
<td>SIGMAR(2), SIGMAV(2)</td>
<td>$\delta R$, $\delta V$</td>
<td>Same at end of next to last burn (km/sec²)</td>
<td>0</td>
</tr>
<tr>
<td>SIGMAR(3), SIGMAV(3)</td>
<td>$\delta R$, $\delta V$</td>
<td>Same in the middle of last burn (km/sec²)</td>
<td>0</td>
</tr>
<tr>
<td>PERT(1)</td>
<td>$\delta a$</td>
<td>Standard deviation of acceleration errors during burns (added each guidance cycle) (km/sec²)</td>
<td>$\frac{.05T}{m_0}$</td>
</tr>
<tr>
<td>PERT(2)</td>
<td>$\delta a$</td>
<td>Standard deviation of acceleration errors during coasts (km/sec²)</td>
<td>$\frac{.5 \times 10^{-4}}{R_e}$</td>
</tr>
<tr>
<td>Name</td>
<td>Symbol</td>
<td>Definition</td>
<td>Default Value</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>MCARLO</td>
<td></td>
<td>Number of Monte Carlo cases to be run</td>
<td>1</td>
</tr>
<tr>
<td>PTB</td>
<td></td>
<td>Time between output samples during burns (sec)</td>
<td>10 sec</td>
</tr>
<tr>
<td>PTC</td>
<td></td>
<td>Time between output samples during coasts (sec)</td>
<td>100 sec</td>
</tr>
</tbody>
</table>

D. **General Parameters**

Included here are the remainder of the parameters which may be set by NAMELIST NAMLS1 input.

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOTARG</td>
<td></td>
<td>-1 Targetting only 0 Targetting and feedback guidance 1 Guidance only using parameters read in by NAMELIST NAMLS2</td>
<td>0</td>
</tr>
<tr>
<td>NAVOFF</td>
<td></td>
<td>0 Convergence status printed whenever output sample is taken in guidance mode 1 No print</td>
<td>1</td>
</tr>
<tr>
<td>IOUTPT</td>
<td></td>
<td>Integer parameter defining output device</td>
<td>6</td>
</tr>
<tr>
<td>EERROR</td>
<td>$\delta_e$</td>
<td>If eccentricity less than EERROR it is set equal to 0</td>
<td>.01</td>
</tr>
<tr>
<td>TERROR</td>
<td>$\delta_t$</td>
<td>If tug or target within this time (sec) tolerance of node or some mean anomaly, considered at node or mean anomaly</td>
<td>10 sec</td>
</tr>
<tr>
<td>RERROR</td>
<td>$\delta_i$</td>
<td>Differences in angles (relative inclination, etc) less than this tolerance will be ignored</td>
<td>.5 degrees</td>
</tr>
<tr>
<td>OBLATE</td>
<td></td>
<td>Weighting factor used in setting oblateness effects (subroutine PERTO)</td>
<td>0.0</td>
</tr>
<tr>
<td>Name</td>
<td>Symbol</td>
<td>Definition</td>
<td>Default Value</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>AXIS(I)</td>
<td>-</td>
<td>Axis of rotation of the earth, must be set in relation to coordinate system</td>
<td>[0 ]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chosen by targeting when oblateness is activated</td>
<td>[0 ]</td>
</tr>
</tbody>
</table>

E. **NOTARG=1, Guidance Only Parameters (NAMSL2)**

The following parameters will define the orbits of the target and vehicle, initial mass of the vehicle (all other vehicle parameters are set by NAMSL1 or default options), the initial costate vector and times array needed to define the burn and coast arcs.

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NBURNS</td>
<td>-</td>
<td>Number of burn arcs</td>
<td>2</td>
</tr>
<tr>
<td>X₀, T₀</td>
<td>x₀, t₀</td>
<td>State of vehicle at start of mission. t₀ time at start of mission.</td>
<td>T₀=-1</td>
</tr>
<tr>
<td>Xₜ, Tₜ</td>
<td>xₜ, tₜ</td>
<td>State of target at time tₜ</td>
<td>Tₜ=-1</td>
</tr>
<tr>
<td>Q₀</td>
<td>q₀</td>
<td>Initial costate</td>
<td>None</td>
</tr>
<tr>
<td>AM₀</td>
<td>m₀</td>
<td>Initial mass</td>
<td>28803.1155 kg</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(63500 lbs)</td>
</tr>
<tr>
<td>TIMES</td>
<td>-</td>
<td>Array of times defining start and end of coast and burn arcs</td>
<td>None</td>
</tr>
</tbody>
</table>
Program Output

The exact program output varies with the setting of the output control parameters NAVOFF, PTB, and PTC. The nature of the output, by subroutine, is as follows:

- **MAIN** - error messages, impulsive approximation summary, program notes of convergence status, and converged targeting summary
- **AUXOUT** - summary of current convergence status
- **PHASE** - error messages, orbit-type message (e.g. 'CIRCULAR/CIRCULAR INCLINED ORBITS'), coast messages (when states must be advanced until proper phasing exists), and phasing-orbit messages (including relative geometry, "desired" phasing orbit, and allowable phasing orbit)
- **GLMNTS** - orbital elements and designation as to whether they are representative of state at start or end of a burn.
- **STATIS** - Monte Carlo summary
- **USTAT** - state, costate, and magnitude of costate vectors

PTB and PTC control the sample collection times in guidance mode, and NAVOFF controls the shutoff of the convergence-status summary (from AUXOUT) during guidance mode. In addition, there exists an internal program variable, IPRINT, which when set to 1 produces voluminous output on each call to GUIDE, detailing state-and-costate-at-predefined-times-on-each-coast-arc-and-orbital-elements at the beginning and end of each burn. Because it gives so much output, and is unlikely to be needed over an entire run, IPRINT must be set within the program.
Interdependence of Subroutines

Note that the dashed lines indicate further calls which are adequately described in the GUIDE document (except for the addition, in subroutine GUIDE, of calls to UCOAST and GLMNTS for output purposes).
Subroutine MAIN

A. Purpose

The MAIN routine controls the overall operation of the targetting and guidance program. It has four major sections. The input section, which reads the input data described in a previous section and calculates the orbital and vehicle parameters needed to perform the targetting and guidance; the phasing and impulsive-initialization section which determines the number of burn arcs, rotates the target orbit into the vehicle orbit plane and calculates the planar impulsive solution for the orbit transfer; the convergence section which first converges from the planar impulsive solution to a finite burn solution and then repeatedly reconverges with the target orbit plane rotated in ten degree steps until the desired relative inclination is obtained; the feedback guidance section which exercises the GUIDE algorithm in a realtime guidance environment, continually reconverging in the presence of perturbations and collecting Monte Carlo statistics on the performance of the algorithm.

B. Major Parameters (Input parameters discussed in Section 3)

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPGΩ, HPGT, HPGX</td>
<td>-</td>
<td>Height at perigee for vehicle, target and transfer orbits (km.)</td>
</tr>
<tr>
<td>HAPΩ, HAPT, HAPX</td>
<td>-</td>
<td>Height at apogee for vehicle, target and transfer orbits (km.)</td>
</tr>
<tr>
<td>aΩ, aT, aX, aP</td>
<td>a</td>
<td>Semi-major axis for vehicle, target, transfer and phasing orbits (km.)</td>
</tr>
<tr>
<td>eΩ, eT, eX, eP</td>
<td>e</td>
<td>Eccentricity for respective orbits</td>
</tr>
<tr>
<td>VAPΩ, VAPT, VAPX, VAPP</td>
<td></td>
<td>Velocity magnitude at apogee for respective orbits (km/sec)</td>
</tr>
<tr>
<td>VPGΩ, VPGT, VPGX, VP GP</td>
<td></td>
<td>Velocity magnitude at perigee for respective orbits (km/sec)</td>
</tr>
<tr>
<td>TAUP, TAUT, TAUX, TAUP</td>
<td>τ</td>
<td>Period for respective orbits (sec.)</td>
</tr>
</tbody>
</table>
| IBOUND | - | 0 - Outbound mission  
| | | 1 - Inbound mission |
C. Method of Computation

After reading the data (as previously discussed), the routine determines whether a two burn mission will be sufficient. If the position and velocity vectors and the mean and true anomalies are not given, the true anomalies are arbitrarily chosen such that a two burn mission is possible. This is accomplished by choosing the vehicle state, for T0=2000 seconds, at a node (perigee for outbound and apogee for inbound) in a coordinate system where perigee is in the $x_1$ direction. This forces the first burn to be centered at 2000 seconds and by choosing the target state at its opposite node (apogee for outbound and perigee for inbound) at TT=2000 + TAUX/2.0 (TAUX is period of desired transfer orbit) a two burn transfer is possible. For all other mission definitions the PHASE routine is called and it determines whether two burns will be sufficient and returns the state vectors defined at the time when the first burn is to begin.

Impulsive Initialization

An impulsive approximation is used as an initial guess for converging to the desired finite burn solutions. It is assumed that the optimal orbit transfer always has a burn centered about the greater apogee and this implies that the transfer orbit has as apogee the larger of the two apogees and as perigee the perigee of the other orbit. By calculating the velocities at apogee and perigee along the transfer orbit, the $\Delta v$'s required are easily determined. By converting these $\Delta v$'s to finite burn times, while assuming that the burns are centered at the respective nodes, and starting the mission 2000 seconds before the node, a reasonable time history for a coast-burn-coast-burn mission is defined. A reasonable estimate of initial costate $\dot{q}_0$ is also needed in order to converge the GUIDE algorithm. By investigating the impulsive case, it is determined that the direction of thrust at the node is parallel to the velocity vector and that the rate of change of thrust direction is anti-parallel to the radius vector. (The reverse directions when decreasing velocity is required, on inbound missions.) By noting that the $|\ddot{q}_0|$ is arbitrary for the boundary value problem only one parameter was left to be determined, the relationship between the $|\ddot{u}|$ and $|\dddot{u}|$. (Note: $\ddot{q}_0^T = (\ddot{u}^T, \dddot{u}^T)$.) Using the fact that the variations in $\ddot{r}, \dddot{v}$ form the same class of solutions as $\ddot{u}, \dddot{u}$, and applying the switching condition that $|\ddot{u}|$ at perigee must equal the $|\dddot{u}|$ at apogee, it was found that the impulsive solution for $\ddot{u}$ and $\dddot{u}$ at apogee and perigee becomes
\[
\begin{align*}
\ddot{u} &= \dot{v} \left( \frac{r_a}{v_p} + \frac{r}{v} \right) \\
\ddot{u} &= -\ddot{r} \left( \frac{v_a}{r} \cdot \frac{r_a + v}{v} \right)
\end{align*}
\]

where \( r_a, v_a \) are the magnitudes of the position and velocity vectors at apogee; \( r_p, v_p \) are position and velocity magnitudes at perigee, and \( r, v \) are position and velocity magnitudes at either apogee or perigee (depending on where \( \theta_0 \) is desired) along the transfer orbit. In the program these formulas are further reduced and the \( |\ddot{u}| \) is chosen to be unit magnitude. The formulas become

\[
\begin{align*}
\ddot{u} &= \frac{\ddot{v}}{v} \\
\ddot{u} &= -\ddot{r} \cdot \text{FACTOR}
\end{align*}
\]

where for perigee the factor becomes

\[
\text{FACTOR} = \frac{(1 + e_x/2 - e^2_x/2) \mu}{r_p^3 \nu \nu_p \nu_p}
\]

and at apogee it is

\[
\text{FACTOR} = \frac{\mu + v_a v_p r_a}{r_a (v_p + v_a)}
\]

When the mission is inbound and velocity needs to be reduced, the sign on both \( \ddot{u} \) and \( \ddot{u} \) is reversed. Since this \( \theta_0 \) is defined for the impulsive case it is good at the node and needs to be propagated back to \( T_0 \), the chosen starting time for the mission. The two burn approximate solution is now completed and the program easily converges from this to the true solution.

The approximate solution for the three burn mission is identical to that of the two burn one, except for insertion of a phasing orbit of period \( \text{TAUP} \). For the approximate solution the phasing orbit is assumed to have the same perigee as the transfer orbit and the vehicle orbit (outbound) or target orbit (inbound). This implies that the burn at perigee is split into two burns and \( \text{TAUP} \) is chosen in subroutine \( \text{PHASE} \) to allow these burns to be as
nearly equal as possible. The typical inbound mission approximate solution thus consists of an initial burn centered at apogee of the vehicle orbit, a coast from apogee to perigee along the transfer orbit, a second burn centered about perigee of the transfer orbit, a second coast of the orbital period (perigee to perigee) along the phasing orbit and a final burn centered again at perigee. The costate vector for the inbound 3 burn planar mission (plane change is added after initial convergence) was initialized using the same formulas as the two burn case and the inbound mission successfully converges.

For the three burn outbound mission, convergence proved to be more difficult. It was discovered that the switching condition along the transfer orbit coast was very sensitive, and that the peaking characteristic of $\dot{u}$ at apogee and perigee was impossible to maintain when the phasing orbit was encountered before the transfer orbit. It was found that by solving the mission backwards and integrating over the transfer orbit first, reasonable convergence was attained. In order to run the GUIDE algorithm backwards from apogee on the target orbit to perigee on the vehicle orbit with increasing mass it was necessary to make the orbits retrograde by changing the sign of their velocity vectors, to change the mass rate from positive to negative, to change the sign on initial $\dot{u}$, to reduce initial mass and to alter the TIMES array. The TIMES array for the backwards three burn outbound mission is initially targetting by choosing it to be

$$
\begin{align*}
\text{TIMES}(1) &= T0 \\
\text{TIMES}(2) &= \text{TIMES}(1) + \text{BURN3} \\
\text{TIMES}(3) &= \text{TIMES}(2) + \text{TAX}/2 - \text{*BURN3 + BURN2)/2.0} \\
\text{TIMES}(4) &= \text{TIMES}(3) + \text{BURN2} \\
\text{TIMES}(5) &= \text{TIMES}(4) + \text{TAUP - (BURN2 + BURN1)/2.0} \\
\text{TIMES}(6) &= \text{TIMES}(5) + \text{BURN1}
\end{align*}
$$

Where BURN1 is the length of the burn at perigee of the vehicle orbit, BURN2 is the length of the burn at perigee of the phasing orbit, BURN3 is the length of the burn at apogee of the transfer orbit and TAX and TAUP are the periods of the transfer and phasing orbits respectively. The initial mass is reduced to

$$
m_0 = m_0 - \dot{m}(\text{BURN1 + BURN2 + BURN3})$$

where $m$ is positive. Q0 is initialized at apogee of the transfer orbit and then the last three components are changed in sign ($\dot{u}$).
Mission Convergence

Using these approximate solutions for the two and three burn missions, the planar missions are converged in less than twenty iterations. At this point the relative inclination, RELINC, between the target and vehicle orbits is tested and if it exceeds some minimum value, the mission is altered to include the desired plane change. The target orbit is rotated in maximum of 10° steps from the vehicle orbital plane, and is reconverged at each step in the process. The two burn missions converged readily using this procedure but it was necessary to alter the three burn missions to two burn ones to obtain good convergence properties. This was accomplished by replacing the lowest orbit (target orbit for inbound and vehicle orbit for outbound) by the phasing orbit found during the planar mission convergence. The inbound mission is converged as a two burn one with the desired end conditions being the phasing orbit rotated about perigee. The outbound mission is converged backwards rotating at each step the target orbit as well as initial costate and converging to the phasing orbit. After inclusion of the total desired angular rotation, the third burn is again introduced into the mission definition and convergence for the three burn mission is attained. The outbound 3-burn mission is then turned around and solved in a forwards fashion using the final costate as initial costate and the burn and coast times derived from the backwards convergence.

Feedback Guidance

At this point targeting is completed and a converged solution exists for guiding the vehicle into the target orbit. In the MAIN routine the major guidance function performed is to control the collection of and print the Monte Carlo statistics generated when doing feedback guidance. The routines BCBCB and CBCB called by MAIN add perturbations into the state of the vehicle and move step by step in time through a full feedback guidance cycle. At several points along each burn and coast arc, error statistics are gathered and an estimate is made of the error in meeting desired end conditions. These statistics are collected over MCARLO separate orbital transfers and a summary printout is obtained from routine STATIS.

This completes the description of the MAIN routine. A math flowchart of it is contained on the next three pages.
START
READ INPUT DATA

NOTARG=1 ?

NO

NO TIME ALONG ORBITS SPECIFIED

NBMURS=2

TO \( TT > 0 \) OR TANOMO, TANOMT \( \geq 0 \) ?

NO

CALL PHASE

DETERMINES IF 3-BURNS ARE NECESSARY, CALCULATES PERIOD OF PHASING ORBIT AND ROTATES TARGET ORBIT INTO VEHICLE ORBIT PLANE

YES

NBMURS=2 ?

NO

NO

3-BURN IMPULSIVE INITIALIZATION

2-BURN IMPULSIVE INITIALIZATION

3-BURN OUTBOUND MISSION

TURN PROBLEM AROUND TO INSURE CONVERGENCE. GO FROM TARGET ORBIT TO VEHICLE ORBIT WITH INCREASING MASS.
RESET \( X_0, Q_0, X_T, T I M E S \) ARRAY AND AMO (INITIAL MASS).

IBACK=1

PRINT IMPULSIVE SOLUTION

CALCULATE SOFT CONSTRAINT WEIGHTS

CALCULATE DESIRED END CONDITIONS AND CONVERGE (IN UP TO 30 ITERATIONS) MISSION

RELINC=0 ?

YES, NO PLANE CHANGE REQUIRED

295
CONVERGE PLANE CHANGE IN MAXIMUM OF 10° STEPS

\[
\text{ANGLE} = \min(\text{ANGLE} + 10°, \text{INCLINATION (RELINC)})
\]

3-BURN NO

NBURNS=2 ? YES 2-BURN

CONVERT TO 2-BURN MISSION USING CONVERGED COPLANAR PHASING ORBIT AS DESIRED END CONDITIONS

IBACK=1 ? YES

BACKWARDS MISSION MUST ROTATE XO, QO ABOUT PERIGEE (XO IS REAL TARGET ORBIT)

NO

3 BURN INBOUND
ROTATE XT-PHASING ORBIT STATE AT TT

CALCULATE DESIRED END CONDITIONS AND CONVERGE PLANE CHANGE MISSION

ANGLE<RELINC ? YES, ADD MORE PLANE CHANGE

NBURNS=2 ? YES 2-BURN TARGETTING COMPLETED

ADD THIRD BURN BACK INTO MISSION DEFINITION

RELINC=0

RETURN TO RECONVERGE TOTAL MISSION
FILE: MAIN

FORTRAN P1

ENGLISH TO METRIC CONVERSIONS

1 POUND THRUST = 0.0444822165 KILOGRAVTS
HENCE, THRUST(KN) = THRUST(LBS) * 0.0444822165

1 POUND MASS = 0.4536 KILOGRAMS
HENCE, MASS(KG) = MASS(LBS) * 0.4536

MASS RATE (KG/SEC) = THRUST(LBS) / ISP(SEC) * 4536 KG/LB

HENCE, ISP(SEC) = THRUST(KN) / 0.0444822165

INITIAL MASS IN KILOGRAMS

AM0 = 28603.1155

THrust IN KILOGRAVTS

THRUST = 667233

SPECIFIC IMPULSE IN SEC

SPFIMP = 440

MASS RATE IN KG/SEC

AMOUT = -1.0

INITIALIZE VARIABLES

C SLT. INPUT DEVICE

INPUT = 1

C SET OUTPUT DEVICE

IOUTPT = 6

SET TIME TOLERANCE (SECONDS) USED TO DETERMINE IF TUG OR TARGET IS CLOSE ENOUGH TO A NODE OR DESIRED MEAN ANOMALY.

ERROR = 10

SET ANGLE TOLERANCE (DEGREES). DIFFERENCES IN ANGLES LESS THAN ERROR WILL BE IGNORED.

C SET ECCENTRICITY TOLERANCE. ORBITS WITH ECCENTRICITIES LESS THAN ERROR WILL BE TREATED AS CIRCULAR.

ERROR = 0.01

SET TIMES TO TT TO -1.0 TO INDICATE A 2-BURN MISSION.

C IF THEY ARE CHANGED BY THE INPUT DATA A 3-BURN MISSION IS ASSUMED.

TO = 1.0

TT = -1.0

SET INPUT VARIABLES TO -1.0 OR 0.0 TO INDICATE THAT THEY HAVE NOT BEEN READ IN THROUGH NAMECL.

CK = 1.0

AG = 0.0

EO = -1.0

AT = 0.0

LT = -1.0

RLLINC = 0.0

HAPU = 1.0

HPGO = -1.0

HPTE = -1.0

FLTO = -1.0

FLTT = -1.0

TANG = -1.0
**FILE: MAIN FORTRAN 91**

- **NAMEXIT=1.0**
- **RUGAG=1.0**
- **DU S 1=1.3**
- **RO(1)=0.0**
- **V0(1)=0.0**
- **RI(1)=0.0**
- **V(1)=0.0**
- **XGB(2)=0.0**
- **XGB(3)=0.0**
- **XGB(4)=0.0**
- **XGB(5)=0.0**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUGAG</td>
<td>1.0</td>
</tr>
<tr>
<td>DU S 1</td>
<td>1.3</td>
</tr>
<tr>
<td>RO(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>V0(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>RI(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>V(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>XGB(2)</td>
<td>0.0</td>
</tr>
<tr>
<td>XGB(3)</td>
<td>0.0</td>
</tr>
<tr>
<td>XGB(4)</td>
<td>0.0</td>
</tr>
<tr>
<td>XGB(5)</td>
<td>0.0</td>
</tr>
</tbody>
</table>

- **NgUrn=2**
- **J1UPOAT=0**
- **NOIS0N=0**
- **NAVUFF=1**
- **NOTARG=0**
- **PTU(0)=-P**
- **PTU=2000**
- **PROS(1)=1.0**
- **PROS(2)=0.0**
- **PROS(3)=0.0**
- **IPHAS=0**
- **IBACK=0**
- **ITURN=0**
- **IBOUND=0**
- **IPRINT=0**

**C DIRECTION OF EARTH'S AXIS IN REFERENCE COORDINATE SYSTEM.**

- **OBLAT=0.0**

- **C STEP SIZE IN GUIDANCE MODE DURING BURN.**
  - **DTYPE(1)=20.**

- **C STEP SIZE IN GUIDANCE MODE DURING COAST.**
  - **DTYPE(2)=100.**

- **C.ZERO ARRAYS**
  - **DU 1 I=1,12**
  - **PVT(1)=0.0**
  - **DO 3 I=1,5**
  - **DO 3 J=1,10**
  - **DO 2 K=1,8**
  - **DATAM(1,J,K)=0.0**
  - **DATAM(1,J,1)=0.0**
  - **DATAM(1,J,2)=0.0**
  - **DATAM(1,J,3)=0.0**
  - **DATAM(1,J,4)=0.0**
  - **DATAM(1,J,5)=0.0**
  - **DATAM(1,J,6)=0.0**
  - **DATAM(1,J,7)=0.0**
  - **DATAM(1,J,8)=0.0**
  - **DATAM(1,J,9)=0.0**
  - **DATAM(1,J,10)=0.0**
  - **DATAM(1,J,11)=0.0**
  - **DATAM(1,J,12)=0.0**

- **C SET TRANSVERSALITY CONDITION TO ZERO.**
  - **TV0=0.0**

- **C SET SEEDS FOR ISM-LOCAL RANDOM NUMBER GENERATOR.**
  - **C THE NEXT LINES MAY BE DROPPED ALONG WITH COMMON BLOCK RNT1, SO LONG AS A RANDOM NORMAL**
  - **C (MEAN 0, VARIANCE 1) NUMBER GENERATOR IS SUBSTITUTED.**
  - **NRRNAIL(1)=0.0**
FILE: MAIN FORTRAN PROGRAM

CAMBRIDGE MONITOR SYSTEM

NRNAAR(2)=0
NRNAAR(3)=35737
NRNAAR(4)=0
C READING INPUT FROM NAMELIST
READ(INPUT,NAMLIST)
C COMPUTE MASS HATL. IF NOT READ IN.
IF (AMODLT.0.0) AMODLT=THRUST/SPE.IMP*101.9716
C PERTURBATIONS IN TUG ACCELERATION DURING GUIDANCE. PERT(1) IS DURING
C A BURN AND PERT(2) DURING A COAST. PERT(3) CAN BE USED FOR INTRODUCING.
C PERTURBATIONS IN THE TARGET ACCELERATION
IF (PERT(1)+PERT(2)*0.01-3) GO TO 6
C
PERT(1)=.05*THRUST/A00
PERT(2)=.00005#UK/(6700.*#2)
C STORE VEHICLE SPECS IN WORKING ARRAY VEH(1,7)
VEH(1,1)=AMO
VEH(1,2)=AMOD
VEH(1,3)=10000.
VEH(1,4)=THRUST
VEH(1,5)=.0
VEH(1,6)=0.0
VEH(1,7)=25.0
C CHECK IF GUIDANCE IS DESIRED WITHOUT PRELIMINARY TARGETING (OPTIMAL)
C SOLUTION ALREADY EXISTS. A PASS WILL BE MADE THROUGH A CONVERGENCE
C LOOP TO INSURE THAT THE SOLUTION IS VALID.
IF (NOTARG.EQ.1) ITUNK=3
IF (ROTARG.EQ.1) GO TO 204
C TARGETING IS DESIRED. DETERMINE IF A 2 OR 3-BURN MISSION IS DESIRED.
C IF TO-GLE, U=0 AND EP=1 THEN 3-BURN ASSUMED.
C ALSO, IF THE TRUE ANOMALIES WERE SPECIFIED, A 3-BURN MISSION IS
C ASSUMED.
IF (TO.GE.0.000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
FILE: MAIN  FORTAN  PL  CAMBRIDGE MONITOR SYSTEM

IF(KOMAG.GT.REARTH.AND.VCOMAG.GE.1.AND.FLTOUG.EQ.0) GO TO 100
RTMAT.REARTH.AND.VRTMAT.GE.1.AND.FLTRUG.EQ.0) GO TO 75
C ALLOWABLE SET OF DATA FOR 2-BURN MISSION WAS NOT READ IN. UMP

C-VARIABLES AND TERMINAL

WRITE(IOUTPT,3000)
3000 FORMAT(' EXECUTION TERMINATING. IMPROPER DATA SPECIFIED FOR 2-BURN MISSION.')

WRITE(IOUTPT,3001) AO,EO,AT,LT,RLLNC

WRITE(IOUTPT,3002) HAP,HPG,HAPT,HRPT,RLLNC

WRITE(IOUTPT,3003) KOMAG,VOMAG,FLTUG,RTMAT,VTMAG,FLTTUG.RLLNC

STOP

C-CONVERT HEIGHTS AT APPOCLE AND PERIGEE INTO ORBITAL ELEMENTS.
50 AO=REARTH+(HAPU+HRPT)/2.
EO=(HAP+REARTH)/2-1.0.
IF(EO.LT.REARTH). EO=0.0.
AT=REARTH+(HAPT+HRPT)/2.
LT=(HAPT+REARTH)/2-1.0.
IF(EL.T.Equer). LT=0.0.
GO TO 110

C CONVERT POSITION VELOCITY AND FLIGHT ANGLES TO ORBITAL ELEMENTS.
75 AO=UK+KOMAG/(2.0+K-KOMAG*K*HAP).
VOMAG=DBS(KOMAG*VOMAG*DSIN(FLTU+DEGCON))
E0=DSQRT(1-HOMAG*2/(AO*UK))
!
IF(EL.T.ELRKR). EL=0.0.
AT=UK+RTMAT/(2.0+UK-VOMAG*2*RTMAT).
HTMAT=DBS(KEART+VOMAG*DSIN(FLTU+DEGCON))
LT=DSQRT(1-HTMAQ*2/(AT*UK))
!
100 HAP=AO+(1+EO)-REARTH.
HPG=AO+(1+EO)-REARTH.
HAPT=AT+(1+LT)-REARTH.
HPGT=AT+(1+LT)-REARTH.
IF(HPGT.GT.0.0).AND.(HPGT.LT.0.0). GO TO 110

WRITE(IOUTPT,3150) HPG,HAPT
3150 FORMAT(' EXECUTION TERMINATING../-/-/- HEIGHT AT PERIGEE OF TUG=1,14.E0",/HEIGHT AT PERIGEE OF TARGET=",14.E0";

STOP

C DETERMINE INITIAL AND FINAL RAD11 TEST FIRST IF AN INBOUND OR
C OUTBOUND MISSION. IF APPOCLE OF THE TARGET ORBIT IS LESS THAN APPOCLE
C OF THE TUG ORBIT, AN INBOUND MISSION IS ASSUMED.

C-CHECK FOR GREATER APPOCLE.
110 IF(HAPG.GT.HAPT) GO TO 125

C OUTBOUND MISSION (FROM PERIGEE TO APPOCLE).

IBOUND=0.
R=REARTH+HPG.
RF=REARTH+HAPT.
GO TO 150.
C. INBOUND MISSION: (FROM APOLLO TO PERIGEE).

125  IBOUND=1
RI=REARTH+HAPX
RF=REARTH+HPG

C. CALCULATE INITIAL AND FINAL VELOCITIES.

150  V=DSORT(UK*(2./RF-1./AT))

C DETERMINE VELOCITIES AT END POINTS OF TUG TARGET ORBITS

VAPX=DSORT(UK*(2./HAPX+REARTH-1./AT))
VAPX=DSORT(UK*(2./HAPX+REARTH-1./AT))
VAPX=DSORT(UK*(2./HAPX+REARTH-1./AT))

C DEFINE STATES IN ARBITRARY COORDINATE SYSTEM, UNLESS ONE IS IMPLIED

C THROUGH THE STATE VECTORS (PHASE WAS CALLED), THE TUG WILL BE LOCATED

C AT PERIGEE/APOLLO AND THE TARGET AT APOLLO/PERIGEE DEPENDING ON THE

C VALUE OF IBOUND. PERIGEE IS IN THE DIRECTION (1,0,0) AND T.E.M.

C VECTOR IN THE DIRECTION (0,0,1)

C IF (PHASE = 1), THE STATE VECTORS WILL BE SET UP IN SUBROUTINE PHASE.

165  IF (PHASE = 0.1) GO TO 175
SIGN=1.0

175  IF (PHASE = 0.1) SIGN = -1.0
R(1)=RI*SIGN
R(2)=R0.0
R(3)=R0.0
V(1)=V0.0
V(2)=V0.0
V(3)=V0.0

C SET UP TRANSFER ORBIT

170  AA=(RI+RF)/2.

C THE INITIAL DELTA V IS DEFINED AS THE VELOCITY AT THE APOLLO/PERIGEE

C TRANSFER ORBIT MINUS THE INITIAL VELOCITY. THE FINAL DELTA V IS

C DEFINED AS THE FINAL VELOCITY MINUS THE VELOCITY AT PERIGEE/APOLLO ON

C THE TRANSFER ELLIPSE.

185  IF (PHASE = 1) GO TO 195

DELTV1=VAPX-V1
DELTV2=V-F-VAPX

VELXFR=VAPX
GO TO 195

DELTV1=VAPX-V1
DELTV2=V-F-VAPX
VELXFR=VAPX

C DETERMINE ORBITAL PERIODS.

195  TAU=2.*PI*DSORT(AU**3/UK)}
FILE: MAIN

TAUX=2.*P1*DSRT(AX**3/UK)
TAUT=2.*P1*DSRT(AX**3/UK)
C DETERMINE FINITE BURNS FROM DESIRED DELTA V'S AND INITIAL "ASS.
BURN1=AMG/AMDOT*(DLXP(-ANDOT*DAAS(DELTV1)/THUST)-1.0)
DELTM=BUKN1*AMDOT
BURN2=(-AMG-DELTM)/AMDOT*(DLXP(-ANDOT*DAAS(DELTVF)/THUST)-1.0)
COAST=TAUX/2.+BUKN1+BUKN2/2.
C SET UP TIMES ARRAY FOR 3-BURN MISSION WITH TIME TO=0 ARBITRARILY 2000
C SECONDS BEFORE THE TUG IS DUE AT THE NODE, UNLESS TO AND TV WERE
C-ORIGINALLY SPECIFIED AS "TO=0.0" IN WHICH CASE...JO...IS...
C RUN BACK BY 2000 SEC. SO LONG AS IT IS *GE`t* 
TIMES(1)=0.0
TIMES(2)=0.0
TIMES(3)=2000.+BURN1/2.+TO
TIMES(4)=TIMES(3)+BURN1
TIMES(5)=TIMES(4)+COAST
TIMES(6)=TIMES(5)+BURN1
C STORE THE TUG AND TARGET STATES IN THE WORKING VARIABLES X,XT
DO 200 I=1,3
X0(I)=X0(1)
XG(I+3)=V0(I)
XT(I)=RT(I)
200 CONTINUE
C.DEFINE IMPULSIVL CUSTATE. U IS IF UNIT MAGNITUDE AND ALONG VELOCITY
C-VECTOE AT THE NODE AND U=0 IF ALONG THE EARTH RADIUS VECTOR.
IF (IINUND.EQ.1) GO TO 2005
SIGN=1.0
FACTOR=(1.+LX/2.0.-LX**2/2.0)*UK/(RI*X3*VELXR)
GO TO 2007
2005 SIGN=-1.0
FACTOR=(UK+VARX*XVELX*(HAPX+REARTH))/((HAPX+REARTH)**3.*VARX*VEGX)
2007 CONTINUE
DO 201 I=1,3
QU(I)=XG(I+3)/SIGN
201 CONTINUE
GU(I+3)=-X0(1)*FACTOR*SIGN
NU=0
C.COAST. TUG JUMPS ARBITRARY 2000 SEC. BEFORE START OF THE 1ST BURN.
CALL COAST(X0,0.,-2000.,XG,0.,PHI1,PHI)
TBEGIN=TIMES(3)
C.DO 10 202
C.THREE-BURN MISSION IS ASSUMED UNLESS THE PHASING WILL ALLOW.
C.
C.*.*.*.*.*.THREE-BURN MISSION ** ** ** **

C

A.3-BURN MISSION IS ASSUMED UNLESS THE PHASING WILL ALLOW.
C-ARE BURN MISSION. CHECK IF THE TUG AND TARGET STATES WERE SPECIFIED.

500 IF(HOMAG*GT.0.0) GO TO 501

MA100387

MA100388

MA100389

MA100390

MA100391

MA100392

MA100393

MA100394

MA100395

MA100396

MA100397

MA100398

MA100399

MA100400

MA100401

C-ANGLES WERE SPECIFIED. (FLIGHT ANGLE DEFINED IN 3-BURN COMMENTS.)

501 IF(HOMAG*GT.REARTH AND VOMAG*GT.0.0) GO TO 500

MA100402

MA100403

MA100404

MA100405

MA100406

MA100407

C INADEQUATE ELEMENTS SPECIFIED. STOP.

WRITE(2,1300) RO,VO,RT,VT,RELEV,AG,E,EU,AT,ET

MA100408

MA100409

C-CONVERT HEIGHTS INTO ORBITAL ELEMENTS.

505 AO=REARTH+HAP+HPO0/2.0

MA100410

MA100411

MA100412

MA100413

MA100414

IF(EL+LT-EERROR) EL=0.0

MA100415

MA100416

MA100417

MA100418

AT=ELARTH+(HAP+HPT)/2.0

MA100419

MA100420

MA100421

MA100422

MA100423

MA100424

MA100425

MA100426

MA100427

MA100428

MA100429

MA100430

MA100431

MA100432

C ELEMENTS WERE SPECIFIED. CHECK WHETHER TRUE ANOMALIES WERE SPECIFIED.

IF(HPG0=GE.0.0 AND HP1T=GE.0.0) GO TO 501

MA100433

MA100434

MA100435

MA100436

MA100437

MA100438

C ANOMALIES WERE NOT SPECIFIED. THE TIMES TO IT WILL NOW BE CONSIDERED.

THE MEAN ANOMALIES (TIMES SINCE PERIGEE OR, IF CIRCULAR, SINCE THE

C-NODE).
FILE: MAIN

C CHECK IF EITHER IS GREATER THAN ITS ORBITAL PERIOD.
TAU = 2. * PI  * DSORT(A0*360/UK)
T AUT = 2. * PI  * DSORT(A0*360/UK)

IF (T0.LT.TA U 0 . AND. TT.LT.TA UT) GO TO 520

C TIMES ARE NOT LESS THAN ONE ORBITAL PERIOD. STOP
WRITE (1001,PT=3102) T0,TT,TAU0,TATU

3102 FORMAT (* EXECUTION TERMINATING. MEAN ANOMALIES EXCEED ORBITAL
1 PERIOD, T0,T T = T0, D14.0, TT = T T, D1 4. 0, TAU0 = TAU 0, D1 4.0,

C TIMES SINCE PERIGEE PASSAGE ARE REASONABLE. DEFINE TUG AND TARGET
C STATES AT PERIGEE AND PROPAGATE AHEAD VIA CALLS TO COAST. 0 THE
C TIMES GIVEN BY THE MEAN ANOMALIES.

520 STATE(1) = A0*(1. - E0)
STATE(2) = 0.0
STATE(3) = 0.0
STATE(4) = 0.0
STATE(5) = DSORT(UK*(2./STATE(1)-1./TAU))
STATE(6) = 0.0

NO=1
CALL COAST(ST ATE,0DUM,0,T0,STATE,0DUM,PH1,PH1)

DO 525 I = 1, 3
R0(I) = STATE(I)
525 V0(I) = STATE(I+3)

STATE(1) = AT*(1. - LT)
STATE(2) = 0.0
STATE(3) = 0.0
VMAGNT = DSORT(UK*(2./STATE(1)-1./AT))
STATE(4) = 0.0
STATE(5) = VMAGNT*DOSIN(REINC#DEGCON)
STATE(6) = VMAGNT*DOSIN(REINC#DEGCON)
CALL COAST(ST ATE,0DUM,0,TT,STATE,0DUM,PH1,PH1)

DO 530 I = 1, 3
RT(I) = STATE(I)
530 VT(I) = STATE(I+3)
C SET BOTH TIMES TO 2000. AFTER PROPAGATING STATES THROUGH THEIR
C MEAN ANOMALIES.
TO = 2000.
TT = 2000.0
GO TO 000

C TRUE ANOMALIES WERE SPECIFIED. SET UP COORDINATE SYSTEM SU H
C THAT THE X(1) AX IS TUGANUG TUG PERIGEE. X(3) IS ALONG
C H, AND X(2) IS X(1) CROSS X(1).

550 RMAG = A0*(1. - E0*2)/((1.+E0)*DCOS(TANMM#DEGCON))
RO(1) = RMAG*DOSIN(TANMM#DEGCON)
RO(2) = RMAG*DOSIN(TANMM#DEGCON)
RO(3) = 0.0
VMAG = DSORT(UK*(A0*(1.-E0*2)))
V0(1) = VMAG*DOSIN(TANMM#DEGCON)
V0(2) = VMAG*(E0+DCOS(TANMM#DEGCON))
V0(3) = 0.0
RMAG = AT*(1.-LT*2)/(1.+LT*DCOS(TANMM#DEGCON))
RT(1) = RMAG*DOSIN(TANMM#DEGCON)
RT(2) = RMAG*DOSIN(TANMM#DEGCON)

C
\textbf{C} \textit{Main Program}
FILE: MAIN  FORTRAN PI  CAMBRIDGE MONITOR SYSTEM

HAPG=A0*(1.+EO)-REAL(1.
HPGU=A0*(1.0)-REAL(1.
HAPT=AT*(1.+LT)-REAL(1.
HPGT=AT*(1.+ET)-REAL(1.
IF(HAP0,GT,HAPT),IBOUND=1
AP=(1AUP*2*UK/(1.+LT)*1.)*1.33333333
IF(IOBOUND+L0.0).HPGU=HPO0
IF(IOBOUND+EQ.1) HPGP=HPGT
EP=1.0-(HPGU+REAL(1.))/AP
HAPP=AP*(1.*EP)-REAL(1.
C CALCULATE VELOCITIES AT END POINTS OF ALL ORBITS
VAP0=DSORT(UK*1.)/REAL(1.))
VPGU=DSORT(UK*1.0)/REAL(1.0))
VAPT=DSORT(UK*1.0)/REAL(1.0))
VPGT=DSORT(UK*1.0)/REAL(1.0))
VAP0=DSORT(UK*1.0)/REAL(1.0))
VPGU=DSORT(UK*1.0)/REAL(1.0))
IF(IOBOUND+EQ.1) GO TO 701
RI=HPGU+REAL(1.
RF=HAPT+REAL(1.
GO TO 702
701 RI=HAPG+REAL(1.
RF=HPGT+REAL(1.
GO TO 702
702 VI=DSORT(UK*2./R1-1./AO))
VF=DSORT(UK*2./RF-1./AT))
AX=(R1+RF)/2.
EX=OAMAX(R1,RF)/AX-1.0
HAPX=AX*(1.+EX)-REAL(1.
HPGX=AX*(1.-EX)-REAL(1.
VAPX=USORT(UK*2./OAMAX(R1,RF)-1./AX))
VPGX=USORT(UK*2./OAMIN(R1,RF)-1./AX))
IF(IOBOUND+EQ.1) GO TO 703
DELTVI=WPGP-VI
DELTVI=VF-VAPX
GO TO 703
703 DELTVI=VF-VAPX
DELTVI=WPGP-VPGX
DELTVF=VF-WPGX
-C-FIND FINITE BURN TIMES
7035 BURN1=A0/AMO*AMO*(EXP(-AMO*OABS(DELTVI)/THUST)=1.0)
DELTAAMBOUTBURN1
BURN2=-(AMO-DELTAAM)/AMO*(EXP(-AMO*OABS(DELTVM)/THUST)=1.0)
DELTAAMBOUTBURN2+DELTAAM
BURN3=-(AMO-DELTAAM)/AMO*(EXP(-AMO*OABS(DELTVF)/THUST)=1.0)
IF(IOBOUND+EQ.1) GO TO 704
COASTIAU=-(BURN1+BURN2)/2.
COAST2=TAUX/2.-((BURN1+BURN2)/2.)
GO TO 705
704 COAST1=TAUX/2.-((BURN1+BURN2)/2.}
COAST2=TAUX/2.-((BURN1+BURN2)/2.}
GO TO 705
-C-SET UP TIMES ARRAY
705 IF(0 TO-20000 LT 0) TT=TT+2000 TO=0
IF(0 TO-20000 LT 0) TO=0.0
IF(TO+LT<001) TO=10-2000

MAIN - 19
T0=0.0; T0=BURN1/2.0+T0
T0=2.0; T0=BURN1/2.0+T0
T0=0.0; T0=BURN1
T0=2.0; T0=BURN1
T0=0.0; T0=BURN1
T0=2.0; T0=BURN1
T0=0.0; T0=BURN1
T0=2.0; T0=BURN1
T0=0.0; T0=BURN1
T0=2.0; T0=BURN1
T0=0.0; T0=BURN1
T0=2.0; T0=BURN1
C STORE STATES IN X0 AND XT
DO 710 I=1,3
X0(I)=X0(I)
X0(1)+3=V0(1)
AT(I)=X(I)
710 XT(I+3)=VT(I)
C SET UP INITIAL G FOR INBOUND MISSION
    DO 720 J=1,3
    QD(I)=X0(I+3)/V0
720 QD(I+3)=X0(I)+3/FAC1
NO=0
C COAST TOU BACK TO START OF FIRST BURN
    CALL COAST(X0,VO,Q0,BURN1/2.0,X0,VO,PHI,PHI)
    IF(ISOUND.EQ.1) GO TO 202
C TO INSURE CONVERGENCE
C
C  TRACK=1
C
C SAVE INITIAL STATE OF TUG
    DO 750 J=1,3
    TUGSAV(I)=X0(I)
750 TIMTUG=T0
C-PROPAGATE TARGET STATE TO FINAL BURN NODE
NO=-1
    CALL COAST(XT,UDUM,TAUX/2.0,X0,UDUM,PHI,PHI)
C-SETUP IMPULSIVE O SOLUTION AT NODE
    FACTOR=(UK+VAPX+VPGX*(HAPX+REARTH))/((HAPX+REARTH)**3*(VAPX+VPGX))
    DO 760 I=1,3
    QD(I)=X0(I+3)/VAPX
760 QD(I+3)=X0(I)+3/FAC2
C-PROPAGATE TO END OF LAST BURN
NO=0
C SET UP TIMES ARRAY
TIMS(1)=T0
TIMS(2)=TIMS(1)+BURN1
TIMS(3)=TIMS(2)+COAST1
TIMS(4)=TIMS(3)+BURN1
TIMS(5)=TIMS(4)+COAST1
TIMS(6)=TIMS(5)+BURN1
C-REDUCE MASS, SET TIMES AND TARGET STATE
T0=TIMS(1)
TT=TIMS(5)
AM0=AMG=(BURN1+BURN2+BURN3)*AM0UT
FILE: MAIN FORTRAN PI  CAROLRIDGE MONITOR SYSTEM

C-REVERSE MASS RATE VELOCITY AND UDOT
AMODT=-AMODT
VEH(1,2)=-VEH(1,2)
00.=90.14.="-XT(J-3)=TUGSAV(J-3)
XT(J)=-TUGSAV(J)
XG(J)=-XO(J)
00.=-00(J)

C WRITE IMPULSIVE SOLUTION
202 WRITE(IOUTPT,3200) NBURNS
3200 FORMAT(//20X,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,**,****
FILE: MAIN FORTRAN 41 CAMBRIDGE MUNITO SYSTEM

-249 -CC(1)=DD(1)
C * * * * * 
C
C IF(ITURNR.EQ.0) WRITE(10,PT,3210)
C IF(IGRAPH .EQ.1) WRITE(10,PT,3213)
C IF(IGRAPH .EQ.2) WRITE(10,PT,3220)
C IF(IGRAPH .EQ.3) WRITE(10,PT,3223)
C 3223 FORMAT(' BEGIN GUIDANCE-ONLY CONVERGENCE
C 3220 FORMAT(' ADD 3RD BURN AND RECONVERGE
C 3213- FORMAT(' BEGIN TURN-AROUND CONVERGENCE
C 3210- FORMAT(' BEGIN COPLANAR CONVERGENCE
C C TRY TO CONVERGE THE COPLANAR MISSION IN 10,50 ITERATIONS.
C
C DO 250 ITER=1,50
C NIP=1
C QMAX=DOTRT(1,1)**2+Q(2)**2+Q(3)**2
C DO 247 Q(1)=Q(1)/QMAX
C CALL GUIDE(0.0)
C CALL AUXOUT
C CALL CKSET(CK)
C QUMAX=Q(0)
C DTMAX=Q(0)
C DO 251 Q(1)=Q(1)+Q(1)*CK
C TIMES(1)=TIMES(1)+DTMAX
C IF(DABS(Q(0)) .GT. QUMAX) QUMAX=DABS(Q(0))
C 251 AMI=AMU
C IF(ITAT=1) AMU=V/LH(1,1)+TIMES(5)-TIMES(5)+TIMES(4)-TIMES(4)
C 1 TIMES(2)-TIMES(1)+AMI OUT
C AMI=(AMU+AMI)**0.5
C IF(DTMAX .LE. 0.0) GO TO 252
C DABS(AMU)*LH(1,1)*LH(1,1)-LH(1,1)**0.5 > 0.0
C 252 CONTINUE
C DIF NOT CONVERGE IN 30 ITERATIONS, DUMP VARIABLES AND STOP.
C WRITE(10,PT,3000)
C 3005 FORMAT(' PLANAR MISSION DID NOT CONVERGE IN 30 ITERATIONS. STOP')
C 3000 CONTINUE
C ADD PLANAR CHANGE
C
C 1SEC0=60
C ANGLE=U.0
C IF(NGO PLANAR CHANGE SKIP ARROUND
C IF(DABS(RELINC)*LT.*ERROR) 60-10-299
C TRANSFORM TO 2-BURN MISSION FOR INSERTION OF PLANET.

IF(NBURNS,NE.,3) GO TO 2940

TOSAV=TMSL(6)
TOSAV=TIMES(6)
TIMES=TT

DO 293 1=1,4

253 TIMES(-7:1)=TIMES(-1:1)

NOR=0

C CALL GUIDE TO GENERATE PHASING ORBIT END CONDITIONS

CALL GUIDE(0,0)

DO 254 1=1,6

254 XPHASE(1)=X(I)

TT=TIMES(0)

25Q-CONTINUE

C PERFORM REQUIRED PLANET CHANGE IN 10 DEGREE STEPS

RELJ=JADS(RELINC)

253-ANGL=DMIN2(JADS(ANGL)+10..RELJ)

ANGL=SIGN(ANGL+RELINC)

DANGL=ANGL-ANGL

ANGL=ANGL

WRITE(10,OUTPUT,4DD) ANGL

400 FORMAT('i ATTEMPT TO CONVERGE *10 DEGREE PLANET CHANGE')

C TEST WHETHER TARGET STAY AT ONE OF NODES (ONLY IF PHASE NOT CALLLD)

IF(IPHASE..NE.0) GO TO 257

C RT, VT AT APOGEE OR PERIGEE, ROTATE THROUGH ANGLE

DO 256 1=1,3

256 X(I)=X(I(1))

XT(I4)=X(I)

XT(I5)=VT(2)*DCLG(ANGL*DCLGL)

XT(I6)=VT(2)*DCLG(ANGL*DCLGL)

GO TO 1000

C CHECK TO SEE IF CONVERGING BACKWARDS

257 IF(IDACK..LE.0) GO TO 2570

C CONVERGING BACKWARDS ROTATE X0 AND X6

C CALL ROTATE(X0,PRCL,DANGL*DEGCON6)

C CALL ROTATE(X6,PRCL,DANGL*DEGCON6)

GO TO 259

C CONVERGING FORWARDS, FIND IF 2 OR 3 BURNS

2570 IF(NBURNS,LE.,3) GO TO 296

C 2-BURN MISSION, ROTATE PRESENT AT

C CALL ROTATE(EXT,PRCL,DANGL*DEGCON6)

GO TO 1000

C 3-BURN FORWARD, ROTATE XPHASE

C CALL ROTATE(XPHASE,PRCL,DANGL*DEGCON6)

C SET UP 3-BURN AS 2-BURN

259 DO 2600 I=1,6

2600 XT(I)=XPHASE(I)

C ON FIRST PASS SET UP AN INITIAL COAST

IF(ISCOD,EQ.0,1) GO TO 1000

ISCD=1

NU=0

TBCF=-500.

CALL COAST(X0,00,TOT,CT,X0,CO,PHI,PHI.)
C REDUCE INITIAL MASS TO REFLECT ESTIMATED THIRD BURN
IF (IBACK.EQ.0) GO TO 1000
AMSAV=AMOUT*(TSAV-TSSAV)

VEH(1,1)=VEH(1,1)+AMSAV

1000 CONTINUE

C SET-UP END CONDITIONS FOR PLANE CHANGE
CALL DVALS(XT,QU,PTV,TN,-1)
DO 261 I=1,6

261 CC(I)=DD(I)
C *

DO 289 I=1,6

289 QQ(I)=Q(I)/UMAG
CALL GUDGE(I,0)
CALL AUXOUT
CALL CKSET(CK)
DQUMAX=0.0
DTMAX=0.0

IPRINT=0
DO 280 I=1,6

280 DD(I)=DD(I)+Q(I)*CK
TML=TIMS(I)*Q(I)*CK
IF (DABS(DD(I))+DTM*0.01) DQUMAX=DD(I)

288 AM=AMU
IF (IBACK.EQ.1) AMO=VEH(1,1)+(TIMS(6)-TIMS(5)+TIMS(4)-TIMS(3))

AM=AMU

1 +TIMS(2)-TIMS(1)) AMOU
AMU=AMU+AMOU

IF (DTMAX.LT.0.05*DQUMAX) DQUMAX=DTM=0.01*AMU

1, DTM=(AMU-AMOU)*TTVEH(1,1)*1.0-6) GO TO 290

290 CONTINUE

C DID NOT CONVERGE IN 30 ITERATIONS. DUMP VARIABLES AND STOP
WRITE(10OUTPT,3006)

3006 FORMAT('OUT-OF-PLANE MISSION DID NOT CONVERGE IN 30 ITERATIONS. STOP')

STOP

291 CONTINUE
WRITE(10OUTPT,3212) ANGLE,ITER

3212 FORMAT('ANGLE DID NOT CONVERGE IN 30 ITERATIONS.')

CGO TO 291

C CHANGE BACK TO 3-BURN IF NECESSARY.
IF (NBOURS.NE.3) GO TO 290

292 TIMES(I)=TIMEX(I+1)

TIMES(5)=TSAV
TIMES(6)=TSAV

DO 293 I=1,3

293 XT(I+3)=TSAV(I+3)

-293 XT(I)=TSAV(I)
FILE: MAIN
FORTRAN 77

CAMBRIDGE MONITOR SYSTEM

IT=TIMSAV
RELINC=0.0
C ADD MASSES TO INITIAL MASSES IF BACKWARDS MISSION
IR(IBACK+2)+1)=VEH(1,1)+-ARNSAV
ITURN=2
C ROTATE XT IF FORWARD MISSION
IF(IBACK+2)=0 TO 200
DO 294 I=1,3
XT(I)=XT(I)
-294 XT(I+3)=VT(I)
CALL ROTATE(XT,PROLL,ANGLE,DEGCON)
GO TO 203
C TEST TO SELECT IF MISSION TURNAROUND IS NECESSARY
295 CONTINUE
IF(IBACK+2)=0 TO 200
C TURNAROUND MISSION AND RECONVERGE:
C
C SET-UP TIMES ARRAY, INITIAL AND XT
WRITE(10,OUTPR+9001)
4001 FORMAT(' TURNAROUND MISSION')
B1=TIMES(0)+TIMES(5)
B2=TIMES(4)+TIMES(5)
B3=TIMES(3)+TIMES(4)
B4=TIMES(2)+TIMES(3)
TALIGN=TT-TIMMLS(0)
TTLGN=TIMLS(0)-T0
T0= Talign+TIMLS
TIMMLS(1)=T0
TIMMLS(2)=TIMMLS(1)+1
TIMMLS(3)=TIMMLS(3)+1
TIMMLS(4)=TIMMLS(4)+2
TIMMLS(5)=TIMMLS(5)+3
TT=T0+TTLGN
AMDT=AMDT
VEH(1,2)=VEH(1,2)
DO 310 I=1,3
QO(I)=Q(I)
QO(I+3)=Q(I+3)
AT(I)=XT(I)
310 XT(I+3)=XT(I+3)
AM0=VEH(1,1)
C COAST TUGSAV TO ALIGN WITH TIMES(1)
NO=1
CALL COAST(TUGSAV,QDUM,TALIGN,XU,QDUM,PH1,PH1)
C RECONVERGE WITH FORWARD MISSION
IBACK=0
RELINC=0.0
ITURN=1
GO TO 203
260 CONTINUE
C VERIFY FINAL ORBIT
DO 262 I=1,3
RTA(1)=X(1)

VTA(1)=X(1+3)

CALL ELMNTS(RTA,VTA,AA,LA,HA,PGA,TAU)

WRITE(OUTPT,3217)RTA,VTA,AA,LA,HA,PGA,TAU

3215 FORMATT ORBIT ACTUALLY ACHIEVED:

1. POSITION=,JO14.6/; VELOCITY=.JO14.6/.

2. SEMI-MAJOR-AXIS=,F10.2/; ECCENTRICITY=,F5.6/.

3. H-VECTOR=,JO14.6/; PERIGEE=.JO14.6/; PERIOD=,10.2)

BURI=TIMES(2)-TIMES(1)

BUR2=TIMES(4)-TIMES(2)

BUR3=TIMES(6)-TIMES(4)

COAS=TIMES(3)-Tu

COAS1=TIMES(1)-T0

COAS2=TIMES(5)-TIMES(2)

COAS3=TIMES(9)-TIMES(4)

IF(NBURNS.EQ.2)WRITE(OUTPT,3217)COAS0,BUR2,COAS3,BUR...

3217 FORMATT CONVERGED COASTS AND BURNS FOR 2-BURN MISSON:

1. INITIAL COAST=,F10.2/; FIRST BURN=,F10.2/; SECOND COAST=MAI...

2=,F10.2/; FINAL BURN=,F10.2)

C IF(NBURNS.EQ.3)WRITE(OUTPT,3218)COAS1,BUR1,COAS2,BUR...}

3218 FORMATT CONVERGED COASTS AND BURNS FOR 3-BURN MISSON:

1. INITIAL COAST=,F10.2/; FIRST BURN=,F10.2/; SECOND COAST=,F10.2/;

THIRD COAST=,F10.2/)

C C-GUIDANCE SECTION

C

C CHECK IF GUIDANCE DESIRED

IF(NUTARG.EQ.-1)STOP

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

IF(DAUS(TI-TIMS(1))LT.TIMRDN) GO TO 405

C 405

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)

C CHECK IF GUIDANCE DESIRED.

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C 3-BURN MISSION. REMOVE ANY INITIAL COAST.

CALL COAST(XO,00,TIMS(1)-Tu,XO,00,PHI,PH1)

TO=TIMS(1)
\[
\begin{align*}
&\text{FILE: MAIN FORTRAN P1} \\
&\text{CAMBRIDGE MONITOR SYSTEM} \\
&\text{MAIN - 21} \\
&406 \quad \text{XT(1)=XTS(1)} \\
&\quad \text{U1=TIMES(2)-TIMES(1)} \\
&\quad \text{C1=TIMES(3)-TIMES(2)} \\
&\quad \text{B2=TIMES(4)-TIMES(3)} \\
&\quad \text{C2=TIMES(5)-TIMES(4)} \\
&\quad \text{B3=TIMES(6)-TIMES(5)} \\
&\quad \text{Tl=TIMES(1)} \\
&\quad \text{T2=TIMES(2)-TIMES(1)} \\
&\quad \text{B2=TIMES(4)-TIMES(3)} \\
&\quad \text{C2=TIMES(5)-TIMES(4)} \\
&\quad \text{B3=TIMES(6)-TIMES(5)} \\
&\quad \text{Tl=TIMES(1)} \\
&\quad \text{T2=TIMES(2)-TIMES(1)} \\
&\quad \text{B2=TIMES(4)-TIMES(3)} \\
&\quad \text{C2=TIMES(5)-TIMES(4)} \\
&\quad \text{B3=TIMES(6)-TIMES(5)} \\
&\quad \text{TIMES(1)=2000.} \\
&\quad \text{TIMES(2)=TIMES(1)+B1} \\
&\quad \text{TIMES(3)=TIMES(2)+C1} \\
&\quad \text{TIMES(4)=TIMES(3)+B2} \\
&\quad \text{TIMES(5)=TIMES(4)+C2} \\
&\quad \text{TIMES(6)=TIMES(5)+B3} \\
&\quad \text{TT=TIMES(6)} \\
&\quad \text{BURNT=ABS(TIMES(6)-TIMES(5)+TIMES(4)-TIMES(3)}+ \\
&\quad \text{TIMES(2)-TIMES(1))} \\
&\quad \text{AMO=VEH(1,1)-BURNT*VEH(1,2) -} \\
&\quad \text{AMDO=AMDOU} \\
&\quad \text{VEH(1,2)=VEH(1,2) -} \\
&\quad \text{C SET UP INITIAL COAST FOR BACKWARDS MISSION.} \\
&\quad \text{NO=0} \\
&\quad \text{CALL-COAST(XG,GO,-SO,...XG,GO,PHI,PHI)} \\
&\quad \text{T0=TO-SOO.} \\
&\quad \text{C SET UP MONTE CARLO RUNS.} \\
&\quad \text{C} \\
&\quad \text{C CALCULATE END CONDITIONS} \\
&\quad \text{410 CALL-VALS(XT,QUUM,PHI,TV,-1)} \\
&\quad \text{DO 411 I=1,6} \\
&\quad \text{411 CC(I)=DD(I)} \\
&\quad \text{C SAVE INITIAL CONDITIONS FOR NEXT MONTE CARLO RUN.} \\
&\quad \text{DO 415 I=1,6} \\
&\quad \text{XOS(I)=XO(I)} \\
&\quad \text{QOS(I)=QO(I)} \\
&\quad \text{VEHS(I)=VEH(I,1)} \\
&\quad \text{TIMES(I)=TIMES(I)} \\
&\quad \text{XTS(I)=XT(I)} \\
&\quad \text{415 CCS(1)=CC(I)} \\
&\quad \text{VEHS(7)=VEH(1,7)} \\
&\quad \text{T0S=T0} \\
&\quad \text{TTS=T1} \\
&\quad \text{AMOS=AMOS} \\
&\quad \text{I6ONS=16000.} \\
&\quad \text{C LOOP FOR MONTE CARLO RUNS.} \\
&\quad \text{IPRINT=1} \\
&\quad \text{DO 420 MONT=1,MCARLO} \\
&\quad \text{IUPDAT=0} \\
&\quad \text{C RESTORE VARIABLES.} \\
&\quad \text{DO 425 J=1,6} \\
&\quad \text{VEH(I,1)=VEHS(I)} \\
&\quad \text{XO(I)=XOS(I)} \\
&\quad \text{QO(I)=QOS(I)} \\
&\quad \text{XT(I)=XTS(I)} \\
&\quad \text{QT(I)=QTS(I)} \\
&\quad \text{TIMES(I)=TIMES(I)} \\
\end{align*}
\]
FILE: MAIN  FORTRAN P1  CAMBRIDGE MONITOR SYSTEM

425  CC(1)=CCS(1)
    VEH(1,7)=VEHS(7)
    DD 426  I=2,11
    DD 426  J=1,7

426  VEH(1,J)=0.0
    ISOUND=ISOUNDS
    TRUCLS=VEH(1+1)
    TCLOCK=0.0
    TACCUM=0.0
    AMO=AMOS
    TO=TOP
    TT=TIM

    IF(NBURNS.EQ.3) CALL RCACB(4,BOUND,TOINT)

420  IF(NBURNS.EQ.2) CALL CCUB
    CALL STATIS(MCA3L)
    STOP
    END
Subroutine AUXOUT

A. Purpose

AUXOUT prints the status of the convergence, from the most recent call to GUIDE.

B. Input/Output Definition

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>X(I)  ( I=1,6 )</td>
<td>( \bar{x} )</td>
<td>Vehicle final state</td>
</tr>
<tr>
<td>XTF(I)  ( I=1,6 )</td>
<td>( \bar{x}_T )</td>
<td>Target state at same time as above</td>
</tr>
<tr>
<td>TIMES(I)  ( I=1,6 )</td>
<td>-</td>
<td>Array of times at ends of coast and burn arcs</td>
</tr>
<tr>
<td>Q0(I)  ( I=1,6 )</td>
<td>( \bar{q}_0 )</td>
<td>Costate at start of mission</td>
</tr>
<tr>
<td>DTIMES(I)  ( I=1,6 )</td>
<td>( \Delta \xi )</td>
<td>Requested corrections to TIMES</td>
</tr>
<tr>
<td>DQ0(I)  ( I=1,6 )</td>
<td>( \Delta \bar{q}_0 )</td>
<td>Requested corrections to costate Q0</td>
</tr>
<tr>
<td>IOUTPT</td>
<td>-</td>
<td>Output device number</td>
</tr>
</tbody>
</table>

Output Parameter

None.

C. Method of Computation

The only variable calculated is the estimate of the total burn remaining

\[
\text{COST} = \left| (\text{TIMES}(2) - \text{TIMES}(1)) + (\text{TIMES}(4) - \text{TIMES}(3)) \\
+ (\text{TIMES}(6) - \text{TIMES}(5)) \right|
\]
SUBROUTINE AUXOUT

IMPLICIT REALS(A-H,O-Z)
COMMON /GIDOIN/XI(6),TT,XU(5),TO,AMO,VEH(10,7),OO(6),TIMES(6),C(6)
COMMON /ADOLVIC/IOUTPUT
COMMON /GIDOUT/DUG(0),DTIMES(6),L(12,12),OC(12),X(6),Y(6)
COMMON /ADOLVOUT/XTF(0),DELTC(6)

WRITE(IOUTPUT,1)X,XIF
1 FORMAT(//,'X(OBTAINED)='',6E14.6','X(DESIRED)='',6E14.6)
COST=DABS(TIMES(0)+TIMES(1)+TIMES(2)+TIMES(3)+TIMES(4))
WRITE(IOUTPUT,2)COST
2 FORMAT(//,'REMAINING SUMP='',14.6')
WRITE(IOUTPUT,3)DUU,DUG,DTIMES
3 FORMAT(1X,'DUU='',6E16.8','DUG='',6E16.8','DTIMES='',6E16.8',')/
RETURN
END
Subroutine BCBCB

A. Purpose

Subroutine BCBCB is used during guidance mode to take the vehicle through the first burn of a 3-burn mission. It operates in either a backwards mode (outbound mission) or a normal mode, and is called by MAIN at the start of each Monte-Carlo run. It in turn calls FORWRD at regular intervals until the end of the first burn, at which time it changes mode (if backwards) to the normal mode and calls CBCB to handle the remaining coasts and burns. BCBCB also modifies the TIMES array on each cycle to reflect the fact that part of the first burn has occurred, calls GUIDE to reconverge the mission with the new (possibly perturbed) vehicle state, and adds the resulting corrections to the TIMES array and costate. On the indicated cycles (IOUT = 1 or next-to-last cycle in the burn arc), subroutine NAVOUT is called to collect the Monte-Carlo statistics. On the last cycle in the burn arc, the call to GUIDE (and the addition of the corrections to TIMES and Q0) is skipped and CBCB is called with an initial step time of zero.

B. Input/Output Definition

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
</table>
| IBOUND          | -      | 0 - outbound mission (implies backwards mode)  
                 |        | 1 - inbound mission |
| TØINT           | -      | In backwards mode, the actual value of TØ |
| TRUEMS          | -      | Vehicle mass before start of burn  
<pre><code>             |        | (normally equivalent to AMØ except when in backwards mode) |
</code></pre>
<p>| XT(I) I=1,6     | xT     | Vehicle state in backwards mode |
| TT              | tT     | Time at start of first burn in outbound case |
| TØ              | tØ     | Time at start of first burn in inbound case |
| IOUTPT          | -      | Output device number |</p>
<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
</table>
| DTYPE(I)        | I=1,2  | I=1; normal guidance step size during burn  
               |        | I=2; not used in BCBCB               |
| TIMES(I)        | I=1,6  | Vector of times at end of each leg (or start of each leg in backwards mode) |

<table>
<thead>
<tr>
<th>Output Parameter</th>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPRINT</td>
<td></td>
<td>Always 0; shuts off printout resulting from calls to GUIDE after first step in first Monte-Carlo run</td>
</tr>
<tr>
<td>MODE</td>
<td></td>
<td>Always 0; restores mission to free-time rendezvous (backwards mode only)</td>
</tr>
<tr>
<td>AMO</td>
<td></td>
<td>Mass at end of first burn</td>
</tr>
<tr>
<td>TIMES(I)</td>
<td>I=1,6</td>
<td>Vector of times at end of arcs, with first burn deleted from the vector (TIMES(1)=0, TIMES(2)=0) and, in backwards mode, the vector restored to its normal form</td>
</tr>
<tr>
<td>Q0(I)</td>
<td>I=1,6</td>
<td>q₀</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Costate at end of first burn</td>
</tr>
<tr>
<td>T0</td>
<td></td>
<td>Time at end of first burn</td>
</tr>
<tr>
<td>TT</td>
<td></td>
<td>Time for which target state is valid</td>
</tr>
<tr>
<td>CC(I)</td>
<td>I=1,6</td>
<td>New end conditions for target (backwards mode only)</td>
</tr>
</tbody>
</table>

C. **Method of Computation**

After zeroing the time accumulator (used to determine when Monte-Carlo statistics are to be collected), saving the vehicle initial mass, and initializing several control integers, BCBCB branches to one of two separate sections of code, depending on whether a normal 3-burn mission is being run. In either case, it is assumed that the first burn begins immediately, with no initial coast.
In the backwards mode, the TIMES array as supplied to BCBCB is already reversed and ready to use, as are T0 and TT. The weights are set to 1.0 since the backwards mode works best with hard constraints and mode is set to 3 to change to a fixed time rendezvous. Subroutine FORWRD is then called every DTYPE(1) seconds during the first burn, with the exception of the last two steps which are approximately equal to each other and less than DTYPE(1)/2, and TIMES(6) is updated. Each time the print accumulator exceeds PTB, subroutine NAVOUT is called to collect Monte Carlo statistics, and the accumulator is reset to zero. M/C statistics are also collected on the next-to-last step in the burn arc. Also, following each call to FORWRD, except the last, GUIDE is called and the corrections are added to Q0 and TIMES, Q0 is maintained at unit magnitude, and the estimate of vehicle final mass is recalculated from the mass rate, current mass, and requested changes in the burn times. On the next-to-last call to FORWRD (NLAST=1), subroutine GUIDE is called repeatedly (with no changes in vehicle state) until the miss in final position is less than 1 kilometer. On the last call to FORWRD, GUIDE is called but no changes are permitted in the TIMES array and Q0 and the weights are restored to their original value. In addition, the flag is set to add the navigation update corrections to vehicle state on the very first call to FORWRD from CBCB. The mission is then turned around to normal mode, and the target end conditions reevaluated. Finally, subroutine CBCB is called to handle the remaining coasts and burns.

In normal mode, BCBCB works in much the same way, except that the states and TIMES array are not reversed, and T0 is updated rather than TIMES(6).
ZERO ACCUMULATOR (FOR PRINTOUT)
STORE INITIAL MASS
SET NOMINAL STEP SIZE (DTYPE(1))
STORE TIME AT END OF FIRST BURN (BACKWARDS MODE ONLY)
SET STEP INDICATOR VARIABLES

SAVE WEIGHTS; FORCE HARD CONSTRAINTS (WEIGHTS=1.0)

BACKWARDS MODE (IBOUND=0)

SET STEP SIZE FOR THIS STEP
CALL FORWARD, INCREMENT TIMES(1)
SET NAVIGATIONAL UPDATE FLAG

CALL GUIDE, INCREMENT TIMES(6)
CALL FORWARD, INCREMENT TIMES(1)
SET NAVIGATIONAL UPDATE FLAG

CALL GUIDE AND CKSET, ADD CORRECTIONS TO TIMES AND Q0
CALL GUIDE, TURN MISSION AROUND; SET TO SOFT CONSTRAINTS (WEIGHTS=1.0);
RESET INITIAL MASS AM0; REEVALUATE END CONDITIONS
SET NAVIGATIONAL UPDATE FLAG
CALL CBCB AND RETURN

LAST STEP IN BURN LEG

CALL GUIDE AND CKSET, ADD CORRECTIONS TO TIMES AND Q0
CALL GUIDE, CALL CKSET ADD CORRECTIONS TO TIMES AND Q0

COLLECT M-C STATISTICS

COLLECT M-C STATISTICS

COLLECT M-C STATISTICS

CALL GUIDE, QTi = q0j i = 1,6

N = N + 1
MISS = END CONDITION ERROR
MISS > 1 KM AND N < 20

B

A

C

N = N + 1
MISS = END CONDITION ERROR
MISS > 1 KM AND N < 20

A

C
SUBROUTINE BCB010 (BOUND, JOINT)
C SUBROUTINE TO TAKE THE TUG THROUGH THE INITIAL BURN OF A 3-BURN
C MISSION IN GUIDANCE MODE. WORKS FOR BOTH INBOUND AND OUTBOUND MISSIONS.
---IMPLICIT REAL*8(A-H,O-Z)---
COMMON /VLOUT/XF(6)
COMMON /UPDATE/UPDAT
COMMON /PHYSX/RESTR,DUM1,DUM2,DUM3
COMMON /ONLINE/IPRINT
COMMON /COUNTER/MODE
COMMON /CNAV/TWOL,PNAV,TACCUM,OUT
COMMON /GIDIN/XT(6),IT,AX(6),TO,AMP,VEH(10,7),Q0(6),TIMES(6),CC(6)
COMMON /IDOUT/DOU(6),UTIMES(6),C(12,12),UC(12),X(6),Q(6),Z(12,12)
COMMON /CINDEX/XARC,IMAX,JMAX1,JMAX2,JLAST,ND,QINT
COMMON /CJ/STRT(3),UTYPE(2),UFAC
COMMON /CMSYS/ST(6)
COMMON /COUT/ST(6)
DIMENSION ATS(6),ATS(6)
C ZERO TIME ACCUMULATOR.
TACCUM=0.0
IDUF=0
C SAVE INITIAL MASS
TULMS=TRUEMS
C SET NOMINAL STEP SIZE IN BURN
DT=UTYPE(1)
C SAVE TIME AT END OF LAST BURN (=TIME AT START OF 1ST BURN...)
IF (BOUND.EQ.0) TDSAVE=TIMES(6)
C SET VARIABLES
M1=0
LAST=0
NPINT=0
IF (BOUND.EQ.1) GO TO 100
C 3-BURN-OUTBOUND MISSION, NO IN-BACKWARDS MODE.
C
C SET WEIGHTS.
C WEIGHT(1)=9 REFLECTS HARD CONSTRAINTS ON BACKWARDS BURN.
CT 1 1 = 1.0
ATS(1)=WEIGHT(1)
1 WEIGHT(1)=1.0
NC=0
C START MAIN GUIDANCE LOOP FOR FIRST BURN.
3  IF (NLAST.EQ.1) LAST=1
IF (DABS(TIMES(6)-TIMES(5)) LE .2.*UTYPE(1)) NLAST=1
IF (NLAST.EQ.0) .GO TO 1
IF (NLAST.EQ.1) DT=DABS(TIMES(6)-TIMES(5))/2.
IF (LAST.EQ.1) DT=DABS(TIMES(6)-TIMES(5))
CALL FORWARD(0,-DT,0)
TIMES(6)=TIMES(6)-DT
IF (LAST.EQ.1) GO TO 0
33 NOP=1
FILE: BCGCH FORTRAN II

CAMBRIDGE MONITOR SYSTEM

BC00056

BC00057

BC00058

BC00059

BC00060

BC00061

BC00062

BC00063

BC00064

BC00065

BC00066

BC00067

BC00068

BC00069

BC00070

BC00071

BC00072

BC00073

BC00074

BC00075

BC00076

BC00077

BC00078

BC00079

BC00080

BC00081

BC00082

BC00083

BC00084

BC00085

BC00086

BC00087

BC00088

BC00089

BC00090

BC00091

BC00092

BC00093

BC00094

BC00095

BC00096

BC00097

BC00098

BC00099

BC00100

BC00101

BC00102

BC00103

BC00104

BC00105

BC00106

BC00107

BC00108

BC00109

BC00110
FILE: BCBCB FORTRAN P1 BCBCB - 7

100 VEH(1+2)=VEH(1,2)  
AMG=TUTLMS-D1+VEH(1,2)  
DO 8 I=1,3  
8 Q0(I)=O(I)  
Q0(I+3)=-O(I+3)  
XTEMP1=XT(1)  
XTEMP2=XT(1+3)  
XT(1)=X0(1)  
XT(I+3)=-X0(I+3)  
X0(I)=XTEMP1  
9 X0(I+3)=-XTEMP2  
C.SET END CONDITIONS.  
CALL-BVALS(XT,GO,PIV,TIV,=1)  
DO 9 I=1,6  
CC(I)=DD(I)  
CALL-CBCB  
RETURN  
C  
C 3-BURN INBOUND MISSION (FORWARD MODE).  
C  
C.SET U0 TO UNIT MAGNITUDE.  
100 QMAG=0.5*SQRT(QO(1)**2+QO(2)**2+QO(3)**2)  
DU=0.1-1=1.6  
101 QO(I)=QO(I)/QMAG  
102 IF(NLAST+3P+1) LAST=1  
IF(TIMES(2)-TIMES(1)*LE.2*4*TYPE(1)-NLAST=1  
IF(NLAST+L4+1) DT=(TIMES(2)-TIMES(1))/2.  
IF(N4+L4+1) DT=TIMES(2)-TIMES(1)  
CALL-FORWD(0,0,1,1)  
IF(LAST.EQ.1) UPDAT=1  
TIMES(1)=TIMES(1)+DT  
IF(LAST.EQ.1) GO TO 106  
NOP=1  
MODEL=0  
CALL-GUIDE(G,0)  
IPRINT=0  
CK=-1.0  
CALL-CKSET(CK)  
C.ADD CORRECTIONS TO GO, TIMES.  
DO 103 I=1,9  
QO(I)=QO(I)+CK*AMG(I)  
103 TIMES(I)=TIMES(I)+DTIMES(I)*CK  
IF(NLAST.EQ.1.0 OR 1001.EQ.1) NPOINT=NPOINT+1  
IF(NLAST.EQ.1.OR 1001.EQ.1) CALL-RAVOUT(1,NPOINT)  
GO TO 106  
106 CALL-CBCB  
RETURN  
END
Subroutine BVAL5

A. Purpose

The new BVAL5 subroutine replaces the BVAL5 and BVAL6 subroutines in GUIDE 71/6. It calculates the miss in end conditions and partial derivatives of the end conditions for either hard or soft constraint missions with up to six end condition constraints and free or fixed terminal time. The subroutine can also be called (for example, for initializing desired \( \mathbf{h} \) and \( \mathbf{e} \)) with \( \text{NBVAL} = -1 \) to calculate the three components of the angular momentum vector \( \mathbf{h} \) and the three components of the eccentricity vector \( \mathbf{e} \), pointing toward peri-gee with magnitude of eccentricity. BVAL5 calls the subroutine COAST to obtain target state \( \mathbf{XTF} \) at the end of the mission, TIMES(6).

B. Input/Output Definition

<table>
<thead>
<tr>
<th>Input Parameter</th>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>XF(I) for I=1 to 3</td>
<td>( r )</td>
<td>Final vehicle position for I=4 to 6</td>
</tr>
<tr>
<td>QF(I) for I=1 to 3</td>
<td>( u )</td>
<td>Final control vector for I=4 to 6</td>
</tr>
<tr>
<td>PTV(I) for I=1 to 12</td>
<td>( \frac{\partial T_V}{\partial y} )</td>
<td>Partial derivatives of ( T_V ) with respect to ( y=(r^T, v^T, u^T, \dot{u})T ) evaluated in BUZZ</td>
</tr>
<tr>
<td>TV</td>
<td>( T_V )</td>
<td>Phasing transversality condition ( \mu(r^T\dot{u})/</td>
</tr>
<tr>
<td>NBVAL</td>
<td>-</td>
<td>Flag parameter indicating whether or not miss in end conditions and their derivatives are to be calculated</td>
</tr>
<tr>
<td>UK</td>
<td>( \mu )</td>
<td>Gravitational constant</td>
</tr>
<tr>
<td>C(I) for I=1 to 3</td>
<td>( h_d )</td>
<td>Desired orbital angular velocity for I=4 to 6</td>
</tr>
<tr>
<td>Input Parameter</td>
<td>Symbol</td>
<td>Definition</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$Z(I,J)$</td>
<td></td>
<td>Partial derivatives of final $y = (r^T,v^T,u^T,u^T)^T$ with respect to</td>
</tr>
<tr>
<td>$I=1$ to $12$</td>
<td>$J=1$ to $JMAX1$</td>
<td>$JMAX1$ independent variables</td>
</tr>
<tr>
<td>$JMAX1$</td>
<td>$JMAX$</td>
<td>Number of independent variables</td>
</tr>
<tr>
<td>$JLAST$</td>
<td>$-$</td>
<td>$JMAX1 + 1$</td>
</tr>
<tr>
<td>$MODE$</td>
<td>$-$</td>
<td>Flag to denote fixed terminal time mission</td>
</tr>
<tr>
<td>$TIMES(6)$</td>
<td>$t_f$</td>
<td>Terminal time</td>
</tr>
<tr>
<td>$TT$</td>
<td>$T$</td>
<td>Target epoch (time at which $x_T(T)$ is valid)</td>
</tr>
<tr>
<td>$XTF(I)$ for $I=1$ to $6$</td>
<td></td>
<td>Target state at time $T$</td>
</tr>
<tr>
<td>$WT(I)$ for $I=1$ to $6$</td>
<td></td>
<td>Diagonal components of weighting matrix ranging from 0.0 to 1.0. $W(I)=1.0$ if the $I$th end condition is a hard constraint. $W(I)=0.0$ if the $I$th end condition is unconstrained.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output Parameter</th>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D(I)$ for $I=1$ to $3$</td>
<td>$h$</td>
<td>Orbital angular velocity</td>
</tr>
<tr>
<td>for $I=4$ to $6$</td>
<td>$e$</td>
<td>Eccentricity vector</td>
</tr>
<tr>
<td>$DELTC(I)$ for $I=1$ to $6$</td>
<td>$\Delta c$</td>
<td>Miss in end conditions</td>
</tr>
<tr>
<td>$XTF(I)$</td>
<td>$x_T(t_f)$</td>
<td>Target state at $t_f$</td>
</tr>
<tr>
<td>$DC(I)$ for $I=1$ to $6$</td>
<td>$DC$</td>
<td>Weighted combination of transversality conditions and misses in end conditions</td>
</tr>
<tr>
<td>$E(I,J)$ for $I=1,6$</td>
<td>$J=1,JMAX1$</td>
<td>Partial derivatives of $S$ with respect to independent variables</td>
</tr>
</tbody>
</table>
C. Method of Computation

Components of the orbital constants $h$ and $v$ are calculated using the expressions

\[
\begin{align*}
    h &= r x v \\
    e &= -\left\{ \frac{r}{|r|} + \frac{(r x v) x v}{\mu} \right\}
\end{align*}
\]  

(1)

The subroutine COAST is called to propagate $x_T(T)$ from $T$ to $t_f$. If a fixed terminal time mission is being flown (indicated by MODE=3), the parameters $JMAX1$ and $JLAST$ are each decremented by 1. This has the effect of eliminating the dependent variable corresponding to the change in the transversality variable across the last burn arc. It also has the effect of eliminating terminal time as an independent variable and of eliminating the appropriate row and column of the $E$ matrix.

The end condition miss vector $\Delta c$ is composed of scaled components of $\Delta h$, $\Delta e$ and $\Delta r$ lying along the $R$ and $K = \frac{H x R}{|H|}$ vectors and a scaled miss in orbital energy $E$.

\[
\Delta c = \begin{pmatrix}
\Delta h & T/K & |H| \\
\Delta E & \left[ R^2 \right] & W \\
\Delta e & T_K \\
\Delta h & T/R & |H| \\
\Delta e & T_R \\
\Delta r & T_K
\end{pmatrix}
\]

(2)

Here, $\Delta h = h_{target} - h$

$\Delta e = e_{target} - e$

and $\Delta c$ is evaluated at $R = r$ and $H = h$. This constraint formulation has excellent convergence properties for well posed orbit injection and rendezvous missions of all geometries. All components of $\Delta c$ are scaled to have the same units as $r$. 
In order to avoid stability problems during the last leg of a mission, the problem is formulated so that a weighted combination of fuel use and miss in end conditions is minimized. The cost functional

\[ J = \int_{t_0}^{f} |\dot{m}| dt + \frac{1}{2} \Delta c^T \Delta c \]  

(3)

is minimized. Here \( W \) is a 6\times6 diagonal weighting matrix and \( |\dot{m}| \) is the rate of fuel consumption during burns. Minimizing this cost functional is equivalent to satisfying the costate equations

\[ p_f = \left( \frac{3\Delta c}{\Delta x} \right)^T W \Delta c \]  

(4)

where \( p_f = (u^T, -u^T) \) or equivalently the equations

\[ (I-w)B^T p_f = w \Delta c \bigg|_{X=x} \]  

(5)

where \( B \) is a nonsingular matrix such that

\[ B(x) \left( \frac{3\Delta c}{\Delta x} \right)^T \bigg|_{X=x} = I \]  

(6)

and \( w \) is a diagonal weighting matrix with ith diagonal component \( w_i \) related to ith diagonal component \( W_i \) of \( W \) by

\[ w_i = \frac{W_i}{1+W_i} \]  

(7)

Whenever an end condition such as phasing is unconstrained, the corresponding diagonal component of \( w \) is zero. For hard constraints, \( w=I \), the vector \( B^T p_f \) is composed of six scaled transversality conditions. The sixth component of \( B^T p_f \) is \( |r|T_v/|h| \) where \( T_v \) is the phasing transversality condition calculated in BUZZ. The components of \( (1-w)B^T p_f \) given in terms of multiplying coefficients \( C_{ij} \) defined in the code are
\[(I-w)B^T_p = \begin{pmatrix}
C_{11}(h^T_u) \\
-C_{21}(r^T_u) - C_{22}(v^T_u) \\
-C_{31}(r^T_u) + C_{32}(r^T_u) - C_{33}(v^T_u) \\
-C_{41}(h^T_u) + C_{42}(h^T_u) \\
C_{51}(r^T_u) + C_{52}(r^T_u) + C_{53}(v^T_u) \\
C_{61}T_v
\end{pmatrix}\] (8)

The DC vector calculated in BVAL5 corresponds to the miss in satisfying Eq. (5)

\[DC = wAc - (I-w)B^T_p\] (9)

Partial derivatives of DC with respect to the independent variables $\zeta$ are calculated via the chain rule.

\[\left(\frac{\partial DC}{\partial \zeta}\right) = \left(\frac{\partial DC}{\partial x}\right) \left(\frac{\partial x}{\partial \zeta}\right) + \left(\frac{\partial DC}{\partial q}\right) \left(\frac{\partial q}{\partial \zeta}\right)\] (10)

The G matrix in BVAL5 corresponds to $\left(\frac{DC}{x}\right)$ neglecting derivatives of scaling factors. From Eq. (8), it can be seen that the second term in Eq. (10) is efficiently evaluated by calculating terms such as $h^T\left(\frac{\partial u}{\partial \zeta}\right)$, $r^T\left(\frac{\partial q}{\partial \zeta}\right)$ and multiplying by the appropriate $C_{ij}$ coefficients.
FILE: CASVUJ FORTRAN PI

BVAL5-6

BVAL5 - 6

THIS FILE CONTAINS SUBROUTINES BVAL5, SOLVE, AND BVAL4 TO BE USED AS A PART OF GUIDL 71/5 AND GUIDL 71/6.

SUBROUTINE BVAL5 CALCULATES D (ANGULAR MOMENTUM AND ECCENTRICITY VECTORS) FROM INPUT STATE XF. IF NVAL DOES NOT EQUAL -1, THEN THE DC VECTOR (WEIGHTED COMBINATIONS OF TRANSVERSALITY AND MISS IN END CONDITIONS) AND THE E MATRIX (PARTIAL DERIVATIVES OF DC WITH RESPECT TO THE JMXX INDEPENDENT VARIABLES) ARE ALSO CALCULATED.

SUBROUTINE BVAL5(XF, QF, RTV, TV, NVAL)

IMPLICIT REAL*(A-H,O-Z)

DIMENSION XF(5), DOT(5), R(3), V(3), DUM2(6)

COMMON /BVLOUT/XF(5), UELTC(0)

COMMON /GID10/XF(5), U10(6), TC, AM, VEH(10,7), UC(0),

1 TIMES(6), UC(0).

COMMON /PHYS/U10(6), U10(6), TC, AM, VEH(10,7), UC(0),

1- DUM4(4), UC.

COMMON /CINDEX/NARC, IARC, JMXX, JM, JMXX, JLAST, NUP, NRKGDS

COMMON /CNODE/ NODE, TNF, TNSTOP

COMMON /CWTZ/ *T(8)

DIMENSION G(6,6), U1(3), UXU(3), UXU(3), QF(3), PTV(1, 1), X(3)

DIMENSION VAU(3), VAU(3), DUV(12), DUV(12), DUV(12), DUV(12), DUV(12)

IF (MOD(N, 3)-60 .GE. 10) THEN

JMXX=JMXX+1

JLAST=JLAST-1

E(JMXX,13)=0.

I NO=1

SUBROUTINE QUASI IS CALLED TO PROPAGATE TARGET STATE TO FINAL TIME SO THAT THE PHASING MISS COMPONENT IN DC(6) CAN BE CALCULATED.

CALL QUASI(XF, DUM2, TIMES(6)-TT, XT, DUM2, DUM1, DUM1)

DO 2 I=1,3

R(I)=XF(I)

U(I)=QF(I)

UD(I)=QF(I+3)

V(I)=XF(I+3)

R2=U(I)*R(I)+R(I)+R(I)+R(I)+R(I)+R(I)

RM=DUM(1,3)

V2=V(I)+V(I)+V(I)+V(I)+V(I)

RTV(1)=V(I)+V(I)+V(I)+V(I)+V(I)+V(I)

RTV2=RTV/UK

RTV2=RTV*RTV

RTV2=RTV*UK

CALCULATE ANGULAR MOMENTUM VECTOR H.

D(I)=R(I)*V(I)-((I)*V(I)

D(I)=R(I)*V(I)-((I)*V(I)

D(I)=R(I)*V(I)-((I)*V(I)

BVA0110

BVA00020

BVA00030

BVA00040

BVA00050

BVA00060

BVA00070

BVA00080

BVA00090

BVA00100

BVA00110

BVA00120

BVA00130

BVA00140

BVA00150

BVA00160

BVA00170

BVA00180

BVA00190

BVA00200

BVA00210

BVA00220

BVA00230

BVA00240

BVA00250

BVA00260

BVA00270

BVA00280

BVA00290

BVA00300

BVA00310

BVA00320

BVA00330

BVA00340

BVA00350

BVA00360

BVA00370

BVA00380

BVA00390

BVA00400

BVA00410

BVA00420

BVA00430.

BVA00440

BVA00450

BVA00460

BVA00470

BVA00480

BVA00490

BVA00500

BVA00510

BVA00520

BVA00530

BVA00540

BVA00550
C  CALCULATE ECCENTRICITY VECTOR
   DO 3 I=1,3
3   D(1+I)=-(R(1+I)*V(1)+V(I)*V(1))
   IF (N(NVAL+60+1) .EQ. 1) RETURN
   H2=V(2)/R2-RTV2
   HM=DSORT(H2)
   CF=0.5#V2-UK#RM
   DO 4 I=1,3
4   XR(I)=(V(I)/R2-TRV*V(I))/HM
   UK2=UK#R2
   UKM=UK/RM-RTV2
   H2MG=H2-RTV2
   CF=0.5#V2-UK#RM
   HC=HM*CF
   C  CALCULATE REQUIRED OUT AND CROSS PRODUCTS
   RTU=V(I+1)*U(I+1)+R(I+1)*U(I+1)
   RTDU=V(I+1)*U(I+1)+R(I+1)*U(I+1)
   VTU=V(1+I)*U(1)+V(I+1)*U(I+1)
   HTU=V(I+1)*U(I+1)+R(I+1)*U(I+1)
   HTUD=V(I+1)*U(I+1)+R(I+1)*U(I+1)
   RXU(I)=(R(I+1)+U(I+1)-R(I+1))#U(I)
   RXU(2)=(R(2+I)+U(2+I)-R(2+I))#U(2)
   RXU(3)=(R(3+I)+U(3+I)-R(3+I))#U(3)
   VXU(I)=V(1+I)-U(1+I)-U(I+1)
   VXU(2)=V(2+I)-U(2+I)-U(I+1)
   VXU(3)=V(3+I)-U(3+I)-U(I+1)
   VXUD(I)=V(I+1)-U(I+1)-U(I)
   VXUD(2)=V(I+1)-U(I+1)-U(I)
   VXUD(3)=V(I+1)-U(I+1)-U(I)
   C  CALCULATE REQUIRED COEFFICIENTS
   C  COEFFICIENTS MULTIPLY
   C  OUT PRODUCTS OF STATE AND CUSTATE IN TRANSVERSALITY CONDITIONS
   C  B- COEFFICIENTS ARE SCALAR MULTIPLIERS IN PARTIALS OF DC WITH
   C  RESPECT TO R AND V
   B11=WT(I+1)##R2
   B12=WT(I+1)##R2
   C11=(1.0-WT(I+1))##R2
   B21=WT(I+2)##RM
   B22=WT(I+2)##UK2
   C21=(1.0-WT(I+2))##UR2##CF
   C22=0.5#C21
   U3=(1.0-WT(I+3))##UK2##HC
   LFACT=U3*RTUD+VU+RNU
   B31=RT(I+3)##UR3##U3+RNU
   LFACT=U3*RT(I+3)##UR3##U3+RNU
   C31=U3*RTV
   C32=(1.0-WT(I+3))##UK2##HM
   C33=0.5#C31
   B41=WT(I+4)/HM
FILE: CASDVJ  FORTRAN  PI
CAMBRIDGE MONITOR SYSTEM

BVA01110
BVA01120
BVA01130
BVA01140
BVA01150
BVA01160
BVA01170
BVA01180
BVA01190
BVA01200
BVA01210
BVA01220
BVA01230
BVA01240
BVA01250
BVA01260
BVA01270
BVA01280
BVA01290
BVA01300
BVA01310
BVA01320
BVA01330
BVA01340
BVA01350
BVA01360
BVA01370
BVA01380
BVA01390
BVA01400
BVA01410
BVA01420
BVA01430
BVA01440
BVA01450
BVA01460
BVA01470
BVA01480
BVA01490
BVA01500
BVA01510
BVA01520
BVA01530
BVA01540
BVA01550
BVA01560
BVA01570
BVA01580
BVA01590
BVA01600
BVA01610
BVA01620
BVA01630
BVA01640
BVA01650

B42=2.*U+1.0-T(4.)*.2*HM  
B43=0.*RAC  
B41=(1.0-T(4.))/H  
B44=3.*RTV  
U5=(1.0-T(5.))*U:2/(H2*CF)  
UFAC=U5*TV(2.+U1+U1+U1)  
B51=TV(3.)*V2+(V1-U5)*RTU*R2*RM-A.5*UK*VTU*H2)  
B42=TV(3.)*RTV*U5*CF*RTU-UFAC  
B53=2.*U5*T(5.)*UR2=U5*(CF*RTU-RTU/R2)  
B54=-2.*U5*T(5.)*RTV*RTFAC  
C51=0.*RUS*H2*MG  
C52=U5*RTV*CF  
C53=0.*RUS*URMG  
B61=TV(6.)*RM  
C61=(1.0-T(6.))/LMHM  
C62=(1.0-T(6.))/UR2/HD  
C  
CALCULATE PARTIALS OF DC WITH RESPECT TO "R AND V."
C  
DO =1=1.3
G(1.1)=b11*x1(1)-c11*x1(1)  
G(1.15)=-b12*x2(1)+c12*x2(1)  
G(2.1)=b11*x1(1)+c11*x1(1)+c21*x2(1)+c21*x2(1)+UK2*RM2/RM2)*CF  
G(2.15)=b12*x2(1)+c12*x2(1)+c21*x2(1)+c21*x2(1)+UK2*RM2/RM2)/CF  
G(3.1)=c31*x1(1)-c32*x1(1)+c31*x1(1)+c31*x1(1)  
G(3.15)=c32*x2(1)-c33*x2(1)+c32*x2(1)+c32*x2(1)  
G(4.1)=d41*x0(1)+d42*x0(1)+d43*x0(1)+d43*x0(1)+d44*x0(1)+d44*x0(1)  
G(4.15)=d42*x0(1)+d43*x0(1)+d43*x0(1)+d43*x0(1)+d44*x0(1)+d44*x0(1)  
G(5.1)=e51*x0(1)+e52*x0(1)+e53*x0(1)+e53*x0(1)+e54*x0(1)  
G(5.15)=e52*x0(1)+e53*x0(1)+e53*x0(1)+e53*x0(1)+e54*x0(1)  
G(6.1)=f61*x0(1)+f62*x0(1)+f63*x0(1)+f63*x0(1)+f64*x0(1)  
G(6.15)=f62*x0(1)+f63*x0(1)+f63*x0(1)+f63*x0(1)+f64*x0(1)  
5  
SUM J=1,JMAXL  
BVA01410
BVA01420
BVA01430
BVA01440
BVA01450
BVA01460
BVA01470
BVA01480
BVA01490
BVA01500
BVA01510
BVA01520
BVA01530
BVA01540
BVA01550
BVA01560
BVA01570
BVA01580
BVA01590
BVA01600
BVA01610
BVA01620
BVA01630
BVA01640
BVA01650

C  
PATTERN OF DC WITH RESPECT TO COSTATE "JMLS."
C  
C (FIRST STEP OF CHN IN RULE)
BVA01500
BVA01510
BVA01520
BVA01530
BVA01540
BVA01550
BVA01560
BVA01570
BVA01580
BVA01590
BVA01600
BVA01610
BVA01620
BVA01630
BVA01640
BVA01650

C  
C ADD IN STATE WITH RESPECT TO STATE TIMES PARI.
BVA01560
BVA01570
BVA01580
BVA01590
BVA01600
BVA01610
BVA01620
BVA01630
BVA01640
BVA01650

C  
C OF STATE WITH RESPECT TO "INDEPENDENT VARIABLES."
BVA01500
BVA01510
BVA01520
BVA01530
BVA01540
BVA01550
BVA01560
BVA01570
BVA01580
BVA01590
BVA01600
BVA01610
BVA01620
BVA01630
BVA01640
BVA01650

C  
C CALCULATE MISS IN SOFT CONSTRAINTS.  
BVA01500
BVA01510
BVA01520
BVA01530
BVA01540
BVA01550
BVA01560
BVA01570
BVA01580
BVA01590
BVA01600
BVA01610
BVA01620
BVA01630
BVA01640
BVA01650

C  
C 1. DELTA H ALONG H CROSS R  
C 2. DELTA. ENERGY  
C 3. DELTA E ALONG H CROSS R  
C 4. DELTA E ALONG R  
C 5. DELTA E ALONG R  
BVA01500
BVA01510
BVA01520
BVA01530
BVA01540
BVA01550
BVA01560
BVA01570
BVA01580
BVA01590
BVA01600
BVA01610
BVA01620
BVA01630
BVA01640
BVA01650
C WHERE DELTA REPRESENTS DESIRED MINUS ACTUAL AND CONSTRAINTS
A RE SCALED TO HAVING UNITS OF LENGTH.

DELTC(1) = (C(1)**4 + C(2)**4 + C(3)**4 + C(4)**4) / H**4
DELTC(2) = (C(6)**4 + C(8)**4 + C(9)**4 + C(10)**4) / H**4
DELTC(3) = (C(12)**4 + C(13)**4 + C(14)**4 + C(15)**4) / H**4
DELTC(4) = (C(16)**4 + C(17)**4 + C(18)**4 + C(19)**4) / H**4
DELTC(5) = (C(20)**4 + C(21)**4 + C(22)**4 + C(23)**4) / H**4
DELTC(6) = (C(24)**4 + C(25)**4 + C(26)**4 + C(27)**4) / H**4

C CALCULATE ADJACENT COMBINATIONS OF TRANSVERSALITY

C CONDITIONS AND MISS-IF-EXICIENT CONDITIONS

DC(1) = C(1) + H**3 + DELTC(1)
DC(2) = C(2) + C(3) + DELTC(2)
DC(3) = C(4) + H**2 + DELTC(3)
DC(4) = C(5) + H**2 + DELTC(4)
DC(5) = C(6) + H**2 + DELTC(5)
DC(6) = C(7) + H + DELTC(6)

RETURN

END

C

SUBROUTINE SOLVE(A,L0,L7)

REAL*8 A(12,25), L0
DO 5 I=1,20
N=I+1
DO 5 1=1,N
A(1,J)=L0
DO 5 J=1,7
B(1,J)=L0
DO 5 1=1,N
A(I,J)=A(I,J)+B(I,J)
DO 5 J=1,7
A(I,J)=A(I,J)+B(I,J)

CONTINUE

END

C

SUBROUTINE BVAL5 A1,F,P1,T1,V1,AT1

IMPLICIT REAL*8(A-H,O-Z)

C THIS IS A FOUR-CONSTRAINT VERSION OF BVAL6, THE MISSION.
C IS TO ACHIEVE AN ORBIT WITH GIVEN VALUES OF SEMIMAJOR AXES.
C ECCENTRICITY, INCLINATION, AND ARGUMENT OF PERIGEE. THE ORBITAL
C CONSTANTS WHICH ARE TRANSMITTED IN C IN THE COMMON BLOCK 8011N.
C ARC MAGNITUDE AND THIRD COMPONENT OF ORBITAL ANGULAR VELOCITY H.
C THE THIRD COMPONENT OF A VECTOR E-POINTING TOWARD PERICENTER WITH...
C**C 1 > CALCULATE D(I) FOR I = 1 TO 4

R2 = 1.0/(XF(1) + XF(2) + XF(3) + XF(4))

R = (R2)**2.0

RTV = XF(1)*XF(2)*XF(3)*XF(4)

V2 = XF(4) + XF(5) + XF(6)

D(1) = (V2*RTV)/(RTV + V2)

D(2) = XF(1)*XF(5)*XF(6)

H2 = 0.0 - (1)

RTVR = RTV/R

FV = H2/UK - 1.0/R

D(4) = RTV*RF

D(4) = RTV*RF /

V2UR = V2/UK

RTV = RTV/UK

D(3) = XF(3) + XF(4) + XF(5) + XF(6)

IF (NVAL = 0.0 - 1) RETURN

C**C 2 > CALCULATE PARTIAL DERIVATIVES E(I,J)  

R3 = XF(3)*RM*RF

V3 = XF(6)

V2H = V2/D(1)

RTVF = RTV/D(1)

R2H = 1.0 - (R2 - D(1))

TR3 = 2.0*RF

CS1 = XF(5) + V2UR + V2/UK

CS2 = 2.0*V3/UK

F = XF(3)*RM*RF

DU 4 I = 1, 3

G(1) = F*XF(1)/3 + CS1*AF(1)

G(4) = CS2*FX(1)/3 + RF

G(6) = CS2*RF

DU -0.1 J = 1, MAX1

C**C 2.2 > CALCULATE R1'S AND V1'S

R2I = XF(1)*Z(1,I) + XF(2)*Z(2,I) + XF(3)*Z(3,I)

R2I = XF(1)*Z(1,I) + XF(2)*Z(2,I) + XF(3)*Z(3,I)

V1 = XF(4)*Z(4,I) + XF(5)*Z(5,I) + XF(6)*Z(6,I)

V2Z = XF(4)*Z(4,I) + XF(5)*Z(5,I) + XF(6)*Z(6,I)

C**C 2.3 > FINISH CALCULATION OF E(I,J)

E(2,J) = XF(3)*Z(2,J) + XF(4)*Z(3,J) + XF(5)*Z(4,J)

E(2,J) = XF(3)*Z(2,J) + XF(4)*Z(3,J) + XF(5)*Z(4,J)

E(3,J) = V2UR*Z(5,J) + V2UR*Z(6,J)

E(4,J) = G(1)*Z(1,J) + G(2)*Z(2,J) + G(3)*Z(3,J) + G(4)*Z(4,J)

E(5,J) = G(5)*Z(5,J) + G(6)*Z(6,J)

SUMS = 0.0

D0 2.K = 1.12
2  SUM=SUM+PTV(K)*Z(K,J)
1  (I,J)=SUM

C***< 3 > CALCULATE MISS IN END CONDITIONS. DEL C

DO J=1,4
  E(J)=C(J)-D(J)
  E(6,J)=TV
  E(5+J)=-(XF(5)*XF(1)+XF(4)*QF(2)+XF(2)*QF(4)+XF(1)*QF(5))
RETURN
END