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

(NASA-CR-120393) TARGETTING AND GUIDANCE PROGRAM DOCUMENTATION Final Report
(International Business Machines Corp.)
61 p HC $6.25 CSCL 22A
Unclas
G3/30 46510
<table>
<thead>
<tr>
<th>Section</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 targetting 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 targetting 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 targetting 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 targetting 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 document¹ 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>AMO</td>
<td>( m_0 )</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_sp</td>
<td>Specific impulse in seconds</td>
<td>440 sec</td>
</tr>
<tr>
<td>AMDOT</td>
<td>( \dot{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 (\( T_{AOMO} \) and \( T_{AOMT} \)) are greater than or equal to zero, they will be used to specify the orbital positions. If true anomalies are not specified and \( T_0 \) and \( T_T \) 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>( R(3), V(3), T_0 )</td>
<td>( \vec{r}_0, \vec{v}_0, t_0 )</td>
<td>Position (km) and velocity (km/sec) vectors at ( t_0 ) for vehicle orbit</td>
<td>( T_0 = -1 )</td>
</tr>
<tr>
<td>( R_T(3), V_T(3), T_T )</td>
<td>( \vec{r}_t, \vec{v}_t, t_t )</td>
<td>Position (km) and velocity (km/sec) vectors at ( t_t ) for target orbit</td>
<td>( T_T = -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>AØ, EØ</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>TANOMØ, TANOMT*</td>
<td>f</td>
<td>True anomalies (not required) (deg)</td>
<td>-1</td>
</tr>
<tr>
<td>TØ, 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

HAPØ, HPGØ  | None  | Height at apogee and perigee for vehicle orbit (km)                       |
HAPT, HPGT  | None  | Height at apogee and perigee for target orbit (km)                        |
RELINC      | 0     | Same as set 2                                                            |
TØ, TT, TANOMØ, TANOMT | -1   | Same as set 2                                                            |

ROMAG, VØMAG, | ROMAG: -1 | Magnitude of position and velocity vectors (km) and flight angle between them for vehicle orbit |
FLTØ  | FLTØ: -1 |
RTMAG, VTMAG, | Same for target orbit |
FLTt  | FLTT: -1 |
RELINC | 0 |
TØ, TT, TANOMØ, TANOMT | -1 | Same as set 2 |
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>Δtₜ</td>
<td>Time between guidance updates during burns (sec)</td>
<td>20 sec</td>
</tr>
<tr>
<td>DELS(2)</td>
<td>Δtₜ</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</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - navigation and acceleration perturbations</td>
<td></td>
</tr>
<tr>
<td>SIGMAR(1),SIGMAV(1)</td>
<td>δᵣ, δᵥ</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>δᵣ, δᵥ</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>δᵣ, δᵥ</td>
<td>Same in the middle of last burn (km/sec²)</td>
<td>0</td>
</tr>
<tr>
<td>PERT(1)</td>
<td>δₐ</td>
<td>Standard deviation of acceleration errors during burns (added each guidance cycle) (km/sec²)</td>
<td>0.05*Tₘ₀</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERT(2)</td>
<td>δₐ</td>
<td>Standard deviation of acceleration errors during coasts (km/sec²)</td>
<td>5.10⁻⁴</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</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 Targetting and feedback guidance</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 Guidance only using parameters read in by NAMELIST NAMLS2</td>
<td></td>
</tr>
<tr>
<td>NAVOFF</td>
<td>-</td>
<td>0 Convergence status printed whenever output sample is taken in guidance mode</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 No print</td>
<td></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>10.0 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>
</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>
</table>
| AXIS(I)   | -          | Axis of rotation of the earth, must be set in relation to coordinate system chosen by targeting when oblateness is activated | \[
\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \] |

### Guidance Only Parameters

- **NBURNS** - Number of burn arcs
- **X₀, T₀** - State of vehicle at start of mission. \( t₀ \) time at start of mission.
  - \( T₀ = -1 \)
- **Xₜ, Tₜ** - State of target at time \( tₜ \)
  - \( Tₜ = -1 \)
- **Q₀** - Initial costate
  - None
- **M₀** - Initial mass
  - 28803.1155 kg (63500 lbs)
- **TIMES** - Array of times defining start and end of coast and burn arcs
  - None
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 targetting 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>HPGØ, HPGT, HPGX</td>
<td>-</td>
<td>Height at perigee for vehicle, target and transfer orbits (km.)</td>
</tr>
<tr>
<td>HAPØ, HAPT, HAPX</td>
<td>-</td>
<td>Height at apogee for vehicle, target and transfer orbits (km.)</td>
</tr>
<tr>
<td>AØ, AT, AX, AP</td>
<td>a</td>
<td>Semi-major axis for vehicle, target, transfer and phasing orbits (km.)</td>
</tr>
<tr>
<td>EØ, ET, EX, EP</td>
<td>e</td>
<td>Eccentricity for respective orbits</td>
</tr>
<tr>
<td>VAPØ, VAPT, VAPX, VAPP</td>
<td></td>
<td>Velocity magnitude at apogee for respective orbits (km/sec)</td>
</tr>
<tr>
<td>VPGØ, VP GT, VPGY, VPGP</td>
<td></td>
<td>Velocity magnitude at perigee for respective orbits (km/sec)</td>
</tr>
<tr>
<td>TAUØ, TAUT, TAUX, TAUP</td>
<td>τ</td>
<td>Period for respective orbits (sec.)</td>
</tr>
<tr>
<td>IBOUND</td>
<td>-</td>
<td>0 - Outbound mission 1 - Inbound mission</td>
</tr>
</tbody>
</table>
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 $|\dot{q}_0|$ is arbitrary for the boundary value problem only one parameter was left to be determined, the relationship between the $|\dot{u}|$ and $|\ddot{u}|$. (Note: $\ddot{q}_0 = (\dot{u}^T, \ddot{u}^T)$.) Using the fact that the variations in $\dot{r}, \dot{v}$ form the same class of solutions as $\dot{u}, \ddot{u}$, and applying the switching condition that $|\dot{u}|$ at perigee must equal the $|\ddot{u}|$ at apogee, it was found that the impulsive solution for $\ddot{u}$ and $\ddot{u}$ at apogee and perigee becomes
\[
\vec{v} = \vec{v} \left( \frac{r_a + \overline{r}}{v_p} \right)
\]

\[
\ddot{\vec{u}} = -\vec{r} \left( \frac{\mu - \dot{r} a + \frac{v}{r}}{r^3 v_p} \right)
\]

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 \( q_0 \) is desired) along the transfer orbit. In the program these formulas are further reduced and the \( |\vec{u}| \) is chosen to be unit magnitude. The formulas become

\[
\ddot{\vec{u}} = \vec{v}
\]

\[
\ddot{\vec{u}} = -\vec{r} \cdot \text{FACTOR}
\]

where for perigee the factor becomes

\[
\text{FACTOR} = \frac{(1 + e_x/2 - e_x^2/2)\mu}{r_p^3 v_p^3} \quad e_x - \text{transfer orbit eccentricity}
\]

and at apogee it is

\[
\text{FACTOR} = \frac{\mu + v_a v_p r_a}{r_a^3 (v_p + v_a)}
\]

When the mission is inbound and velocity needs to be reduced, the sign on both \( \vec{u} \) and \( \ddot{\vec{u}} \) is reversed. Since this \( q_0 \) 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 \( T_{AUP} \). 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 \( T_{AUP} \) is chosen in subroutine 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 $|\tilde{u}|$ 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

\[
\begin{align*}
\text{T Imes}(1) &= \text{TA} \\
\text{TIMES}(2) &= \text{T IMES}(1) + \text{BUR N3} \\
\text{TIMES}(3) &= \text{T IMES}(2) + \text{TAUX/2} - \text{*BUR N3 + BURN 2)/2.0} \\
\text{TIMES}(4) &= \text{T IMES}(3) + \text{BUR N2} \\
\text{TIMES}(5) &= \text{T IMES}(4) + \text{TAUP} - \text{(BUR N2 + BURN 1)/2.0} \\
\text{TIMES}(6) &= \text{T IMES}(5) + \text{BUR N1}
\end{align*}
\]

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 TAUX and TAUP are the periods of the transfer and phasing orbits respectively. The initial mass is reduced to

\[
m_0 = m_0 - m(BURN1 + BURN2 + BURN3)
\]

where $m$ is positive. $\text{Q0}$ 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 ?

YES (NO TARGETTING)

READ KNOWN CONVERGENCE

NO

TO, TT > 0 OR TANOMO, TANOMT > 0 ?

NO

TIME ALONG ORBITS SPECIFIED

NBURNS=2

YES

CALL PHASE

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

NO

2-BURN IMPULSIVE INITIALIZATION

NBURNS=2 ?

YES

3-BURN IMPULSIVE INITIALIZATION

YES

3-BURN OUTBOUND MISSION

TURN PROBLEM AROUND TO INSURE CONVERGENCE. GO FROM TARGET ORBIT TO VEHICLE ORBIT WITH INCREASING MASS. RESET XO, QO, X, TIMES 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

IBOUND=1

YES INBOUND MISSION

RELINC=0 ?

YES, NO PLANE CHANGE REQUIRED

NO
CONVERGE PLANE CHANGE | IN MAXIMUM OF 10° STEPS

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

3-BURN NO \( \text{NBURNS} = 2 \)? YES 2-BURN

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

\[ \text{IBACK} = 1 \text{? YES} \]

BACKWARDS MISSION MUST ROTATE \( \text{XO}, \text{QO} \) ABOUT PERIGEE (\( \text{XO} \) IS REAL TARGET ORBIT)

\[ \text{IBACK} = 1 \text{? YES} \]

3 BURN INBOUND ROTATE XT-PHASING ORBIT STATE AT TT

CALCULATE DESIRED END CONDITIONS AND CONVERGE PLANE CHANGE MISSION

\[ \text{ANGLE} < \text{RELINC} \text{? YES, ADD MORE PLANE CHANGE} \]

\[ \text{NBURNS} = 2 \text{? YES} \]

2-BURN TARGETTING COMPLETED

ADD THIRD BURN BACK INTO MISSION DEFINITION

\[ \text{RELINC} = 0 \]

RETURN TO RECONVERGE TOTAL MISSION
RECONVERGE MISSION

3-BURN OUTBOUND
MISSION WHICH HAS BEEN CONVERGED BACKWARDS, TURN AROUND AND RECONVERGE

IBACK=0

203

ALL TARGETTING COMPLETED

PRINT PERTINENT DATA ABOUT ACHIEVED FINAL ORBIT

NOTARG=-1 ? YES NO FEEDBACK GUIDANCE DESIRED

STOP

G GUIDANCE SECTION

NBURNS=2 ? YES NO

REMOVE ANY INITIAL COAST FROM 3-BURN MISSIONS

3-BURN OUTBOUND TURN AROUND MISSION TO INSURE CONVERGENCE

IBOUND=1 ?

YES CALCUATE END CONDITIONS AND SAVE INITIAL CONDITIONS NEEDED TO PROCESS FUTURE MONTE CARLO CASES

MONTE CARLO LOOP

SET INITIAL CONDITIONS

NBURNS=3 - CALL BCBCB
NBURNS=2 - CALL CBCB

CALL STATIS
PRINT: MONTE CARLO STATISTICS

STOP
FILE: MAIN FORTRAN P1 CAMBRIDGE MONITOR SYSTEM

C ** ENGLISH TO METRIC CONVERSIONS **

C 1 POUND THRUST = 0.00444822165 KILONEWTONS
C HENCE: THRUST(KN) = THRUST(LB) * 0.00444822165
C
C 1 Lb OF MASS = 0.4536 KILOGRAMS
C HENCE: MASS(KG) = MASS(LBS) * 0.4536
C
C ** MASS RATE (KG/SEC)= THRUST(LBS) / ISP (SEC) * 4536 KG/LB
C
C ** INITIAL MASS IN KILOGRAMS
AMO=28603.1155
C THRUST IN KILOWATTS
THRU=66.7233
C SPECIFIC IMPULSE IN SEC.
SPF[LMP]=440.
C MASS RATE IN KG/SEC
AMOUT=-1.0
C
C ** INITIALIZE VARIABLES **
C
C SLT. INPUT DEVICE
INPUT=1
C SET OUTPUT DEVICE
OUTPT=6
C SET TIME TOLERANCE (SECONDS) USED TO DETERMINE IF TUG OR TARGET IS
C CLOSE ENOUGH TO A NODE OR DESIRED MEAN ANOMALY.
ERRCTR=10.
C SET ANGLE TOLERANCE (DEGREES) DIFFERENCES IN ANGLES LESS THAN ANG=ERRCTR
C WILL BE IGNORED.
ANG=5.
C SET ECCENTRICITY TOLERANCE ORBITS WITH ECCENTRICITIES LESS THAN
C ERRCTR WILL BE TREATED AS CIRCULAR.
ERRCTR=0.01
C IF TIMES TO TT TO -1.0 TO INDICATE A 2-BURN MISSION.
TT=-1.0
C IF THEY ARE CHANGED BY THE INPUT DATA A 3-BURN MISSION IS ASSUMED.
TT=-1.0
C SET INPUT VARIABLES TO -1.0 OR 0.0 TO INDICATE THAT THEY HAVE NOT
C BEEN READ IN THROUGH NAMELT.
CK=1.0
AG=0.0
EG=-1.0
AT=0.0
LT=-1.0
RLINC=0.0
HAP=1.0
MPG0=-1.0
HAPF=-1.0
MPGF=-1.0
FTO=1.0
FTTF=-1.0
TAN40=-1.0

MAIN - 10
FILE: MAIN  FORTRAN 77  CAMBRIDGE MONITO  SYSTEM

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TANUMT</td>
<td>1.0</td>
</tr>
<tr>
<td>RUMAG</td>
<td>-1.0</td>
</tr>
<tr>
<td>DU_5_1</td>
<td>1.3</td>
</tr>
<tr>
<td>RUT(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>VOT(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>RT(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>VT(1)</td>
<td>0.0</td>
</tr>
<tr>
<td>NJOIRNS</td>
<td>2</td>
</tr>
<tr>
<td>IUPODAT</td>
<td>50</td>
</tr>
<tr>
<td>NOISON</td>
<td>0</td>
</tr>
<tr>
<td>NAVUFF</td>
<td>1</td>
</tr>
<tr>
<td>NOTARG</td>
<td>0</td>
</tr>
<tr>
<td>PTU=</td>
<td>50</td>
</tr>
<tr>
<td>PTC=</td>
<td>2000.0</td>
</tr>
<tr>
<td>PRED(1)</td>
<td>1.0</td>
</tr>
<tr>
<td>PRED(2)</td>
<td>0.0</td>
</tr>
<tr>
<td>PRED(3)</td>
<td>0.0</td>
</tr>
<tr>
<td>IPHASE</td>
<td>0</td>
</tr>
<tr>
<td>IBACK</td>
<td>0</td>
</tr>
<tr>
<td>ITURN</td>
<td>0</td>
</tr>
<tr>
<td>IBOUND</td>
<td>0</td>
</tr>
<tr>
<td>IPRINT</td>
<td>0</td>
</tr>
</tbody>
</table>

.C DIRECTION OF EARTH'S AXIS IN REFERENCE COORDINATE SYSTEM.
.C TUG ORBIT IS CONSIDERED EQUATORIAL. TO USE STATES IN ANOTHER CARTESIAN.
.C-SYSTEM-THE VECTOR-AXIS MUST BE CHANGED TO REFLECT THE NEW SYSTEM.

.C OBALLENCE
.OJLATE=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_1 = 1.12
.PTV(1) = 0.0
.DO_3_1 = 1.5
.DO_3_2 = 1.10
.DO_2_2 = 1.8

.DATAM(1,J,K) = 0.0
.DATO(1,J,K) = 0.0

.DATAM(1,J,2) = 0.0
.DU_4_1 = 1.5

.PERT(1) = 0.0
.SIGMA(1) = 0.0

.NCAR(1) = 1

.C SET TRANSVERSALITY CONDITION TO ZERO.
.TV=0.0

.C-SET-SEEDS FOR ISM-LOCAL RANDOM NUMBER GENERATOR. THE NEXT 4 LINES MAY BE DROPPED ALONG WITH COMMON BLOCK RNI, SO LONG AS A RANDOM NORMAL
.C-MEAN 0, VARIANCE 1) NUMBER GENERATOR IS SUBSTITUTED.
.NRAN(1) = 0.
C PERTURBATIONS IN THE ACCELERATION DURING GUIDANCE. PERT(1) IS DURING A BURN AND PERT(2) DURING A COAST. PERT(3) CAN BE USED FOR INTRODUCING MORE PERTURBATIONS IN THE TARGET ACCELERATION.

If (PERT(1)+PERT(2)*GT.1.D=0) GO TO 6

C

PERT(1) = .05#THrust/AM0
PERT(2) = .00005#UK/(6700.*32)

C CONTINUE

C STORE VEHICLE SPECS IN WORKING ARRAY VEH(1,7)

VEH(1,1) = AM0

VEH(1,2) = AM0
VEH(1,3) = 10000.
VEH(1,4) = THRUST
VEH(1,5) = 0.0
VEH(1,6) = 0.0
VEH(1,7) = 25.0

C CHECK IF GUIDANCE IS DESIRED WITHOUT PRELIMINARY TARGETING (OPTIMAL SOLUTION ALREADY EXISTS). A PASS WILL BE MADE THROUGH A CONVERGENCE LOOP TO INSURE THAT THE SOLUTION IS VALID.

IF (NOTARG.EQ.1) ITUNRE=3

C TANGENTIAL IS DESIRED; DETERMINE IF A 2 OR 3-BURN MISSION IS DESIRED.

C IF TO GEO U=0 AND EQUATORIAL BURN IS ASSUMED.

C ALSO, IF THE TRUE ANOMALIES WERE SPECIFIED; A 3-BURN MISSION IS ASSUMED.

IF (TO.GE.0.0..AND.TI.GE.0.0) GO TO 500

IF (TAN0UT.GE.0.0..AND.TAN0MT.GE.0.0) GO TO 500

C

C * * * * * TWO-BURN MISSION SPECIFIED * * * * * * *

C CHECK FOR COMPLETE SET OF 2-BURN INPUT DATA.

C CHECK FIRST WHETHER SEMI-MAJOR AXIS (AU) AND ECLIPTICITY (EU) OF INITIAL ORBIT AND (AT,ET) OF FINAL ORBIT WERE SPECIFIED.

IF (AU.GT.1.EARTH AND UG0.GE.0.0..AND.UGL.GE.REARTH .AND. ET ..)

160.00) GO TO 100

C (AO,EU) AND (AT,ET) WERE NOT SPECIFIED. CHECK IF HEIGHTS AT APUGE:

C AND PERIGEE (KM) WERE SPECIFIED.

IF (APU.GE.0.0..AND.HPG0.GE.0.0..AND.HAPT.GE.0.0..AND.HPGT.GE.0.0.)

1 GO TO 50

C HEIGHTS NOT SPECIFIED. CHECK IF MAGNITUDES OF POSITION AND VELOCITY.

C AND FLIGHT ANGLES WERE SPECIFIED.

C THE FLIGHT ANGLE IS DEFINED AS THE ANGLE BETWEEN THE POSITION AND V-VECTOR, MEASURED IN THE DIRECTION OF ORBITAL MOTION. IT IS

C DEGREES BETWEEN PERIGEE AND APUGE. ANGLE 45 DEGREES BETWEEN

C APUGE AND PERIGEE, AND DETERMINES THE ANGULAR MOMENTUM THROUGH VMAG.

C VMAGDABS(SIN(PFL))
FILE: MAIN

1 IF(ROMAG.GT.REARTH AND VMAG.GE.1 AND FLTT.GE.0.0) GO TO 75
2 RTMAG.GE.REARTH AND VMAG.GE.1 AND FLTT.GE.0.0) GO TO 75
3 C VARIABLES AND TERMINAL
4 WRITE(IOUTPT,3000)
5 3000 FORMAT(* EXCMISION TERMINATING. LEPROPER DATA SPECIFIED FOR 2-BURN*MAIN)
6 3 MISSION*)
7 WRITE(IOUTPT,3001) AO,EO,AT,LT,RELLINO
9 25 WRITE(IOUTPT,3002) HAPG,HPGT,RELLINC
11 1 WRITE(IOUTPT,3003) ROMAG,VMAG,FLTU,RTMAG,VMAG,FLTT,RELLINC
13 STOP
2 C CONVERT HEIGHTS AT-APOGEE AND PERIGEE INTO ORBITAL ELEMENTS.
30 AO=REARTH+(HAPG*HAPG)/2.
40 EO=(HAPG*REARTH)/AO-1.0
50 IF(Eeo.LT.ERROR) EO=EO-EO.
60 AT=REARTH+(HAPG*HAPG)/2.
70 LT=(HAPG*REARTH)/AT-1.0
80 IF(LT.LT.ERROR) LT=EO-EO.
90 GO TO 110
10 C CONVERT POSITION, VELOCITY AND FLIGHT ANGLES TO ORBITAL ELEMENTS.
125 AO=UK+ROMAG/(2.*REKT-VMAG**2.*RTMAG)
130 HUMAG=DBS((ROMAG*VMAG)**2*sin(FLTU+DEGCON))
135 EO=DSQRT(1.-HUMAG**2/(AO*UK))
140 IF(Eeo.LT.ERROR) EO=EO-EO.
145 AT=UK+RTMAG/(2.*UK-VMAG**2.*RTMAG)
150 HTMAG=DBS((RTMAG*VMAG)**2*sin(FLTU+DEGCON))
155 LT=DSQRT(1.-HTMAG**2/(AT*UK))
160 IF(LT.LT.ERROR) LT=EO-EO.
170 HAPG=AU+(1.+EO)-REARTH
175 HAPG=AK+(1.+EO)-REARTH
180 HAPG=AT+(1.+EO)-REARTH
185 HPGT=AT+(1.+EO)-REARTH
190 IF(HPGT.GT.0.0 AND HPGT.GT.0.0) GO TO 110
200 WRITE(IOUTPT,3150) HAPG,HPGT
3150 FORMAT(* EXCMISION TERMINATING. */ HEIGHT AT PERIGEE OF TUG=*
1.1,D14.0","/ HEIGHT AT PERIGEE DE TARGET=",D14.0)
STOP
C C DETERMINE INITIAL AND FINAL RADII. TEST FIRST IF AN INGUON OR
D OUTBOUND MISSION. IF APOGEE OF THE TARGET ORBIT IS LESS THAN APOGEE
D OF THE TUG ORBIT, AN INGUON MISSION IS ASSUMED.
C CHECK FOR GREATER APOGEE.
110 IF(HAPG.GT.HAPT) GO TO 120
C OUTBOUND MISSION (FROM PERIGEE TO APOGEE).
225 IBOUND=0
230 RI=REARTH+HAPG
235 RF=REARTH+HAPT
240 GO TO 150
FILE:  MAIN  SUBROUTINE  V1  CAMBRIDGE  MONITOR  SYSTEM

C INBOUND MISSION (FROM APOLLO TO PERIGEE).
125  BOUND=1
   R1=REARTH+HAPX
   RF=REARTH+HAPX
C CALCULATE INITIAL AND FINAL VELOCITIES.
150  V=DSORT(UK#(2.*(RI-1./AX)))
C DETERMINE VELOCITIES AT END POINTS OF TUG, TARGET ORBITS
   VAPX=DSORT(UK#(2./*(HAPX+REARTH)-1./AX))
   VAPX=DSORT(UK#(2./*(HAPX+REARTH)-1./AX))
   VAPX=DSORT(UK#(2./*(HAPX+REARTH)-1./AX))
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 1BOUND. PERIGEE IS IN THE DIRECTION (1.0,0) AND THE
C VECTOR IN THE DIRECTION (0.0,1)
C IF PHASE=1, THE STATE VECTORS WILL BE SET UP IN SUBROUTINE PHASE.
   IF(PHASE.EQ.1) GO TO 175
   SIGN=1.0
   IF(1BOUND.EQ.1) SIGN=-1.0
   R0(1)=R1*SIGN
   R0(2)=0.0
   R0(3)=0.0
   V0(1)=0.0
   V0(2)=V1*SIGN
   V0(3)=0.0
   AT(1)=RF*SIGN
   AT(2)=0.0
   AT(3)=0.0
   VT(1)=0.0
   VT(2)=-V1*SIGN
   VT(3)=0.0
C SET UP TRANSFER ORBIT
   A=(RI+RF)/2.
   EX=MAX1(R1,RF)/AX-1.0
   HAPX=AX#(1.+EX)-REARTH
   HAPX=AX#(1.+EX)-REARTH
   VAPX=DSORT(UK#(2./*(HAPX+REARTH)-1./AX))
   VAPX=DSORT(UK#(2./*(HAPX+REARTH)-1./AX))
C DEFINE DELTA VS (+ 0K =)
   C THE INITIAL DELTA VS IS DEFINED AS THE VELOCITY AT THE APOLLO/PERIGEE
   C ON THE TRANSFER ORBIT MINUS THE INITIAL VELOCITY. THE FINAL DELTA VS IS
   C DEFINED AS THE FINAL VELOCITY MINUS THE VELOCITY AT PERIGEE/APO
C THE TRANSFER ELLIPSE.
   IF(1BOUND.EQ.1) GO TO 180
   DELTV1=VAPX-V1
   DELTVF=VF-VAPX
   VELXRF=VAPX
   GO TO 195
180  DELTV1=VAPX-V1
   DELTVF=VF-VAPX
   VELXRF=VAPX
C DETERMINE ORBITAL PERIODS.
- 195  TAU=2.*PI#DSORT(AX#3/UK)
**FILE: MAIN FORTRAN 77**

**CAMBRIDGE MONITOR SYSTEM**

---

\[ \text{TAUX} = 2.0 \times 10^{-6} \text{s} \]

\[ \text{TAUT} = 2.0 \times 10^{-5} \text{s} \]

**C** DETERMINE FINITE BURNS FROM DESIRED DELTA V'S AND INITIAL \( \Phi \).

\[ \text{BURN1} = \text{AMG} / \text{AMDUT} \times (\text{DEL} / \text{AMDUT} \times \text{DABS(DELTV1) / THRUST}) - 1.0 \]

\[ \text{DELTAM} = \text{BURN1} \times \text{AMDUT} \]

\[ \text{BURN2} = (-\text{AMG} \times \text{DELTAM}) / \text{AMDUT} \times (\text{DEXT} / \text{AMDUT} \times \text{DABS(DELTVF) / THRUST}) - 1.0 \]

\[ \text{COAST} = \text{TAUX} / 2.0 \times (\text{BURN1} + \text{BURN2}) / 2.0 \]

**C** SET UP TIMES ARRAY FOR 2-BURN MISSION WITH TIME TO=0 ARBITRARILY 2000

**C** SECONDS BEFORE THE TUG IS DUE AT THE NODE, UNLESS TO AND T1 WERE

**C** ORIGINALLY SPECIFIED AS >= 0.0 OR (PHASE=1) IN WHICH CASE TO-1.0 IS

**C** RUN BACK BY 2000 SEC, SO LONG AS IT IS <= 0.0

\[ \text{T1} = 2.0000 \times \text{TAUX} / 2.0 \]

\[ \text{T0} = 0.0 \]

**IF** (PHASE=0.0) **GO TO** 196

\[ \text{T1} = 2.0000 \times \text{TAUX} / 2.0 \]

\[ \text{T0} = 0.0 \]

**IF** (TO-2000.0 LT 0.0) **SHIFT**=2000.0-T0

**IF** (TO-2000.0 LT C.0) \( \text{TT} = \text{TT} + 2000.0 - \text{T0} \)

\[ \text{TT} = \text{TT} + 2000.0 - \text{T0} \]

**IF** (SHIFT.GT.0.0) **WRITE** (IOUTPT,3050) **SHIFT**, \( \text{TT} \)

**GO TO** 197

\[ \text{TT} = 2.0000 + \text{TAUX} / 2.0 \]

\[ \text{T0} = 0.0 \]

**IF** (PHASE=0.0) **GO TO** 197

**GO TO** 197

**TIME** (3)=2000.0 + BURN1/2 + T0

**TIME** (4)=**TIME** (3) + BURN1

**TIME** (5)=**TIME** (4) + COAST

**TIME** (6)=**TIME** (5) + BURN1

**C** STORE THE TUG AND TARGET STATES IN THE WORKING VARIABLES X, XT

\[ \text{DD} = 200.0 \times 1.0 \times \text{X(0)} \]

\[ \text{XG(1)} = \text{X(1)} \]

\[ \text{XT(1)} = \text{X(2)} \]

\[ \text{XT(3)} = \text{V(1)} \]

**C** DEFINE IMPULSIVE COSTATE U IS UNIT MAGNITUDE AND ALONG VELOCITY

**C** VECTOR AT THE NODE AND U=0.15ALONG THE EARTH RADIUS-VECKR.

\[ \text{IF} \left( \text{BURN1} \leq 0.0 \right) \text{GO TO} 2005 \]

**SIGN**=1.0

\[ \text{FACTOR} = \left( 1.0 + \text{LK} / 2.0 - \text{LK}^{2} / 2.0 \right) \times \text{UK} / \left( \text{RI} \times \text{X3} \times \text{VELX} \times \text{RI} \right) \]

**GO TO** 2007

**SIGN**=-1.0

\[ \text{FACTOR} = \left( \text{UK} \times \text{VARX} \times \text{VARX} \times \text{VARX} \times \text{VARX} \right) / \left( \text{RI} \times \text{X3} \times \text{VELX} \times \text{RI} \right) \]

**GO TO** 2007

**CONTINUE**

\[ \text{DD} = 201.0 \times 1.0 \]

\[ \text{GO(1)} = \text{XG(1)} / \text{SIGN} \]

\[ \text{GO(1)+3} = -\text{GO(1)} \times \text{FACTOR} \times \text{SIGN} \]

\[ \text{GO(0)} = 0.0 \]

**C** COAST-TUG-JACK ARBITRARY 2000.0 SEC BEFORE START OF THE 1ST BURN

**CALL** COAST(X0,0.0,-2000.0,0.0,0.0,PHI1,PHI)

**BEGIN**=**TIME** (3)

**GO TO** 191

**END**

---

**C** A 3-BURN MISSION IS ASSUMED, UNLESS THE PHASING WILL ALLOW.
C-2 BURN MISSION. CHECK IF THE TUG AND TARGET STATES WERE SPECIFIED.
500 NDUNMS = 3
   IF (HUMAG.GT.0.0) GO TO 501
   ROPMAG = OSORT (K0 (1) * R0 (2) * R0 (3) * 2)
   VOPMAG = OSORT (V0 (1) * R0 (2) * R0 (3) * 2)
   NTMAG = OSORT (NT (1) * R0 (2) * R0 (3) * 2)
   VTMAG = OSORT (VT (1) * R0 (2) * R0 (3) * 2)
   IF (HUMAG.GT.0.0) AND VUMAG.GT.0.0 AND.
      NTMAG.GT.0.0) GO TO 500
   C-STATES NOT SPECIFIED. CHECK IF ORBITAL ELEMENTS WERE.
   IF (HAPL.GE.0.0 AND HAPG.GE.0.0 AND HAPT.GE.0.0)
      1. AND HPG.GE.0.0) GO TO 510
   C-CHECK IF HEIGHTS AT APOLLO AND PERIGEE (KM) WERE SPECIFIED.
   IF (HAPL.GE.0.0 AND HAPG.GE.0.0 AND HAPT.GE.0.0)
      1. AND HPG.GE.0.0) GO TO 500
   C-CHECK IF MAGNITUDES OF POSITION AND VELOCITY AND FLIGHT.
   C-ANGLES WERE SPECIFIED. (FLIGHT ANGLE DEFINED IN 2-BURN COMMENTS.)
501  IF (HUMAG.GE.0.0 AND VUMAG.GE.0.0 AND FLTT.GE.0.0 AND.
      1. INTMAG.GE.0.0) GO TO 500
   C INADEQUATE ELEMENTS SPECIFIED. STOP.
   WRITE (IOUTPT,3100) RT, VT, RELINC, A0, E0, AT, ET
   3100 FORMAT (1, EXECUTION TERMINATING. IMPROPER DATA SPECIFIED FOR 3-BURN.)
   L-MISSION./* / SET 1 = / / RT =130.0, / VT =13014.0, / E0 =13014.0, / AT =13014.0, / ET =13014.0, /
   1. GO TO 25
   C C-CONVERT HEIGHTS INTO ORBITAL ELEMENTS.
   505  A0 = REARTH + (HAPL + HPG) / 2.0
      E0 = (HAPG + REARTH) / A0 = 1.0
      AT = REARTH + (HAPT + HPG) / 2.0
      ET = (HAPT + REARTH) / AT = 1.0
   1. IF (ET.GT.MAXERR) ET = 0.0
   GO TO 510
   C-CALCULATE ORBITAL ELEMENTS INTO ORBITAL ELEMENTS.
   507  A0 = UKHUMAG / 2.0, UK = VUMAG - 2 * UKHUMAG
      HUMAG = DABS (HUMAG + VUMAG) * SIN (FLTT # DEGCON)
      E0 = OSORT (1.0, HUMAG) * (A0 * UK)
      IF (ET.GT.MAXERR) ET = 0.0
      AT = UK + NTMAG / 2.0, NT = VTMAG - 2 * NTMAG
      HTMAG = DABS (HTMAG + VTMAG) * SIN (FLTT # DEGCON)
      ET = OSORT (1.0, HTMAG) * (AT * UK)
   1. IF (ET.GT.MAXERR) ET = 0.0
   S10 HPG = A0 + (1.0 - U) * REARTH
   HP = AT + (1.0 - U) * -REARTH
   C ELEMENTS WERE SPECIFIED. CHECK WHETHER TRUE ANOMALIES WERE SPECIFIED.
   IF (HPG.GE.0.0 AND HP.GE.0.0) GO TO 511
   WRITE (IOUTPT,3150) HP, HP, STOP
   3150 FORMAT (1, THE TIMES TO IT WILL NOW BE CONSIDERED.
   C THE TRUE ANOMALIES (TIMES SINCE PERIGEE OR, IF CIRCULAR, SINCE THE
   C-NODE.}
C - CHECK IF EITHER IS GREATER THAN ITS URBITAL PERIOD.
   TAUO=2.*PI*DOSRT(AO*WS/1K)
   TAUT=2.*PI*DOSRT(AO*WS/1K)
   IF(TO.LT.TAUO.AND.TT.LT.TAUT) GO TO 520
C TIMES ARE NOT LESS THAN ONE URBITAL PERIOD. STOP
   WRITE(1001P+3102) TO,TT,TAUO,TAUT
5102 FORMAT('EXECUTION TERMINATING. MEAN-ANOMALIES EXCEED URBITAL PERIOD.'
   1 PERIOD:*'1/1',TO='1',D14.0,TT='1',D14.0,TAUO='1',D14.0,TAUT='1',D14.0,10)
   STOP
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)=AO*(1.-EO)
   STATE(2)=0.0
   STATE(3)=0.0
   STATE(4)=0.0
   STATE(5)=DOSRT(4K*1./STATE(1)-1./TAU)
   STATE(6)=0.0
   NO=1
   CALL COAST(STATE,QUUM,T0,STATE,QUUM,PHI,PHI)
525 DO 530 I=1,3
   RT(I)=STATE(I)
   VO(I)=STATE(I+3)
   STATE(1)=AT*(1.-LT)
   STATE(2)=0.0
   STATE(3)=0.0
   VMAGT=DOSRT(4K*1./STATE(1)-1./AT))
   STATE(4)=0.0
   STATE(5)=VMAGT*DSIN(2*ELINC*DEGCON)
   STATE(6)=VMAGT*DCOS(2*ELINC*DEGCON)
   CALL COAST(STATE,QUUM,TT,STATE,QUUM,PHI,PHI)
530 DO 530 I=1,3
   RT(I)=STATE(I)
   VO(I)=STATE(I+3)
C SET BOTH TIMES TO 200. AFTER PROPAGATING STATES THROUGH THEIR
C MEAN-ANOMALIES.
   TT=200.
   TF=200.
   GO TO 560
C
C TRUE ANOMALIES WERE SPECIFIED. SET UP COORDINATE SYSTEM SU H
C THAT THE X(1) AXIS IS TUG AND TUG. PERIGEE. X(3) IS ALONG
C H, AND X(2) IS X(3) CRUSS X(1).
550 RMAG=AO*(1.-EO**2)/(1.+EO*DCOS(TANOM*10*DEGCON))
   R0(1)=RMAG*DCOS(TANOM*10*DEGCON)
   R0(2)=RMAG*DSIN(TANOM*10*DEGCON)
   R0(3)=0.0
   VMAG=DSQRT(4K*(AO*(1.-EO**2)))
   V0(1)=VMAG*DSIN(TANOM*10*DEGCON)
   V0(2)=VMAG*(EO*DCOS(TANOM*10*DEGCON))
   V0(3)=0.0
   RMAG=AT*(1.-LT**2)/(1.+LT*DCOS(TANOM*10*DEGCON))
   RT(1)=RMAG*DCOS(TANOM*10*DEGCON)
   RT(2)=RMAG*DSIN(TANOM*10*DEGCON)
RT(3)=0.0
VMAG=DSQRT(UR/(RT(1)**2+RT(2)**2+RT(3)**2))
VT(1)=VMAG*DSIN(RT(2)/VMAG*DEGCON)
VT(2)=VMAG*DCOS(RT(2)/VMAG*DEGCON)
VT(3)=0.0
C ADD INCLINATION TO TARGET ORBIT
RT(1)=RT(1)
RT(3)=RT(2)*DSIN(RELINC*DEGCON)
VT(1)=VT(1)
VT(3)=VT(2)*DSIN(RELINC*DEGCON)
VT(2)=VT(2)*DCOS(RELINC*DEGCON)
TT=2000.0
GO TO 700
C TUG AND TARGET STATES WERE SPECIFIED CHECK IF THE TIMES ARE THE SAME
C THE STATES MUST BE SPECIFIED AT THE SAME TIME WHEN PHASE IS CALLED
600 IF(TAUS(TT)-T.T..1) GT TO 700
C COAST TARGET STATE BACKWARDS OR FORWARDS IN TIME AS REQUIRED
C SUBROUTINE COAST EXPLOITS A 6-VECTOR OF STATES
DO 601 I=1,3
STATE(I)=RT(I)
601 STATE(I+3)=VT(I)
NO=1
CALL COAST(STA,..GAUM...T.T...STATE,...GAUM,...PHI,...PHI)
TT=TT
DO 602 I=1,3
RT(I)=STATE(I)
602 VT(I)=STATE(I+3)
C SECONDS ALLOW FOR POSSIBLE INITIAL COASTS
IF(TO.T.T..GT..2000.) GO TO 700
TEMP=2000.-TO
TT=TT+TEMP
TO=2000.
WRITE(OUTPUT,3120) TEMP,TT,IT
3120 FORMAT(I.1H5,T10.5,F15.5)
IT=IN
1.. INCREASED TO AND TT BY .1.*F7.2.** TO AND TT NOW EQUAL 
.2E10.5., 1.E10.5.)
C CALL PHASE TO DETERMINE IF RENDEZVOUS IS POSSIBLE WITH 2 OR 3
C BURNS FOR CIRCULAR-TO-CIRCULAR CO-PLANAR MISSIONS A 2-BURN
C MISSION IS ALWAYS POSSIBLE FOR OTHER GEOMETRIES A 2 OR 3 BURN
C MISSION MAY BE POSSIBLE EXECUTION WILL TERMINATE IN PHASE IF NO
C MISSION IS POSSIBLE WITHIN THE ALLOTTED TIME.
700 CALL PHASE(KO,.VO,.VT,.NBURNS,.COAST,.TAU)
IPHASE=1
TAU0=2.*PI*DSQRT(TAU**3/UK)
TAU1=2.*PI*DSQRT(TAU**3/UK)
TO=TO+TCOAST
TT=TO
IF(NBURNS.LT.2) GO TO 100
C DETERMINE WHETHER AN INBOUND OR OUTBOUND MISSION IS REQUIRED
C FORMING HEIGHTS AT APOGEE AND PERIGEE
FILE: MAIN    FORTRAN I    CAMBRIDGE MONITOR SYSTEM

HAPG=A0*(1.+EL)-REARTH
HPGU=A0*(1.-EL)-REARTH
HAPT=AT*(1.+LT)-REARTH
HPGT=AT*(1.-LT)-REARTH
IF(HAPGT.HAP1).IBOUND=1
AP=(TAU0**2/UK)/((4.*EL*EL))**.53
IF(BOUND.0.-HPGU).MPGU=MPGU
IF(BOUND.EQ.1.-HPGT).MPGT=MPGT
EP=1.0-(HPGU+REARTH)/AP
HAPP=AP*(1.+EP)-REARTH

C CALCULATE VELOCITIES AT END POINTS OF ALL ORBITS.

VAPD=USORT(UK*(1.-LO)/(A0*(1.+LO)))
VPGU=USORT(UK*(1.+EL)/(A0*(1.-LO)))
VAPT=USORT(UK*(1.+EL)/(AT*(1.+LT)))
VPGT=USORT(UK*(1.+EL)/(AT*(1.-LT)))
VAPP=USORT(UK*(1.+EP)/(AP*(1.+EP)))
VPGP=USORT(UK*(1.+EP)/(AP*(1.-EP)))
IF(BOUND.EQ.1) GO TO 701

R1=HPGU+REARTH
RF=HAPT+REARTH
GO TO 702

702

VI=USORT(UK*(2.*R1-1./A0))
VF=USORT(UK*(2.*RF-1./AT))

AX=(R1+RF)/2.
EX=OMAX1(R1,RF)/AX-1.0

HAPX=AX*(1.+EL)-REARTH
HPGX=AX*(1.-EL)-REARTH
VAPX=USORT(UK*(2.*R1/OMAX1(R1,RF)-1./AX))
VPGX=USORT(UK*(2.*R1/OMIN1(R1,RF)-1./AX))
IF(BOUND.EQ.1) GO TO 703

Deltv1=VPGP-V1
Deltv1=V1-VAPX
GO TO 7035

Deltv1=VAPX-V1
Deltv1=VAPX-VPGX
Deltv1=VPGX-VPGP

C FIND FINITE BURN TIMES.

7035

BURN1=AN0/AMOU*DEAP(-AMOU*OABS(DELTV1)/THUST)-1.0
DELTA1=ANOU*BU2NO1
BURN2=-(AMO-DELTA1)/AMOU*DEAP(-AMOU*OABS(DELTV1)/THUST)-1.0
DELTA2=ANOU*BU2NO2
BURN3=-(AMO-DELTA2)/AMOU*DEAP(-AMOU*OABS(DELTV1)/THUST)-1.0
IF(BOUND.EQ.1) GO TO 704

COAST1=TAU0-(BURN1+BURN2)/2
COAST2=TAUX-COAST1-(BOUND+BURN3)/2
GO TO 705

704

COAST1=TAUX-(BOUND+BURN2)/2
COAST2=TAUX-(BURN2+BURN3)/2

C SET UP TIMES ARRAY

705

IF(TO-2000.-LT.0.-0.) TT=TT+2000.-TO
IF(TO-2000.-LT.0.-0.) TO=0.0
IF(TO.GT.0.-001.) TO=10.-2000.
FILE: MAIN FORTRAN 77

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TO=2000.0-BURN1/2.0+TO</td>
<td>Initial time setting</td>
</tr>
<tr>
<td>BEGIN=TO</td>
<td>Beginning of time step</td>
</tr>
<tr>
<td>TIMES(1)=TO</td>
<td>Time step 1</td>
</tr>
<tr>
<td>TIMES(2)=TO+BURN1</td>
<td>Time step 2</td>
</tr>
<tr>
<td>TIMES(3)=TIMES(2)+COAST1</td>
<td>Time step 3</td>
</tr>
<tr>
<td>TIMES(4)=TIMES(3)+BURN2</td>
<td>Time step 4</td>
</tr>
<tr>
<td>TIMES(5)=TIMES(4)+COAST2</td>
<td>Time step 5</td>
</tr>
<tr>
<td>TIMES(6)=TIMES(5)+BURN3</td>
<td>Time step 6</td>
</tr>
</tbody>
</table>

C STORE STATES IN X0 AND X1

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D0=710.1=1.3</td>
<td>Initial time for first burn</td>
</tr>
<tr>
<td>X0(1)=X0(1)</td>
<td>State variable initialization</td>
</tr>
<tr>
<td>X0(1+3)=V0(1)</td>
<td>State variable initialization</td>
</tr>
<tr>
<td>AT(1)=1(1)</td>
<td>Target state variable initialization</td>
</tr>
</tbody>
</table>

710 XT(1+3)=VT(1) | Initial state for in-bound mission |

C SET UP INITIAL G FOR INBOUND MISSION

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FACTOR=(UK+VAPX+VPGX*(HAPX+REARTH))/((HAPX+REARTH)*<em>3</em>(VAPX+VPGX))</td>
<td>Mass factor calculation</td>
</tr>
</tbody>
</table>

D0.720 1=1.3 | Initial time for first burn |

Q0(1)=X0(1+3)/V1 | Initial state for in-bound mission |

720 Q0(1+3)=X0(1)/FACTOR | Initial state for in-bound mission |

NO=0 | No additional initialization required |

C COAST TUG BACK TO START OF FIRST BURN

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL COAST(X0,QU,-BURN1/2.0,X0,QU,PHI,PHI)</td>
<td>Coasting function call</td>
</tr>
<tr>
<td>IF(BOUND.EQ.1) GO TO 202</td>
<td>Conditional branch</td>
</tr>
</tbody>
</table>

C TO INSURE CONVERGENCE

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C=BACK=1</td>
<td>Convergence check</td>
</tr>
</tbody>
</table>

C SAVE INITIAL STATE OF TUG

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D0.750 1=1.3</td>
<td>Initial time for first burn</td>
</tr>
</tbody>
</table>

750 TUGSAV(1)=X0(1) | State variable initialization |

TIMTUG=TO | Time step for coasting |

C PROPAGATE TARGET STATE TO FINAL BURN NODE

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D0.760 1=1.3</td>
<td>Initial time for last burn</td>
</tr>
</tbody>
</table>

Q0(1)=X0(1+3)/VAPT | Initial state for last burn |

760 U0(1+3)=X0(1)/FACTOR | Initial state for last burn |

NO=0 | No additional initialization required |

C CALL COAST(X0,QU,-BURN3/2.0,X0,QU,PHI,PHI) | Coasting function call |

C SET UP TIMES ARRAY

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMES(1)=TO</td>
<td>Time step 1</td>
</tr>
<tr>
<td>TIMES(2)=TIMES(1)+BURN3</td>
<td>Time step 2</td>
</tr>
<tr>
<td>TIMES(3)=TIMES(2)+COAST2</td>
<td>Time step 3</td>
</tr>
<tr>
<td>TIMES(4)=TIMES(3)+BURN3</td>
<td>Time step 4</td>
</tr>
<tr>
<td>TIMES(5)=TIMES(4)+COAST1</td>
<td>Time step 5</td>
</tr>
<tr>
<td>TIMES(6)=TIMES(5)+BURN1</td>
<td>Time step 6</td>
</tr>
</tbody>
</table>

C REDUCE MASS, SET TIMES AND TARGET STATE

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TO=TIMES(1)</td>
<td>Target time for first burn</td>
</tr>
<tr>
<td>TT=TIMES(6)</td>
<td>Target time for last burn</td>
</tr>
</tbody>
</table>

AM0=AM0-(BURN1+BURN2+BURN3)*AMOUT | Mass reduction calculation |
FILE: MAIN FORTRAN PI

C REVERSE MASS RATE VELOCITY AND UDOT
AMOBT=-AMOBT
V(EH(1,2)=-V(EH(1,2)
00.790=4.5
XT(J-3)=TUGSAV(J-3)
XT(J)=TUGSAV(J)
XO(J)=XO(J)
00.790=0(J)

790
C WRITE IMPULSIVE SOLUTION

3202 WRITE(100PT,3200) NBURNS
3203 FORMAT(//,20X,'**'**'**'**12.'-BURN IMPULSIVE APPROXIMATION SUMMAMAI000672
12Y**'**'**'**'**CRITICAL ELEMENTS'**'**'**12X'.'-SEMAI000673.
12Z AXIS ECCENTRICITY HEIGHT(APOGEE) HEIGHT(PERIGEE) PERIOD V(APMAI000674
30GEE) V(PERIGEE) **14X*(KILOMETERS)*16X*(KILOMETERS) (KILOMAI000675
ETERS) **(SECONDS) **(KM/SEC) **(KM/SEC)
3201 WRITE(100PT,3201) A00.0,TAU0.0,TAUH0.0,TIT0.0,TV0.0,TVH0.0
3202 WRITE(100PT,3202) F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J),F(J)
MAIN - 22

FILE: MAIN FORTRAN A1 CAMBRIDGE MUNITIO SYSTEM

-249  CC(1)=DD(1)
C ** **
C
C IF(1TURNR.EQ.0) WRITE(10,IOUTPT,3210)
C IF(1TURNR.EQ.1) WRITE(10,IOUTPT,3213)
C IF(1TURNR.EQ.2) WRITE(10,IOUTPT,3220)
C IF(1TURNR.EQ.3) WRITE(10,IOUTPT,3223)

3223 FORMAT(' BEGIN GUIDANCE-ONLY CONVERGENCE')
3220 FORMAT(' ADD 3RD BURN AND RECONVERGE')
3213 FORMAT(' BEGIN TURN-AROUND CONVERGENCE')
3210 FORMAT(' BEGIN COPLANAR CONVERGENCE')

C TRY TO CONVERGE THE COPLANAR MISSION IN 30 ITERATIONS.

C

DO 250 ITEK=1,30
NDF=1
QMAX=QSORT(001)**2+Q0(2)**2+Q0(3)**2
DO 247 00(1)=00(1)/QMAX

CALL GUID(0.0)
CALL AUXUT
CALL CKSET(CK)
DUMAX=0.0
DTMAX=0.0
DO 251 1=0

Q0(1)=Q0(1)+Q0(1)*CK
TIMES(1)=TIMES(1)+DTMAX(1)*CK
IF(DABS(D00(1)).GT.DQUMAX) DUMAX=DABS(D00(1))

251 IF(DABS(DTMAX(1)).GT.DTMAX) DTMAX=DABS(DTMAX(1))

AM=AMU
IF(IBACK.EQ.1) AMG=VLAN(1,1)+TIMES(3)-TIMES(5)+TIMES(4)-TIMES(3)
1-1.TIMES(2)-TIMES(1)+ANOUT
AMG=(AMG+AM)/2.0
IF(DTMAX.LT.0.0) D=A/0.0
DUMAX=Q0(1)* timelines(6). AND. DUMAX.LT.0.0

3211 FORMAT(' COPLANAR MISSION CONVERGED IN',13,' ITERS')
3214 FORMAT(' TURN-AROUND ACHIEVED IN',13,' ITERS')
3221 FORMAT(' 3RD BURN ADDED AND CONVERGED IN',13,' ITERS')

250 CONTINUE
C DID NOT CONVERGE IN 30 ITERATIONS. DUMP VARIABLES AND STOP.

3009 FORMAT(' PLANAR MISSION DID NOT CONVERGE IN 30 ITERATIONS. STOP')

252 CONTINUE
IF(1TURNR.EQ.0) WRITE(10,IOUTPT,3211) IT
IF(1TURNR.EQ.1) WRITE(10,IOUTPT,3214) IT

3222 FORMAT(' GUIDANCE-ONLY CONVERGENCE ACHIEVED IN',13,' ITERS')

C ADD PLANAR CHANGE
C
C ISEC=0
C ANGLE=0.0
C IF NO PLANAR CHANGE SKIP AROUND
C IF(DABS(RELINC.LT.ERROR)-60 TO 295
FILE: MAIN

C TRANSFORM TO 2-BURN MISSION FOR INSERTION OF PLANE CHANGE

C CALL GUIDE TO GENERATE PHASING ORBIT END CONDITIONS

C PERFORM REQUIRED PLANE CHANGE IN 10 DEGREE STEPS

C CHECK TO SEE IF CONVERTING BACKWARDS

C CONVERGING BACKWARDS ROTATE X0 AND Y0

C CONVERGING FORWARDS FIND IF 2 OR 3 BURNS

C 2-BURN MISSION ROTATE PRESENT AT

C ON FIRST PASS SET UP AN INITIAL COAST

C SET UP 3-BURN AS 2-BURN

C CALL ROTATE(X0,Y0,PRI,DEG,DEGCON,6)

C CALL COAST(X0,Y0,TCTR,CO,PHI,PHI)
C REDUCE INITIAL MASS TO REFLECT ESTIMATED THIRD BURN
IF(1BACK.EQ.0) GO TO 1000
AMSAV=AMDUM*(T6SAV-T5SAV)
VEH(1,1)=VEH(1,1)+AMSAV
1000 CONTINUE
C SET UP END CONDITIONS FOR PLANES CHANGE
CALL DVALS(XT,QU,PTV,TV,-1)
DO 261 I=1,6
-CC(1)=CC(I)
1 C ** **
C CTRY TO CONVERGE WITH PLANES CHANGE IN LE 30 ITERATIONS.
DO 290 I=1,30
DOP=1
UMAG=DSORT(DQ(1)**2+DQ(2)**2+DQ(3)**2)
DO 289 I=1,6
-CALL AUXOUT
CALL CKSET(CK)
DUMAX=0.0
DTHMAX=0.0
1 IPRINT=0
DO 288 I=1,6
DO(1)=DO(1)+DO(1)*CK
-TIMES(I)=TIMES(I)+.STIMES(1)*CK
-IF(DABS(DQ(I)),0.7,.DQMAX=DO(1)
288 IF(DABS(DQ(I)),.D1.DTMAX)DTHMAX=DABS(DTIMES(I))
AMI=AMI
-IF(1BACK.EQ.1)AM=AM(1,1)+(TIMES(6)-TIMES(5)+TIMES(4)-TIMES(3))
1+TIMES(2)-TIMES(1))AMOUT
-AM=AM(AMI+1,1)
-IF(DTHMAX.LT.1.0) DQ(MAX(TIMES(I))) AND DOOMAX.LT.0.01 AMOUT
1.1DAM=AMI+AMOUT+TIM(1,1)*1.0-6) GO TO 291
290 CONTINUE
C DID NOT CONVERGE IN 30 ITERATIONS. DUMP VARIABLES AND STOP
WRITE(IOUTPT,3006)
-3006 FORMAT(OUT-OF-PLANE MISSION DID NOT CONVERGE IN 30 ITERATIONS.)

STOP

C CHANGE BACK TO J-BURN IF NECESSARY.
-IF(NBURNS.NE.3) GO TO 200
-DO 292 I=1,4
-TIMES(I)=TIMES(I+2)
TIMES(5)=T5SAV
TIMES(6)=T6SAV
DO 293 I=1,6
XT(I+3)=TUGSAV(I+3)
-293 XT(I)=TUGSAV(I)

C TRY TO CONVERGE WITH J-BURN IN 13 ITERATIONS.
DO 3212 I=1,13
DOP=1
UMAG=DSORT(DQ(I)**2+DQ(I+1)**2+DQ(I+2)**2+DQ(I+3)**2)
DO 312 I=1,6
-CALL AUXOUT
CALL CKSET(CK)
DUMAX=0.0
DTHMAX=0.0
1 IPRINT=0
DO 311 I=1,6
DO(I)=DO(I)+DO(I)*CK
-TIMES(I)=TIMES(I)+.STIMES(I)*CK
-IF(DABS(DQ(I)),0.7,.DQMAX=DO(I)
311 IF(DABS(DQ(I)),.D1.DTMAX)DTHMAX=DABS(DTIMES(I))
AMI=AMI
3006 IF(1BACK.EQ.1)AM=AM(1,1)+(TIMES(I+6)-TIMES(I+5)+TIMES(I+4)-TIMES(I+3))
1+TIMES(I+2)-TIMES(I+1))AMOUT
-AM=AM(AMI+1,1)
-IF(DTHMAX.LT.1.0) DQ(MAX(TIMES(I))) AND DOOMAX.LT.0.01 AMOUT
1.1DAM=AMI+AMOUT+TIM(I,1)*1.0-6) GO TO 291
291 CONTINUE
C CHANGE BACK TO J-BURN IF NECESSARY.
-IF(NBURNS.NE.3) GO TO 200
-DO 292 I=1,4
-TIMES(I)=TIMES(I+2)
TIMES(5)=T5SAV
TIMES(6)=T6SAV
DO 293 I=1,6
XT(I+3)=TUGSAV(I+3)
-293 XT(I)=TUGSAV(I)
FILE: MAIN
FORTRAN 77
CAMBRIDGE MONITOR SYSTEM

IT=TIMSAV.
RELINC=0.0
C ADD MASSES TO INITIAL MASS IF BACKWARDS MISSION
IR(IBACK,EQ,1)=VEH(1,1)=VEH(1,1)-ARMSAV
ITRUN=2
C ROTATE XT IF FORWARDS MISSION
IF(IBACK,NE,1) GO TO 203
DO 294 I=1,3
XT(I)=RT(I)
294-XT(I+3)=VT(I).
CALL ROTATE(XT,PROE,ANGLE,DEGCON,0)
GO TO 203
C TEST-TO-SEL IF MISSION TURNAROUND IS NECESSARY
295 CONTINUE
IF(IBACK,NE,1) GO TO 200
C TURNAROUND MISSION AND RECONVERGE
C
C SET-UP TIMES ARRAY, U0, AND XT
WRITE(IOUTPR,4001)
4001 FORMAT(' TURNAROUND MISSION')
   B1=TIMES(6)-TIMES(3)
   C1=TIMES(5)-TIMES(4)
   B2=TIMES(4)-TIMES(3)
   C2=TIMES(3)-TIMES(2)
   B3=TIMES(2)-TIMES(1)
   TALIGN=TT-TIMES(6)
   TTLGN=TIMES(6)-TT
   TO=TALIGN+TTLGN
   TIMES(1)=TO
   TIMES(2)=TIMES(1)+B1
   TIMES(3)=TIMES(2)+C1
   TIMES(4)=TIMES(3)+B2
   TIMES(5)=TIMES(4)+C2
   TIMES(6)=TIMES(5)+B3
   TT=TO+TTLGN
   AMDUT=AMDUT
   VEH(1,2)=-VEH(1,2)
   DO 310 I=1,3
   QO(I)=Q(I)
   QO(I+3)=-Q(I+3)
   AT(I)=XO(I)
310-XT(I+3)=-XO(I+3).
   AM0=VEH(1,1)
   CALL TUGSAV(ITUGSAV,TUGSAL,TTIMG,TAUX,R0,RO,AM0)
C COAST TUGSAV TO ALIGN WITH TIMES(1)
   NO=1
   CALL COAST(TUGSAV,0,0,0,0,0,0,AM0)
   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)
VT(1)=X(1)+3

CALL ELMNTS(RTA,VT,AA,LA,HA,PA,TAU)

WRITE(10,OUTPT,3215) RTA,VT,AA,LA,HA,PA,TAU

3215 FORMAT(1X,13H0,1X,13H0,1X,13H0,1X,13H0,1X,13H0,1X,13H0,1X,13H0)

1. POSITION=*,3014.6,* VELOCITY=*,3014.6,*
2. SEMI-MAJOR AXIS=*,F10.2,* ECCENTRICITY=*,F8.6,*
3. H-VECTOR=*,3014.6,* PERIGEE=*,3014.6,* PERIOD=*,10.2)

BURN1=TIMES(2)-TIMES(1)

BURN2=TIMES(4)-TIMES(3)

BURN3=TIMES(6)-TIMES(5)

COAS1=TIMES(3)-TO

COAS2=TIMES(5)-TIMES(4)

IF(NBURNS.EQ.3) WRITE(10,OUTPT,3217)COAS1,BURN1,COAS2,BURN2,COAS3,BURN3

3217 FORMAT(1X,13H0,1X,13H0,1X,13H0,1X,13H0,1X,13H0,1X,13H0)

1. INITIAL COAST=*,F10.2,* FIRST BURN=*,F10.2,* SECOND_COAST=*,F10.2,*

2. =*,F10.2,* FINAL_BURN=*,F10.2,*

C

C-GUIDANCE-SECTION

C

C-CHECK-(IF-GUIDANCE-DISABLED)

IF(NUTARG.EQ.-1) STOP

C CHECK IF 2 OR 3 BURNS.

IF(NBURNS.EQ.2) GO TO 410

C BURN MISSION. REMOVE ANY INITIAL COAST.

IF(DAUS(TI.TIMES(1))LT.TERMRTK) GO TO 409

NO=0

CALL COAST(X0,00,TIMES(1)-TO,X0,00,PHI,PH1)

TO=TIMES(1)

AMO=VHEL(1,1)

TINT=TIMES(1)

IF(BOUND.EQ.1) GO TO 410

C TURN-AROUND OUTBOUND-3-BURN MISSION

C

C SET UP STATE AND TIME ARRAYS.

DU=0.1*1.1

ATS(1)=XU(1)

ATS(1+3)=XU(1+3)

GTS(1)=QQ(1)

GTS(1+3)=QQ(1+3)

X0(1+3)=X(1+3)

XT(1+3)=ATS(1+3)

X0(1)=X(1)

QQ(1)=QQ(1)
FILE: MAIN FORTRAN 11 CAMBRIDGE MONITOR SYSTEM

406 XT(1)=XTS(1)
B1=TIMES(2)-TIMES(1)
C1=TIMES(3)-TIMES(2)
B2=TIMES(4)-TIMES(3)
C2=TIMES(5)-TIMES(4)
B3=TIMES(6)-TIMES(5)

TIMES(1)=0.001
TIMES(2)=TIMES(1)+B1
TIMES(3)=TIMES(2)+C1
TIMES(4)=TIMES(3)+B2
TIMES(5)=TIMES(4)+C2
TIMES(6)=TIMES(5)+B3

TO=TIMES(1)
TT=TIMES(6)

BURN=OABS(TIMES(6)-TIMES(5)+TIMES(4)-TIMES(3)+
TIMES(2)-TIMES(1))
AM0=VEH(1,1)-BURN+VEH(1,2)
AM0D=AM0UT

VEH(1,2)=-VEH(1,2)

C SET UP INITIAL COAST FOR BACKWARDS MISSION.
NO=0
CALL COAST(XO,QQ,-300..XQ,QQ,PHI,PHI)

C SET UP MONTE CARLO RUNS.

C

C CALCULATE END CONDITIONS

DO 410 1=1,10

411 CC(I)=DD(I)

CSAVE INITIAL CONDITIONS FOR NEXT MONTE CARLO RUN.

DO 415 1=1,6

415 CCS(I)=CC(I)

VEH(7)=VEH(1,7)

TU0=TO

TTS=TT

AMUS=AM0

16OONS=1BOUND

C LOOP FOR MONTE CARLO RUNS.

IPRINT=1

DO 420 MONTL=1,MCARLU

IUPDAT=0

C RESTORE VARIABLES.

DO 425 1=1,6

VEH(1,1)=VEH(1)

XQ(I)=X0S(I)

QQ(I)=QQS(I)

XT(I)=XTS(I)

QT(I)=QTS(I)

TIMES(I)=TIMES(I)
FILE: MAIN  FORTRAN PI  CAMBRIDGE MONITOR SYSTEM

425  CC(1)=CCS(1)
    VEH(1,7)=VEHS(7)
    DO 426 I=1,10
426  VEH(1,0)=0.0
    ISOUND=ISOUNS
    TRUCMS=VEH(1,1)
    TCLOCK=0.0
    TACCUM=0.0
    AMO=AMOS
    TO=TCOS
    TT=TI5
    IF(NBURNS.EQ.3) CALL MCANL(BOUND,POINT)
    IF(NBURNS.EQ.2) CALL MCUB
    CALL STATIS(MCANLD)
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 \bar{t} )</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

\[
\text{COST} = |(\text{TIMES}(2) - \text{TIMES}(1)) + (\text{TIMES}(4) - \text{TIMES}(3)) + (\text{TIMES}(6) - \text{TIMES}(5))|
\]
FILE: AUXOUT    FORTRAN PL    CAMBRIDGE MONITOR SYSTEM

SUBROUTINE AUXOUT
IMPLICIT REAL*8(A-H,O-Z)
COMMON /GIDIN/XT(10),TT,XU(10),TO,AMO,VEH(10,7),QU(6),TIES(6),C16
COMMON /ADDLN/10UTP1
COMMON /GIDOUT/DOC(6),DTIM3ES(6),C(12),DC(12),X(6),Y(6),
1Z(12,12),D(6),DHIT(4),SM
COMMON /AVLOUT/ATRF(6),DELTG(6),
WRITE(IOMPT+1)X,XTF
1 FORMAT(//,'X(OBTAINED)='6E14.6,'X(DESIRED)='6E14.6)
WRITE(10UTP,2)COST
2 FORMAT('REMAINING SURN='6E14.6)
WRITE(10UTP,3)QU,DOC+TIM3,DTIM3ES
3 FORMAT(1X,'QU='6E16.8,'DOC='6E16.8,'6X,'I=',
16E16.8,'4X,'DF='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 QØ) 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>
<tbody>
<tr>
<td>IBOUND</td>
<td>-</td>
<td>0 - outbound mission (implies backwards mode)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - inbound mission</td>
</tr>
<tr>
<td>TØINT</td>
<td>-</td>
<td>In backwards mode, the actual value of TØ</td>
</tr>
<tr>
<td>TRUEMS</td>
<td>-</td>
<td>Vehicle mass before start of burn (normally equivalent to AMØ except when in backwards mode)</td>
</tr>
<tr>
<td>XT(I) I=1,6</td>
<td>-</td>
<td>Vehicle state in backwards mode</td>
</tr>
<tr>
<td>TT</td>
<td>t_T</td>
<td>Time at start of first burn in outbound case</td>
</tr>
<tr>
<td>TØ</td>
<td>t_0</td>
<td>Time at start of first burn in inbound case</td>
</tr>
<tr>
<td>IOUTPT</td>
<td>-</td>
<td>Output device number</td>
</tr>
</tbody>
</table>
### Input Parameter

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
</table>
| I=1,2  | I=1; normal guidance step size during burn  
|         | I=2; not used in BCBCB |
| TIMES(1) I=1,6 | Vector of times at end of each leg (or start of each leg in backwards mode) |

### Output Parameter

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPRINT</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>Always 0; restores mission to free-time rendezvous (backwards mode only)</td>
</tr>
<tr>
<td>AMQ</td>
<td>Mass at end of first burn</td>
</tr>
<tr>
<td>TIMES(1) 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>
</tbody>
</table>
| Q0(I) I=1,6 | q0  
|         | Costate at end of first burn |
| T0     | Time at end of first burn |
| TT     | Time for which target state is valid |
| CC(I) I=1,6 | New end conditions for target (backwards mode only) |

### 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 FORWARD 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 FORWARD, 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 FORWARD (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 FORWARD, 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 FORWARD 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 coast 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)
N = 0

SET STEP SIZE FOR THIS STEP
CALL FORWARD, INCREMENT TIMES(6)

LAST STEP IN BURN LEG
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

UPDATE MASS
N = N + 1
MISS = END CONDITION ERROR

N > 1 KM AND N < 20
MISS > 1 KM

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

COLLECT M-C STATISTICS
IOU T=1
COLLECT M-C STATISTICS

UPDATE MASS
N = N + 1
MISS 2 END CONDITION ERROR

N < 1 KM AND N < 20
MISS < 1 KM

CALL GUIDE
QT1 = Q0
i = 1, 6
SUBROUTINEBCBIC(ROUND,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/3VLOUT/XF(6)
COMMON/UPDATE/UPDAT
COMMON/CPHYS/UR,K,EARTH,DUAI,DU32,DU3
COMMON/ONDLINE/PRINT
COMMON/COUNT/USE
COMMON/CNAV/RULMS,TRAV,TACCUM,OUT
COMMON/3GIDIN/XT(6),IT,TD(6),TA,AM,A,M,VEH(10,7),QU(6),TIMES(6),CC(6)
COMMON/3UPDJ/DOU(6),UTIMES(6),L(12,12),DC(12),X(6),Q(6,5),Z(12,12)
COMMON/3NIN/SM(5)

COMMON/CINDEX/XARC,4MAX,JM,JMAX1,JLAST,NO,NOP,NR,IGOS

COMMON/CJT/WEIGHT(6)
COMMON/CUTST/ZL(5)
COMMON/C4AQ/SAVE/ATS(6),UTS(6)

DIMENSION ATS(6)

C ZERO TIME ACCUMULATOR.
TACCUM=0.0

IDNF=0

C SAV INITIAL MASS
TULMS=RULMS

C SET NOMINAL STEP SIZE IN BURN
DT=TYPE(1)

C SAVETIME AT END OF LAST BURN (=TIME AT START OF 1ST BURN OWARDS)
IF(1BOUNDA.*O) T0SAVE=TIMES(6)

C SET VARIABLES

N_LAST=0

LAST=0

NPOINT=0

IF(1BOUND=0.1) GO TO 100

C C 3-BURN, OUTBOUND MISSION, NO. IN BACKWARDS MODE.

C 3

C \C SET WEIGHTS.
C WEIGHT(1)=1 REFLECTS HARD CONSTRAINTS ON BACKWARDS BURN.

DU 1 1=1.5

WTS(1)=WEIGHT(1)

1

WEIGHT(1)=1.0

NC=0

C START MAIN GUIDANCE LOOP FOR FIRST BURN.

3

IF(N_LAST=0.1) LAST=1

IF(DABS(TIMES(6)-TIMES(5))LE.2.*UTYPE(1)) N_LAST=1

IF(N_LAST=0.1) JOUT=1

IF(N_LAST=0.1) DT=DABS(TIMES(6)-TIMES(5))/2.

IF(N_LAST=0.1) DT=DABS(TIMES(6)-TIMES(5))

CALL FORWD(0,-DT,0)

TIMES(6)=TIMES(6)-DT

IF(LAST=0.1) GO TO 0

33

NOP=1
FILE: BCBO

C SET INDEX FOR OUTPUT.

C APPLY CORRECTIONS TO UT. KEEP UT (TRUE UT) AT UNIT MAGNITUDE.

C APPLY CORRECTIONS TO ALL TIMES EXCEPT THE LAST, SINCE IT IS REALLY

C THE CLOCK TIME IN BACKWARDS MODE.

C END OF FIRST TURN. TURN MISSION AROUND.

C RESTORE THE WEIGHTS.

C CHANGE MISSION BACK TO NORMAL MODE.

BCBO056
BCBO057
BCBO058
BCBO059
BCBO060
BCBO061
BCBO062
BCBO063
BCBO064
BCBO065
BCBO066
BCBO067
BCBO068
BCBO069
BCBO070
BCBO071
BCBO072
BCBO073
BCBO074
BCBO075
BCBO076
BCBO077
BCBO078
BCBO079
BCBO080
BCBO081
BCBO082
BCBO083
BCBO084
BCBO085
BCBO086
BCBO087
BCBO088
BCBO089
BCBO090
BCBO091
BCBO092
BCBO093
BCBO094
BCBO095
BCBO096
BCBO097
BCBO098
BCBO099
BCBO100
BCBO101
BCBO102
BCBO103
BCBO104
BCBO105
BCBO106
BCBO107
BCBO108
BCBO109
BCBO110
FILE: BCBEB  FORTRAN P1

1. VEH(1:2)=-VEH(1:2).
   AMG=TULTMS=0*VEH(1:2)
   DO 8 I=1,3
   QO(I)=QT(I)
   QO(I+3)=-OT(I+3)
   XTemp1=XT(I)
   XTemp2=XT(I+3)
   XT(I)=X0(I)
   XT(I+3)=-X0(I+3)
   8 X0(I)=XTemp1

2. C. SET END CONDITIONS.
   CALL HVALS(XT(6),PIX. TV. =1)
   DU 9 I=1,6
   9 CC(I)=DD(I)
   CALL.CBCB
   RETURN

3. C
   C 3-BURN INBOUND MISSION (FORWARD MODE).
   C
   C
   C SET DU TO UNIT MAGNITUDE.
   100 UMA=DSQRT(QO(I)**2+QO(2)**2+QO(3)**2)
   DU-101 I=1,6
   101 QO(I)=QO(I)/UMA
   102 IF(NLAST. END.1) LAST=1
   IF(TIMES(2)-TIMES(1). LE. 2.-4DTYPE(1))=NLAST=1
   IF(NLAST. END.1) DT=(TIMES(2)-TIMES(1))/2.
   IF(NLAST. END.1) DT=TIMES(2)-TIMES(1)
   CALL FORWARD(U.0.0.1)
   IF(LAST. EQ.1) UPDATE=1
   TIMES(1)=TIMES(1)+DT
   IF(LAST. EQ.1) GO TO 106
   NOP=1
   MODE=0
   CALL GUIDE(U.0.0)
   IPRINT=0
   CK=-1.0
   CALL.CKSET(CK)
   C ADD CORRECTIONS TO DU. TIMES.
   DU 103 I=1,6
   QO(I)=QO(I)+CK*DU(I)
   103 TIMES(1)=TIMES(1)+DTIMES(1)*CK
   IF(NLAST. EQ.1 .OR. IOUT. EQ.1) NPOINT=NPOINT+1
   IF(NLAST. EQ.1 .OR. IOUT. EQ.1) CALL HAVOUT(1, NPOINT)
   GO TO 100
   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 h and e) with NBVAL=-1 to calculate the three components of the angular momentum vector h and the three components of the eccentricity vector e, pointing toward peri-gpee with magnitude of eccentricity. BVAL5 calls the subroutine COAST to obtain target state 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</td>
</tr>
<tr>
<td>for I=4 to 6</td>
<td>v</td>
<td>Final vehicle velocity</td>
</tr>
<tr>
<td>QF(I) for I=1 to 3</td>
<td>u</td>
<td>Final control vector</td>
</tr>
<tr>
<td>for I=4 to 6</td>
<td>ū</td>
<td>Final (du/dt)</td>
</tr>
<tr>
<td>PTV(I) for I=1 to 12</td>
<td>(\frac{\partial T_Y}{\partial y})</td>
<td>Partial derivatives of (T_Y) with respect to (y=(r^T,v^T,u^T,\dot{u}^T)^T) evaluated in BUZZ</td>
</tr>
<tr>
<td>TV</td>
<td>(T_Y)</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</td>
</tr>
<tr>
<td>for I=4 to 6</td>
<td>(e_d)</td>
<td>Desired eccentricity vector</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)) (I=1) to 12 (J=1) to JMAX1</td>
<td>(\frac{\partial y}{\partial z})</td>
<td>Partial derivatives of final (y = (r^T, v^T, u^T, T^T)) with respect to JMAX1 independent variables</td>
</tr>
<tr>
<td>JMAX1</td>
<td>JMAX</td>
<td>Number of independent variables</td>
</tr>
<tr>
<td>JLAST</td>
<td>JMAX1 + 1</td>
<td></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>(x_T(T))</td>
<td>Target state at time (T)</td>
</tr>
<tr>
<td>WT(I) for (I=1) to 6</td>
<td>(w)</td>
<td>Diagonal components of weighting matrix ranging from 0.0 to 1.0. ((W(I)=1.0) if the Ith end condition is a hard constraint. (W(I)=0.0) if the Ith 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) (J=1) to JMAX1</td>
<td>(\frac{\partial DC}{\partial \zeta})</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

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

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 $\text{MODE}=3$), the parameters $J_{\text{MAX1}}$ and $J_{\text{LAST}}$ 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 $E$.

$$
\Delta c = \begin{pmatrix}
    \Delta h^T K |H| \\
    \Delta E \left( \frac{R^2}{\mu} \right) \\
    \Delta e^T K \\
    \Delta h^T R |H| \\
    \Delta e^T R \\
    \Delta r^T K
\end{pmatrix}
$$

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

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

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

\[ J = \int_{t_0}^{t_f} |\dot{m}| dt + \frac{1}{2} \Delta c^T W \Delta c \quad (3) \]

is minimized. Here \( W \) is a 6x6 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

\[ \hat{p}_f = \left( \frac{\partial \Delta c}{\partial x} \right)^T W \Delta c \quad (4) \]

where \( \hat{p}_f^T = (u^T, -u^T) \) or equivalently the equations

\[ (I-w)B^T p_f = w \Delta c \quad \text{at} \quad X=X \quad (5) \]

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

\[ B(x)^T \left( \frac{\partial \Delta c}{\partial x} \right)^T \bigg|_{X=X} = I \quad (6) \]

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

\[ w_i = \frac{W_i}{1+W_i} \quad (7) \]

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 \( (I-w)B^T p_f \) given in terms of multiplying coefficients \( C_{ij} \) defined in the code are
The DC vector calculated in BVAL5 corresponds to the miss in satisfying Eq. (5)

\[
DC = wAc - (I-w)B^T \rho_f
\]  

(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 \( \frac{DC}{x} \) 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: CASUJ FORTRAN P1

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

SUBROUTINE BVALS CALLS BVALS (ANGULAR MOMENTUM AND ECCENTRICITY VECTORS), FROM INPUT STATE XF. IF NTVAL = 1, THEN THE D VECTOR (WEIGHTED COMBINATIONS OF TRANSVERSALITY AND MISS IN END CONDITIONS) AND THE E MATRIX (PARTIAL DERIVATIVES OF D VECTOR WITH RESPECT TO THE JMAX) INDEPENDENT VARIABLES ARE ALSO CALCULATED.

SUBROUTINE BVALS(XF, G, RTV, TV, NTVAL)
IMPLICIT REAL(*,*)
DIMENSION XF(3), U(3), V(3), W(3), X(3), Y(3), Z(3), DUM(2)
COMMON /BVALOUT/ XTF(3), DUM(2)
COMMON /G1010/ XTF(3), DUM(2), TO, AMO, VEH(10,7), UC(6),
1 TIMES(5), C(5),
2 COMMON /PHYS/ UK, KE, RARTH, RHO, DEGRA, OMEGA, G
COMMON /G1010/ DUM(2), DUM(2), DC12, Y(12), Z(12),
3 DUM(4),
4 COMMON /INDEX/ NARC, IARC, JMAX, JMAX1, JLAST, NO, NOP, NRKGDS
COMMON /GMODE/ M0, IFREV, ITSTOP
COMMON /CTWT/ +T(8)
DIMENSION G(8,6), UC(6), DUM(2), U(3), X(3), Y(3), Z(3)
DIMENSION VAU(3), VAU(3), DRU(12), DUM(2), DUM(2),
5 DC12, Y(12), Z(12),
6 JMAX1=JMAX1-1
7 JLAST=LAST-1
8 E(JMAX1+1)=0
9 N=1
0 SUBROUTINE GAUS IS CALLED TO PROPAGATE TARGET STATE TO FINAL TIME SUCH THAT THE PHASING MISS COMPONENT IN DC(6) CAN BE CALCULATED.
1 CALL GAUS(XF, DUM2, TIMES(6)-TT, XTF, DUM2, DUM1, DUM1)
2 DO 2 I=1,3
3 R(I)=XF(I)
4 V(I)=GF(I)
5 UD(I)=UF(I)
6 2 V(I)=VF(I)
7 R2=1.0/((R(1)+R(2)+R(3))
8 RM=USQRT(R2)
9 V2=V(1)+V(1)+V(2)+V(3)+V(3)
10 RTV=1.0+R(2)+R(3)+R(2)+R(3)
11 R-TV=RTV/UK
12 RTV2=R-TV
13 R-TV=RTV
14 2.0/2.0
15 DO 2 I=1,3
16 D(I)=R(I)*V(I)+V(I)*V(I)
17 D(2)=R(I)*V(I)+V(I)*V(I)
18 D(3)=R(2)*V(2)+V(2)*V(2)
19 D(4)=R(2)*V(2)+V(2)*V(2)
20 D(5)=R(2)*V(2)+V(2)*V(2)
21 D(6)=R(2)*V(2)+V(2)*V(2)
22 DC(6)=-RTV2(SELECT, TIMES(6)-TT, XTF, DUM2, DUM1, DUM1)

CAMBRIDGE MONITOR SYSTEM

BVA00010
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
FILE: CAS0 UV FORT RAN 11 CAMBRIDGE MONITOR SYSTEM

CALCULATE ECCENTRICITY VECTOR

DO 3 I=1,3

3 D(1+I)=-((R-M-V2U)*R(1)-(VU*V(1)

IF (MVAVL.EQ.0.) RETURN

H2=V2/R2-RTV2

HM=D3ORT(H2)

CF=10.5*V2-UK*RM

VUK=V2-RM*UK

DO 4 I=1,3

4 X(1)=V(1)/R2=(TV+X(1))/HM

HU=HM*UK

UK2=UK*RM

UKM=UK/UK-RTV2

H2MG=H2-RTV2

CF=10.5*V2-UK*RM

HC=HM*CF

CALCULATE REQUIRED OUT AND CROSS PRODUCTS.

RTU=R(1)*U(1)+R(2)*U(2)+R(3)*U(3)

RTUO=R(1)*UC(1)+R(2)*UC(2)+R(3)*UC(3)

VTU=V(1)*U(1)+V(2)*U(2)+V(3)*U(3)

HTU=0(1)*U(1)+0(2)*U(2)+0(3)*U(3)

HTUO=0(1)*UC(1)+0(2)*UC(2)+0(3)*UC(3)

RXU(1)=R(2)*U(3)-R(3)*U(2)

RXU(2)=R(3)*U(1)-R(1)*U(3)

RXU(3)=R(1)*U(2)-R(2)*U(1)

RXUO(1)=R(2)*UC(3)-R(3)*UC(2)

RXUO(2)=R(3)*UC(1)-R(1)*UC(3)

RXUO(3)=R(1)*UC(2)-R(2)*UC(1)

VXU(1)=V(2)*U(3)-V(3)*U(2)

VXU(2)=V(3)*U(1)-V(1)*U(3)

VXU(3)=V(1)*U(2)-V(2)*U(1)

VXUO(1)=V(2)*UC(3)-V(3)*UC(2)

VXUO(2)=V(3)*UC(1)-V(1)*UC(3)

VXUO(3)=V(1)*UC(2)-V(2)*UC(1)

CALCULATE REQUIRED COEFFICIENTS. C COEFFICIENTS MULTIPLY

C OUT PRODUCTS OF STATE AND CUSTATE IN TRANSVERSALITY CONDITIONS.

C 0-COEFFICIENTS ARE SCALAR MULTIPLIERS IN PARTIALS OF DC WITH

C RESPECT TO R AND V.

B11=WT(1)*TV/H2

B12=WT(1)*UH2*RC

C11=(1.0-WT(1))*K2

B21=WT(2)*HM

B22=WT(2)/UO2

C21=(1.0-WT(2))*UR2/CF

C22=0.5*K2

U3=(1.0-WT(3))*K3/HC

LFA=U3*(RTUO+CTU+NTU)

B31=K3*(HM)*UO-CTU

B32=WT(3)*RTV+UUK3*CTU+U2*RM

B33=WT(3)*RTV/(H2*RM)-U3*RTU

B34=WT(3)*H2MG/HU-CTU

C31=U3*RTV

C32=(1.0-WT(3))*UK2/HM

C33=0.5*K2

B41=WT(4)/HM
FILE: CASDVRJ FORTRAN PI CAMBRIDGE MONITOR SYSTEM

1. BVAL5 - 8

BVAL5 - 8
FILE:

BVA01120
BVA01130
BVA01140
BVA01150
BVA01160
BVA01170
BVA01180
BVA01190
BVA01200
BVA01210
BVA01220
BVA01230
BVA01240
BVA01250
BVA01260

C  CALCULATE PARTIALS OF DC WITH RESPECT TO R AND V.

DO 6 = 1, 6

G(1,1) = b11 + u1(1) - c11 * VU(1)
G(2,1) = -b12 * (1) + c12 * U(1)
G(2,2) = b22 * (1) + c22 * U(1)
G(3,1) = -d31 * V(1) - b31 * U(1) - c31 * U(1)
G(3,2) = b32 * V(1) + c32 * U(1)
G(5,1) = b51 * (1) - b52 * V(1) - c51 * U(1) - c52 * U(1)
G(5,2) = b53 * V(1) + c53 * U(1)
G(6,1) = b61 * (1) - c61 * V(1) - c62 * V(1)
G(6,2) = c61 * V(1) - c62 * V(1)

5 DRU(J) = R(1) * Z(J) + R(2) * Z(J) + R(3) * Z(J)

DVRU(J) = V(1) * Z(J) + V(2) * Z(J) + V(3) * Z(J)

C  PARTIAL OF COSTATE WITH RESPECT TO INDEPENDENT VARIABLES.

C (FIRST STEP OF CIN IN RULE)

E(1,1) = -c11 * HU(J)
E(2,1) = c21 * DRU(J) + c22 * DVRU(J)
E(3,1) = c31 * DRU(J) + c32 * DVRU(J)
E(4,1) = c41 * (1) * Z(J) + V(2) * Z(J) + D(J) * Z(12, J) - c42 * HU(J)
E(5,1) = c51 * DRU(J) + c52 * DVRU(J) - c53 * HU(J)
E(6,1) = c61 * (1) * Z(J) + V(2) * Z(J) + c62 * HU(J)

C  ADD IN STATE WITH RESPECT TO STATE TIMES PARTIAL

C  OF STATE WITH RESPECT TO INDEPENDENT VARIABLES.

DO 6 = 1, 6

C  CALCULATE MISS IN STATE CONSTRAINTS

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
FILE: CALLJVJ / FORTRAN I1

CAMBRIDGE MONITOR SYSTEM

WHERE DELTA REPRESENTS DESIRED MINUS ACTUAL AND CONSTRAINTS
ARE SCALED TO HAVE UNITS OF LENGTH.

\[ \text{DELT}(1) = (C(1) - \text{DELT}(1)) / H^M \]
\[ \text{DELT}(2) = (C(1) - C(2) + \text{DELT}(2)) / H^M \]
\[ \text{DELT}(3) = (C(1) - C(3) + \text{DELT}(3)) / H^M \]
\[ \text{DELT}(4) = (C(1) - C(4) + \text{DELT}(4)) / H^M \]

\( \text{CALCULATE SCALED COMBINATIONS OF TRANSVERSALITY} \)

\[ \text{DC}(1) = C(1) + \text{DELT}(1) \]
\[ \text{DC}(2) = C(2) + \text{DELT}(2) \]
\[ \text{DC}(3) = C(3) + \text{DELT}(3) \]
\[ \text{DC}(4) = C(4) + \text{DELT}(4) \]

\( \text{SUBROUTINE SOLVE(A,L,D,L7)}}

REAL*8 A(12,25)
DO 5 N=1,LO
M=N+1
IF(I.EQ.N) GO TO 7
DO 2 J=M,L7
Q=A(I*J+N) / A(I*J+N)
A(I*J+N) = A(I*J+N)
2 A(I*J+N) = A(I*J+N)
DO 3 = 1,LO
IF(I.EQ.N) GO TO 5
DO 4 = K,LO
4 A(I*K) = A(I*K) - A(I*J) / A(I*K)
3 CONTINUE
DO 6 = 6,LO
6 CONTINUE
RETURN
END

C

C SUBROUTINE BVAL4 X, YF, PT4, TV, TVBVAL

IMPLICIT REAL*8(A-N,1-2)
C
C THIS IS A FOUR-CONSTRAINT VERSION OF BVAL4. THE MISSION
C IS TO ACHIEVE AN ORBIT WITH GIVEN VALUES OF SEMIMAJOR AXIS.
C ECCENTRICITY, INCLINATION, AND ARGUMENT OF PERIGEE. THE ORBITAL
C CONSTANTS WHICH ARE TRANSMITTED IN C IN THE COMMON BLOCK G8IN
C ARE MAGNITUDE AND THIRD COMPONENT OF ORBITAL ANGULAR VELOCITY H.
C THE THIRD COMPONENT OF A VECTOR L-POINTING TOWARD PERICENTER. W1

BVAL5 - 9
**FILE: CASU**

**MAGNITUDE OF ELECTRICITY AND THE THIRD COMPONENT OF \( H \times E \)**

**COMM**

- /CPHY/ UK, XARTH, KHOO, KHHA, OMEGA, OBLATE
- /CINDEX/ HAC, IAHC, MAX, JMAX, JLAST, NU, NUP, NKGOS
- /GRID\( N \times T \times R \times U \times G \), TVA, TVM, TVL, TVC \( (6, 10, 7) \), O\( (6) \), TIMES \( (6) \), C\( (6) \)

**COMMON**

- /GRID\( D \times U \times O \times U \times G \), D\( (6, 9) \), TIMES \( (6) \), E \( (12, 13) \), Z \( (12, 12) \), D \( (6, 12) \)

**DIMENSION X \( (6) \), Y \( (6) \), Z \( (6) \), D \( (6) \)**

**C<<< 1 > calculate D \( (1) \) FOR 1 = 1 TO 4

\[
R2 = \frac{u}{(xf(1) * xf(1) + xf(2) * xf(2) + xf(3) * xf(3))}
\]

**R3 = (R2) \( ^2 \)**

\[
RTX = xf(1) * xf(4) + xf(2) * xf(5) + xf(3) * xf(6)
\]

\[
V2 = xf(4) * xf(4) + xf(5) * xf(5) + xf(6) * xf(6)
\]

\[
D(1) = (v2/z) * RTV = RTV
\]

\[
D(2) = xf(1) * xf(5) - xf(2) * xf(4)
\]

\[
H2 = D(1) * D(1)
\]

\[
RTV = RTV * RM
\]

\[
FV = h2/uk - 1/2
\]

D \( (4) = RTV * xf(3) \)

\[
D(4) = RTV * xf(3) * xf(3) / RM + h2 * xf(3) / UK
\]

\[
V2UR = v2/uk - RM
\]

\[
RTV = RTV / UK
\]

\[
D(3) = xf(3) * v2ur - xf(4) * RTV
\]

**IF (D(3) < 0) RETURN**

**C<<< 2 > calculate partial derivatives E \( (1, j) \)

\[
R3R = xf(3) * RM * R2
\]

\[
U3 = xf(6) / (v2/uk - 1/2)
\]

\[
V2H = v2/(d(1))
\]

\[
RTVH = RTV / (d(1))
\]

\[
R2H = 1/2 / (R2 * D(1))
\]

\[
TR3 = 2.0 * xf(3) / UK
\]

\[
CS1 = xf(6) * (v2ur + v2/uk) - RTV * R3R
\]

\[
CS2 = 2.0 * v3u / R2
\]

\[
F = xf(3) * RM * 2.0 * RTV * xf(6)
\]

\[
DU 4 1 = 13
\]

\[
G(1) = F * xf(1) + CS1 * xf(1)
\]

\[
A(1) = CS1 * xf(13) + CS1 * xf(1)
\]

\[
G(3) = G(3) + \text{RTV}
\]

\[
G(6) = G(6) + \text{FV}
\]

**DU - 1 J = 1 MAX1

**C<<< 2 > calculate V's and V's

\[
R21 = xf(1) * xf(2) * xf(3) * xf(4) * xf(5) * xf(6)
\]

\[
R24 = xf(1) * xf(4) * xf(5) * xf(6)
\]

\[
V21 = xf(4) * xf(4) * xf(4) * xf(5) * xf(5) * xf(5)
\]

\[
V24 = xf(4) * xf(4) * xf(4) * xf(5) * xf(5) * xf(5)
\]

**C<<< 2 > finish calculation of E \( (1, j) \)

\[
E(2, j) = xf(3) * (1, j) + xf(4) * (2, j) + xf(5) * (3, j)
\]

\[
E(3, j) = xf(6) * (1, j) + xf(6) * (2, j) + xf(6) * (3, j)
\]

\[
E(4, j) = g(1) * (1, j) + g(2) * (2, j) + g(3) * (3, j) + g(4) * (4, j)
\]

**SUM = 0

**OD 2 = K = 1, 12

**BVAL5-10**
FILE: CSBVJ FORTRAN P1

CAMBRIDGE MONITOR SYSTEM,

1. \( (0, J) = \text{SUM} \)

2. \( \text{SUM} = \text{SUM} + PTV(K) \times Z(K, J) \)

C***< Do the calculations in END CONDITIONS Del C

3. \( E(1, J) = C(1) - D(1) \)

4. \( E(6, J) = TV \)

5. \( E(s+13) = (X^F(3) \times X^F(1) - X^F(4) \times QF(2) - X^F(2) \times QF(4) + X^F(1) \times QF(5)) \)

RETURN
END