NBODY - A MULTIPURPOSE TRAJECTORY OPTIMIZATION COMPUTER PROGRAM

by William C. Strack
Lewis Research Center
Cleveland, Ohio 44135

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • FEBRUARY 1974
### Title and Subtitle
NBODY - A MULTIPURPOSE TRAJECTORY OPTIMIZATION COMPUTER PROGRAM

### Abstract
Documentation of the NBODY trajectory optimization program is presented in the form of a mathematical development plus a user's manual. Optimal multistage-launch-vehicle ascent trajectories may be determined by variational thrust steering during the upper phase. Optimal low-thrust interplanetary spacecraft trajectories may also be calculated with solar power or constant power, all-propulsion or embedded coast arcs, fixed or optimal thrust angles, and a variety of terminal end conditions. A hybrid iteration scheme solves the boundary-value problem, while either transversality conditions or a univariate search scheme optimize vehicle or trajectory parameters.
# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUMMARY</td>
<td>1</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>SOLAR SYSTEM MODEL</td>
<td>4</td>
</tr>
<tr>
<td>EPHEMERIDES</td>
<td>4</td>
</tr>
<tr>
<td>PHYSICAL DATA</td>
<td>4</td>
</tr>
<tr>
<td>VEHICLE MODELS</td>
<td>4</td>
</tr>
<tr>
<td>ELECTRICALLY PROPELLED VEHICLES</td>
<td>5</td>
</tr>
<tr>
<td>LAUNCH VEHICLES AND HIGH-THRUST OR BALLISTIC SPACECRAFT</td>
<td>10</td>
</tr>
<tr>
<td>PROGRAM LOGICAL STRUCTURE</td>
<td>11</td>
</tr>
<tr>
<td>LEVEL 1 - TRAJECTORIES AND VARIATIONAL NECESSARY CONDITIONS</td>
<td>12</td>
</tr>
<tr>
<td>Trajectory Equations</td>
<td>12</td>
</tr>
<tr>
<td>Primary gravitational body attraction</td>
<td>13</td>
</tr>
<tr>
<td>Perturbing body gravitational attraction</td>
<td>14</td>
</tr>
<tr>
<td>Aerodynamic forces</td>
<td>14</td>
</tr>
<tr>
<td>Thrust acceleration</td>
<td>16</td>
</tr>
<tr>
<td>Optimal Thrust Control</td>
<td>18</td>
</tr>
<tr>
<td>Optimum continuously variable thrust angle</td>
<td>20</td>
</tr>
<tr>
<td>Optimum choice of fixed thrust angles</td>
<td>21</td>
</tr>
<tr>
<td>Boundary Conditions</td>
<td>22</td>
</tr>
<tr>
<td>Flight time</td>
<td>22</td>
</tr>
<tr>
<td>Initial conditions</td>
<td>23</td>
</tr>
<tr>
<td>Arrival end conditions</td>
<td>24</td>
</tr>
<tr>
<td>Transversality conditions for optimum spacecraft parameters</td>
<td>26</td>
</tr>
<tr>
<td>The Two-Point Boundary-Value Problem</td>
<td>27</td>
</tr>
<tr>
<td>NOPT=0</td>
<td>28</td>
</tr>
<tr>
<td>NOPT=1</td>
<td>28</td>
</tr>
<tr>
<td>NOPT=2</td>
<td>28</td>
</tr>
<tr>
<td>NOPT=3</td>
<td>29</td>
</tr>
<tr>
<td>NOPT=4</td>
<td>29</td>
</tr>
<tr>
<td>NOPT=5</td>
<td>29</td>
</tr>
<tr>
<td>NOPT=6</td>
<td>29</td>
</tr>
<tr>
<td>NOPT=7</td>
<td>29</td>
</tr>
<tr>
<td>Additional features of the NOPT options</td>
<td>30</td>
</tr>
<tr>
<td>Partial Derivatives</td>
<td>31</td>
</tr>
<tr>
<td>Finite difference method</td>
<td>31</td>
</tr>
</tbody>
</table>
Page intentionally left blank
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analytical method</td>
<td>32</td>
</tr>
<tr>
<td>Iterator for Boundary-Value Problems</td>
<td>38</td>
</tr>
<tr>
<td>Convergence criterion</td>
<td>38</td>
</tr>
<tr>
<td>Linear correction scheme (Newton-Raphson)</td>
<td>39</td>
</tr>
<tr>
<td>Univariate search scheme</td>
<td>40</td>
</tr>
<tr>
<td>Integration Method</td>
<td>41</td>
</tr>
<tr>
<td>Runge-Kutta scheme</td>
<td>41</td>
</tr>
<tr>
<td>Trajectory interrupt</td>
<td>44</td>
</tr>
<tr>
<td>Choice of coordinate systems</td>
<td>46</td>
</tr>
<tr>
<td>Origin shift</td>
<td>46</td>
</tr>
<tr>
<td>Orbit element integration</td>
<td>46</td>
</tr>
<tr>
<td>Coordinate transformations</td>
<td>48</td>
</tr>
<tr>
<td>Earth-fixed coordinate frame</td>
<td>50</td>
</tr>
<tr>
<td>LEVEL 2 - DIRECT OPTIMIZATION OF VEHICLE AND MISSION</td>
<td></td>
</tr>
<tr>
<td>PARAMETERS</td>
<td>53</td>
</tr>
<tr>
<td>SWEEP SCHEMES FOR RUNNING SIMILAR CASES</td>
<td>55</td>
</tr>
<tr>
<td>Manual Sweeps</td>
<td>56</td>
</tr>
<tr>
<td>Automatic Sweeps</td>
<td>57</td>
</tr>
<tr>
<td>Multidimensional Sweeps</td>
<td>57</td>
</tr>
<tr>
<td>PROGRAM DESCRIPTION</td>
<td>58</td>
</tr>
<tr>
<td>PROGRAM STRUCTURE</td>
<td>58</td>
</tr>
<tr>
<td>INPUT</td>
<td>60</td>
</tr>
<tr>
<td>Vehicle Model</td>
<td>61</td>
</tr>
<tr>
<td>Analytic Spiral Escape Maneuver at Departure</td>
<td>63</td>
</tr>
<tr>
<td>Analytic High-Thrust Departure of Electric Vehicle</td>
<td>63</td>
</tr>
<tr>
<td>Analytic High-Thrust Capture Retromaneuver of Electric Vehicle</td>
<td>64</td>
</tr>
<tr>
<td>Departure Time</td>
<td>64</td>
</tr>
<tr>
<td>Initial Position and Velocity</td>
<td>64</td>
</tr>
<tr>
<td>Thrust Program Options</td>
<td>65</td>
</tr>
<tr>
<td>Trajectory Integration Controls</td>
<td>66</td>
</tr>
<tr>
<td>Output Controls</td>
<td>67</td>
</tr>
<tr>
<td>Trajectory Interrupt Controls</td>
<td>68</td>
</tr>
<tr>
<td>Level 1 Boundary-Value Problem</td>
<td>68</td>
</tr>
<tr>
<td>Level 2 Optimization</td>
<td>70</td>
</tr>
<tr>
<td>Parameter Sweeps</td>
<td>70</td>
</tr>
<tr>
<td>PROGRAM OUTPUT</td>
<td>71</td>
</tr>
<tr>
<td>Full Output Mode</td>
<td>71</td>
</tr>
<tr>
<td>One-Line Summary</td>
<td>73</td>
</tr>
<tr>
<td>EXAMPLE PROBLEMS</td>
<td>Page</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
</tr>
<tr>
<td>EXAMPLE 1 - JUPITER RENDEZVOUS USING THE MULTIDIMENSIONAL SWEEP FEATURE</td>
<td>73</td>
</tr>
<tr>
<td>EXAMPLE 2 - 0.1-AU SOLAR PROBE WITH A SWEEP ON SPECIFIC IMPULSE</td>
<td>73</td>
</tr>
<tr>
<td>EXAMPLE 3 - JUPITER CAPTURE MISSION WITH HIGH-THRUST DEPARTURE AND CAPTURE AND OPTIMUM VEHICLE PARAMETERS</td>
<td>77</td>
</tr>
<tr>
<td>SINGLE-STAGE LAUNCH VEHICLE WITH CHEMICAL AND NUCLEAR PROPULSION</td>
<td>82</td>
</tr>
<tr>
<td>APPENDIXES</td>
<td></td>
</tr>
<tr>
<td>A - SYMBOLS</td>
<td>96</td>
</tr>
<tr>
<td>B - SUBPROGRAM GLOSSARY</td>
<td>102</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>104</td>
</tr>
</tbody>
</table>
NBODY - A MULTIPURPOSE TRAJECTORY OPTIMIZATION COMPUTER PROGRAM

by William C. Strack

Lewis Research Center

SUMMARY

This report documents the NBODY computer program. NBODY calculates the performance and trajectories for a variety of space vehicles such as low-thrust electric spacecraft and multistage launch vehicles (three degrees of freedom). Thrust, n-body, and aerodynamic forces may be simulated through flexible vehicle and solar system models. The thrust steering program, for example, may be specified or optimized for maximum performance by using variational techniques. If coast arcs are permitted, the engine on-off times may be optimized also. The low-thrust spacecraft options include solar or nuclear power, two-body or n-body simulation, fixed or optimum thrust angles, analytic spiral escape or high-thrust departure and/or capture, and fixed or optimized vehicle parameters such as specific impulse (constant), initial acceleration, and launch speed. Parameter optimization is done with transversality conditions or a search procedure, depending on the particular set of parameters.

The trajectory integration is carried out with a variable-step-size, fourth-order Runge-Kutta technique with double-precision accumulation but single-precision derivative evaluation. Boundary-value problems are solved with a general-purpose iterator using a hybrid univariate search and linear correction scheme (modified multivariable Newton-Raphson scheme). The program is written FORTRAN IV and occupies about 20 K of IBM 7094 core storage exclusive of standard library routines. Both the mathematical description of the program and detailed operating instructions with examples are included in this report.

INTRODUCTION

The computer program NBODY is used to generate trajectories for low-thrust interplanetary spacecraft and high-thrust launch vehicles. It was originally developed as a general-purpose program for a wide variety of space mechanics problems (ref. 1). During the mid-1960's its evolution was directed toward the optimum low-thrust problem...
as the potential of electric rockets came to be more widely recognized. Today NBODY
is used mainly to calculate trajectories for electrically propelled spacecraft although it
still retains its earlier multipurpose capability, such as the calculation of multistage-
launch-vehicle trajectories. Extending NBODY’s capability to cover optimal low-thrust
trajectories required a considerable increase in the size and complexity of the program
and, although the revised program has been extensively exercised, no formal documenta-
tion was made available. This report provides such documentation in the form of a
user’s manual. It describes the assumed vehicle and solar system models, summarizes
the basic equations and program logic, and provides operating instructions.

The solar system model may be specified during input. It could be a simple two-
dimensional model with only a single point-mass gravitational body. Or it could be a
more accurate but more complicated model involving three dimensions, n bodies, and
a realistic Earth (atmosphere, rotation, and oblateness). The planetary positions are
determined by analytical time-dependent orbital elements.

The vehicle model is also specified during input and is generally either a low-thrust
electric spacecraft or a multistage launch vehicle, although any thrust level and as many
as 10 stages are permitted in either case. All vehicles are assumed to be point masses
and to operate at constant specific impulse. Either a constant or solar-electric power
profile may be chosen. The thrust direction program may be defined by the user or
optimized by the program to yield maximum net mass. The thrust attitude may be op-
timized over an infinite set of angles (a continuous thrust program) or over a finite set
of specified angles. In either case, the engine operation mode may be selected as con-
tinuously on or on-off with optimal switch times.

The NBODY program numerically integrates trajectories by using a fourth-order
Runge-Kutta scheme with automatic step-size control. The only exceptions occur when
the user wishes to calculate the planetary phases of an interplanetary trajectory for an
electric spacecraft with approximate closed-form solutions. This is common practice
when either a high-thrust or low-thrust Earth departure maneuver is part of a problem
since significant simplification results with only a relatively small sacrifice in accuracy
(refs. 2 to 4). For example, an electric spacecraft may be assumed to be boosted to
at least escape velocity by a launch vehicle of known performance. The numerical tra-
jectory integration begins in heliocentric space just outside the Earth’s sphere of
influence. Alternatively, a closed-form low-thrust spiral may be assumed for the
Earth-escape maneuver, with numerical integration again commencing just outside the
Earth’s sphere of influence.

Either flyby or orbiter trajectory modes may be selected for interplanetary prob-
lems. For orbiters, some or all of the arrival hyperbolic excess speed may be removed
with a closed-form, high-thrust retromaneuver.

For launch vehicle studies, the trajectory simulation consists of a zero angle-of-
attack phase followed by an optimally steered upper phase. This simulation is in accord
with the usual practice of limiting the angle of attack during atmospheric flight to reduce structural and heating loads.

Many trajectory problems require finding a set of initial conditions that permit a terminal set of conditions to be satisfied. This troublesome nonlinear two-point boundary-value problem is normally solved with an iterative linear correction scheme. Past experience here at the Lewis Research Center has shown that it is often helpful to program several iteration schemes to improve the chances of obtaining a solution. NBODY uses a univariate search scheme when the terminal set of conditions are far from being satisfied and a Newton-Raphson scheme when the solution is not far away. If either scheme fails to converge rapidly, an automatic shift to the other scheme takes place. The partial derivatives needed by the Newton-Raphson scheme may always be generated in NBODY by finite differencing. However, it is faster and more accurate to generate these partial derivatives by numerical integration. This option is available in NBODY for many typical problems but not all, since this method cannot be programmed to accommodate arbitrary end conditions as can the finite differencing method. Therefore, if the user selects a set of end conditions different from those for which the numerically integrated partials have already been programmed, he must either reprogram several sections of NBODY or resort to the finite difference method.

The problem of optimizing the thrust program to maximize gross payload is solved by using variational calculus. This method requires guessing a set of initial values for the adjoint variables (equivalent to guessing the initial thrust direction and its first derivative) that yield a trajectory not too far from the solution trajectory that satisfies certain end conditions. The user may also choose to optimize the central travel angle, the magnitude and direction of any hyperbolic excess speeds due to high-thrust launch or retrobraking of an electric spacecraft, the spacecraft specific impulse (assumed to be constant), and the initial mass flow rate. These options are also handled by variational calculus through transversality conditions. Or, if the user wishes, he may choose to optimize these or any other arbitrary variables with a simple search scheme. Transversality conditions are preferred whenever possible because their use has marked convergence speed and accuracy advantages.

It is often desirable to generate many solutions over a range of some arbitrary parameter, and provision has been made in NBODY to automatically "sweep" from one solution to others. Since problems often arise for which good starting guesses of the adjoint variables are lacking, a feature has been provided that sweeps a known solution of a related problem to the solution of the sought problem by a continuous transformation.

The NBODY program is written in many different subprograms in an effort to retain as much flexibility as possible. It is therefore possible to modify the program (the solar system model, vehicle model, etc.) with a minimum amount of difficulty. Its primary advantages compared to other trajectory programs are its relatively small size and broad capability. It is not specifically tailored to two-body, low-thrust interplanetary
trajectories as are HILTOP (ref. 5) and CHEBYTOP II (ref. 6) or to launch vehicle trajectories such as the program reported in reference 7. Still, even as a general-purpose tool, it has proven itself capable of handling most of the problems for which such special-purpose programs are designed. It is sized for running on a computer having 32 000 words of core storage.

SOLAR SYSTEM MODEL

EPHEMERIDES

Ephemeris data are needed in two-body problems if the user instructs the program to calculate initial and final end conditions to be identical with those of specified gravitational bodies. Ephemeris data are also needed in n-body problems where the perturbing bodies' positions need to be known at each point along the spacecraft's trajectory. Elliptic orbits are used to approximate the true paths. To increase the accuracy of this approximation, the orbital elements are computed as a function of the departure Julian date in accordance with the relations presented in reference 8. Prestored elliptic data for the solar system planets are referenced to the mean equinox and ecliptic of date. Data for bodies in addition to the planets (e.g., the Moon) may be added by amending subroutines WORDER and WORBEL. Also, for interplanetary problems involving an Earth departure specified in equatorial coordinates, the prestored elliptic ephemerides would have to be converted to a consistent equatorial framework by amending subroutine WORBEL.

PHYSICAL DATA

The assumed values of several astronomical constants and the planetary masses and sphere-of-influence radii are given in table I. These values are consistent with the Jet Propulsion Laboratory values given in reference 8. The 1962 U.S. Standard Atmosphere model (ref. 9) is programmed for the Earth in subroutine WICAO. Other atmosphere models may be simulated by altering this subroutine.

VEHICLE MODELS

The discussion of vehicle models is separated into two major parts: (1) electrically propelled low-thrust spacecraft and (2) non-electric-type vehicles including launch vehicles, high-thrust spacecraft, and ballistic spacecraft. Actually, while it is con-
convenient and logical to separate the discussion in this way, it should be noted that the program makes no internal distinction between these two types of vehicles. The inputs for mass, specific impulse, and so forth, are loaded into the same storage locations; and the integrated equations are identical. Thus, the user never inputs a single indicator that "tells" the program which vehicle model to use; instead, he supplies only the type of input data applicable to a particular vehicle model. For example, one normally does not think in terms of a multistage low-thrust electric vehicle; however, if one inputs three stage times, specific impulses, and so forth (as one might do in the case of a launch vehicle), the program will calculate a three-phase low-thrust trajectory. In general, then, all the features discussed in this section apply to either vehicle model. It is clearer, however, to discuss them in two, logically distinct sections.

ELECTRICALLY PROPELLED VEHICLES

The electric spacecraft is assumed to be composed of the following components:

1. Electric propulsion system, \( m_{ps} \)
2. Propellant mass, \( m_p \)
3. Tankage mass, \( m_t \)
4. Structure mass, \( m_s \)
5. Retropropulsion mass, \( m_r \)
6. Net spacecraft mass (gross payload), \( m_n \)

The net spacecraft mass refers to everything aboard the spacecraft not specified in this list. It includes the scientific instruments, communications equipment, control system, and so forth. The spacecraft mass at departure \( m_0 \) is just the sum of all these components,

\[
m_0 = m_{ps} + m_p + m_t + m_s + m_r + m_n
\]  

(1)

which can also be written in a form that facilitates scaling,

\[
\frac{m_n}{m_0} = 1 - \frac{m_{ps}}{m_0} - \frac{m_p}{m_0} - \frac{m_t}{m_0} - \frac{m_s}{m_0} - \frac{m_r}{m_0}
\]  

(2)

(All symbols are defined in appendix A.) This is the form actually programmed since the net mass ratio is usually the criterion to be maximized. The propulsion system mass ratio is computed from the electrical power available at 1 AU from the Sun \( P_r \) and the specific powerplant mass \( \alpha_{ps} \):
\[
\frac{m_{ps}}{m_0} = \frac{\alpha_{ps} P}{m_0} = -\frac{\dot{m}_0 c^2}{2\eta \frac{P_0}{P_r} m_0}
\]  
(3)

Here \( P_0 / P_r \) is the ratio of initial power to the 1-AU power; \( \dot{m}_0 \) is the initial flow rate; \( c \) is the input jet exhaust speed; and \( \eta \) is the overall propulsion system conversion efficiency, assumed to be a function of the jet exhaust speed,

\[
\eta = \frac{bc^2}{c^2 + d^2}
\]  
(4)

where \( b \) and \( d \) are input constants that reflect the assumed technology level. If the efficiency is constant, for example, \( b = \eta \) and \( d = 0 \).

The propellant mass is determined by integrating the mass flow rate \( \dot{m} \) over the entire trajectory

\[
m_p = - \int_{t_0}^{t} \dot{m} \, dt = - \int_{t_0}^{t} \epsilon \left( \frac{P}{P_r} \right) \dot{m}_0 \, dt
\]  
(5)

where \( \epsilon \) is a step function equal to unity if the engines are on and equal to zero if they are off. The initial flow rate \( \dot{m}_0 \) may be inputted directly or, if the user prefers, computed from an input value of the initial thrust-weight ratio \( f / m_0 g \)

\[
\dot{m}_0 = -a_0 \frac{m_0}{c} = -\left( \frac{f}{m_0 g} \right) \frac{m_0 g}{c}
\]  
(6)

or from an input value of the initial power \( P_0 \)

\[
\dot{m}_0 = -2\eta P_r \frac{P_0}{c^2}
\]  
(7)

The power ratio \( P / P_r \) may be chosen at input to simulate nuclear electric propulsion, in which case \( P / P_r = 1 \); or it may be chosen to simulate solar electric propulsion, in which case it is a function of the distance \( r \) from the Sun.
This relation is derived in reference 10; however, any other preferred model may be substituted by altering subroutine WPOWER.

The tankage mass is assumed to be proportional to the propellant mass

\[ \frac{m_t}{m_0} = k_t \left( \frac{m_p}{m_0} \right) \]  

and the structure mass is assumed to be proportional to the initial mass

\[ \frac{m_s}{m_0} = k_s \]  

Both \( k_t \) and \( k_s \) are input constants.

The retropropulsion mass component is really two components, one representing the retropropellant \( m_{rp} \) and the other representing tankage, engine, and other retropropulsion structure \( m_{rt} \) (assumed proportional to \( m_{rp} \)). Hence,

\[ \frac{m_{rp}}{m_0} = \left[ 1 - \frac{m_p}{m_0} - j \left( \frac{m_{ps}}{m_0} + \frac{m_t}{m_0} \right) \right] \left( 1 - e^{-\Delta v_r/c_r} \right) \]  

\[ \frac{m_{rt}}{m_0} = k_{rt} \frac{m_{rp}}{m_0} \]  

\[ \frac{m_r}{m_0} = \frac{m_{rp}}{m_0} + \frac{m_{rt}}{m_0} \]  

where \( j \) is a jettison indicator equal to unity if the electric propulsion system and tankage mass components are to be jettisoned prior to the retromaneuver and equal to zero if they are not, \( c_r \) is the retropropulsion jet exhaust speed (input), and \( \Delta v_r \) is the
magnitude of the retropropulsion velocity increment. The latter is assumed to be an
impulsive velocity change,

$$\Delta v_r = v_r - v_{c,r} \sqrt{1 + e_r}$$  \hspace{1cm} (14)$$

where \(v_r\) is the planetocentric velocity at periapsis before the retrofire (input), \(v_{c,r}\)
is the planetocentric circular orbit velocity at periapsis (input), and \(e_r\) is the eccen-
tricity of the planetocentric elliptic orbit (input). Note that \(v_{c,r}\) and \(e_r\) specify the
desired planetary orbit, while \(v_r\) controls the amount of high-thrust braking and is
usually free for optimization.

It often happens that a user wishes to include the Earth escape phase as part of the
overall optimization of the net spacecraft mass \(m_n\). However, it is exceedingly diffi-
cult to obtain solutions to such problems because of the extreme sensitivity of the as-
sociated two-point boundary-value problem. To avoid this difficulty, two options are
available in NBODY that involve departure-phase approximations that are generally re-
garded as sufficiently accurate for preliminary analysis. The first option is a high-
thrust launch to at least escape energy, and the second option is a low-thrust escape
spiral. If the high-thrust option is chosen, the net spacecraft mass ratio is redefined
in terms of the launch vehicle's payload capability in a low Earth circular parking orbit
\(m_{\text{ref}}\)

$$\frac{m_n}{m_{\text{ref}}} = \frac{m_0}{m_{\text{ref}}} \frac{m_n}{m_0}$$  \hspace{1cm} (15)$$
The launch vehicle's mass ratio is assumed to obey the following relation:

$$\frac{m_0}{m_{\text{ref}}} = (1 + k_l)e^{-(v_l-v_{c,l})/c_l} - k_l$$  \hspace{1cm} (16)$$

where \(v_l\) is the launch velocity relative to the Earth, \(v_{c,l}\) is the circular orbit veloc-
ity of the low Earth parking orbit (e.g., at 185-km altitude), and \(c_l\) and \(k_l\) are input
constants characterizing the launch vehicle performance. This equation is a curve fit to
published launch vehicle performance curves, assuming impulsive velocity addition be-
yond the initial low Earth orbit. While \(c_l\) and \(k_l\) appear to be the launch vehicle's
exhaust speed and propellant-sensitive hardware fraction, they are in fact, merely
curve-fit parameters that only coincidently may be close to the actual values for these
parameters. To obtain their values from a specified performance curve, select two \(v_l\)
values, note the corresponding $\frac{m_0}{m_{\text{ref}}} \cdot \text{ref}$ values, and solve equation (16) for $c_l$ and $k_l$. A particularly simple solution exists if the velocity increments $\Delta v = v_l - v_{c, l}$ are chosen such that $\Delta v_2 = 2 \Delta v_1$, as illustrated in sketch (a), then

$$k_l = \frac{\left(\frac{m_0}{m_{\text{ref}}}_1\right)^2 - \left(\frac{m_0}{m_{\text{ref}}}_2\right)}{1 + \left(\frac{m_0}{m_{\text{ref}}}_2\right)^2 - 2\left(\frac{m_0}{m_{\text{ref}}}_1\right)}$$

(17)

and

$$c_l = \ln \left[ \frac{1 + k_l}{\left(\frac{m_0}{m_{\text{ref}}}_1\right)^2 + k_l} \right]$$

(18)

(a)
Using this simple scheme ordinarily results in an adequate representation of launch vehicle performance for preliminary design analyses. The launch velocity \( v_l \) is an input variable subject to internal change if it is selected as an optimization variable, as is usually the case. In effect, choosing \( v_l \) is equivalent to choosing the initial spacecraft mass \( m_0 \). Note that \( m_{\text{ref}} \) does not need to be specified before trajectories are integrated if the initial acceleration \( a_0 \) is input directly or if \( \dot{m}_0 \) and \( m_0 \) are input together. In such cases the net mass ratios are evaluated and the absolute net mass calculated afterwards, if so desired, by multiplying by any \( m_{\text{ref}} \). On the other hand, if the initial power \( P_0 \) is input, it is also necessary to input \( m_{\text{ref}} \) to determine \( \dot{m}_0 \) and \( m_0 \) (eqs. (7) and (16)). Solutions obtained with this option may be scaled by keeping \( P_0/m_{\text{ref}} \) constant.

If the closed-form tangential, low-thrust spiral escape option is chosen, equation (15) is still used but with a different form for \( m_0/m_{\text{ref}} \):

\[
\frac{m_0}{m_{\text{ref}}} = 1 - \xi \left( 1 - e^{-v_{\text{c},i}/c} \right)
\]

where \( \xi \) is an empirical correction factor dependent on the thrust-weight ratio \( a_0/g \), curve fitted from reference 11:

\[
\xi = 0.28988 - 0.14084 \left( \frac{a_0}{g} \right) - 0.010483 \left( \frac{a_0}{g} \right)^2 - 0.00028355 \left( \frac{a_0}{g} \right)^3
\]

The low-thrust spiral escape option is permitted when inputting either the initial flow rate \( \dot{m}_0 \) or the initial thrust-weight ratio \( a_0/g \), but not when inputting the initial power \( P_0 \).

LAUNCH VEHICLES AND HIGH-THRUST OR BALLISTIC SPACECRAFT

This section discusses features normally associated with non-electric-type vehicles such as launch vehicles and ballistic or high-thrust spacecraft. As many as 10 trajectory phases are permitted, each specified by its flight time \( t_f \), initial mass \( m_0 \), vacuum specific impulse \( I \), and mass flow rate \( \dot{m}_0 \). These phases may be defined by actual vehicle staging or by a change in thrust vector control. Atmospheric flight may be simulated by also including the aerodynamic reference area \( S_{\text{ref}} \), the engine exit area \( A_e \), and lift and drag coefficient tabular data.

Between phases the vehicle mass may remain unchanged, be set to a new value, or decremented a fixed amount. The payload ratio is the same as that defined as net space-
craft mass for electric vehicles (eq. (2)) with the absence of the electric powerplant and planetary retropropulsion terms.

The thrust magnitude of each phase is assumed to be

\[ f = -m_0I_g - pA_e \]  \hspace{1cm} (21)

where \( p \) is the atmospheric pressure and \( A_e \) is the input engine exit area. Instead of inputting the mass flow rate \( \dot{m}_0 \), the user may input the initial vacuum thrust-weight ratio \( f/m_0g \), in which case \( \dot{m}_0 \) is calculated internally from

\[ \dot{m}_0 = \frac{\left( \frac{f}{m_0g} \right)m_0g + pA_e}{I_g} \]  \hspace{1cm} (22)

The vehicle drag coefficient is composed of a parasitic component \( C_{D0} \) and an induced component \( C_{DI} \). These coefficients are assumed to be quadratic functions of Mach number \( M \),

\[ C_D = C_{D0} + C_{DI} \]  \hspace{1cm} (23)

\[ C_{D0} = a_1 + a_2M + a_3M^2 \]  \hspace{1cm} (24)

\[ C_{DI} = (a_4 + a_5M + a_6M^2)c_L^2 \]  \hspace{1cm} (25)

where the \( a_i \) coefficients are input in sets that apply to specific intervals of \( M \). The lift coefficient \( C_L \) is determined in a similar manner

\[ C_L = (a_7 + a_8M + a_9M^2)\sin \alpha \]  \hspace{1cm} (26)

Here \( \alpha \) is the vehicle angle of attack, which is identical to the angle between the thrust and relative velocity vectors because of the implied assumption that the engine thrust is always aligned along the vehicle longitudinal axis.

**PROGRAM LOGICAL STRUCTURE**

The program NBODY is structured logically in what may be called three levels of operation. By analogy, these levels may be thought of as a set of three nested DO
loops in FORTRAN programming. In the first level (analogous to an innermost DO loop), trajectories are integrated and an iteration scheme is available to solve two-point boundary-value problems. Thus, trajectories are found that satisfy specified terminal constraints such as fixed position and velocity. Every trajectory is integrated, including those for purely ballistic spacecraft. In addition to finding trajectory solutions, four vehicle-related parameters - specific impulse, initial mass flow rate, launch hyperbolic excess speed (equivalent to initial spacecraft mass), and high-thrust retropropulsion velocity increment (equivalent to retropropellant) - may be optimized in level 1 by incorporating the required transversality conditions into the two-point boundary-value problem.

Level 2 (analogous to a middle DO loop) permits direct optimization of either trajectory- or vehicle-related variables. The user selects the optimization criterion and the set of independent variables from among all variables computed by the program. If the user alters the net mass equation already programmed without also altering the vehicle-related transversality conditions, he must use level 2 to optimize specific impulse, acceleration, and so forth. Each time a level 2 independent variable is changed, the level 1 trajectory calculations are repeated, including the two-point boundary-value iteration.

Level 3 (analogous to an outermost DO loop) involves running several cases in succession with parameter sweep capability. Level 1 and level 2 calculations are repeated each time a level 3 variable is altered. Thus, variables that are optimized in level 2 may be reoptimized during a sweep on, for example, mission time.

LEVEL 1 - TRAJECTORIES AND VARIATIONAL NECESSARY CONDITIONS

Trajectory Equations

The forces included in the trajectory simulation are gravitational forces of the Sun and the planets, thrust forces, and aerodynamic forces. These forces are vectorially summed as a resultant total force on the assumed point-mass vehicle relative to a primary center of attraction. The vector equation of motion is

\[ \ddot{R} = -\nabla u - \sum_{i=2}^{n} \mu_i \left( \frac{R - R_i}{|R - R_i|^3} + \frac{R_i}{r_i^3} \right) + \frac{D}{m} + \frac{L}{m} + aT \]  

(Total) = (Primary body) + (Perturbing bodies) + (Drag) + (Lift) + (Thrust)
The convention of using capital letters to denote vectors and lower case letters to denote scalars is adopted in this report except where it interferes with well-known symbols.

Here $\mathbf{R}$ is the vehicle's position vector (and $r$ is the magnitude of $\mathbf{R}$) relative to the origin of the coordinate system - located at the center of the primary body as shown in sketch (b); $\mathbf{R}_i$ is the position vector of the $i^{\text{th}}$ perturbing body; $\mu_i$ is the body's gravitational constant; $m$ is the vehicle mass; and $\mathbf{T}$ is a unit vector in the thrust direction.

![Diagram of primary gravitational body attraction](image)

**Primary gravitational body attraction.** - The first term $\nabla u$ in the acceleration equation denotes the gradient of the gravitational potential function $u = u(x, y, z)$ of the primary body. A point-mass body may be selected, in which case $u = -\mu/r$ and

$$\nabla u = \left[ \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial u}{\partial z} \right] = \frac{\mu}{r^3} \mathbf{R}$$ \hspace{1cm} (28)

Or, in the case of the Earth, an oblate potential function may be selected (from ref. 8)

$$u = -\frac{\mu}{r} \left[ 1 - \frac{J_2}{2} \left( \frac{a_e}{r} \right)^2 (3 \sin^2 \varphi - 1) - \frac{J_3}{2} \left( \frac{a_e}{r} \right)^3 (5 \sin^3 \varphi - 3 \sin \varphi) \right. \right. \hspace{1cm} (29)$$

\begin{align*}
&\quad \quad \quad - \left. \left. \frac{J_4}{8} \left( \frac{a_e}{r} \right)^4 (35 \sin^4 \varphi - 30 \sin^2 \varphi + 3) \right] \right]
\end{align*}
where $a_e$ is the equatorial radius of the Earth; $\varphi$ is the vehicle's geocentric latitude relative to the Earth's equatorial plane; and $J_2$, $J_3$, and $J_4$ are zonal harmonic coefficients whose values are given in table I (from ref. 8). In this case,

$$
\frac{\mathbf{a}}{\partial x} = \frac{\mu}{r^2} \left[ 1 - \frac{3}{2} J_2 \left( \frac{a_e}{r} \right)^2 \left( 5 \sin^2 \varphi - 1 \right) - \frac{5}{2} J_3 \left( \frac{a_e}{r} \right)^3 \left( 7 \sin^2 \varphi - 3 \right) \sin \varphi \right]
$$

$$
- \frac{35}{8} J_4 \left( \frac{a_e}{r} \right)^4 \left( 9 \sin^4 \varphi - 6 \sin^2 \varphi + \frac{3}{7} \right) x - y
$$

$$
\frac{\mathbf{a}}{\partial z} = \frac{\mu}{r^2} \left[ 1 - \frac{3}{2} J_2 \left( \frac{a_e}{r} \right)^2 \left( 5 \sin^2 \varphi - 3 \right) - \frac{5}{2} J_3 \left( \frac{a_e}{r} \right)^3 \left( 6 - 7 \sin^2 \varphi - \frac{3}{5} \frac{1}{\sin^2 \varphi} \right) \sin \varphi \right]
$$

$$
- \frac{35}{8} J_4 \left( \frac{a_e}{r} \right)^4 \left( 9 \sin^4 \varphi - 10 \sin^2 \varphi + \frac{15}{7} \right) \frac{z}{r}
$$

Note that the oblateness forces are referenced to the Earth's equatorial plane. Thus, if oblateness terms are to be considered, the integrating inertial coordinate frame must be chosen as an equatorial frame or the user must program a coordinate transformation.

**Perturbing body gravitational attraction.** - The second term in equation (27) represents the acceleration caused by $n - 1$ perturbing bodies. The bodies are all assumed to be point masses, and their positions are determined by ephemerides as explained previously in the section **SOLAR SYSTEM MODEL**.

**Aerodynamic forces.** - Aerodynamic forces are split into drag and lift components with the usual definitions. Drag is opposite the relative wind vector and lift is perpendicular to the relative wind as shown in sketch (c). Since drag is opposite the relative wind velocity $V_r$

$$
D = -(C_D q S_{ref}) \frac{V_r}{v_r}
$$

where $v_r$ is the magnitude of $V_r$ and the dynamic pressure $q$ is a function of atmospheric density $\rho$:

$$
q = \frac{1}{2} \rho v_r^2
$$
The relative velocity $V_r$ is referenced to the primary body, which is assumed to rotate counterclockwise about the $z$-axis. Hence,

$$v_{r,x} = v_x + \omega_r y$$  \hspace{1cm} (34a)

$$v_{r,y} = v_y - \omega_r x$$  \hspace{1cm} (34b)

$$v_{r,z} = v_z$$  \hspace{1cm} (34c)

where the subscripts refer to the $x, y, z$ components and $\omega_r$ is the rotation rate of the primary body and its atmosphere.

To compute the lift vector in the $x, y, z$ inertial frame, it is convenient to first define the relative angular momentum vector per unit mass $H_r$

$$H_r = R \times V_r$$  \hspace{1cm} (35)

and then define another vector $B$ such that

$$B = V_r \times H_r$$  \hspace{1cm} (36)

Note that $H_r$ is normal to the $R \times V_r$ plane; $B$ is within the $R \times V_r$ plane; and $V_r$, $H_r$, and $B$ form an orthogonal set as shown in sketch (c). The lift vector $L$ can be resolved along $V_r$, $H_r$, and $B$ as follows:
\[ L \cdot V_r = 0 \]  
(37a)

\[ L \cdot H_r = (l \sin \beta)H_r \]  
(37b)

\[ L \cdot B = (l \cos \beta)B \]  
(37c)

where \( \beta \) is the out-of-orbit (relative orbit) thrust angle (sketch (c)) and the lift magnitude \( l \) is

\[ l = C_L q S_{ref} \]  
(38)

Solving these equations for \( L \) yields

\[ L = l \sin \beta \frac{H_r}{|H_r|} + l \cos \beta \frac{B}{|B|} \]  
(39)

The lift and drag coefficients \((C_D \text{ and } C_L)\) are tabular input data as explained in the section VEHICLE MODELS.

**Thrust acceleration.** The fourth term in equation (27) is the thrust acceleration \( a_T \). The thrust acceleration magnitude is, in general,

\[ a = -\epsilon \left( \frac{c_m 0}{m} \right) \left( \frac{P}{P_r} \right) - \frac{p A_e}{m} \]  
(40)

where the second term is absent for exoatmospheric flight and the power ratio \( P/P_r \) is unity except for solar electric propulsion, as explained earlier in the section VEHICLE MODELS. The engine on-off switch parameter \( \epsilon \) is unity for engine-on operation and zero for engine-off operation. It is needed in this equation only if the user selects the optimum thrust-coast profile option, in which case \( \epsilon \) is calculated internally by the program.

The unit thrust vector \( T \) determines the thrust direction, and the user selects either an optimum \( T \) program or a specified \( T \) program. For a specified \( T \) program, the angle between the thrust force and the relative velocity (sketch (c)) is assumed to be a quadratic function of time

\[ \alpha(t) = a_{10} + a_{11} t + a_{12} t^2 \]  
(41)
where the $a_i$ coefficients are input in sets that apply to specific time intervals. The out-of-plane thrust component is determined by the angle $\beta$, previously defined in the discussion of the lift force as an input constant (sketch (c)). The unit thrust vector can be resolved along the $V_r$, $H_r$, and $B$ axes similar to the resolution of the lift force along these axes

\[
T \cdot V_r = V_r \cos \alpha 
\]

\[
T \cdot H_r = H_r \sin \alpha \sin \beta 
\]

\[
T \cdot B = B \sin \alpha \cos \beta 
\]

Thus,

\[
T = \cos \alpha \frac{V_r}{|V_r|} + \sin \alpha \sin \beta \frac{H_r}{|H_r|} + \sin \alpha \cos \beta \frac{B}{|B|} 
\]

Instead of referencing the thrust angle to the relative velocity vector, the user may alternatively select to reference it to the circumferential direction as shown in sketch (d). In this case he specifies $\alpha_c$ - the angle from the forward circumferential direction to the thrust vector - as a function of time as in equation (41). The program then subtracts the path angle $\gamma$ from $\alpha_c$ to determine $\alpha$:

\[
\alpha = \alpha_c - \gamma 
\]
and resolves the thrust, as before, using equation (43). This option is most useful for interplanetary missions where $\omega_T = 0$ (hence, $V$ replaces $V_T$ in sketch (d)) and the thrust orientation is often conveniently given in terms of the circumferential direction. A brief summary of frequently encountered thrust programs is given in the following table:

<table>
<thead>
<tr>
<th>Thrust program</th>
<th>Required angles, deg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tangential thrust (forward)</td>
<td>$\alpha = 0, \beta = 0$</td>
</tr>
<tr>
<td>Tangential thrust (rearward)</td>
<td>$\alpha = 180, \beta = 0$</td>
</tr>
<tr>
<td>Circumferential thrust (forward)</td>
<td>$\alpha_c = 0, \beta = 0$</td>
</tr>
<tr>
<td>Radial thrust (outward)</td>
<td>$\alpha_c = 90, \beta = 0$</td>
</tr>
<tr>
<td>Radial thrust (inward)</td>
<td>$\alpha_c = -90, \beta = 0$</td>
</tr>
<tr>
<td>Normal thrust (upward)</td>
<td>$\alpha_c = 0, \beta = 90$</td>
</tr>
<tr>
<td>Normal thrust (downward)</td>
<td>$\alpha_c = 0, \beta = -90$</td>
</tr>
</tbody>
</table>

If the user selects an optimum T program, variational calculus is employed to determine $T(t)$. This is rather involved and is the subject of the following section. The user could, of course, closely approximate the optimum $T(t)$ without variational calculus by using the built-in generalized search procedure to optimize $\beta$ and the $a_i$ coefficients in the $a(t)$ equation. Since $\beta$ is programmed as a constant, this could only be done for two-dimensional cases. Moreover, such a direct optimization procedure is generally inadequate unless the independent function (e.g., $a(t)$) is subdivided many times, which in turn greatly increases the number of independent variables and slows down the search procedure significantly. Nevertheless, it may be applicable whenever the optimum thrust angle is fairly constant or when the thrust angle is constrained.

Optimal Thrust Control

Since the application of optimal control theory is especially complicated if oblateness and aerodynamic forces are present, these effects are not included in the optimal control formulation. This is quite acceptable for interplanetary transfers, of course, and usually so for preliminary launch vehicle studies. If the user should request optimal thrust control with oblateness or aerodynamic forces present, the equations of motion will account for such forces but the optimal control law will not. Hence, the trajectory will not be truly optimal.

With these restrictions the optimal control law formation is based on a simplified version of the equations of motion, namely

18
\[ \dot{V} = -\frac{\mu}{r^3} R - \sum_{i=2}^{n} \mu_i \left( \frac{R - R_i}{|R - R_i|^3} + \frac{R_i}{r_i^3} \right) - \frac{c}{m} \dot{m} T \] (45a)

\[ \dot{R} = V \] (45b)

where \( V \) is the vehicle's absolute velocity. The mass equation is

\[ \dot{m} = \epsilon \dot{m}_0 \zeta \] (45c)

where \( \zeta \) is the power ratio \( P/P_r \). These three equations define seven state variables - the three components of position and velocity, and the vehicle mass. The four parameters \( \dot{m}_0, c, v_l, \) and \( v_r \) may also be treated as state variables in order to optimize them with the variational method. To do this, four more state equations are appended to the preceding set.

\[ (\dot{\dot{m}}_0) = \frac{d}{dt} (\dot{m}_0) = 0 \] (45d)

\[ \dot{c} = 0 \] (45e)

\[ \dot{v}_l = 0 \] (45f)

\[ \dot{v}_r = 0 \] (45g)

The necessary conditions for maximizing the net spacecraft mass are determined by variational principles (ref. 12). These conditions determine the optimum thrust orientation; engine on-off switch times; and values of \( \dot{m}_0, c, v_l, \) and \( v_r \). Part of these conditions are differential equations known as Euler-Lagrange or adjoint equations which are

\[ \dot{\Lambda} = -\Lambda \] (46a)

\[ \dot{\Lambda}_r = \sum_{i=1}^{n} \frac{\mu_i}{r_i^3} \left[ \Lambda - \frac{3}{r_i^2} (\Lambda \cdot R_i) R_i \right] + \left( \frac{\epsilon \dot{m}_0 \zeta' \kappa}{r} \right) R \quad (R_1 = R) \] (46b)
\[ \dot{\lambda}_m = - \frac{\dot{m} c \lambda}{m^2} \quad (46c) \]

\[ \dot{\lambda}_{m0} = \frac{\dot{m} \kappa}{m_0} \quad (46d) \]

\[ \dot{\lambda}_c = \frac{\dot{m} \lambda}{m} \quad (46e) \]

\[ \dot{\lambda}_{v_l} = 0 \quad (46f) \]

\[ \dot{\lambda}_{v_r} = 0 \quad (46g) \]

where

\[ \kappa = \frac{c \lambda}{m} - \lambda_m \quad (47) \]

The subscripts here denote which state variable the adjoint variables \( \lambda_i \) are associated with. The term \( \Lambda_T \) is a vector with components \( \lambda_{x}, \lambda_{y}, \) and \( \lambda_{z} \). Likewise, \( \Lambda \) is a three-component vector associated with velocity (it is not subscripted in keeping with its usual notation) and is customarily referred to as the primer vector.

**Optimum continuously variable thrust angle.** - Variational theory shows that the optimum value of \( T \) is determined by the primer vector:

\[ T = \frac{\Lambda}{\lambda} \quad (48) \]

where \( \lambda \) is the magnitude of \( \Lambda \). The engine on-off indicator \( \epsilon \) is determined as follows:

\[ \epsilon = 0 \quad \text{if} \quad \kappa < 0 \quad (49a) \]

\[ \epsilon = 1 \quad \text{if} \quad \kappa > 0 \quad (49b) \]

The theoretical possibility of \( \kappa = 0 \) over a finite time interval very seldom occurs in practice, and this case presents no problems. If \( \kappa = 0 \) at the initial time, \( \dot{\kappa} \) is inter-
rogated to determine the proper initial value of $\epsilon$. The adjoint equations must be integrated along with the equations of motion to determine the optimum values of $T$ and $\epsilon$. So far, we have not discussed how the optimization of $m_0$, $c$, $v_l$, and $v_r$ is accomplished, nor how the initial values of the adjoint variables $\lambda_i$ are determined. Values for these variables are determined by the transversality conditions, which are dependent on the form of the net mass equation and the desired end conditions. Hence, a separate discussion of these conditions is presented later. It is sufficient for the moment to note that the adjoint equations are independent of these conditions. If any of the variables $m_0$, $c$, $v_l$, or $v_r$ are fixed instead of free for optimization, its corresponding state equation (eq. (45)) and adjoint equation (eq. (46)) are deleted. Conversely, the adjoint equations for any free variables are retained as they are needed in the evaluation of the transversality conditions.

**Optimum choice of fixed thrust angles.** - For two-dimensional problems, the user may choose to restrict the thrust angle to a finite set of fixed input values $\alpha_i$ referenced either to the velocity vector or to the circumferential direction, as explained earlier. To determine which of the input angles should be used at any instant of time, the variational equations just presented are employed, with the exceptions that $T \cdot \Lambda$ is substituted at every occurrence of the scalar $\lambda$ (eqs. (46c), (47), and several subsequent equations) and that equation (48) is replaced with

$$T \cdot \Lambda = \max_i (T_i \cdot \Lambda)$$

When this criterion is used, the program automatically switches the thrust angle whenever the difference between the two largest values of $T_i \cdot \Lambda$ reaches zero. The value of $T_i$ depends on $\alpha_i$; and since this option is programmed only for the two-dimensional case of trajectories in the x-y plane,

$$\Lambda \cdot T_i = \lambda_x \cos \psi_i + \lambda_y \sin \psi_i$$

where

$$\psi_i = \frac{\pi}{2} + \theta - (\alpha_i + \gamma)$$

Here $\psi_i$ is the thrust angle referenced to the x-axis and $\theta$ is the travel angle as shown in sketch(e). If the thrust angle is referenced to the forward circumferential direction ($\alpha_c$ in sketch (d)), then $(\alpha_c)_i$ replaces $\alpha_i + \gamma$ in equation (52).
Boundary Conditions

Every problem involves certain boundary conditions that must be satisfied to obtain the desired solution. In some cases these are fixed, such as the initial position and velocity vectors. But in other cases, some of the boundary conditions are in the form of transversality conditions. These conditions arise whenever an end condition is not fixed but left open for optimization. The transversality equations are derived from variational theory (ref. 12) and are presented herein without proof.

**Flight time.** - The time at departure \( t_0 \) is an input constant \( \bar{t}_0 \)

\[
t_0 = \bar{t}_0 \quad \text{(Departure time)}
\]  

The departure time is usually zero except for interplanetary problems involving actual planetary positions as a function of date. The arrival (or final) time is also a constant

\[
t_a = t_0 + t_f \quad \text{(Arrival time)}
\]  

where \( t_f \) is an input mission time. It should be noted at this point that although the boundary conditions are often stated herein in terms of input constants, such as \( t_f \), the user always has the power to override this choice and declare such boundary conditions to be variable. For example, in typical launch vehicle problems the flight time \( t_f \) is not a fixed constant but an independent variable used in an iteration scheme to obtain certain orbit insertion conditions. How this is done is explained in the section The Two-Point Boundary Value Problem.

The flight time for low-thrust interplanetary missions must be defined in detail if
any of the closed-form planetocentric simulations are involved. The planetocentric flight
times are ignored for both high-thrust closed-form simulations; that is, the launch ve-

cle boost phase and the planetary capture phase (if any). In these cases, \( t_0 \) and \( t_a \)
refer to the heliocentric portion of the flight, only. In the case of the closed-form low-
thrust tangential escape spiral, \( t_0 \) and \( t_a \) again refer only to the heliocentric portion
of the mission. But the program also calculates the spiral time required for the vehicle
to reach escape velocity

\[
\begin{align*}
t_s &= \xi \left( \frac{c}{a_0} \right) \left( 1 - e^{-v_c, t'/c} \right) \\
&= \xi \left( \frac{c}{a_0} \right) \left( 1 - e^{-v_c, t'/c} \right)
\end{align*}
\]

(55)

where \( \xi \) is an empirical function of \( a_0 \) previously defined in equation (20). This time
is added to the user-supplied heliocentric flight time \( t_f \) and printed out as total mission
time. This method of handling the low-thrust spiral is acceptable since the known opti-
mum trajectory for low-thrust escape from a circular orbit approximates a tangential
thrust spiral (ref. 2). The main drawback in this method is that the total mission time
\( (t_f + t_s) \) is a dependent rather than an independent variable. (The user selects \( t_f \),
but the program calculates \( t_s \).) This is because \( t_s \) depends on \( c \) and \( a_0 \) and these two
variables are frequently changing during the course of an optimization iteration. In a
typical case, \( t_s \) will be between 50 and 350 days.

Initial conditions. - At departure the vehicle position and velocity may be assumed
fixed

\[
\begin{align*}
R_0 &= R_0 \quad \text{(Initial position)} \\
V_0 &= V_0 \quad \text{(Initial velocity)}
\end{align*}
\]

(56) (57)

where \( R_0 \) and \( V_0 \) are constants that are inputted directly or calculated by the program
using an ephemeris. In the latter option, \( R_0 \) and \( V_0 \) are identical with a specified
planet's position and velocity at time \( t_0 \).

For interplanetary low-thrust problems that begin in heliocentric space with the
closed-form launch vehicle simulation, the velocity equation is modified to read

\[
V_0 = \bar{V}_0 + v_s, d(\Lambda)_{0} \quad \text{(Optimum launch orientation)}
\]

(58)

where
\[ v_{s,d}^2 = v_l^2 - 2v_{c,l}^2 \left( 1 - \frac{r_l}{r_{s,d}} \right) \] (59)

Here \( v_{s,d} \) is the spacecraft speed relative to the departure planet as it passes through a sphere of influence of radius \( r_{s,d} \). If \( r_{s,d} = \infty \), \( v_{s,d} \) is identical to the often used hyperbolic excess speed. The launch vehicle burnout velocity \( v_l \) is assumed to occur at radius \( r_l \), where the circular orbit speed is \( v_{c,l} \). Both \( v_{c,l} \) and the ratio \( r_l/r_{s,d} \) are input constants. The launch vehicle burnout velocity \( v_l \) may be fixed or, as is often the case, optimized to yield maximum net spacecraft mass. If \( v_l \) is optimized, a transversality condition can be used for this purpose; and one is given later (eq. (72)) in the section Transversality conditions for optimum spacecraft parameters. The spacecraft velocity at the sphere of influence \( v_{s,d} \) is added to the planet's velocity \( \dot{V}_0 \) in the primer direction \( \Lambda/\lambda \) in order to minimize propellant expenditure (ref. 13).

Arrival end conditions. - There are several sets of programmed arrival end conditions. The simplest of these is requiring the vehicle position and velocity to match those of a given target:

\[ R_a = \overline{R}_a \] (60)

\[ V_a = \overline{V}_a \] (61)

Here \( \overline{R}_a \) and \( \overline{V}_a \) are the target position and velocity vectors. They are inputted directly or calculated by the program using an ephemeris for a specified target body.

For problems involving the analytic high-thrust capture maneuver, the arrival velocity equation must be modified to account for the braking maneuver,

\[ \tilde{V}_a = V_a + v_{s,a} \left( \frac{\Lambda}{\lambda} \right)_a = \overline{V}_a \] (Optimum retromaneuver orientation) (62)

where

\[ v_{s,a}^2 = v_{r}^2 - 2v_{c,r}^2 \left( 1 - \frac{r_r}{r_{s,a}} \right) \] (63)

Here \( v_{s,a} \) is the vehicle speed relative to the target planet as the vehicle passes through the sphere of influence of radius \( r_{s,a} \). Orienting the planetocentric path so that \( v_{s,a} \) is directed in the primer direction at arrival \( (\Lambda/\lambda)_a \) is a transversality result (ref. 13). The retrofire is assumed to take place at radius \( r_r \), where the circular
orbit speed is $v_c, r$. Both $v_c, r$ and the ratio $r/r_{s, a}$ are input constants. The velocity just prior to retrofire $v_r$ is either fixed or open for optimization. Optimizing $v_r$ effectively optimizes the amount of high-thrust braking into the specified capture orbit. The transversality condition for optimum $v_r$ is given in the next section.

For two-dimensional trajectories in the $x$-$y$ plane, the user may elect to specify the polar coordinates of a target point,

\[ r_a = r_a \quad \text{(Arrival radius)} \]  
\[ \tilde{v}_a = v_a \quad \text{(Arrival velocity)} \]  
\[ \tilde{\gamma}_a = \gamma_a \quad \text{(Arrival path angle)} \]  
\[ \theta_a = \theta_a \quad \text{(Central travel angle)} \]  

where $r_a, v_a, \gamma_a,$ and $\theta_a$ are input constants. The tilde on the arrival velocity $\tilde{v}_a$ and path angle $\tilde{\gamma}_a$ indicates that these variables are not necessarily the arrival values. They are the arrival values if an analytic braking maneuver is not used. But if it is used, $\tilde{v}_a$ and $\tilde{\gamma}_a$ refer to values evaluated after the $v_{s, a}$ term is added to $v_a$ in equation (62).

A frequently used variation of the polar set for launch vehicle and parametric interplanetary problems is leaving the travel angle $\theta$ open for optimization since $\theta$ is seldom constrained in these cases. In this case the $\theta$ equation (eq. (67)) is replaced with the transversality equation

\[ \left( \lambda_1 v_y - \lambda_2 v_x + \lambda_4 y - \lambda_5 x \right)_a = 0 \quad \text{(Optimum travel angle)} \]  

where $\lambda_1$ and $\lambda_2$ are the components of $\Lambda$, and $\lambda_4$ and $\lambda_5$ are the components of $\Lambda_r$. Fortunately, this expression is also a constant of the motion (for two-body problems only, strictly speaking). It therefore can be invoked at the departure point to solve for $\lambda_5$ (or any other $\lambda_i$) directly instead of solving equation (68a) at the arrival point by iteration. Also, this transversality condition may be generalized to three-dimensional problems to optimize the arrival latitude and longitude,

\[ \left( \Lambda \times V + \Lambda_r \times R \right)_a = 0 \]  

In this case it is used to determine the initial values of $\lambda_3$, $\lambda_5$, and $\lambda_6$.

In some problems, such as interplanetary flybys, the arrival velocity vector is left open for optimization, which leads to the transversality condition
\[
\frac{(\lambda m_a)}{\lambda_m a} = 0 \quad \text{(Optimum arrival velocity)} \quad (69)
\]

This equation replaces the \( V_a \) equation if the arrival end conditions are specified in rectangular coordinates (eqs. (60) to (62)) or the \( \tilde{V}_a \) and \( \tilde{y}_a \) equations if polar coordinates are specified (eqs. (64) to (67)).

**Transversality conditions for optimum spacecraft parameters.** - There are four variables associated with low-thrust spacecraft that may be open for optimization: The initial mass flow rate \( \dot{m}_0 \); the exhaust speed \( c \); the launch vehicle burnout speed \( v_l \); and for capture missions, the spacecraft speed just prior to the braking retrofire \( v_r \) (equivalent to the amount of retrofire propellant). The transversality conditions for these variables are

\[
T_{\dot{m}_0} = \frac{(\lambda m_a)}{(\lambda m)_0} \left( \frac{A_2 m_{ps} - m_a}{A_1 m_0 - m_0} \right) + 1 = 0 \quad \text{(Optimum \( \dot{m}_0 \))} \quad (70a)
\]

\[
T_c = \frac{(\lambda m_a)}{(\lambda c)_a} \left( \frac{m_{ps} A_2 (2 - \frac{1}{\eta} \frac{d\eta}{dc})}{A_1 c_0} \right) + 1 = 0 \quad \text{(Optimum \( c \))} \quad (70b)
\]

\[
T_{v_l} = \frac{m_n (v_{s, d})}{A_1 (v_l)} \left( \frac{(\lambda m_a)}{(\lambda m)_0} \left( \frac{1 + k_l \frac{m_{ref}}{m_0}}{c_l} \right) - 1 \right) = 0 \quad \text{(Optimum \( v_l \))} \quad (70c)
\]

\[
T_{v_r} = \left( \frac{(\lambda m_a)}{(\lambda v_r)} \right) \left( \frac{v_{s, a}}{v_r} \right) \left( \frac{1}{A_1 c_r} \right) \left[ (1 + k_r) \left( \frac{m_a - j(m_{ps} + m_t)}{m_a - j(m_{ps} + m_t)} - m_r \right) - 1 \right] = 0 \quad \text{(Optimum \( v_r \))} \quad (70d)
\]

where

\[
A_1 = 1 + k_t - \frac{m_r (1 + j k_t)}{m_a - j(m_{ps} + m_t)} \quad (71)
\]

\[
A_2 = 1 - \frac{j m_r}{m_a - j(m_{ps} + m_t)} \quad (72)
\]
and \( m_a \) is the arrival mass at the target before any retrofire (i.e., \( m_a = m_0 - m_p \)). It has been assumed in these equations that

\[
(\lambda_{v_c})_0 = (\lambda_{m_0})_a = \lambda_{v_L} = \lambda_{v_R} = 0
\]  \( (73) \)

since they are arbitrary and choosing the value zero yields the simplest expressions. Unlike the trajectory transversality conditions, these four spacecraft transversality conditions are dependent on the particular definition of net spacecraft mass. Hence, the user is cautioned that any modification he makes to the net mass equations (eqs. (2), (3), and (9) to (16)) invalidates these four transversality equations.

The Two-Point Boundary-Value Problem

In some nonvariational and in nearly all variational trajectory problems, a two-point boundary-value problem arises that must be solved with iterative methods. In variational problems, for example, it is necessary to guess values for the initial adjoint variables \( \lambda_i \) and then use an iterative scheme that adjusts the \( \lambda_i \) until a given set of end conditions are satisfied. The NBODY user has the power to create any boundary-value problem he chooses. He does this at input time by specifying a particular set of independent variables \( x_i \) and a particular set of dependent variables \( y_i \), where \( i = 1, 2, \ldots, n \) \((n \leq 10)\). He must be careful, of course, to create a well-defined boundary-value problem by selecting \( y_i \) that really do depend on \( x_i \). He must also input guesses for the \( x_i \) and specify his desired values \( \bar{y}_i \) for \( y_i \). The program then adjusts the \( x_i \) until \( y_i = \bar{y}_i \) by using techniques discussed in the section Iterator for Boundary-Value Problems.

Users of NBODY specify \( x_i \), \( y_i \), and \( \bar{y}_i \) by loading values into the program arrays IA, IB, and DESIRE. The array DESIRE is simply \( \bar{y}_i \). The array IA is a list of program locations relative to the beginning of COMMON where the \( x_i \) are stored. The array IB is a similar list for the dependent variables \( y_i \). Table II is intended to assist users in this task by giving the COMMON locations of the anticipated candidates for \( x_i \) and \( y_i \). If the user does not find his selections for \( x_i \) or \( y_i \) in table II, he must consult the complete COMMON map given in table III. To save the user the trouble of looking up the IA and IB indexes for frequently encountered problems, the program will fill these arrays automatically if the user selects special values for the program control variable NOPT. Since the description of how to use NOPT is too lengthy to include as a part of the input instructions given later, its use is detailed here and summarized in table IV.
**NOPT=0.** - NOPT=0 is the option for nonoptimal control - only the equations of motion are integrated, not the adjoint equations. The user must fill the IA, IB, and DESIRE arrays himself through input. If the program finds IA empty, it assumes that a boundary-value problem does not exist and calculates only a single trajectory.

**NOPT=1.** - All nonzero values for NOPT specify a variational problem involving the adjoint equations. The NOPT=1 option is useful for rendezvous problems where the vehicle is required to match a target's velocity $\vec{V}_a$ and position $\vec{R}_a$. The IA array is automatically filled by the program with the COMMON locations of the initial values for the adjoint variables $\lambda_i$ ($i = 1, 2, \ldots, 6$), where $\lambda_1$, $\lambda_2$, and $\lambda_3$ are the components of the primer vector $\Lambda$ and $\lambda_4$, $\lambda_5$, and $\lambda_6$ are the components of the position adjoint vector $\Lambda_r$. The IB array is automatically filled with the locations of the components of the modified arrival velocity $\vec{V}_a$ and position $\vec{R}_a$. The modified arrival velocity (eq. (62)) is used here so that cases involving the analytic high-thrust braking maneuver may be included as a generalization. If this braking maneuver is not used, $\vec{V}_a = V_a$. The user must fill the DESIRE list with the components of the target velocity and position ($\vec{V}_a$ and $\vec{R}_a$) unless he selects the ephemeris option. In this case, DESIRE is filled by the program using the selected target body's ephemeris. For two-dimensional problems in the x-y plane, only the x and y components of $\Lambda$, $\Lambda_r$, $\vec{V}_a$, $\vec{R}_a$, $\vec{V}_a$, and $\vec{R}_a$ are used.

The preceding discussion assumes that the engines may be shut down and restarted whenever a zero is attained by the on-off switching function $\kappa$. Thus, coast arcs will occur in the optimum trajectory. The user is cautioned here that he must select a value of initial mass flow rate $\dot{m}_0$ (or initial thrust-weight ratio $a_0/g$) large enough to permit a solution. If he selects too small a value of $\dot{m}_0$, not enough propulsive effort can be expended within the allotted mission time to accomplish the mission. In this case, the program will eliminate all coast arcs and fail to converge to a solution. Simultaneously, the adjoint variables will tend toward infinity - a sure signal to the user that $\dot{m}_0$ is too small.

If the user selects the all-propulsion constraint (COAST=F), the initial mass flow rate $\dot{m}_0$ (or $a_0/g$ if inputted) is treated as an independent variable and replaces $\lambda_1$ in the IA list. Thus, in this case, the input value of $\dot{m}_0$ is merely a first guess and will be changed by the program in the process of iterating to a solution trajectory. The converged value of $\dot{m}_0$ will be the smallest possible value that may be used to reach the target at the specified mission time. The use of a larger value of $\dot{m}_0$ (or $a_0/g$) would result in a lower payload all-propulsion trajectory.

**NOPT=2.** - Valid only for two-dimensional trajectories in the x-y plane, NOPT=2 is a polar coordinate option that utilizes the end conditions defined by equations (64) to (67). The user must fill the DESIRE list with values for the target's radius $\bar{r}_a$, speed $\bar{v}_a$, path angle $\gamma_a$, and the central travel angle $\varnothing_a$. The IA list is automatically filled
with the locations of $\lambda_1$, $\lambda_2$, $\lambda_4$, and $\lambda_5$ (the initial values of the adjoint variables).

The IB list is automatically filled with the locations of the modified arrival conditions $r_{a'}$, $v_{a'}$, $\gamma_{a'}$, and $\theta_{a'}$. Again, $\gamma_{a'}$ and $\gamma_{a}$ differ from $v_{a}$ and $\gamma_{a}$ only in cases involving the analytic high-thrust braking maneuver. And as explained in the NOPT=1 option, the initial mass flow rate or initial thrust-weight ratio is substituted for $\lambda_1$ in the IA list for all-propulsion missions.

**NOPT=3.** - The NOPT=3 option is identical to the NOPT=2 option with the exception that the optimum-travel-angle transversality condition (eq. (68a)) replaces the fixed-travel-angle condition (eq. (67)). Actually, as explained earlier, equation (68a) also applies at the departure point and may be used to solve for $\lambda_5$. Hence, only three end conditions require iteration (eqs. (64) to (66)). The user must load only the target values $r_{a'}$, $v_{a'}$, and $\gamma_{a}$ into the DESIRE list and also guess values for $\lambda_1$, $\lambda_2$, and $\lambda_4$. For all-propulsion missions, the initial propellant flow rate (or the initial thrust-weight ratio) replaces $\lambda_1$ in the IA list as previously explained.

**NOPT=4.** - The NOPT=4 option is useful for two-dimensional flybys and involves end conditions, equations (64), (67), and (69). The user must fill the DESIRE list with a target radius $r_{a}$, two consecutive zeros for the two components of $(\lambda_{1}/\lambda_{m})_{a}$, and the central travel angle $\bar{\theta}_{a}$. The two zeros do not have to be loaded since the program default values are set to zero, but the loading order must be maintained (i.e., $\bar{\theta}_{a}$ must be loaded as the fourth element of DESIRE). The IA list is automatically filled with the $\lambda_1$, $\lambda_2$, $\lambda_4$, and $\lambda_5$ locations, while the IB list is filled with the location of $r_{a'}$, $(\lambda_{1}/\lambda_{m})_{a}$, $(\lambda_{2}/\lambda_{m})_{a}$, and $\theta_{a}$. The optimum arrival velocity end condition $\Lambda_{a} = 0$ is modified here by scaling with the arrival value of the mass adjoint variable $\lambda_{m}$. One of the transversality conditions requires $(\lambda_{m})_{a} = 1$ and, because of the homogeneity of the adjoint equations, we are free to scale all the $\lambda_i$ without changing the trajectory. In effect then, the user need only supply a target radius $r_{a}$; the desired travel angle $\bar{\theta}_{a}$; and guesses for $\lambda_1$, $\lambda_2$, $\lambda_4$, and $\lambda_5$. For all-propulsion trajectories, the comments under the NOPT=1 option apply here also.

**NOPT=5.** - NOPT=5 is the optimum-travel-angle flyby option and is similar to the NOPT=4 option except that the optimum-travel-angle end condition is invoked to calculate $\lambda_5$ in the manner described for the NOPT=3 option. The user is required only to load a target radius $r_{a}$ and guess the initial values of $\lambda_1$, $\lambda_2$, and $\lambda_4$. The all-propulsion mission comments of the NOPT=1 option also apply here.

**NOPT=6.** - In the NOPT=6 option the user must fill the IA and IB lists himself. The only difference between this option and the NOPT=0 option is that with this option the thrust control is optimal. Thus, the adjoint equations are integrated, and the user must load initial values for the adjoint variables $\lambda_i$.

**NOPT=7.** - The NOPT=7 option is identical to the NOPT=6 option with the addition
of the appropriate optimum angle condition (eq. (68a) or (68b)). Thus, the user must fill
the IA, IB, and initial adjoint variable lists. However, he need not furnish a value for
\( \lambda_5 \) for two-dimensional problems (or \( \lambda_3, \lambda_5, \) and \( \lambda_6 \) for three-dimensional problems)
since equation (68) is used at the departure point to determine them.

Additional features of the NOPT options. - Since choosing reasonable initial values
for the adjoint variables \( \lambda_i \) is often a difficult task, a somewhat simpler scheme is
provided for the common two-dimensional case. The user may elect instead to input
variables that have more physical significance; namely, the departure thrust angle \( \psi_0 \),
its derivative \( \dot{\psi}_0 \), the engine on-off switch function at departure \( \kappa_0 \), its derivative \( \dot{\kappa}_0 \),
and the magnitude of the primer vector at departure \( \lambda_0 \). Sketch (e) defines \( \psi \), and
equation (47) defines \( \kappa \). Under this option, the initial values of the adjoint variables are
calculated with the following equations:

\[
\begin{align*}
\left( \lambda_1 \right)_0 &= (\lambda \cos \psi)_0 \\
\left( \lambda_2 \right)_0 &= (\lambda \sin \psi)_0 \\
\left( \lambda_4 \right)_0 &= \left( \lambda_2 \dot{\psi} - \frac{\lambda_1 \dot{\kappa}}{c \lambda} \right)_0 \\
\left( \lambda_5 \right)_0 &= \left( -\lambda_1 \dot{\psi} - \frac{\lambda_2 m \dot{\kappa}}{c \lambda} \right)_0 \\
\left( \lambda_7 \right)_0 &= \left( \frac{c \lambda}{m} - \kappa \right)_0
\end{align*}
\]

Here \( \left( \lambda_7 \right)_0 \) is the initial value of the mass adjoint variable \( \lambda_m \). It is usually convenient
to let the program set \( \lambda_0 = 1 \) by default since it represents a scale factor here and
since all values of \( \lambda_0 \) result in identical trajectories. It is useful to set \( \lambda_0 \neq 1 \) when
attempting to reproduce a previously computed trajectory for which the initial value of
\( \lambda \) is not unity. (It avoids scaling by the user in such a situation.) If one of the optimum-
travel-angle options is selected (NOPT=3, 5, or 7), it is unnecessary for the user to
load \( \kappa_0 \) since the program will compute its value using a variation of the transversality
condition,
\[ \dot{\kappa}_0 = \left[ \frac{\lambda_2(v_x - y\dot{\psi}) - \lambda_1(v_y + x\dot{\psi})}{\frac{m}{c\lambda} (\lambda_2 x - \lambda_1 y)} \right]_0 \] (79)

In most NOPT options, inputting the alternative set \( \psi_0, \dot{\psi}_0, \kappa_0, \dot{\kappa}_0, \) and \( \lambda_0 \) instead of \( \lambda_1, \lambda_2, \lambda_4, \lambda_5, \) and \( \lambda_7 \) will result in its use for only the first trajectory of the boundary-value-problem iteration sequence. The remaining trajectories are begun by the program using the adjoint variables directly. However, the NOPT=6 and 7 options permit using the alternative set through the iteration sequence, if preferred, simply by filling the IA list with the locations of whichever members of this set are chosen.

Partial Derivatives

The boundary-value problem consists of driving a set of \( n \) dependent variables \( y_i \) to desired values \( \bar{y}_i \) by adjusting a set of \( n \) independent variables \( x_i \). The iterator that does this needs a partial derivative matrix \( G \) whose elements are \( \frac{\partial y_i}{\partial x_j} \) evaluated for the current approximate solution set of \( x_i \). There are two methods used in NBODY to generate \( G \):

1. A finite difference method
2. An analytical method

**Finite difference method.** - This method consists of computing \( n \) perturbation trajectories about a reference trajectory - one for each \( x_i \). Then the elements of \( G \) are formed in an approximate way by differencing the results of the perturbation trajectories with the reference trajectory.

\[
\frac{\partial y_i}{\partial x_j} \approx \frac{\Delta y_i}{\Delta x_j} \equiv \frac{y_i - y_i^0}{x_j - x_j^0} \] (80)

The superscript zero denotes the reference trajectory values. This method has the advantage of being quite general and straightforward. It does suffer, however, from two standpoints: (1) it is relatively slow in comparison with the analytical method, and (2) it is often not easy to select appropriate perturbation sizes \( \Delta x_j \). The latter difficulty manifests itself in highly nonlinear, sensitive problems, where a large \( \Delta x_j \) results in an excessively large error in \( \Delta y_i \) and where a small \( \Delta x_j \) results in too much numerical noise in \( \Delta y_i \). To help alleviate this difficulty, NBODY is programmed to monitor \( \Delta y_i \) and to adjust \( \Delta x_j \) accordingly. If \( \Delta y_i \) is judged to be too small or too large, the
perturbation trajectory is repeated; therefore, more than \( n \) perturbation trajectories are frequently necessary.

**Analytical method.** The analytical method of generating the partial derivatives is faster and more accurate than the finite difference method. The partial derivatives are generated by integrating an additional set of differential equations along with the state and adjoint equations. Thus, the problem of choosing perturbation sizes is avoided. However, the method has the serious disadvantage of not being general—that is, a change in the definition of the end conditions or payoff criterion generally requires deriving and programming new partial derivative equations. In the NBODY program these equations are currently programmed for a limited set of options; namely, variational problems not involving any of the transversality equations for optimum \( \dot{m}_0, c, v_L, \) or \( v_r \). Thus, the user must resort to the finite difference method if his problem is nonvariational or if any of these four variables are to be optimized by using transversality conditions. (They may also be optimized by using an ordinary search scheme, as discussed later.) In particular, the program will use the analytical scheme only if \( 1 \leq \text{NOPT} \leq 5 \). (The NOPT operations are discussed in detail in the preceding section and are summarized in Table IV.) If the user wishes to include analytical partial derivatives for \( \dot{m}_0, c, v_L, \) or \( v_r \) (or any others), he may do so by amending subroutines WDERIV, WALTER, WLOOK, WBEGIN, and WOUT.

The differential equations used to generate the partials are derived by differentiating the state and adjoint equations (eqs. (45) and (46)) with respect to an arbitrary variable. When \( \delta \) is used to denote a partial with respect to the arbitrary variable, the resulting equations are

\[
\frac{d}{dt} (\delta V) = - \sum_{j=1}^{n-1} \frac{\mu_j}{r_j^3} \left[ \delta R - \frac{3}{r_j^2} (R_j \cdot \delta R_j) R_j \right]
\]

\[
- \frac{c \dot{m}_0}{m \lambda} \xi \epsilon \left\{ \delta \Lambda + \left[ \frac{\delta c}{c} \frac{\dot{m}_0}{m_0} - \frac{\delta m}{m} + \frac{\zeta'(R \cdot \delta R)}{\zeta r} - \frac{\Lambda \cdot \delta \Lambda}{\lambda^2} \right] \right\}
\]

(81a)

\[
\frac{d}{dt} (\delta R) = \delta V
\]

(81b)

\[
\frac{d}{dt} (\delta m) = \epsilon \xi \dot{m}_0 + \epsilon \dot{m}_0 \xi^e \frac{R \cdot \delta R}{r}
\]

(81c)
\[
\frac{d}{dt}(\delta \Lambda) = -\delta \Lambda_r \tag{81d}
\]

\[
\frac{d}{dt}(\delta \lambda_m) = \sum_{j=1}^{n-1} \frac{\mu_j}{r_j^3} \left( \delta \lambda - \frac{3}{r_j^2} \left( R_j \cdot \delta R \Lambda + (\Lambda \cdot R_j) \delta R + \left[ \left( R_j \cdot \delta \Lambda \right) + (\Lambda \cdot \delta R) - \frac{5}{r_j^2} (\Lambda \cdot R_j)(R_j \cdot \delta R) \right] R_j \right) \right) \tag{81e}
\]

\[
+ \frac{\dot{m}_0}{r} R \left[ \frac{\delta \dot{m}_0}{m_0} + \frac{R \cdot \delta R}{R} \left( \frac{c}{r} \right) \right] \delta R + \left[ \frac{\Lambda}{m} \frac{\delta c}{c} + \frac{\delta (\Lambda \cdot \delta \Lambda)}{m \lambda} - \frac{\delta c}{m^2} \frac{\delta m - \delta \lambda m}{m} \right] R + \delta R \tag{81f}
\]

\[
\frac{d}{dt}(\delta \dot{m}_0) = \frac{d}{dt}(\delta \dot{a}_0) = 0 \tag{81g}
\]

\[
\frac{d}{dt}(\delta c) = 0 \tag{81h}
\]

\[
\frac{d}{dt}(\delta \lambda_c) = \frac{\epsilon \dot{m}_0 \xi \lambda}{m} \left( \frac{\delta \dot{m}_0}{\dot{m}_0} - \frac{\delta m}{m} + \frac{\Lambda \cdot \delta \lambda}{\lambda} + \frac{R \cdot \delta R}{r} \xi' \right) \tag{81i}
\]

\[
\left[ \frac{\delta \lambda}{\dot{m}_0} \right]_0^a = - \frac{1}{\dot{m}_0} \left[ \lambda_m \delta m + m \delta \lambda_m + \lambda \dot{m}_0 \delta \dot{m}_0 \right]_0 \tag{81j}
\]

Each time the engine is switched on or off \((\kappa = 0)\), discontinuities appear in several of these partials due to the presence of the factor \(\epsilon\) in the state and adjoint equations. In general, the jumps are given by

\[
\Delta \delta y = (\delta y)^+ - (\delta y)^- = \frac{(\dot{y})_{\text{engines off}} - (\dot{y})_{\text{engines on}}}{\dot{\kappa}} \delta \kappa \tag{82}
\]

where
\[
\delta \kappa = \frac{c}{m \lambda} (\Lambda \cdot \delta \Lambda) - \frac{\delta \lambda}{m^2} \frac{m \lambda}{m} \delta m + \frac{\lambda}{m} \delta c 
\]  
(83)

\[
\dot{k} = - \frac{c(\Lambda \cdot \Lambda_r)}{m \lambda} 
\]  
(84)

Specifically, the nonzero jumps are

\[
\Delta \delta V = - \left( \frac{\dot{m}_0 \xi \delta \kappa}{\Lambda \cdot \Lambda_r} \right) \Lambda 
\]  
(85a)

\[
\Delta \delta m = \frac{\dot{m}_0 \xi m \lambda \delta \kappa}{c(\Lambda \cdot \Lambda_r)} 
\]  
(85b)

\[
\Delta \delta \lambda_m = - \frac{\dot{m}_0 \xi \lambda^2 \delta \kappa}{m(\Lambda \cdot \Lambda_r)} 
\]  
(85c)

\[
\Delta \delta \lambda_c = \frac{\dot{m}_0 \xi \lambda^2 \delta \kappa}{c(\Lambda \cdot \Lambda_r)} 
\]  
(85d)

A jump discontinuity also occurs in \( \delta V \) at the arrival point if the analytic high-thrust braking maneuver option is selected,

\[
\Delta \delta V = \frac{v_s}{\lambda a} \left( \delta \lambda - \frac{\Lambda \cdot \delta \Lambda}{\lambda^2} \right) \a 
\]  
(86)

Recall that \( \delta \) denotes a partial derivative with respect to any variable \( x \). The \( x_1 \) of main interest, of course, are those variables that are defined as independent variables in the two-point boundary-value problem - namely, the initial values of the adjoint variables (plus \( \dot{m}_0 \) for all-propulsion cases). Actually, there are nine \( x_1 \) programmed in NBODY:

\[
x_1 = \left( \lambda_1 \right)_0 \quad \text{(Initial value of } x\text{-component of } \Lambda) 
\]  
(87a)
\[ x_2 = (\lambda_2)_0 \] (Initial value of y-component of \( \Lambda \)) \hspace{1cm} (87b)

\[ x_3 = (\lambda_3)_0 \] (Initial value of x-component of \( \Lambda \)) \hspace{1cm} (87c)

\[ x_4 = (\lambda_4)_0 \] (Initial value of x-component of \( \Lambda_r \)) \hspace{1cm} (87d)

\[ x_5 = (\lambda_5)_0 \] (Initial value of y-component of \( \Lambda_r \)) \hspace{1cm} (87e)

\[ x_6 = (\lambda_6)_0 \] (Initial value of x-component of \( \Lambda_r \)) \hspace{1cm} (87f)

\[ x_7 = (\lambda_m)_0 \] (Initial value of \( \lambda_m \)) \hspace{1cm} (87g)

\[ x_8 = c \] (Specific impulse) \hspace{1cm} (87h)

\[ x_9 = \dot{m}_0 \] (Initial mass flow rate) \hspace{1cm} (87i)

The initial value of \( \lambda_m \) is included in this list since it is needed for the evaluation of \( \delta (\Lambda/\lambda_m) \) used in the flyby end condition \( (\Lambda/\lambda_m)_a = 0 \). The specific impulse \( c \) is also included but is not required in the present version of NBODY. If the user prefers other \( x_i \) (such as \( \psi_0, \dot{\psi}_0, \kappa_0, \dot{\kappa}_0 \)), he must alter subroutines WDERIV, WBEGIN, WOUT, WLOOK, and WINTEG.

Having defined the list of \( x_i \), it is now possible to calculate the initial values of the partials - that is, the partial derivatives have values at the departure point according to the conditions imposed at departure. For example, the value of \( \delta V \) at departure depends on the magnitude of the boost velocity supplied by the launch vehicle (if any) and the boost velocity orientation. Thus, by differentiating equation (58), the first four of the following set of initial-value equations may be derived. The others are derived similarly. The subscripts on the partials denote which \( x_i \) in the preceding list is the independent variable \( (e.g., \delta V_1 = \text{partial derivative of } V \text{ with respect to } (\lambda_1)_0) \). Also, \( \hat{i}, \hat{j}, \hat{k} \) are unit vectors along the \( x, y, z \) axes, respectively. The following, then, are the initial values of the partial derivatives:
\[
\delta V_1 = \left[ \frac{v_s d}{\Lambda} \left( i - \frac{\Lambda}{\Lambda^2} x \right) \right]_0 \\
\delta V_2 = \left[ \frac{v_s d}{\Lambda} \left( j - \frac{\Lambda}{\Lambda^2} y \right) \right]_0 \\
\delta V_3 = \left[ \frac{v_s d}{\Lambda} \left( k - \frac{\Lambda}{\Lambda^2} z \right) \right]_0 \\
\delta V_i = 0 \quad i = 4, \ldots, 9 \\
\delta R_i = 0 \quad i = 1, \ldots, 9 \\
\delta m_i = 0 \quad i = 1, \ldots, 9 \\
\delta \Lambda_1 = \hat{i} \\
\delta \Lambda_2 = \hat{j} \\
\delta \Lambda_3 = \hat{k} \\
\delta \Lambda_i = 0 \quad i = 4, \ldots, 9 \\
\delta (\Lambda_r)_1 = \begin{cases} 
0 & \text{for fixed travel angle} \\
\left( \frac{v_y + \lambda_1 \delta v_y - \lambda_2 \delta v_x}{x} \right) \hat{j} & \text{for optimum travel angle (two dimensions only)} 
\end{cases}
\]
\[
\delta (\Lambda_r)_2 = \begin{cases} 
0 & \text{for fixed travel angle} \\
\left(-v_x + \lambda_1 \delta v_2 - \lambda_2 \delta v_2 \right)_{0} & \text{for optimum travel angle (two dimensions only)}
\end{cases}
\]

\[
\delta (\Lambda_r)_3 = 0 
\]

\[
\delta (\Lambda_r)_4 = \begin{cases} 
\hat{i} & \text{for fixed travel angle} \\
\hat{i} + \left(\frac{\text{Y}}{x} \right)_{0} & \text{for optimum travel angle (two dimensions only)}
\end{cases}
\]

\[
\delta (\Lambda_r)_5 = \hat{j} 
\]

\[
\delta (\Lambda_r)_6 = \hat{k} 
\]

\[
\delta (\Lambda_r)_i = 0 \quad i = 7, 8, 9 
\]

\[
\delta (\lambda_m)_i = 0 \quad i = 1, \ldots, 9 
\]

\[
\delta (c)_8 = 1 
\]

\[
\delta (c)_i = 0 \quad i = 1, \ldots, 7, 9 
\]

\[
\delta (\hat{\mathbf{m}}_0)_9 = 1 
\]

\[
\delta (\hat{\mathbf{m}}_0)_i = 0 \quad i = 1, \ldots, 8 
\]
\[ \delta (\lambda_{c})_i = 0 \quad i = 1, \ldots, 9 \]  
\[ \delta (\lambda_{m0})_i = 0 \quad i = 1, \ldots, 9 \]

After the partial derivative equations are integrated, it is necessary to transform them into another set if the end conditions are in terms of polar coordinates (two dimensions only):

\[ \delta r = \frac{R \cdot \delta R}{r} \quad \text{(Radius magnitude)} \]  
\[ \delta v = \frac{V \cdot \delta V}{v} \quad \text{(Velocity magnitude)} \]  
\[ \delta \theta = \frac{x \delta v - y \delta x}{r^2} \quad \text{(Polar travel angle)} \]  
\[ \delta \gamma = \delta \theta - \frac{v_x \delta v_y - v_y \delta v_x}{v^2} \quad \text{(Path angle)} \]

Also, for the flyby end condition \( \left(\frac{\Lambda}{\lambda_m}\right)_a = 0 \) we need the following transformation equation:

\[ \delta \left(\frac{\Lambda}{\lambda_m}\right) = \left(\frac{1}{\lambda_m}\right) \delta \Lambda - \left(\frac{\delta \lambda_m}{\lambda_m^2}\right) \Lambda \]

**Iterator for Boundary-Value Problems**

**Convergence criterion.** - Before each trajectory integration, an N-vector of independent variables \( X \) is selected that yields, after the integration, a particular N-vector of dependent variables \( Y \). That is, the end condition vector \( Y \) is a function of \( X \),

\[ Y = f(X) \]
The boundary-value problem is to determine the solution to this equation if the \( Y \) vector is known—that is, given \( Y \) find the \( X \) that satisfies

\[
\overline{Y} = f(\overline{X})
\]  
(92)

A solution is judged to be found if the square root of the sum of the squares of the weighted residuals \( \Delta Y = \overline{Y} - Y \) in less than a tolerance criterion \( \tau \):

\[
\tau = \sqrt{\Delta Y^T W^2 \Delta Y} < \tau
\]  
(93)

The weighting matrix \( W \) is diagonal and positive definite. The diagonal elements of \( W \) consist of the weighting factors \( 1/w_i \) either selected by the user, or by default, calculated by the program as follows:

\[
w_i = \begin{cases} 
\overline{y}_i & \text{if } \overline{y}_i \neq 0 \\
1 & \text{if } \overline{y}_i = 0 \\
360 & \text{if } \overline{y}_i = 0 \text{ and } y_i \text{ is path angle}
\end{cases}
\]  
(94)

In the majority of cases the default weighting factors will result in approximately equal emphasis on all residuals, and the error \( \tau \) will be of the order

\[
\tau \approx \max_i \left| 1 - \frac{y_i}{\overline{y}_i} \right|
\]  
(95)

The convergence criterion \( \overline{\tau} \) may be selected by the user or defaulted to \( 10^{-4} \).

**Linear correction scheme (Newton-Raphson).** - Starting with a guess \( X_i \) that yields an error \( \tau_i \), the problem is to choose a new value \( X_{i+1} \) such that \( \tau_{i+1} < \tau_i \). In general, the end condition vectors are related by

\[
Y_{i+1} = Y_i + G \Delta X + \text{(Higher order terms)}
\]  
(96)

where \( \Delta X = X_{i+1} - X_i \) and \( G \) is the partial derivative matrix \( \partial Y/\partial X \). By ignoring the higher order nonlinear terms, we may estimate \( \Delta X \) by setting \( Y_{i+1} = \overline{Y} \) as follows:

\[
\Delta X = G^{-1}(\overline{Y} - Y_i)
\]  
(97)
If \( X_i \) is close to the solution value \( \bar{X} \), this estimate will usually result in \( \tau_{i+1} < \tau_i \) and the process is repeated until convergence is obtained \((\tau < \tau_\ast)\). However, \( \Delta X \) may be too large if \( X_i \) is not close to \( \bar{X} \), and the new error value may exceed the old value. When this occurs, the trajectory is repeated using a smaller value of \( \Delta X \). In particular,

\[
\Delta X = \chi G^{-1}(\bar{Y} - Y_i)
\]

where \( \chi \) is an inhibitor whose value lies between zero and unity \((0 < \chi \leq 1)\). Several cutbacks in the size of \( \chi \) may be necessary before \( \tau_{i+1} < \tau_i \). The program reduces \( \chi \) by a factor of 2 for each cutback and restores its value to unity upon satisfying \( \tau_{i+1} < \tau_i \). Thus, each iteration cycle is initially attempted with \( \chi = 1 \). This method of controlling \( \chi \) has proven to be just as effective as more elaborate inhibitor controllers in terms of reducing overall computation time.

The partial derivative matrix \( G \) is generated either by finite differencing or by numerical integration, as explained in the section Partial Derivatives, and is updated each time \( X_i \) is improved.

This convergence scheme is generally quite satisfactory providing the initial estimate of \( X \) is reasonably close to \( \bar{X} \). If \( X \) is not close to \( \bar{X} \), however, the inhibitor \( \chi \) may be forced to very small values in order to improve \( X \). This situation is undesirable since \( X \) is improved very slowly. To alleviate this difficulty, another error-reducing scheme is programmed to handle cases of large errors.

**Univariate search scheme.** - This scheme is called upon when the initial guess \( X \) is poor. In particular, it is used if \( \tau > \tau_\ast \), where the value of \( \tau_\ast \) is selected by the user or defaulted to unity (experience is the best guide for selecting \( \tau_\ast \)). It is also called upon if the linear correction scheme bogs down because of inaccurate partial derivatives or highly nonlinear behavior. Each member of \( X \) is varied - one at a time - to reduce \( \tau \). The individual searches are conducted by increasing the step increments until a minimum in \( \tau \) is detected. Rather than attempting to pinpoint the minimum, the search proceeds to vary the next variable as soon as \( \tau \) begins to increase after having been in a downward trend. After the search cycles through all the variables, it begins over again with reduced initial step increments.

Although this technique has the capability to reduce large errors quickly, it is unacceptably slow in the neighborhood of the solution. Thus, whenever \( \tau < \tau_\ast \), the univariate scheme is abandoned in favor of the linear correction scheme. This switch also occurs if the univariate scheme fails to halve the error \( \tau \) within 15N trajectory simulations. Actually, control may be passed between these two schemes several times in difficult problems. If it is determined that neither scheme is working well, the linear correction scheme is activated without inhibitor control \((\chi = 1)\) in a final effort to sal-
vage the iteration. With reasonable first guesses, however, this hybrid technique is a powerful iterator that combines the advantages of both schemes.

Integration Method

Runge-Kutta scheme. - All the state equations, adjoint equations, and partial derivative equations (if used) are numerically integrated simultaneously by using a fourth-order Runge-Kutta method. This method for a single equation of the form \( \dot{y} = f(t, y) \) may be described as follows:

\[
y_{n+1} = y_n + \frac{1}{6} (k_1 + 2k_2 + 2k_3 + k_4)
\]

where

\[
k_1 = hf(t_n, y_n)
\]

\[
k_2 = hf\left(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right)
\]

\[
k_3 = hf\left(t_n + \frac{h}{2}, y_n + \frac{k_2}{2}\right)
\]

\[
k_4 = hf(t_n + h, y_n + k_3)
\]

Four function evaluations are required for each integration step of size \( h \). One of the disadvantages of this method is the absence of a simple yet accurate method to estimate the truncation error propagated along the trajectory. The truncation error is the difference between the true value of \( y \) and the value obtained with the integration formulas. If an accurate estimate of the truncation error were available, one could use it to control the step size in a manner that would maintain a specified accuracy level. In the absence of a rigorous and efficient step-size controller, an approximate but very efficient step-size controller is programmed that experience has shown to be stable and well-behaved in difficult situations. In particular, it reduces the step size in regions where the time derivative of the \( f \) function is changing rapidly.

The basis of the technique is the assumption that the truncation error \( \delta \) is proportional to the fifth power of \( h \),

41
\[ \delta(t, h) = \Psi(t) h^5 \]  

In general \( \Psi \) varies with time \( t \) in some unknown fashion. We further assume that, over the time span of two steps, the logarithm of \( \Psi \) varies linearly with \( t \). This assumption allows us to predict a value of \( h \) that will result in a desired error \( \bar{\delta} \) if we know the values of \( \ln \Psi \) at the two previous time points (sketch f),

\[ \ln h_n = \frac{1}{5} (\ln \bar{\delta} - \ln \Psi_n) \]  

(102)

where

\[ \ln \Psi_n = \ln \Psi_{n-1} + (\ln \Psi_{n-1} - \ln \Psi_{n-2}) \frac{h_{n-1}}{h_{n-2}} \]  

(103)

The values of \( \ln \Psi_{n-1} \) and \( \ln \Psi_{n-2} \) are computed with the same basic formula except that the computed error \( \bar{\delta} \) is used in place of the desired error \( \bar{\delta} \),

\[ \ln \Psi = \ln \bar{\delta} - 5 \ln h \]  

(104)
Evaluating this equation at times $t_{n-1}$ and $t_{n-2}$ requires determining the error $\delta$. Instead of attempting to calculate $\delta$ precisely, a very simple estimate is generated with the following Lagrangian interpolation formula:

$$
\hat{y}(t) = \hat{y}_{n-2} \frac{(t - t_{n-1})(t - t_n)}{h_{n-2}(h_{n-2} + h_{n-1})} - \hat{y}_{n-1} \frac{(t - t_{n-2})(t - t_n)}{h_{n-2}h_{n-1}} + \hat{y}_n \frac{(t - t_{n-2})(t - t_{n-1})}{h_{n-1}(h_{n-2} + h_{n-1})}
$$

Integrating this equation between $t_{n-1}$ and $t_n$ yields

$$
\Delta y = y_n - y_{n-1} = \frac{1}{6} \left[ \frac{h_{n-1}^3 \hat{y}_{n-2}}{h_{n-2}(h_{n-2} + h_{n-1})} + \frac{h_{n-1}(h_{n-1} + 3h_{n-2})\hat{y}_{n-1}}{h_{n-2}} + \left( \frac{2h_{n-1}}{h_{n-2} + h_{n-1}} \right) \hat{y}_n \right]
$$

This low-order integration formula may be evaluated quite efficiently since all the required data (the derivatives in particular) are already available from the Runge-Kutta integration. Thus, at each integration step the error $\delta$ may be estimated by differencing the value of $\Delta y$ obtained by the Runge-Kutta formulas with the value obtained with this low-order method.

$$
\delta_r = \left| \frac{(\Delta y)_{\text{Runge-Kutta}} - (\Delta y)_{\text{Low-order scheme}}}{y_n} \right|
$$

This definition of error $\delta_r$ is not the same as the previous definition of $\delta$: (1) because $\delta_r$ represents the difference in answers between two integration schemes instead of the true error $\delta$ and (2) because $\delta_r$ is a relative error since the $\Delta y$ increments are divided by a normalization factor $y_n$.

Since many independent variables are integrated simultaneously, there are many values of $\delta_r$ calculated at each step (one for each state and adjoint variable). Only the maximum value of $\delta_r$ is used to calculate the next step size. Obviously, inaccurate predictions of step size can occur - particularly when the maximum value of $\delta_r$ shifts from one variable to another or when any sudden change occurs. Whenever the error is excessive ($\delta_r > \delta_{\text{limit}}$), the step is recomputed with a smaller value of $h$, which is calculated by updating the $\ln \Psi$ data (using the excessive error in eq. (104)). Two consecutive failures at satisfying $\delta_r > \delta_{\text{limit}}$ result in a restart of the integration procedure at the time of failure. The start (and restart) procedure is to take two identical sized
steps before checking the relative error $\delta_r$. This is necessary because no values of $\ln \psi$ are yet available. In this procedure the value of $\ln \psi_{n-2}$ is set equal to $\ln \psi_{n-1}$ and the low-order integration formula (eq. (106)) is replaced by a simplified form (Simpson's Rule) because $h_{n-2}$ equals $h_{n-1}$.

$$\Delta y = \frac{h_{n-1}}{3} (\dot{y}_{n-2} + 4\dot{y}_{n-1} + \dot{y}_n)$$

The program user selects the level of accuracy and initial step size as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FORTRAN name</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference relative error, $\delta_r$</td>
<td>EREF</td>
<td>$10^{-4}$</td>
</tr>
<tr>
<td>Limit relative error, $\delta_{\text{limit}}$</td>
<td>ERLIMIT</td>
<td>$3\times10^{-4}$</td>
</tr>
<tr>
<td>Initial step size, $h_1$</td>
<td>STEP</td>
<td>$t_f/100$</td>
</tr>
</tbody>
</table>

The default initial step size is $1/100^{th}$ of the mission time $t_f$. If the estimate of $h_1$ is too large, the program automatically reduces it until the limit error criterion is satisfied. If $h_1$ is grossly underestimated ($\delta_r << \delta_{\text{limit}}$), the step is accepted and the next step size is increased substantially, but as a precaution no step is permitted to be greater than three times the size of the previous step.

Comparisons with exact solutions have shown that $\delta_r = 10^{-4}$ is sufficiently accurate for most parametric studies requiring only three or four figures of accuracy. With a 36-bit word length computer, roundoff error will ordinarily exceed truncation error if $\delta_r < 10^{-7}$. Furthermore, very little accuracy difference exists between $\delta_r = 10^{-6}$ and $\delta_r = 10^{-7}$. Hence, as a general guideline, setting $\delta_r < 10^{-6}$ is not recommended since little improvement in accuracy can be gained at the expense of much greater computer execution time. Decreasing $\delta_r$ by an order of magnitude will result roughly in doubling the number of integration steps and execution time.

Most of the integration process is computed in single precision on 36-bit word length computers ($8 \frac{1}{3}$ significant figures), although the variables being integrated are accumulated in double precision. That is, the derivatives $\dot{y}$ are evaluated in single precision, but the integration variables $y$ are accumulated in double precision. This is nearly as fast and compact as complete single-precision integration and approaches the accuracy afforded by complete double-precision integration since usually $\Delta y << y$.

**Trajectory interrupt.** - It is often necessary to interrupt the integration process before the trajectory terminates to allow some specific action to be taken. Interrupts
for printouts at selected time or step intervals are an obvious example. Figure 1 illustrates several other interrupt situations. For problems involving more than one phase (fig. 1(a)), the phase-defining data are changed at each phasing point. When phases are identical to physical vehicle stages, this amounts to reinitializing the mass, specific impulse, propellant flow rate, and so forth. These data are read in during a single input at the beginning of a case and stored in arrays. Interrupts also occur whenever a trajectory passes through a sphere of influence in order to translate the coordinate system origin to the center of another body (fig. 1(b)). A third type of interrupt occurs under the optimal thrust option and involves switching the engines either on or off when $\kappa = 0$ (fig. 1(c)). Several of the partial derivatives (if integrated) are discontinuous whenever the engines shut down or start up. Yet a fourth type of interrupt occurs under the optimum-fixed-thrust-angle option (fig. 1(d)). In this case the thrust angle changes discontinuously whenever the difference between the two largest values of $A \cdot T_i$ vanishes. The unit thrust vector $T_i$ is dependent on the thrust angle $\alpha_i$ according to equations (50) to (52).

The last type of interrupt provides the user with a means to force phase points or printouts to occur whenever a user-selected variable attains a user-selected target value (fig. 1(e)). This flexibility circumvents many awkward situations such as trying to guess the firing time of a launch vehicle’s first stage so that the first-stage burnout occurs at some desired altitude. In this example the user can specify directly that the first stage should cease when the vehicle’s altitude reaches some input value. The user commands the program to search for an interrupt by loading the COMMON location of the specified variable into LOOKX and the target values into XLOOK. Table III is a map of COMMON which the user refers to in order to determine LOOKX. If desired, the search for an interrupt may be delayed until some side condition is met; namely, $C(\text{LOOKSW}) > \text{SWLOOK}$, where $C$ refers to COMMON, and LOOKSW and SWLOOK are both input parameters. Whenever an interrupt does occur (i.e., when $C(\text{LOOKX})=\text{XLOOK}$), a printout is issued and the program interrogates the input parameter ENDX to determine whether to continue the current phase (ENDX=0), terminate the current phase and begin the next (ENDX=1), or terminate the entire trajectory (ENDX=-1). After the first interrupt, the search will continue for more interrupts of the same kind unless a minus sign was attached to the LOOKX entry as a trigger to cease searching.

The LOOKX interrupt search feature is programmed to accommodate five simultaneous searches. Thus, LOOKX, XLOOK, ENDX, LOOKSW, and SWLOOK are actually five-element arrays whose values may be set either by the user or the program as follows:

LOOKX (1) always available to the user
LOOKX (2) always available to the user

LOOKX (3) available to the user unless the option of finding the best thrust angle from a set of fixed angles is selected

LOOKX (4) available to the user unless perturbing bodies are involved (n-body problems)

LOOKX (5) available to the user unless the optimal engine on-off timing option is selected

Elements 3, 4, and 5 of these arrays may be filled by the program if certain options are selected so that the user must be careful to avoid interfering with preprogrammed searches when he sets up his input. Ordinarily, no more than two user-selected searches are required, and it is always permissible to use the first two elements of these arrays. The interrupts for normal printout and time-specified phase points do not require the use of the LOOKX search scheme.

Choice of coordinate systems. - The basic coordinate system is a Cartesian inertial system with its origin at the center of the primary gravitational body. It has no specific reference axis or reference plane. This system is useful for problems that do not refer to NBODY's built-in ephemeris data. For example, if the user wishes to input the departure and arrival points directly without reference to NBODY's built-in ephemeris, this coordinate system is ideal. However, if the user calls on NBODY to supply ephemeris date, the coordinate system is defined by the mean equinox and ecliptic of date - the x-y plane lies in the ecliptic plane of date and the x-axis points toward the mean equinox of date. By modifying subroutine WORBEL, however, the user may redefine the coordinate system. If, for example, he wishes to use the 1950 mean equatorial system, he would simply supply elliptic ephemeris data in that system instead of the system just defined.

Origin shift. - To minimize integration error in n-body problems, it is necessary to shift the origin of the coordinate system occasionally. These shifts take place whenever the vehicle penetrates a body's sphere of influence. Values of the programmed sphere-of-influence radii are given in table I. These shifts translate the origin to the center of the dominant gravitating body while keeping the coordinate axes aligned. (There is no rotation of axes.) A printout message is issued each time the origin is shifted, and the integration procedure is restarted.

Orbit element integration. - In many problems where the perturbation forces are relatively small it is advantageous to integrate a set of six orbit elements instead of the rectangular coordinates because fewer integration steps are required for the same accuracy. The NBODY user may select orbit element integration as an option only for non-variational problems (problems not involving the adjoint equations for optimal thrust control). Instead of integrating the  \( \dot{R} \) and  \( \dot{V} \) equations the following set of equations is integrated (ref. 14):
\[
\dot{e} = \sqrt{\frac{\mu}{\rho}} \left[ (\sin \nu) \mathcal{R} + \frac{1}{e} \left( \frac{\rho - 1 - e^2}{\rho} \right) \mathcal{V} \right] \quad \text{(Eccentricity)} \tag{109a}
\]

\[
\dot{\omega} = \sqrt{\frac{\mu}{\rho}} \left[ \frac{\sin \nu}{e} \left( 1 + \frac{\rho}{p} \right) \mathcal{R} - \left( \frac{\cos \nu}{e} \right) \mathcal{V} - \left( \frac{\rho}{p} \right) \sin u \cot i \right] \quad \text{(Argument of pericenter)} \tag{109b}
\]

\[
\dot{\Omega} = \left( \frac{r}{\sqrt{\mu} \sin i} \right) \mathcal{N} \quad \text{(Longitude of ascending node)} \tag{109c}
\]

\[
i = \left( \frac{r}{\sqrt{\mu} \cos u} \right) \mathcal{N} \quad \text{(Inclination)} \tag{109d}
\]

\[
\dot{M} = n + \sqrt{\frac{\mu}{\rho}} \left[ 1 - e^2 \right] \left[ \left( \frac{\cos \nu}{e} - 2 \frac{\rho}{p} \right) \mathcal{R} - \frac{\sin \nu}{e} \left( 1 + \frac{\rho}{p} \right) \mathcal{V} \right] \quad \text{(Mean anomaly)} \tag{109e}
\]

\[
\dot{p} = \left( 2r \sqrt{\frac{\mu}{\rho}} \right) \mathcal{V} \quad \text{(Semilatus rectum)} \tag{109f}
\]

where

\[
u = \omega + \nu \quad \text{(Argument of latitude)} \tag{110}
\]

\[
n = \pm \sqrt{\frac{\mu}{p^3}} \left[ 1 - e^2 \right]^{\frac{3}{2}} + \text{if } e < 1, - \text{if } e > 1 \tag{111}
\]

Here \( \nu \) is the true anomaly; \( n \) is the mean angular motion; and \( \mathcal{R}, \mathcal{V}, \) and \( \mathcal{N} \) are the radial, circumferential, and normal perturbative acceleration components, respectively:

\[
\mathcal{R} = a_x (\cos u \cos \Omega - \sin u \sin \Omega \cos i) + a_y (\cos u \sin \Omega + \sin u \cos \Omega \cos i) + a_z (\sin u \sin i) \tag{112a}
\]

\[
\mathcal{V} = a_x (-\sin u \cos \Omega - \cos u \sin \Omega \cos i) + a_y (-\sin u \sin \Omega + \cos u \cos \Omega \cos i) + a_z (\cos u \sin i) \tag{112b}
\]
\[ N = a_x \sin \Omega \sin i - a_y \cos \Omega \sin i + a_z \cos i \]  

(112c)

Here \( a_x, a_y, \) and \( a_z \) are the components of the perturbative acceleration along the \( x, y, \) and \( z \) axes (i.e., the sum of the four rightmost terms of eq. (27)). The true anomaly \( \nu \) requires solving Kepler's equation iteratively for the eccentric anomaly \( E \) (or \( F \))

\[
M = E - e \sin E \quad (e < 1)
\]

(113a)

\[
M = -F + e \sinh F \quad (e > 1)
\]

(113b)

before substituting \( E \) (or \( F \)) into

\[
\cos \nu = \frac{\cos E - e}{1 - e \cos E} \quad (e < 1)
\]

(114a)

\[
\cos \nu = \frac{\cosh F - e}{1 - e \cosh F} \quad (e > 1)
\]

(114b)

This set of orbit element equations experiences numerical difficulties under any of the following conditions: (1) \( e \sim 1 \); (2) \( e \sim 0, \rightangle \neq 0, \angle \neq 0 \); (3) \( i \sim 0 \) and \( N \neq 0 \); and (4) whenever a vehicle approaches an asymptote while on a hyperbolic orbit. In these situations the program will temporarily shift from orbit element integration to rectangular coordinate integration until the difficulty subsides. A printout message is issued whenever such an integration shift takes place.

**Coordinate transformations.** - A transformation from orbit element to rectangular coordinates is sometimes required for numerical reasons and also because the user may wish to input orbit elements but to integrate rectangular coordinates. The transformation equations are given following sketch (g), which illustrates the geometry.
\[ x = r(\cos \Omega \cos u - \sin \Omega \sin u \cos i) \] (115a)

\[ y = r(\sin \Omega \cos u + \cos \Omega \sin u \cos i) \] (115b)

\[ z = r(\sin u \sin i) \] (115c)

\[ \dot{x} = -\sqrt{\frac{\mu}{p}} (N \cos i \sin \Omega + Q \cos \Omega) \] (115d)

\[ \dot{y} = \sqrt{\frac{\mu}{p}} (N \cos i \cos \Omega - Q \sin \Omega) \] (115e)

\[ z = \sqrt{\frac{\mu}{p}} (N \sin i) \] (115f)

where

\[ r = \frac{p}{1 + e \cos \nu} \] (116)

\[ N = e \cos \omega + \cos u \] (117)

\[ Q = e \sin \omega + \sin u \] (118)

The inverse transformation equations are

\[ p = \frac{h^2}{\mu} \] (119a)

\[ i = \tan^{-1} \left( \frac{\sqrt{h_x^2 + h_y^2}}{h_z} \right) \] (119b)

\[ \Omega = \tan^{-1} \left( \frac{h_x}{-h_y} \right) \] (119c)

\[ \omega = \tan^{-1} \left[ \frac{z \sin i + (y \cos \Omega - x \sin \Omega) \cos i}{x \cos \Omega + y \sin \Omega} \right] - \nu \] (119d)
\[ e = \sqrt{1 + \frac{p}{\mu}} \]  \hspace{1cm} (119e)

\[ M = \tan^{-1}\left(\frac{\sin E}{\cos E}\right) - e \sin E \quad (e < 1) \]  \hspace{1cm} (119f)

\[ M = -\ln\left(-\sin E + \sqrt{\sin^2 E + 1}\right) - e \sin E \quad (e > 1) \]  \hspace{1cm} (119g)

where

\[ h_x = y \dot{z} - z \dot{y} \]  \hspace{1cm} (120a)

\[ h_y = z \dot{x} - x \dot{z} \]  \hspace{1cm} (120b)

\[ h_z = x \dot{y} - y \dot{x} \]  \hspace{1cm} (120c)

\[ h^2 = h_x^2 + h_y^2 + h_z^2 \quad \text{(Unit angular momentum)} \]  \hspace{1cm} (121)

\[ \sin E = \frac{\sqrt{1 - e^2 \sin \nu}}{1 + e \cos \nu} \]  \hspace{1cm} (122)

\[ \cos E = \frac{e + \cos \nu}{1 + e \cos \nu} \]  \hspace{1cm} (123)

\[ \nu = \tan^{-1}\left[\frac{hR \cdot V}{\mu (p - r)}\right] \]  \hspace{1cm} (124)

\[ r^2 = x^2 + y^2 + z^2 \]  \hspace{1cm} (125)

**Earth-fixed coordinate frame.** For many launch vehicle problems it is convenient to specify the departure conditions in terms of an Earth-fixed frame of reference. The Earth-fixed equatorial frame is related to a space-fixed frame as shown in sketch (h). The Earth-fixed position vector is specified by the radius \( r \), north latitude \( \varphi \), and east longitude \( \theta \).
It is convenient to translate the Earth-fixed velocity vector \( \mathbf{V}_r \) to the end of the position vector and project it on the local horizontal. Then it is specified by its magnitude \( v_r \), path angle or elevation angle above the local horizontal \( \gamma \), and the north azimuth \( \sigma \) as shown in the sketch. The transformation between the Earth-fixed spherical coordinates and the space-fixed Cartesian coordinates is

\[
\begin{align*}
x &= r \cos \varphi \cos \theta \\
y &= r \cos \varphi \sin \theta \\
z &= r \sin \varphi \\
x' &= v_r (\Phi \cos \vartheta - \cos \gamma \sin \sigma \sin \theta) - y \omega_r \\
y' &= v_r (\Phi \sin \vartheta + \cos \gamma \sin \sigma \cos \theta) + x \omega_r \\
z' &= v_r (\sin \varphi \sin \gamma + \cos \varphi \cos \sigma \cos \gamma)
\end{align*}
\]

where \( \omega_r \) is the Earth's rotation rate and

\[
\Phi = \cos \varphi \sin \gamma - \sin \varphi \cos \gamma \cos \sigma
\]

Since this transformation is not the mean-ecliptic and equinox-of-date system, the inclusion of n-body effects is not permitted for launch vehicle problems which use the Earth-centered coordinates for input unless the user alters subroutine WORBEL to redefine the coordinate system, as explained in the previous section Choice of coordinate systems.
Two problems emerge if one attempts to use these Earth-fixed coordinates for a launch vehicle starting from rest and aimed straight vertically. First, if \( v_r = 0 \), defining the thrust direction relative to the velocity vector results in an undefined thrust direction at lift-off. And, secondly, the lift-off thrust should be aligned with the sensible gravity direction, which is not identical to the radial direction (\( \gamma = 90^\circ \)) in the case of an oblate or rotating Earth. To avoid the first difficulty, the launch vehicle is assumed to rise vertically for a short time \( t_v \) and atmospheric forces are ignored, which leads to a closed-form solution for the changes in relative velocity \( \Delta v_r \) and radius \( \Delta r \),

\[
\Delta v_r = c_0 \ln \left( \frac{m_0}{m} \right) - gt_v \tag{128}
\]

\[
\Delta r = v_0 t_v + c_0 \frac{m_0}{m_0} \left[ 1 - \frac{m}{m_0} \left( 1 + \ln \frac{m_0}{m} \right) \right] - \frac{1}{2} g t_v^2 \tag{129}
\]

where

\[ m = m_0 + \dot{m}_0 t_v \tag{130} \]

\[ c_0 = c + \frac{pA e}{m_0} \tag{131} \]

The subscript 0 refers to values at lift-off. The numerical integration is begun just after this short vertical rise with an instantaneous tilt of the velocity vector to the desired path angle \( \gamma \) (generally between 85\(^\circ\) and 89.5\(^\circ\)) and azimuth \( \sigma \).

To avoid the second difficulty (vertical direction not identical to sensible gravity direction), small corrections are made to the latitude \( \varphi \) used in the preceding transformation equations so that the rocket will be aligned with the sensible gravity direction when \( \gamma = 90^\circ \). In effect, this helps avoid the problem of having a low-acceleration (\( 1 \approx \dot{a}_0 / g \approx 1.2 \)) launch vehicle turn quickly and crash into the ground just because the vehicle's velocity and thrust vectors are not properly aligned to the net external force field. The correction for an oblate Earth model is to replace the geocentric latitude \( \varphi \) with a simple approximation to the geodetic latitude \( \varphi^* \) as illustrated in sketch (i).
This equation is derived by comparison of the oblate potential function written in the geocentric framework (eq. (29)) with a similar function written in the geodetic system, ignoring terms higher than \( J_2 \). Here \( J_2 \) is the second zonal harmonic coefficient and \( a_e \) is the Earth's mean equatorial radius. The correction for centrifugal force is

\[
\phi^* \approx \tan^{-1} \left\{ \frac{15}{2} \frac{J_2 a_e^2}{r^2} \left( \sin^2 \phi - 0.6 \right) - 1 \right\} \tan \phi
\]

(132)

This equation is derived by comparison of the oblate potential function written in the geocentric framework (eq. (29)) with a similar function written in the geodetic system, ignoring terms higher than \( J_2 \). Here \( J_2 \) is the second zonal harmonic coefficient and \( a_e \) is the Earth's mean equatorial radius. The correction for centrifugal force is

\[
\Delta \phi \approx \frac{\omega^2 r \cos \phi \sin \phi}{g}
\]

(133)

If both effects are present, \( \phi \) is replaced by \( \phi^* + \Delta \phi \) when applying the transformation equations.

**LEVEL 2 - DIRECT OPTIMIZATION OF VEHICLE AND MISSION PARAMETERS**

After the solution to the boundary-value problem (assuming there is one) is obtained \( Y = \overline{Y} \), the program control passes to the level 2 optimizer if there are any additional vehicle or mission parameters to be optimized. The level 2 optimizer is a general-purpose iterator that extremizes a user-specified payoff criterion \( \Gamma \) over a field of user-specified variables \( Z \). It operates on one variable at a time in the same fashion as the univariate search scheme (in fact, the same subroutine is utilized for both schemes). Each time it changes one of the \( Z \) variables, the boundary-value problem of level 1 must be resolved, as illustrated in sketch (j).
The level 2 search cycles over each $Z$ variable in sequence, records the extremum of $\Gamma$ for the first complete iteration, and then repeats the complete iteration a second time. The $\Gamma$ value at the end of the second complete iteration is compared with the value obtained from the first iteration and the entire process repeated until

$$\frac{\Gamma_{i+1} - \Gamma_i}{\Gamma_{i+1}} < 0.001$$  \hspace{1cm} (134)$$

where $i$ refers to values obtained after a complete iteration cycle on all variables.

Each time $Z$ is changed in level 2, the level 1 independent variable vector $\overline{X}$ is also estimated with the functional form $\overline{X} = \overline{X}(Z)$ by using a linear extrapolation. In this regard, it is desirable to avoid such large changes in $Z$ that the resulting $\overline{X}$ estimate is so poor that it hinders convergence of the level 1 boundary-value problem. Thus, $\Delta Z$ is constrained in the sense that if $\Delta Z$ produces an initial level 1 error $\tau$ greater than 0.3, $Z$ is reduced until this constraint is satisfied.

The user specifies the payoff criterion $\Gamma$ by loading its COMMON location into IBB. The most frequently used criterion are given in the following table:

<table>
<thead>
<tr>
<th>Typical level 2 payoff criterion, $\Gamma$</th>
<th>COMMON location (for IBB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Final mass, $m_f$</td>
<td>2159</td>
</tr>
<tr>
<td>Net spacecraft mass, $m_n/m_{ref}$</td>
<td>437</td>
</tr>
<tr>
<td>Payload ratio for launch vehicle problems, $m_n/m_0$</td>
<td>437</td>
</tr>
</tbody>
</table>
The latter two criterion have the same IBB location since they are both calculated with equations (2) and (15), although \( \frac{m_0}{m_{\text{ref}}} = 1 \) for launch vehicles.

The user selects the level 2 independent variable list \( Z \) by loading the COMMON locations of his chosen set into the IAA array during input. Thus, if \( C \) denotes COMMON storage, \( Z = C(\text{IAA}) \). A list of likely candidates for \( Z \) is given in the following table, along with their COMMON locations:

<table>
<thead>
<tr>
<th>Typical level 2 independent variable, ( Z )</th>
<th>COMMON location (for IAA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electric vehicles:</td>
<td></td>
</tr>
<tr>
<td>Specific impulse, ( I )</td>
<td>418</td>
</tr>
<tr>
<td>Initial mass flow rate, ( \dot{m}_0 )</td>
<td>363</td>
</tr>
<tr>
<td>Only one of ( \dot{m}_0 ) these may</td>
<td></td>
</tr>
<tr>
<td>Initial thrust-weight ratio, ( a_0/g )</td>
<td>408</td>
</tr>
<tr>
<td>Initial electric power level, ( P_0 )</td>
<td>397</td>
</tr>
<tr>
<td>Launch vehicle burnout velocity, ( v_L )</td>
<td>429</td>
</tr>
<tr>
<td>Vehicle velocity just prior to capture retro-</td>
<td>430</td>
</tr>
<tr>
<td>fire, ( v_r )</td>
<td></td>
</tr>
<tr>
<td>Departure date, ( t_0 )</td>
<td>11</td>
</tr>
<tr>
<td>Mission time, ( t_f )</td>
<td>1</td>
</tr>
<tr>
<td>Launch vehicles:</td>
<td></td>
</tr>
<tr>
<td>Stage firing times, ( (t_i) )</td>
<td>1, 2, \ldots , 10</td>
</tr>
<tr>
<td>Elevation angle at launch, ( \gamma )</td>
<td>48</td>
</tr>
<tr>
<td>Either vehicle: Desired final conditions in</td>
<td></td>
</tr>
<tr>
<td>level 1, ( \bar{Y} )</td>
<td>866, \ldots , 875</td>
</tr>
</tbody>
</table>

Any or all of the set \( I, \dot{m}_0, v_L, v_r \) may be optimized either in level 2 or in level 1 if the payoff criterion is net spacecraft mass ratio. Level 1 is recommended in this case because it is faster and more accurate. Choosing any other independent variables or payoff criterion requires the use of the level 2 optimizer.

**SWEEP SCHEMES FOR RUNNING SIMILAR CASES**

Studies frequently require a set of answers over a range of some parameter \( s \). The basic problem that arises in generating such a set of answers is to make reasonable estimates of the independent variables (\( X \) and \( Z \)) as \( s \) is "swept" from \( s_1 \) to \( s_n \). If \( s \) is varied too quickly, the successive boundary-value solutions \( X \) cannot be estimated with sufficient accuracy to avoid nonconvergence or unacceptably slow convergence. And if \( s \) is varied too slowly, much computer time will be wasted solving intermediate problems. Thus, the successive values of \( s \) must be chosen carefully to avoid both com-
putational difficulties. Two sweeping schemes are programmed which have the following descriptions:

<table>
<thead>
<tr>
<th>Manual sweeping scheme</th>
<th>Automatic sweeping scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>User must guess each new $s$.</td>
<td>Program guesses each new $s$.</td>
</tr>
<tr>
<td>Sweep is terminated if $\Delta s$ is oversize.</td>
<td>Sweep recovers from oversize $\Delta s$.</td>
</tr>
<tr>
<td>Output occurs for every value of $s$.</td>
<td>Output occurs on selected values of $s$.</td>
</tr>
<tr>
<td>Scope includes levels 1 and 2.</td>
<td>Scope includes level 1 only.</td>
</tr>
<tr>
<td>Program estimates $\bar{X}$ and $\bar{Z}$.</td>
<td>Program estimates $\bar{X}$.</td>
</tr>
</tbody>
</table>

The main advantage of the manual scheme is that it covers both level 1 and level 2, instead of just level 1 as is the case with the automatic scheme. However, the automatic scheme is much more convenient to use and is recommended in all cases except those involving level 2 optimization.

**Manual Sweeps**

In this method a group of cases are executed sequentially and the user selects each new value of the sweeping parameter $s$. This is implemented by submitting a separate data set for each case, as illustrated in figure 2. The first case consists of all the usual data plus a value for NSWEEP which is the COMMON location of the sweep parameter $s$. The only data entered for the subsequent cases are new values of $s$. The program will execute the first case and then the second case starting with the converged solution $(\bar{X}_1$ and $\bar{Z}_1)$ of the first case. Since the manual sweep cannot recover if any of the cases fails to converge, the best policy is to select a small increment in $s$ for the second case (e.g., $s_2 = 1.01 s_1$). The third and remaining cases are started with a linear extrapolation of the two previous solutions for $\bar{X}$ and $\bar{Z}$. The user increases $\Delta s$ to whatever value he believes is satisfactory and may expect several trial-and-error attempts in sensitive problems if he chooses large steps in $s$. It is often more productive to pick small increments in $s$ and accept the extra output and computer execution time.

If the entire sweep is executed to completion, a second sweep may be initiated where the first sweep terminated by resetting the value of NSWEEP in the first data set of the second sweep. Additional sweeps may be appended to the second sweep in a similar fashion.
Automatic Sweeps

The automatic sweep scheme eliminates the guesswork in selecting the $\Delta s$ increments. The user only needs to specify which variable is to be "swept" and the particular values of $s$ for which he desires a full trajectory printout. Only a single data set is required for this scheme, as shown in figure 3. The program will execute the first problem and then proceed to sweep $s$ toward the first printout value $s_1$. All intermediate $s$-values will be noted in the output, but only $s_1$ (and subsequently $s_2, s_3, \ldots, s_n$) will trigger a full trajectory printout. The program attempts to select $\Delta s$ increments that will result in an initial boundary-value problem error of 0.1 each time $s$ is changed. If the error is greater than 0.3 or the boundary-value problem iteration falters, the iteration is terminated and a smaller $\Delta s$ is selected before reiteration. Thus, this scheme recovers from poor estimates of $\Delta s$. As with the manual scheme, the $\bar{X}$ estimates are produced with a linear extrapolation of the previous two solutions. As an additional option, the user may override the linear extrapolation with an $n^{th}$-order least-squares curve-fit extrapolation of the previous $m$ solutions by also inputting

MORDER order of the curve fit, $n$

MAXPTS number of points used in the curve fit, $m$

Experience has shown that the linear extrapolation (and sometimes a quadratic) usually is more productive overall than high-order extrapolations. Finally, the automatic sweep scheme cannot be used in cases of level 2 optimization - only the manual scheme can.

Multidimensional Sweeps

The sweeping method is also very useful in obtaining the solution of a boundary-value problem when one does not have a reasonable estimate of the solution vector $\bar{X}$, but does have the solution to a related problem. In such a case, the related problem can often be transformed into the sought problem by a continuous transformation of the set of independent variables $S$ that differ between the two problems.

The manual sweeping scheme can handle this situation, albeit rather awkwardly, simply by performing multiple sweeps in tandem. Alternatively, the user may vary the entire set of parameters $S$ in parallel by changing each element of $S$ in each data set of a single sweep. The extrapolation of $\bar{X}$ and $\bar{Z}$ will be based on only one element (specified by NSWEEP), however, so this method is prone to failure unless the user is very careful in choosing successive values of $S$. 
The automatic sweeping scheme is better equipped to handle a multidimensional sweep. The user loads the COMMON locations of all elements of S into the IAA vector and the corresponding sought values of S into the SVALUE vector. This is a simple extension of the definitions given in figure 3 for a single-dimensional sweep except that SVALUE represents a set of single values for n parameters instead of n values of a single parameter. The program automatically sweeps all elements of S simultaneously to the target values loaded in SVALUE. It does this by the linear transformation

\[ S = S_1 + l_s (S_2 - S_1) \quad 0 \leq l_s \leq 1 \]  

(135)

where \( S_1 \) is the initial value of S loaded as normal input, \( S_2 \) is the sought value of S loaded in SVALUE, and \( l_s \) is a scalar. The program solves the initial problem for \( S_1 \) and then sweeps \( l_s \) from 0 to 1, which completes the multidimensional parallel sweep since \( S = S_2 \) when \( l_s = 1 \). Other transformations may be better than this linear one in situations where a constraint (such as maintaining a circular orbit) preserves the similarity of the problems, but this one is general and works reasonably well in most cases.

**PROGRAM DESCRIPTION**

**PROGRAM STRUCTURE**

The entire NBODY program is written in the FORTRAN IV (7090 compiler, version 5) language and occupies about 22,000 core storage locations on an IBM 7094. Peripheral equipment is assigned as follows:

1. Logical unit 5: All input is taken from this unit from a single READ command in the main program.

2. Logical unit 6: All output is written on this unit from many of the subroutines.

Most of the variables that are transferred between NBODY's 35 subroutines are located in "labeled COMMON blocks" as follows:

<table>
<thead>
<tr>
<th>COMMON block name</th>
<th>Description of variables in block</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME</td>
<td>time-related variables such as departure time, mission time, and integration step size</td>
</tr>
<tr>
<td>FIXED</td>
<td>fixed physical constants such as ( \pi ) and ( g )</td>
</tr>
<tr>
<td>ENTER</td>
<td>input variables assigned either true or false values</td>
</tr>
<tr>
<td>LAT</td>
<td>input variables for the Earth-fixed spherical coordinate system option</td>
</tr>
</tbody>
</table>

58
COMMON Description of variables in block
block name

LOOK variables related to the interrupt search scheme
CASES bookkeeping variables for running successive cases
OUTPUT output option variables
LOCATE indexes that give locations of variables relative to the beginning of COMMON
IGRATE integration scheme controls and the increments $\Delta y$
COFV variables associated with the optimal thrust control formulation
ROCKET variables that describe the vehicle
TRAJEC variables that describe the trajectory
ITERAT iteration scheme control variables
BODIES variables that describe the gravitational bodies
AERODY variables associated with aerodynamics
SAVE bookkeeping variables that must be saved each time a trajectory is repeated

HD values of the integration variables at the current time (at time $t_n$ in eq. (99)) in double precision

H values of the integration variables at the current time plus any Runge-Kutta subinterval increment (at time $t_n', t_n + (h/2), or t_n + h$ in eq. (100))

HDOT derivatives of the integration variables

The program's built-in flexibility regarding free choices of optimization variables, criterion of merit, interrupt parameters, sweep parameters, and so forth, is implemented by specifying the COMMON locations of these variables. Hence, it is important that these labeled COMMON blocks be loaded in the same sequence as just given. This loading will be handled automatically by many computer software packages, but in others it is necessary to always load the main program (or block data subprogram) first just to ensure the proper loading sequence. The user is also cautioned against changing these COMMON blocks without also changing the prestored indexes in LOCATE. It is generally recommended that any user-supplied additional COMMON be appended after the last block (HDOT) or defined as "unlabeled COMMON." Appendix B is a glossary of the variables appearing in COMMON, along with their relative locations.
An overall flow diagram of the program is given in figure 4. The user's input data set is read by a NAMELIST-type read command in the main program. Control is then passed to subroutine WSTAGE, which initiates phase 1 (same as vehicle stage 1 in many cases) by supplying the appropriate phase data such as initial mass, specific impulse, and so forth, to the integrator. After more initializing in subroutines WORDER and WBEGIN, the trajectory integration is carried out by subroutine WINTEG. The derivatives of the integration variables are computed in WDERIV, the time step sizes are calculated in WSTEP, and the relative integration error is evaluated in WERROR. After the phase 1 trajectory arc is computed, control is returned to WSTAGE for the initialization of phase 2 and it, and any remaining trajectory phases, are computed in a similar manner. After the last trajectory phase is computed, control is passed to subroutine WOPT, which controls the iteration of the boundary-value problem, the level 2 optimization schemes, and the automatic sweep scheme. The program control is passed back to WSTAGE each time a new trajectory must be computed during these processes. When level 1, level 2, and any automatic sweep are all completed, control is finally sent back to the main program for the next case's input data set (if any). The main program also performs the extrapolation on the level 1 (\(\tilde{X}\)) and level 2 (\(\tilde{Z}\)) independent variables if the manual parameter sweep option is selected.

There are many other subprograms that perform specific tasks, and appendix B provides a definition of every subprogram's function. The small TIMLFT routine is of particular concern since it would probably be deleted or rewritten at other installations. It is a convenience routine for batch sequence operation that warns the program when its allotted execution time is almost over. Thus, some useful information can be extracted before an imminent termination by triggering a final trajectory printout. A complete subprogram call sequence diagram is given in figure 5.

**INPUT**

The input data sets are read by a single NAMELIST read command in the main program, and successive cases may be stacked in tandem indefinitely. All variables are input in SI units using floating-point, single-precision format unless otherwise noted. In the list of operating instructions that follows, the input variable names are written entirely in capital letters. The default value of all variables is zero (or F, false, for logical variables) unless otherwise noted. The dimensionality of the coordinate system is specified as follows:

\[
\text{NDEM}=2 \quad \text{two-dimensional model (default value)}
\]

\[
\text{=3} \quad \text{three-dimensional model}
\]
The set of gravitational bodies is specified by a list of indexes:

NUMBOD=index of the origin body, index of the first perturbing body, ..., index of
nth perturbing body (0 ≤ n ≤ 6); default value: 1, 6*0

The first index refers to the origin body at the departure date, and the remaining indexes
are all of perturbing bodies in random order. The vehicles initial coordinates are reference
to the origin body. The permissible indexes and corresponding body names are

1  Sun  7  Saturn
2  Mercury  8  Uranus
3  Venus  9  Neptune
4  Earth-Moon  10  Pluto
5  Mars  11  Earth
6  Jupiter

The physical model for the Earth may be selected as follows:

OBLATE=T  oblate Earth model
    =F  spherical Earth model (default value)
ROTATE=T  rotating Earth
    =F  nonrotating Earth (default value)

The atmospheric Earth model is automatically programmed for the 1962 U.S. Standard
Atmosphere. Altering this model or adding another planet's atmosphere requires re-
programming subroutine WICAO.

Vehicle Model

The program provides the capability to simulate an n-stage vehicle (1 ≤ n ≤ 10). The term "stage" really refers to "trajectory phase" since a "stage" change does not necessarily mean that a vehicle stage is discarded. It may only mean that the thrust steering control is switched from a tangential program to an optimal program, for example. The vehicle related inputs are as follows (1 ≤ i ≤ 10):

VMASS(i)>0  initial mass of stage i, m_0, kg (default value: 1, 9*0)
VMASS(i) = 0 vehicle mass is continuous between stage i - 1 and stage i
<0 vehicle mass decreases between stage i - 1 and stage i by the amount specified for stage i, \( m_0 \), kg

ISP(i) vacuum specific impulse, \( I \), sec

TB(i) > 0 stage flight time, \( t_f \), sec
<0 total flight time of i stages, \( \sum_{j=1}^{i} t_f \), sec

NOPT(i) preprogrammed optimal-thrust-control end condition options (see table IV for a summary or preceding text for complete discussion)

Choose\[\begin{align*}
\text{PFLOW(i)} & \\
\text{TW(i)} & \\
\text{POWER(i)} & 1
\end{align*}\]
propellant flow rate at 1 AU, \( -\dot{m}_0 \), kg/sec
initial thrust-weight ratio (at 1 AU), \( a_0/g \)
initial electric power, \( P_0 \), kW

SOLAR=T propulsion power depends on solar distance (eq. (8))
= F propulsion power is constant (default value)

KE propellant tankage factor, \( k_t \)

STRUCT structural mass factor, \( k_s \)

ALFPOW specific mass of electric propulsion system, \( \alpha_{ps} \), kg/kW

BE, DE overall powerplant efficiency \( \eta \) factors, b and d (default value: \( BE=0.75 \), \( DE=14350 \).

DISPO=T electric propulsion system and tankage mass are jettisoned just prior to high-thrust retromaneuver (\( j = 1 \))
= F electric propulsion system and tankage mass are not jettisoned prior to high-thrust retromaneuver (\( j = 0 \)) (default value)

The last six entries are normally used only for single-stage electric vehicles. For n-stage vehicles, they are applicable to the entire flight as a whole (e.g., the tankage factor \( k_t \) is applied to all stages taken together). The number of stages is taken to be the number of nonzero flight times that are inputted.

\[\text{1This option is valid only when using the analytical launch vehicle simulation and requires that NOPT be equal to 0, 6, or 7. Also instead of inputting VMASS, the reference mass } m_{\text{ref}} \text{ must be loaded into BOOSTM.}\]
The following group of inputs is required only if aerodynamic forces are to be included in the simulation:

- **REFA(i)** aerodynamic reference area of \( i \)th stage, \( S_{\text{ref}, i} \), \( m^2 \)
- **AEXIT(i)** engine exit area, \( A_e, m^2 \)
- **CD0C** set of parasite drag coefficient data (eq. (24)); \( M_1, (a_1, a_2, a_3)_1, M_2, (a_1, a_2, a_3)_2, M_3, \ldots, M_n \); the coefficients \( (a_1, a_2, a_3)_i \) apply to the Mach number interval \( (M_i, M_{i+1}) \)
- **CDIC** set of induced drag coefficient data (eq. (25)); \( M_1, (a_4, a_5, a_6)_1, M_2, (a_4, a_5, a_6)_2, M_3, \ldots, M_n \)
- **CLC** set of lift coefficient data (eq. (26)); \( M_1, (a_7, a_8, a_9)_1, M_2, (a_7, a_8, a_9)_2, M_3, \ldots, M_n \)

The user may install his own method of handling the aerodynamic data by modifying subroutine WAERO.

**Analytic Spiral Escape Maneuver at Departure**

A tangential-thrust spiral escape from a departure planet circular orbit will be simulated for electrically propelled vehicles (eqs. (19) and (55)) if the following are input:

- **SPIR=T**
- **VC1** speed in initial circular orbit, \( v_{c,l}, m/sec \)

**Analytic High-Thrust Departure of Electric Vehicle**

The launch vehicle is assumed to impart a speed \( v_l \) to the electric vehicle at a distance \( r_l \) from the departure planet’s center. The inputs are

- **VB1** launch vehicle’s burnout speed, \( v_l, m/sec \)
- **RRAT1** departure planet’s sphere-of-influence radius ratio, \( r_l/r_s, d \)
- **VC1** circular orbit speed at \( r_l \), \( v_{c,l}, m/sec \)
- **VJET1, K1** curve-fit parameters defining launch vehicle’s performance (eq. (16)), \( c_l, m/sec \), and \( k_l \)
Analytic High-Thrust Capture Retromaneuver of Electric Vehicle

If an electric vehicle is to be braked into a planetary capture orbit at the arrival planet with a high-thrust retrorocket, input the following:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB2</td>
<td>planetocentric vehicle speed just before retrofire at periapsis radius $r_r$, $v_r$, m/sec</td>
</tr>
<tr>
<td>RRAT2</td>
<td>arrival planet’s sphere-of-influence radius ratio, $r_r/r_s$, a</td>
</tr>
<tr>
<td>VC2</td>
<td>circular orbit speed at $r_r$, $v_c$, m/sec</td>
</tr>
<tr>
<td>VJET2</td>
<td>retrojet exhaust speed, $c_r$, m/sec</td>
</tr>
<tr>
<td>K2</td>
<td>retropropulsion tankage factor, $k_{rt}$</td>
</tr>
<tr>
<td>ECC2</td>
<td>eccentricity of capture ellipse, $e_r$</td>
</tr>
</tbody>
</table>

Departure Time

The departure time $t_d$ need only be specified in problems involving ephemerides. It is input as a Julian date in Greenwich time as follows:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTOFFJ</td>
<td>whole Julian day number (default value: 2 440 000.)</td>
</tr>
<tr>
<td>TOFFT</td>
<td>fraction of Julian day</td>
</tr>
</tbody>
</table>

Initial Position and Velocity

The vehicle coordinates at departure may be specified in any of these sets:

1. Rectangular coordinates (double-precision variables):
   - $R$ $x, y, z$ components of position vector $R_0$, m
   - $V$ $x, y, z$ components of velocity vector $V_0$, m/sec

2. Orbit elements (double-precision variables, sketch (g)):
   - $E$ eccentricity, e
   - OMEGA argument of pericenter, $\omega$, rad
   - NODE longitude of ascending node, $\Omega$, rad
   - INCL orbit inclination to reference plane, $i$, rad
   - MA mean anomaly, $M$, rad

64
(3) Earth-fixed spherical coordinates (sketch (h)):

LAT  northern latitude, $\varphi$, deg
LONG eastern longitude from Greenwich, $\theta$, deg
ALT0 altitude above sea level, $r_0 - r_e$, m
VEL0 relative velocity, $v_r$, m/sec
ELEV elevation angle, $\gamma$, deg
AZI azimuth, eastward from north, $\sigma$, deg

TKICK duration of short, vertical, nondrag ascent to facilitate starting (eq. (129)), $t_v$, sec

Alternatively, the user may instruct the program to use ephemerides to compute the departure (and desired arrival) coordinates. This option is intended for Sun-centered two-body problems only where the departure coordinates are taken to be identical to the specified departure planet coordinates (and likewise for the arrival conditions). The option is invoked by setting

EPHEM=T
NUMBOD=1, index of departure planet, index of arrival planet

Thrust Program Options

The previously defined array NOPT determines whether a variational thrust program (NOPT(i)=0) or a nonvariational thrust program (NOPT(i)=0) applies to the $i^{th}$ stage. The inputs depend on which of these two types is selected:

(1) Nonvariational thrust program (NOPT(i)=0):

ALFCOE a set of thrust angle coefficient data (eq. (42)); $t_1$, $\left( a_{10}, a_{11}, a_{12} \right)_1$, $t_2$, $\left( a_{10}, a_{11}, a_{12} \right)_2$, ..., $t_n$, the coefficients $\left( a_{10}, a_{11}, a_{12} \right)_i$ apply to the time interval $(t_i, t_{i+1})$; if $a_{11} = a_{12} = 0$, the thrust angle $\alpha$ equals $a_{10}$.
ALPHAC=T  thrust angle $\alpha$ is referenced to local horizontal

   =F  thrust angle $\alpha$ is referenced to velocity vector

BETA  out-of-orbit-plane thrust angle (sketch (c)), $\beta$, deg

If the user prefers another method of specifying the thrust program, he may do so simply by modifying subroutine WVREL.

(2) Variational thrust program (NOPT(i)≠0):

COAST=T  coast arcs permitted (default)

   =F  coast arcs not permitted

KBODYS  number of gravitating bodies included in variational equations (It may be desirable to limit this number to 1 even though n bodies affect the equations of motion; default value: 1.)

LAMDA  seven element array of initial values of the adjoint variables (Lagrange multipliers): the three components of $\Lambda$ ((kg)(sec)/m), the three components of $\Lambda_r$ (kg/m), and $\lambda_m$

As an alternative to inputting LAMDA, the following set of variables may be input for two-dimensional problems only (sketch (d) and eqs. (74) to (79)):

PS  initial thrust direction relative to x-axis, $\psi_0$, deg

DPS  time derivative of thrust angle, $\dot{\psi}_0$, deg/sec

KAPPA  thrust on-off switching function, $\kappa_0$

DKAPPA  time derivative of on-off switching function, $\dot{\kappa}_0$, sec⁻¹

LAM  scale factor, $\lambda_0$, (kg)(sec)/m (default value: 1.)

The program will always use the latter set if PS≠0. If the thrust angle at any given moment is to be picked from a specified set of angles $\alpha_i$ instead of varying continuously, input

ALF  set of angles $\alpha_i$ (i ≤ 5) referenced according to the value of ALPHAC, deg

Trajectory Integration Controls

The input initial coordinates for any problem may be (1) rectangular coordinates, (2) orbit elements, or (3) Earth-fixed spherical coordinates as previously explained. Regardless of which set of input coordinates is selected, the user may also choose be-
between rectangular coordinate integration or orbit element integration for nonvariational problems. Only rectangular coordinates may be integrated for variational problems. The input coordinates - integration coordinates option is defined by MODEI as follows:

<table>
<thead>
<tr>
<th>Trajectory integration coordinates</th>
<th>Input coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rectangular</td>
<td>Rectangular</td>
</tr>
<tr>
<td>Modei=2</td>
<td>Modei=-1</td>
</tr>
<tr>
<td>Modei=-2</td>
<td>Modei=4</td>
</tr>
<tr>
<td>Orbit elements (non-variational only)</td>
<td>Orbit elements</td>
</tr>
<tr>
<td>Modei=-2</td>
<td>Modei=1</td>
</tr>
<tr>
<td>Modei=-4</td>
<td></td>
</tr>
<tr>
<td>Earth fixed</td>
<td></td>
</tr>
</tbody>
</table>

The following controls affect the accuracy and efficiency of the trajectory integration:

- **EREF** reference relative error, \( \delta_r \) (default value: \( 10^{-4} \))
- **ERLIMT** limit relative error, \( \delta_{\text{limit}} \) (default value: \( 3 \times 10^{-4} \))
- **STEP(i)** initial step size for \( i \)th stage, \( \left( h_1 \right)_i \), sec (default value: \( t_f/100 \))

Output Controls

For problems that involve neither a level 1 boundary-value problem iteration nor a level 2 optimization search, the user selects the frequency of trajectory printout as follows:

- **STEPS** number of trajectory integration steps between printouts
- **DELMAX** time interval between printouts, sec (default value: \( \text{DELMAX} = 50 \text{ days} \))

By default, the first and last trajectories of level 1 and level 2 iteration sequences will be printed out in full, and a one-line summary will be printed out for each intermediate trajectory. After inspecting a computer run, it is occasionally desirable to repeat the run with a request for more trajectories to be printed out in full (to examine odd behavior, for example). This request will be fulfilled if the following is input:

\( \text{NOUT} = n_1, n_2, \ldots, n_l \) \((l \leq 5)\), where each \( n_i \) is the sequence number of the specific trajectory for which printout is desired. These sequence numbers appear as the leftmost entry in the one-line summary printouts (default value: \( \text{NOUT} = 1, 4*0 \)).
Trajectory Interrupt Controls

As explained in the section **Trajectory interrupt** (p. 44), the trajectory may be interrupted occasionally in order to take some specific action. The program may do this automatically in some cases (such as when the engine is turned on or off), but the user may also cause this to happened by inputting the following:

**LOOKX(i)** location relative to COMMON of interrupt parameter \( i \) entered in fixed point format; table III contains a map of these locations; a minus sign on LOOKX(i) will cause the interrupt search to terminate after the first interrupt, otherwise interrupts will continue to occur each time XLOOK(i)=C(LOOKX(i)), where C=COMMON (default value: consult text)

**XLOOK(i)** value that interrupt parameter \( i \) must attain to trigger an interrupt

**ENDX(i)=-1** flight is terminated at interrupt

\(-0\) flight continues after interrupt (default value)

\(-1\) stage is terminated, but flight continues

If the interrupt search is to be delayed until an arbitrary criterion \( y > \bar{y} \) is satisfied, input

**LOOKSW(i)** location relative to COMMON of the delay parameter \( y \) entered in fixed-point format (default value: location of time, t)

**SWLOOK(i)** value that the delay parameter must exceed before interrupt may occur, \( \bar{y} \)

All these interrupt inputs are five-element arrays. The first two elements are always available to the user, but the latter three may not be, as explained in the **Trajectory interrupt** section.

**Level 1 Boundary-Value Problem**

The program will recognize that a two-point boundary-value problem exists if the following are input:

**IA(i)** COMMON location of the \( i^{th} \) independent variable \( x_i \) in fixed-point format \((0 \leq i \leq 10)\)

**IB(i)** COMMON location of the \( i^{th} \) dependent variable \( y_i \) in fixed-point format \((0 \leq i \leq 10)\)
DESIRE(i) desired value of the \( i^{th} \) dependent variable \( \bar{y}_i \) (0 \( \leq \) i \( \leq \) 10)

WEIGHT(i) weighting factor of the \( i^{th} \) residual, \( w_i \) (0 \( \leq \) i \( \leq \) 10); (default value: \( \bar{y}_i \) if \( \bar{y}_i \neq 0 \), 1.0 if \( \bar{y}_i = 0 \), 360 if \( \bar{y}_i = 0 \) and \( \bar{y}_1 \) is path angle)

TOLER convergence criterion, \( \tau \) (default value: \( 10^{-4} \))

ERSTAR relative error value above which the univariate search scheme is used and below which the linear corrective scheme is used, \( \tau^* \) (default value: 1.0)

NBVP trajectory phase number where boundary-value problem begins in fixed-point format (default value: \( j \), where \( j \) is the number of the first stage having NOPT(j)\( \neq \)0)

MAXNUM maximum number of trajectories allowed before execution is terminated (default value: 500.)

The IA and IB vectors are filled automatically by the program if 1 \( \leq \) |NOPT| \( \leq \) 5, as indicated in table IV. For other cases, the COMMON locations may be selected from table II or table III. Also, DESIRE is calculated by the program as the arrival planet's velocity and position if EPHEM=T, as explained in the section Initial Position and Velocity.

Occasionally, the situation arises that successive iterations fluctuate between n coast phases and m coast phases. Convergence difficulty is often experienced in the region of such a boundary, especially when finite difference partials are used. This type of difficulty is avoided if solutions are sought away from such a boundary and an extrapolation is accepted in the boundary's immediate vicinity. An alternative method that sometimes works is to ignore phase shifts near the boundary by setting TSKIP equal to \( t_1, t_2 \) (phase shifts are ignored in the time interval \( (t_1, t_2) \), sec) until convergence is obtained and then releasing this constraint (TSKIP=0, 0) to determine whether n or m phases are optimal.

If any of the vehicle-related variables \( \dot{m}_0 \), \( c \), \( v_l \), or \( v_r \) are to be optimized in level 1, the appropriate COMMON locations are automatically loaded into IA and IB vectors simply by inputting:

\begin{align*}
\text{OPTA}=T & \quad \text{for optimum } \dot{m}_0 \ (\text{or its equivalent, } f/m_0g) \\
\text{OPTC}=T & \quad \text{for optimum } c \\
\text{OPTVB1}=T & \quad \text{for optimum } v_l \\
\text{OPTVB2}=T & \quad \text{for optimum } v_r
\end{align*}
Level 2 Optimization

User-specified variables \( z_i \) will be optimized in level 2 if the following are input:

- **IAA(i)**: COMMON location of the \( i^{th} \) optimization variable \( z_i \) (\( 0 \leq i \leq 10 \))
- **IBB**: COMMON location of the external criterion \( \Gamma \) (default value: location of payload)
- **TOL2**: relative tolerance on \( \Gamma \) to be satisfied for convergence: positive for a maximization problem, negative for a minimization problem (default value: 0.001)
- **MAXNUM**: maximum number of iteration trajectories allowed before execution is terminated (total of level 2 and level 1, if any; default value: 500)
- **PERT2(i)**: initial perturbation size for \( z_i \) (\( 0 \leq i \leq 10 \)), expressed as a fraction of \( z_i \) (default value: 0.001)

Parameter Sweeps

For manual sweeps the user simply inputs successive data sets in tandem, as shown in figure 2, and identifies the sweep parameter \( s \) in the first data set:

- **NSWEEP**: COMMON location of the sweep parameter \( s \) (see table IV for likely candidates)

For automatic sweeps on a single parameter \( s \), the user inputs:

- **IAA**: COMMON location of the sweep parameter \( s \)
- **SVALUE(i)**: sequential set of values of \( s \) for which a full trajectory printout is desired (\( 1 \leq i \leq 10 \))
- **MAXPTS=2**

For multidimensional automatic sweeps on \( n \) sweep parameters, input the following:

- **IAA(i)**: COMMON locations of the sweep parameters \( s_i \) (\( 1 \leq i \leq 10 \))
- **SVALUE(i)**: desired set of \( s_i \) values; each \( SVALUE(i) \) corresponds to \( IAA(i) \)
- **MAXPTS=2**

The automatic sweep schemes may be applied only to level 1 (not level 2). The estimation procedure of the level 1 independent variable array \( X \) is defaulted to a linear
extrapolation of the previous two solutions. The order and number of data points used in this procedure may be changed as explained in the section Automatic Sweeps.

**PROGRAM OUTPUT**

The frequency of output is controlled by the input variables NOUT, STEPS, and DELMAX, as explained in the input instructions. Each trajectory is noted on the printout in either (1) a full output mode or (2) a one-line summary mode.

### Full Output Mode

The full output mode produces information blocks at specified intervals of flight time (DELMAX) or integration step number (STEPS) with the following format:

<table>
<thead>
<tr>
<th>STEP=</th>
<th>ECCENTRICITY=</th>
<th>OMEGA=</th>
<th>V=</th>
<th>R=</th>
<th>REFER=</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME=</td>
<td>SEMILATUS R. =</td>
<td>TRU A=</td>
<td>VX=</td>
<td>X=</td>
<td>RMASS=</td>
</tr>
<tr>
<td>DAYS=</td>
<td>MEAN ANOMALY=</td>
<td>NODE=</td>
<td>VY=</td>
<td>Y=</td>
<td>REV S. =</td>
</tr>
<tr>
<td>ALFA=</td>
<td>PATH ANGLE=</td>
<td>INCL=</td>
<td>VZ=</td>
<td>Z=</td>
<td>DELT=</td>
</tr>
</tbody>
</table>

- **STEP** current integration step number to the left of the plus sign and a count of the step-size cutbacks on the right
- **TIME** current flight time, \( t \), sec
- **DAYS** flight time, \( t \), days
- **ALFA** angle between thrust and velocity vectors, \( \alpha \), deg
- **ECCENTRICITY** orbit eccentricity, \( e \)
- **SEMILATUS R.** semilatus rectum of orbit, \( p \), m
- **MEAN ANOMALY** mean anomaly, \( M \), rad
- **PATH ANGLE** path angle, \( \gamma \), deg
- **OMEGA** argument of pericenter, \( \omega \), rad
- **TRU A** true anomaly, \( \nu \), rad
- **NODE** longitude of ascending node, \( \Omega \), rad
- **INCL** orbit inclination, \( i \), rad
- **V, VX, VY, VZ** velocity and its components, \( V \), m/sec
R, X, Y, Z  
radius and its components, R, m

REFER  
name of reference body followed by integration mode

RMASS  
vehicle mass, kg

REVS.  
revolutions past x-axis

DELT  
current integration step size, h, sec

In the case of atmospheric flight the following two lines are appended to the preceding:

BETA=  
out-of-plane thrust angle, \( \beta \), deg

ALT. =  
altiltude, \( m \)

R PATH ANGLE  
path angle relative to Earth (may be rotating), deg

MACH NUMBER  
Mach number, \( M \)

DRAG, LIFT  
drag and lift acceleration magnitudes, \( |D|/m, l/m; m/sec^2 \)

VR  
velocity relative to rotating Earth, \( V_r \), m/sec

CD  
drag coefficient

G  
net force acting along longitudinal axis of vehicle, Earth g's

Q  
dynamic pressure, \( q \), N/m\(^2\)

PUSH  
thrust acceleration magnitude, \( a \), m/sec\(^2\)

HEAT  
heating rate per unit mass, W/m\(^2\)/sec

In the case of an n-body problem, additional lines of printout give the vehicle-to-perturbing-body position vectors in terms of their magnitudes in meters, followed by the three \( x, y, z \) direction cosines (e.g., EARTH \( R=9.25E8 \ 0.580 \ 0.743 \ 0.335 \)). In the case of variational thrust steering programs, the following two lines are added to the basic output block:

PSI=  
thrust angle relative to x-axis, \( \psi \), deg; and its derivative, \( \dot{\psi} \), deg/sec

DPSI=  

THETA=  

DK=  

K=  

L7=  

L1=  

L2=  

L3=  

L4=  

L5=  

L6=  

PSI, DPSI  
K, DK  
THETA  

central travel angle, \( \theta \), deg
L1, ... , L7  Lagrange multipliers (adjoint variables): the components of $\Lambda$ and $\Lambda_r$, and also $\lambda_m$

The full trajectory output occurs for the first trajectory and the converged solution trajectory. During automatic sweeps, full output will occur for each converged solution that corresponds to the SVALUE list.

One-Line Summary

Each trajectory generated during the level 1 boundary-value iteration is noted in a one-line summary table. The table heading is

<table>
<thead>
<tr>
<th>RUN</th>
<th>ERROR</th>
<th>TIME</th>
<th>INDEPENDENT VARIABLES</th>
<th>DEPENDENT VARIABLES</th>
</tr>
</thead>
</table>

The trajectory number is listed under RUN, the number of engine on-off switch points under N, the level 1 boundary-value error $\tau$ under ERROR, and flight time in seconds under TIME. The remaining columns list the values of the independent variable vector $X$ and the dependent variable vector $Y$. During sweeps or level 2 optimizations, this table is interrupted each time a solution is attained with a one-line notation of the current IAA and IBB values. Finally, the following letters may appear between the RUN and N columns:

- **E** indicates the beginning of a univariate search sequence
- **N** indicates that a new partial derivative matrix $G$ is being generated for the linear correction scheme
- **P** indicates that both search schemes have bogged down and that control will now pass to the linear correction scheme without inhibitor ($\chi = 1.0$) in a last-ditch effort to achieve convergence

There are other printout messages that are intended to be self-explanatory.

EXAMPLE PROBLEMS

EXAMPLE 1 - JUPITER RENDEZVOUS USING THE MULTIDIMENSIONAL SWEEP FEATURE

This example illustrates how the multidimensional sweep method can be used to assist in finding the solution of a problem. The mission is a 500-day Jupiter rendezvous
commencing in a circular orbit at 1 AU. The heliocentric travel angle is fixed at $133^\circ$. The spacecraft's initial thrust-weight ratio is $2 \times 10^{-4}$, the specific impulse is 5000 seconds, coasting is permitted, and the thrust is constant. The final conditions being sought are

1. Radius, $7.778 \times 10^{11}$ meters (Jupiter's distance from Sun)
2. Velocity, 13 062.5 meters per second (Jupiter's circular orbit speed)
3. Path angle, $0^\circ$ (rendezvous condition)
4. Travel angle, $133^\circ$ (assumed)

Suppose we already know the optimum-thrust-angle solution to a similar problem; namely, one that has the same vehicle parameters but different target conditions:

1. Radius, $7.0528 \times 10^{11}$ meters
2. Velocity, 16 570 meters per second
3. Path angle, $30^\circ$
4. Travel angle, $138^\circ$

Since these conditions (especially the path angle) are significantly different than the sought conditions, we may expect trouble if we straightforwardly attempt to begin our search with the same set of adjoint variables. Therefore, we use the multidimensional sweep scheme to gradually transform the known solution to the sought solution. The input is as follows:

- \(TB=4.32E7\) mission time, sec
- \(R=1.49597893E11, 0, 0\) initial position vector on x-axis, m
- \(V=0, 29784.7, 0\) initial position vector in y-direction
- \(VMASS=1000\) initial vehicle mass, kg
- \(TW=2,E-4\) initial thrust-weight ratio
- \(ISP=5000\) specific impulse, sec
- \(COAST=T\) coast arcs permitted
- \(STEPS=100\) output every 100th integration step
- \(NOPT=2\) fixed-travel-angle rendezvous option
- \(SVALUE=7.778E11, 13062.5, 0, 133\) sought values of final conditions
- \(DESIRE=7.0528E11, 16570, 30, 138\) current values of final conditions
- \(WEIGHT(3)=365\) better weighting factor for \(\gamma\) than the default value of 30
LAMDA=3, 3431, 4. 40876, 0, 1. 072E-6, 3. 70E-7, 0, 63. 3065

IAA=866, 867, 868, 869

COMMON locations of the sweep variables (the DESIRE array)

MAXPTS=2

number of points used to extrapolate \( \bar{x} \)

For simplicity, this input list contains no vehicle model variables other than those necessary to generate a trajectory. High-thrust chemical propulsion options have also been ignored - for the same reason.

The output is shown below. Since whole sequence of problems are actually solved in the process of transforming the given solution into the sought solution, the first trajectory printout is followed by the one-line trajectory summaries and, finally, the complete trajectory printout of the sought solution. The one-line trajectory summaries are interrupted each time an intermediate solution is found to present the value of the first sweep variable (final radius in this case) and several other parameters. The computer execution time on the IBM 7094II is 1. 1 minutes.

EXAMPLE I - JUPITER RENDEZVOUS

SAVE INITIAL DATA FOR STAGE 1 OF CASE 1.

REFERENCE BODY IS SUN

2 DIMENSIONS 15G DIFF.EQNS. \( T=2.00000000E-04 \)

STIM= 0.00000000 15= 5000.0000

PFILE=. A. 00000000E-05  REFA = 0

AEQX = 0

LAMDA= 3. 3431, 4. 40876, 0, 1. 072E-6, 3. 70E-7, 0,

63. 3065

correct solution values of the adjoint variables for DESIRE

The output is shown below. Since whole sequence of problems are actually solved in the process of transforming the given solution into the sought solution, the first trajectory printout is followed by the one-line trajectory summaries and, finally, the complete trajectory printout of the sought solution. The one-line trajectory summaries are interrupted each time an intermediate solution is found to present the value of the first sweep variable (final radius in this case) and several other parameters. The computer execution time on the IBM 7094II is 1. 1 minutes.

EXAMPLE I - JUPITER RENDEZVOUS
EXAMPLE 2 - 0.1-AU SOLAR PROBE WITH A SWEEP ON SPECIFIC IMPULSE

This example considers a mission to 0.1 AU using a Titan IIID/Centaur to launch a 10-kilowatt solar-electric spacecraft. A sequence of solutions are sought for specific-impulse values from 2600 to 4000 seconds. From previous experience (ref. 10), it is known that permitting coast flight adds very little to the performance but much to the convergence difficulty for these missions. Hence, we assume optimum thrust steering with the no-coast constraint for simplicity. The Earth-escape phase is simulated analytically, and the electric spacecraft begins its heliocentric flight on the x-axis. The level 1 boundary-value problem is set up such that \( \psi_0, \psi_0, \) and \( v_\ell \) will be iterated to satisfy the optimum flyby conditions at 0.1 AU: \( r_a = 0.1 \) AU and \( \left( \Lambda / \lambda_m \right)_a = 0 \). The launch velocity \( v_\ell \) is used here instead of \( K_0 \) because the power level is fixed at 10 kilowatts. Technically, this leaves \( K_0 \) open for optimization; however, we shall ignore this optimization since the payoff criterion \( m_p / m_{ref} \) is quite insensitive to \( K_0 \).

The input required for this example is as follows:

- **ISP=2600** spacecraft specific impulse, \( I, \) sec
- **TB=4.32E7** mission time, \( t_f, \) sec; equal to 500 days
- **NOPT=7** manual specification of level 1 boundary-value problem with optimum-travel-angle option
- **POWER=10** initial spacecraft electric power, \( P_0, \) kW
- **SOLAR=T** solar power option
- **KE=0.03** low-thrust tankage factor, \( k_t \)
- **STRUCT=0** low-thrust structure factor, \( k_s \)
- **ALFPOW=30** specific powerplant mass, \( a_{ps}, \) kg/kW
- **BE=0.8** overall powerplant efficiency factors, \( b \) and \( d \)
- **DE=15700** reference mass of launch vehicle, \( m_{ref}, \) kg
- **BOOSTM=15500** launch velocity, \( v_\ell, \) m/sec
- **VB1=13375** sphere-of-influence radius ratio, \( r_s / r_\ell \)
- **RRAT1=150** circular orbit speed at 160-n mi launch altitude, \( v_c, \) m/sec
- **VC1=7810** launch vehicle performance parameter, \( c_\ell, \) m/sec
- **VJET1=3811** launch vehicle performance parameter, \( k_\ell \)
R=1.49597893D11, 0, 0 initial heliocentric position vector, R_0, m
V=29765.2, 0 heliocentric velocity of Earth, V_0, m/sec
COAST=F coast arcs not permitted
PS=-88.27 initial thrust angle, \( \psi_0 \), deg
DPS=5.58E-6 initial thrust angle rate, \( \dot{\psi}_0 \), deg/sec
KAPPA=28.3 initial engine on-off switch function, \( \kappa_0 \)
LAM=4.19145 initial magnitude of primer, \( \lambda_0 \), (kg)(sec)/m
EREF=1.E-3 integration scheme relative error control, \( \delta_r \)
ERLIMT=3.E-3 limit relative integration error, \( \delta_{\text{limit}} \)
DELMAX=8640000 output frequency, sec; every 100 days
IA=343, 344, 429 COMMON locations of \( \psi_0, \dot{\psi}_0, v_l \)
IB=480, 363, 364 COMMON locations of \( r, \lambda_1/\lambda_m, \lambda_2/\lambda_m \)
DESIRE=1.49597893E10, 0, 0 desired values of arrival conditions, \( \bar{y} \)
WEIGHT=1.496E11 weighting factor \( w_1 \) for radius
TOLER=0.001 convergence tolerance, \( \tau \)
IAA=418 COMMON location of sweep variable, specific impulse
SVALUE=3000, 3500, 4000 values of specific impulse for which full trajectory
printout is desired
MAXPTS=2 number of points used in extrapolation of \( \bar{X} \)

The output of this example follows. Note that the \( 2\frac{1}{2} \)-revolution solution was found.
(There are also solutions for \( 1/2, 1\frac{1}{2}, 3\frac{1}{2}, \) etc., revolutions.) Full printouts occur for
the first trajectory and for the solutions with I of 3000, 3500, and 4000 seconds. The
computer execution time on the IBM 7094II is 0.9 minute.
EXAMPLE 3 - JUPITER CAPTURE MISSION WITH HIGH-THRUST DEPARTURE
AND CAPTURE AND OPTIMUM VEHICLE PARAMETERS

This example illustrates (1) the analytic high-thrust approximations for the departure and capture phases; (2) the use of transversality conditions to optimize the initial mass flow rate, specific impulse, high-thrust launch velocity, and high-thrust retro-braking; and (3) the alternate method of specifying the initial values of the adjoint variables. We will specify a two-dimensional solar system model with only the Sun's gravitational force acting on the spacecraft. The spacecraft will start its heliocentric path on the x-axis at 1 AU with a velocity equal to Earth's circular velocity plus an incremental velocity from the high-thrust launch vehicle. The launch vehicle is assumed to inject the electric spacecraft at 185-kilometer altitude, after which it coasts to a sphere of influence of radius 150 times the launch radius. Hence,

\[ R_0 = (1.49597893 \times 10^{11}, 0, 0) \text{ m} \]
\[ V_0 = (0, 29765.2, 0) \text{ m/sec} \]
\[ v_{c,1} = 7795 \text{ m/sec (circular speed at 185 km)} \]
\[ r_s, d/r_l = 150 \]

The launch vehicle performance simulates the Atlas/Centaur/SLV-3C:

\[ k_l = 0.369 \]
\[ c_l = 4001 \text{ m/sec} \]

Instead of specifying the reference mass of the launch vehicle in low Earth orbit, a nondimensional approach will be used to permit simple scaling to any reference mass. This is done by specifying that the initial heliocentric mass of the electric vehicle is some convenient number - 1000 kilograms - and letting the program print out the appropriate mass ratios. The electric vehicle assumptions are

1. Specific powerplant mass, \( \alpha_{ps} \), 34 kg/kW
2. Structure mass factor, \( k_s \), 0.1
3. Tankage mass factor, \( k_t \), 0.1
4. Specific impulse, \( I \), 3650 sec
5. Initial thrust-weight ratio, \( f/m_{0g} \), \( 3.73 \times 10^{-5} \)
6. Powerplant efficiency, the default efficiency curve
(7) Type of power source, solar panels using built-in power profile
(8) Thrust program, optimum angle with coast arcs permitted

Since the specific impulse and initial thrust-weight ratio are to be optimized, the values for I and f/m₀g quoted simply serve as first estimates. Likewise, the launch velocity \( v_L \) and the spacecraft velocity just prior to retrofire must also be estimated, although both will be optimized:

1. Launch velocity, \( v_L \), 11540 m/sec
2. Velocity just before retrofire, \( v_r \), 43200 m/sec

After 1200 days of flight time, the high-thrust retropropulsion unit is assumed to brake the entire spacecraft into a parabolic orbit about Jupiter with a periapsis of 2 Jupiter radii. The Jovian sphere of influence for this maneuver is assumed to be 345 times the periapse radius. Hence,

\[
t_0 = 0
\]
\[
t_f = 1.368 \times 10^8 \text{ sec (1200 days)}
\]
\[
e_r = 1
\]
\[
v_{c,r} = 30500 \text{ m/sec (circular speed at 2 Jupiter radii)}
\]
\[
r_{s,a}/r_r = 345
\]

The retropropulsion unit parameters are

\[
c_r = 2940
\]
\[
k_{rt} = 0.2
\]

Instead of guessing initial values of the adjoint variables \( \Lambda, \Lambda_r, \) and \( \lambda_m \), we will use the alternate set of thrust program variables; namely, the thrust angle \( \psi_0 \), its derivative \( \dot{\psi}_0 \), and the engine on-off switch function \( \kappa_0 \):

\[
\psi_0 = 103^\circ
\]
\[
\dot{\psi}_0 = 8.97 \times 10^{-6} \text{ deg/sec}
\]
\[
\kappa_0 = 29
\]
A value of \( \kappa_0 \) is not required because the central travel angle \( \theta_a \) is left open for optimization. The other desired target conditions are assumed to be

1. Jupiter's heliocentric radius, \( r_a \), \( 7.778 \times 10^{11} \) m
2. Jupiter's orbit speed, \( v_a \), \( 13050 \) m/sec
3. Jupiter's path angle, \( \gamma_a \), \( 0^0 \)

These three conditions plus the four transversality conditions for optimum \( c, f/m_0 g, v_l \), and \( v_r \) comprise a seven-variable level 1 boundary-value problem. The presence of the vehicle-related transversality conditions requires generating the partial derivative matrix \( G \) with the finite difference method. Hence, the NOPT=7 option must be used. In this option the COMMON locations of \( r_a, \hat{r}_a, \hat{\gamma}_a \) must be loaded into the IB vector and the locations of \( \psi_0, \hat{\psi}_0 \), and \( \kappa_0 \) into the IA vector. (The locations of the four vehicle-related variables and their transversality conditions are set by the program by inputting OPTA=T, etc.)

The nondefault input values are given here in the same order as presented in the input instructions:

\[
\begin{align*}
VMASS &= 1000, \quad ISP=3650, \quad TB=1.0368E8, \quad NOPT=7, \\
TW &= 3.73E-5, \quad SOLAR=T, \quad KE=0.1, \quad STRUCT=0.1, \\
ALFPOW &= 34, \quad VB1=11540, \quad RRAT1=150, \quad VC1=7795 \\
VJET1 &= 4001, \quad K1=0.369, \quad VB2=43200, \quad RRAT2=345 \\
VC2 &= 30500, \quad VJET2=2940, \quad K2=0.2, \quad ECC2=1, \\
R &= 1.49597893D11, \quad 0, \quad 0, \quad V=0, \quad 29765.2, \quad 0, \\
PS &= 103, \quad DPS=8.97E-6, \quad KAPPA=29, \quad EREF=1.E-3, \\
ERLIMT &= 3.E-3, \quad DELMAX=3.456E7, \quad IA=343, \quad 344, \quad 345, \\
IB &= 480, \quad 493, \quad 479, \quad DESIRE=7.778E11, \quad 13050, \quad 5*0, \\
OPTA &= T, \quad OPTC=T, \quad OPTVB1=T, \quad OPTVB2=T
\end{align*}
\]

The output for this example is reproduced on the following pages.
### Phase 3 - Jupiter Orbiter

**Save Initial Date for Stage 1 of Case 1.**

**Reference Date is SUN**

2 Dimensions 85 Diff. Eqns. T/5 3.7279499E+05 ISPH 3650.000000 PHEL 1.02010785E+05 REFRA 0 AEXIT 0

<table>
<thead>
<tr>
<th>Step</th>
<th>Omega</th>
<th>Time</th>
<th>Semi Latus</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
<th>Path Angles</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.244392</td>
<td>1.03809E+08</td>
<td>0.6121790</td>
<td>3.9231377</td>
<td>1.4959799E+11</td>
<td>-7.01E-08</td>
<td>0.6121790</td>
<td>3.9231377</td>
<td>1.4959799E+11</td>
<td>-7.01E-08</td>
<td>0.6121790</td>
<td>3.9231377</td>
<td>1.4959799E+11</td>
<td>-7.01E-08</td>
<td>0.6121790</td>
<td>3.9231377</td>
</tr>
<tr>
<td>1</td>
<td>0.013945</td>
<td>1.0149E+08</td>
<td>0.6121790</td>
<td>3.9231377</td>
<td>1.4959799E+11</td>
<td>-7.01E-08</td>
<td>0.6121790</td>
<td>3.9231377</td>
<td>1.4959799E+11</td>
<td>-7.01E-08</td>
<td>0.6121790</td>
<td>3.9231377</td>
<td>1.4959799E+11</td>
<td>-7.01E-08</td>
<td>0.6121790</td>
<td>3.9231377</td>
</tr>
</tbody>
</table>

#### Trajectory Interrupt: (Clocks) = 0

**Step 1:**
- Omega: 0.479615
- Semi Latus: 0.639412

**Step 2:**
- Omega: 0.479615
- Semi Latus: 0.639412

**Step 3:**
- Omega: 0.481624
- Semi Latus: 0.639412

**Step 4:**
- Omega: 0.479615
- Semi Latus: 0.639412

**Step 5:**
- Omega: 0.479615
- Semi Latus: 0.639412

**Step 6:**
- Omega: 0.479615
- Semi Latus: 0.639412

**Phase 1 Complete.** Delv: 873.3 Mass Ratio: 0.7382 *** Total Delv: 837.3 Total Mass Ratio: 0.7852 Payload Ratio: 0.0167

**Keplerian Orbit Data:**
- Mean Anomaly: 7.80581E+11
- Eccentricity: 7.80626E+11
- Semi-major Axis: 7.806108E+11

**M+3 7, Mass-F, EPHEM-F, NPSEVF, KOMDSF, RESF= 1.000000 NS= 0, R= 437**

**Run Number:**
- Time: 103.0000

**1 Independent Variables:**
- **Primary Variables:**

---

**85**
The optimization of $f/m_0g$, $c$, $v_L$, and $v_R$ could also have been accomplished with the level 2 optimization scheme instead of with transversality conditions. In this case, the OPTA, OPTC, OPTVB1, and OPTVB2=T statements would be deleted from the input list and, instead, the following would be needed:

$$\text{IAA} = 408, 418, 429, 430$$

These numbers are the COMMON locations of the four vehicle-related variables to be optimized. By default, the optimization criterion is payload ratio $m_n/m_{ref}$ ($\text{IBB}=437$). Also, it would be more economical in this case to change NOPT=7 to NOPT=3 so that the partial derivative matrix $G$ would be integrated rather than computed by finite differencing. The computer execution time on the IBM 7094II is 0.9 minute.

SINGLE-STAGE LAUNCH VEHICLE WITH CHEMICAL AND NUCLEAR PROPULSION

The performance of an advanced, hypothetical single-stage Earth shuttle is sought. The shuttle uses conventional chemical propulsion during lift-off and ascent through the atmosphere but switches to nuclear propulsion for the upper trajectory phase. This upper phase terminates with injection into a 444,165-meter (240-n mi) circular parking orbit. A zero angle-of-attack thrust program is assumed for the chemical boost phase and an optimal steering program for the nuclear upper phase. To be specific, assume that the shuttle has the following description:

Vehicle:

1. Gross lift-off mass, $2 \times 10^6$ kg
2. Maximum cross-sectional area, 100 m$^2$
3. Drag coefficient, $C_D = 0.4 + 0.6M^2 (0 \leq M \leq 1)$;
   $1.15306 - 0.16326M + 0.010204M^2 (1 < M \leq 8); 0.5 (M > 8)$

Chemical engine:

1. Vacuum specific impulse, 425 sec
2. Ratio of thrust to lift-off weight, 1.25
3. Exit area, 40 m$^2$
4. Specific weight, 0.02

Nuclear engine:

1. Vacuum specific impulse, 1200 sec
2. Propellant flow rate, 140 kg/sec
3. Specific weight, 1/3
The payoff criterion is payload delivered into orbit, and for this calculation it will be assumed that the tankage factor $k_t$ is 0.1 and the structure factor $k_s$ is 0.053. This particular value of structure factor is simply the total engine mass divided by the gross lift-off mass:

\[
\frac{(\text{Chemical engine mass}) + (\text{Nuclear engine mass})}{\text{Gross lift-off mass}} = \frac{0.02 m_0 \left( \frac{f_0}{m_0 g} \right) + \frac{1}{3} (mD)_{\text{nuclear}}}{m_0} = 0.02(2\times10^6)(1.25) + \frac{1}{3}(140)(1200) = \frac{2\times10^6}{0.053} = 0.053
\]

Of course, in any real shuttle there would be many additional items (such as radiation shielding, reentry structure, landing engines, and fuel) that should also be subtracted from the injected weight to calculate net payload, but these items are simply lumped together with net payload and called gross payload in this illustration.

A rotating, spherical Earth model is assumed and, for convenience, a due-eastward launch from an equatorial site is assumed so that the calculations need be done in only two dimensions. The launch site is also assumed to be at the Greenwich meridian (zero longitude) and 10 meters above mean sea level. The short vertical rise segment $t_v$ is assumed to be 20 seconds. After 20 seconds, the vehicle is instantaneously tilted to 90° azimuth (eastward launch) and to an elevation angle initially assumed to be 89.40° but later optimized for maximum gross payload. The elevation angle $\gamma$ strongly affects the amount of trajectory lofting and must be carefully chosen to avoid paths that go straight up ($\gamma$ too close to 90°) and paths that fall back to Earth ($\gamma$ too far from 90°). Lift-off acceleration and vertical rise duration strongly affect the proper choice of $\gamma$, and experience dictated the choice of $t_v = 20$ seconds and $\gamma = 89.40°$.

The level 1 boundary-value problem is set up as follows:
With this set of conditions plus the usual optimal-travel-angle assumption, the NOPT(2)=7 option is needed for the second stage. (Option NOPT(1)=0 is required for the first stage.)

A level 2 optimization scheme is set up so that the initial elevation angle $\gamma$ and the amount of chemical propulsion $(t_f)_1$ are optimized to yield maximum gross payload. The initial guesses for the level 1 and level 2 independent variables are

Level 1:

\[ \psi_0 = 54^\circ \]
\[ \dot{\psi}_0 = 0.053 \text{ deg/sec} \]
\[ (t_f)_2 = 1100 \text{ sec} \]

Level 2:

\[ \gamma = 89.4^\circ \]
\[ (t_f)_1 = 220 \text{ sec} \]

Finally, the use of the trajectory interrupt feature is illustrated by requiring a trajectory step printout to occur if the path angle attains zero before orbit injection. This occurs whenever the acceleration level is low enough to produce lob-type trajectories.

The input for this case is given here:

NUMBOD=11, ROTATE=T, VMASS=2. E6, ISP=425, 1200,
TB=220, 1100, NOPT=0, 7, TW=1.25, PFLOW(2)=140,
KE=0.1, STRUCT=0.053, REFA=100, AEXIT=40,
CDOC=0, 0.4, 0, 0.6, 1, 1.15306, -0.18326, 0.010204, 8, 0.5, 0, 0, 100,
LAT=0, LONG=0, ALT0=10, ELEV=89.4, AZI=90, TKICK=20,
The small perturbation size (0.003) for the elevation angle is necessary to prevent non-convergence difficulties during the level 2 search procedure. The output for this case is presented below. Note that two trajectory phases are indeed listed: the zero angle-of-attack, chemical propulsion, atmospheric phase; and the optimum steering, nuclear propulsion vacuum phase. The level 1 boundary-value problem concerns only phase 2, while the level 2 optimization is over both phases. The IBM 7094 computer execution time is 4.5 minutes.
### Table: Step-by-Step Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Time Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Altitude Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Page Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

---

### Table: Step-by-Step Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Time Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Altitude Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Page Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

---

### Table: Step-by-Step Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Time Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Altitude Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Page Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

---

### Table: Step-by-Step Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Time Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Altitude Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>

### Table: Page Data

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Date</th>
<th>Page</th>
<th>Path</th>
<th>Ret</th>
<th>Alt.</th>
<th>Pst.</th>
<th>L10</th>
<th>L20</th>
</tr>
</thead>
<tbody>
<tr>
<td>55-4</td>
<td>1305</td>
<td>25-11</td>
<td>1127</td>
<td>00</td>
<td>0.046</td>
<td>0.029</td>
<td>0.006</td>
<td>0.07</td>
<td>0.14</td>
</tr>
</tbody>
</table>
Lewis Research Center,  
National Aeronautics and Space Administration,  
Cleveland, Ohio, September 11, 1973,  
502-04.
APPENDIX A

SYMBOLS

\[ A_e \] engine exit area, \( m^2 \)
\[ a \] thrust acceleration magnitude, \( m/sec^2 \)
\[ a_e \] Earth's equatorial radius, \( m \)
\[ a_x, a_y, a_z \] components of total perturbating acceleration, \( m/sec^2 \)
\[ a_1, \ldots, a_{12} \] curve-fit coefficients
\[ B \] \( V_r \times H_r \)
\[ b \] electric thruster efficiency parameter
\[ C \] vector constant of motion, \( kg \)
\[ \omega \] perturbative acceleration in circumferential direction, \( m/sec^2 \)
\[ C_D \] total drag coefficient
\[ C_{DI} \] induced drag coefficient
\[ C_{DO} \] parasite drag coefficient
\[ C_L \] lift coefficient
\[ c \] jet exhaust speed of vehicle, \( m/sec \)
\[ c_l \] launch vehicle performance parameter, \( m/sec \)
\[ c_r \] jet exhaust speed of high-thrust retroengine, \( m/sec \)
\[ D \] vehicle drag force vector, \( N \)
\[ d \] electric thruster efficiency parameter, \( m/sec \)
\[ E \] eccentric anomaly, \( rad \)
\[ e \] orbit eccentricity
\[ e_r \] eccentricity of planetary capture orbit
\[ F \] eccentric anomaly equivalent for hyperbolic orbits, \( rad \)
\[ f \] thrust force magnitude, \( N \)
\[ G \] partial derivative matrix for two-point boundary-value problem
\[ g \] universal gravitational constant, \( m/sec^2 \)
\[ H_r \] relative angular momentum per unit mass vector, \( m^2/sec \)
absolute angular momentum magnitude, m^2/sec; and integration step size, sec

specific impulse, sec

orbit inclination, rad

unit vectors along x, y, z axes

zonal harmonic oblateness coefficients

retrorocket jettison indicator

launch vehicle performance parameter

retrosystem tankage factor

structure factor

tankage factor

Runge-Kutta subinterval increments

lift force vector, N

lift force magnitude, N

transformation factor used in multidimensional sweeps

Mach number; and mean anomaly, rad

vehicle mass, kg

net spacecraft mass, kg

propellant mass, kg

propulsion system mass, kg

retrosystem mass, kg

retropropellant mass, kg

retrosystem tankage mass, kg

reference mass in planetary orbit, kg

structure mass, kg

tankage mass, kg

perturbative acceleration normal to orbit plane, m/sec^2

mean motion

instantaneous electric power available from power source, W
electric power available from power source at 1-AU distance from Sun, W
atmospheric pressure, N/m²; and semilatus rectum, m
e sin ω + sin u
dynamic pressure, N/m²
position vector of vehicle, m
perturbative acceleration in outward radial direction, m/sec²
distance from origin to vehicle, m
radius of launch vehicle at injection, m
radius of retrofire maneuver at arrival planet, m
sphere-of-influence radius of arrival planet, m
sphere-of-influence radius of departure planet, m
vector of sweep parameters
aerodynamic reference area, m²
sweep parameter
unit vector in thrust direction
transversality conditions for \( \dot{m}_0, c, v_l, \) and \( v_r \)
time, sec
flight duration of a stage, sec
duration of low-thrust escape spiral maneuver, sec
time of short vertical rise for launch vehicles, ignoring atmosphere
gravitational potential function, m²/sec²; and argument of latitude, rad
absolute vehicle velocity vector, m/sec
vehicle speed relative to a planet, m/sec
vehicle speed, m/sec
circular orbit speed about departure planet at radius \( r_l \), m/sec
circular orbit speed about arrival planet at radius \( r_r \), m/sec
launch speed of spacecraft when analytic launch vehicle simulation is invoked, m/sec
\[ v_r \] spacecraft speed just prior to an analytic high-thrust retrofire maneuver, m/sec; and relative spacecraft speed, m/sec

\[ \Delta v_r \] retrofire speed increment, m/sec

\[ v_{s,a,s,d} \] vehicle speed as it passes through arrival (departure) planet's sphere of influence, m/sec

\[ W \] weighting matrix for end condition residuals

\[ w_i \] diagonal elements of W

\[ X \] vector of level 1 independent variables

\[ X, Y, Z \] inertial Cartesian coordinate axes

\[ x, y, z \] components of vehicle position, m

\[ x_i \] \( i^{th} \) element of X

\[ Y \] vector of level 1 dependent variables

\[ y_i \] \( i^{th} \) element of Y

\[ y_n \] value of integration variable at \( n^{th} \) step

\[ Z \] vector of level 2 optimization variables

\[ z_i \] \( i^{th} \) element of Z

\[ \alpha \] angle between thrust vector and velocity vector (numerically identical with angle of attack), deg

\[ \alpha_c \] angle between thrust vector and circumferential direction, deg

\[ \alpha_{ps} \] specific weight of propulsion system, kg/kW

\[ \beta \] out-of-orbit thrust angle, deg

\[ \Gamma \] level 2 optimization criterion

\[ \gamma \] vehicle path angle, rad

\[ \delta \] integration scheme truncation error

\[ \delta_{\text{limit}} \] acceptable limit value of \( \delta_r \)

\[ \delta_r \] relative truncation error (between fourth-order Runge-Kutta scheme and lower order scheme)

\[ \delta() \] partial derivative with respect to arbitrary variable

\[ \epsilon \] engine on-off indicator

\[ \zeta \] ratio \( P/P_r \)

\[ \eta \] thruster efficiency
\( \theta \) central travel angle, rad

\( \varphi \) east longitude relative to Greenwich, rad

\( \kappa \) engine on-off switching function

\( \Lambda \) vector of velocity-related adjoint variables (primer vector), (kg)(sec)/m

\( \Lambda_r \) vector of position-related adjoint variables, kg/m

\( \lambda \) magnitude of primer vector \( \Lambda \), (kg)(sec)/m

\( \lambda_c \) adjoint variable for engine exhaust speed \( c \), (kg)(sec)/m

\( \lambda_m \) adjoint variable for mass

\( \lambda_{m_0} \) adjoint variable for initial mass flow rate \( \dot{m}_0 \), sec

\( \lambda_{V_l} \) adjoint variable for analytic launch vehicle speed \( v_l \), (kg)(sec)/m

\( \lambda_{V_r} \) adjoint variable for analytic retrofire speed \( v_r \), (kg)(sec)/m

\( \lambda_1, \ldots, \lambda_7 \) components of \( \Lambda \), components of \( \Lambda_r \), and \( \lambda_m \) in that order

\( \mu \) gravitational constant, m\(^3\)/sec\(^2\)

\( \nu \) true anomaly, rad

\( \xi \) empirical factor used in spiral escape equations

\( \rho \) atmospheric density, kg/m\(^3\)

\( \sigma \) azimuth measured eastward from north, rad

\( \tau \) boundary-value-problem error criterion

\( \tau^* \) value of \( \tau \) separating univariate scheme domain from linear correction scheme domain

\( \Phi \) \( \cos \varphi \sin \gamma - \sin \varphi \cos \gamma \cos \sigma \)

\( \phi \) geocentric latitude, rad

\( \phi^* \) geodetic latitude, rad

\( \chi \) inhibitor for linear correction scheme

\( \Psi \) time-dependent term of Runge-Kutta truncation error

\( \psi \) angle between thrust vector and x-axis, rad

\( \Omega \) longitude of ascending node, rad

\( \omega \) argument of periapsis, rad

\( \omega_r \) rotation rate of Earth, rad/sec
Subscripts:

\( a \)  
arrival value

\( x, y, z \)  
x, y, z components of vector

\( 0 \)  
departure value

Superscripts:

\( 0 \)  
reference trajectory value

\( . \)  
derivative with respect to time \( t \)

\( \prime \)  
derivative with respect to radius \( r \)

\( - \)  
desired value

\( \sim \)  
modified arrival planet value
APPENDIX B

SUBPROGRAM GLOSSARY

WAERO computes lift and drag acceleration

WALSO provides auxiliary computation after each integration step, such as a check for zero vehicle mass and determining the optimum fixed-thrust-angle switch function

WALTER alters the independent variables \( X \) of level 1

WBEGIN initializes variables and controls for the boundary-value problem involved with optimal thrust steering

WCREEP univariate search scheme

WCURVE least squares \( n \)-order curve fit to \( m \) points

WDERIV evaluates derivatives of integration variables

WELIPS computes position and velocity of a body in elliptic orbit

WEPHEM computes \( n \)-body accelerations, and position and velocity of bodies

WELEM transforms rectangular coordinates to orbit elements

WENDST auxiliary computations between trajectory phases

WERROR computes relative integration errors between fourth-order Runge-Kutta scheme and low-order scheme

WGauss Gauss-Jordan elimination solution to a set of linear equations

WICAO U. S. Standard Atmosphere 1962 model

WINTEG Runge-Kutta (fourth order) integrator, also low-order integrator

WLOOK computes jump discontinuities or takes other appropriate action at trajectory interrupt points

WMARCH automatic parameter sweep scheme

WNR finite difference method (generalized Newton-Raphson) of generating partial derivatives for boundary-value problem

WOBLAT computes oblateness acceleration

WOPT master controller of level 1 boundary-value-problem iteration, level 2 variable optimization, and the automatic parameter sweep scheme

WORBEL computes analytic time-series approximate ephemerides
WORDER sorts gravitational body list so that a body's position from the reference is dependent upon positions already computed for other bodies

WOUT basic trajectory output routine

WPENAL computes boundary-value-problem error function

WPOWER computes the solar power ratio $P/P_r$ and its derivatives with respect to distance

WPUSH computes thrust acceleration for nonoptimal steering

WQUAD curve-fit model based on $n$ quadratic functions pieced together

WRXV computes unit angular momentum

WSTAGE prepares data for use in integrator by updating key variables (mass, specific impulse, etc.) with current trajectory phase data

WSTEP computes integration step size and searches for trajectory interrupts

WTUDES transforms Earth-fixed spherical coordinates to space-fixed inertial rectangular coordinates

WREL computes velocity relative to a rotating planet and the nonoptimal thrust angle

WXFER tests for and translates the coordinate system origin when a sphere of influence is penetrated

TIMLFT calculates the amount of computer execution time remaining (in 1/60-sec units) before execution is terminated by the system monitor. This is a Lewis Research Center non-FORTRAN routine that uses the $\$IBFTC$ card time estimate for batch sequencing operation. A dummy FORTRAN version is substituted for other users unless otherwise requested. The function of this routine is to provide a warning that the job is about to be prematurely terminated, thus giving the program an opportunity to print out the best unconverged trajectory instead of being "thrown-off" without gaining any useful information.
REFERENCES


### TABLE I. - ASSUMED PHYSICAL DATA

<table>
<thead>
<tr>
<th>Constant</th>
<th>Assumed value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Astronomical unit, m</td>
<td>1.495978730×10^{11}</td>
</tr>
<tr>
<td>Gravitational constant of the Sun, AU^3/day^2</td>
<td>2.959122083×10^{-4}</td>
</tr>
<tr>
<td>Earth's rotation rate, ωₚ, rad/sec</td>
<td>7.29211515×10^{-5}</td>
</tr>
<tr>
<td>Equatorial Earth radius, aₑ, m</td>
<td>6378 160</td>
</tr>
<tr>
<td>J₂ zonal harmonic coefficient for Earth</td>
<td>1082.7×10^{-6}</td>
</tr>
<tr>
<td>J₃ zonal harmonic coefficient for Earth</td>
<td>-2.56×10^{-6}</td>
</tr>
<tr>
<td>J₄ zonal harmonic coefficient for Earth</td>
<td>-1.58×10^{-6}</td>
</tr>
<tr>
<td>Body Reciprocal mass</td>
<td>Sphere-of-influence radius, m</td>
</tr>
<tr>
<td>Sun</td>
<td>1.0×10^{20}</td>
</tr>
<tr>
<td>Mercury</td>
<td>1.0×10^{3}</td>
</tr>
<tr>
<td>Venus</td>
<td>6.14×10^{6}</td>
</tr>
<tr>
<td>Earth-Moon</td>
<td>9.25×10^{8}</td>
</tr>
<tr>
<td>Mars</td>
<td>5.78×10^{8}</td>
</tr>
<tr>
<td>Jupiter</td>
<td>4.81×10^{10}</td>
</tr>
<tr>
<td>Saturn</td>
<td>5.46×10^{10}</td>
</tr>
<tr>
<td>Uranus</td>
<td>5.17×10^{10}</td>
</tr>
<tr>
<td>Neptune</td>
<td>8.61×10^{10}</td>
</tr>
<tr>
<td>Pluto</td>
<td>3.81×10^{10}</td>
</tr>
<tr>
<td>Earth</td>
<td>9.25×10^{8}</td>
</tr>
<tr>
<td>Moon</td>
<td>1.60×10^{8}</td>
</tr>
</tbody>
</table>

^aEarth reference.
TABLE II. - COMMON LOCATIONS OF ANTICIPATED CANDIDATES FOR BOUNDARY-VALUE VARIABLES

(a) Independent variables, $X$, for $IA$ vector

<table>
<thead>
<tr>
<th>Variable</th>
<th>FORTRAN name</th>
<th>COMMON location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage flight times, $(t_f)_i$, sec</td>
<td>TB</td>
<td>1, 2, ..., 10</td>
</tr>
<tr>
<td>Elevation angle for launch vehicles, $\gamma$, deg</td>
<td>ELEV</td>
<td>48</td>
</tr>
<tr>
<td>Initial thrust angle relative to x-axis, $\psi_0$, deg</td>
<td>PS</td>
<td>343</td>
</tr>
<tr>
<td>Initial thrust-angle rate, $\dot{\psi}_0$, deg/sec</td>
<td>DPS</td>
<td>344</td>
</tr>
<tr>
<td>Initial value of engine on-off switch function, $\kappa_0$</td>
<td>KAPPA</td>
<td>345</td>
</tr>
<tr>
<td>Time derivative of $\kappa_0$, $\dot{\kappa}_0$, sec$^{-1}$</td>
<td>DKAPPA</td>
<td>346</td>
</tr>
<tr>
<td>Initial values of adjoint variables, $\Lambda_i$, $\lambda_i$, $\lambda_m$</td>
<td>LAMDA</td>
<td>347 to 353</td>
</tr>
<tr>
<td>Stage initial propellant flow rates, $(-m_0)_i$, kg/sec</td>
<td>PFLOW</td>
<td>397</td>
</tr>
<tr>
<td>Initial power level, $P_0$, kW</td>
<td>POWER</td>
<td>397</td>
</tr>
<tr>
<td>Stage initial thrust-weight ratios, $f/m_0g$</td>
<td>TW</td>
<td>408 to 417</td>
</tr>
<tr>
<td>Launch speed of electric spacecraft, $v_1$, m/sec</td>
<td>VB1</td>
<td>429</td>
</tr>
<tr>
<td>Spacecraft speed just prior to high-thrust retromaneuver, $V_R$, m/sec</td>
<td>VB2</td>
<td>430</td>
</tr>
<tr>
<td>Nonvariational thrust program coefficients, $a_{10}$, $a_{11}$, $a_{12}$</td>
<td>ALFCOE</td>
<td>1458 to 1507</td>
</tr>
<tr>
<td>$x$, $y$, $z$ components of initial velocity, $V_0$, m/sec</td>
<td>V</td>
<td>2161, 2163, 2165</td>
</tr>
<tr>
<td>$x$, $y$, $z$ components of initial position, $R_0$, m</td>
<td>R</td>
<td>2167, 2169, 2171</td>
</tr>
</tbody>
</table>

(b) Dependent variables, $Y$, for $IB$ vector

<table>
<thead>
<tr>
<th>Variable</th>
<th>FORTRAN name</th>
<th>COMMON location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orbit elements, $e$, $\omega$, $\Omega$, $i$, $M$, $p$</td>
<td>ORBELS</td>
<td>447 to 452</td>
</tr>
<tr>
<td>Energy per unit mass, $J/kg$</td>
<td>ENERGY</td>
<td>462</td>
</tr>
<tr>
<td>Path angle, $\gamma$, deg</td>
<td>PATH</td>
<td>479</td>
</tr>
<tr>
<td>Radius, $r$, m</td>
<td>RADIUS</td>
<td>480</td>
</tr>
<tr>
<td>Central travel angle, $\theta$, deg</td>
<td>THETA</td>
<td>485</td>
</tr>
<tr>
<td>True anomaly, $\nu$, deg</td>
<td>TRU</td>
<td>486</td>
</tr>
<tr>
<td>$x$, $y$, $z$ components of position, $R$, m</td>
<td>X, Y, Z</td>
<td>487 to 489</td>
</tr>
<tr>
<td>$x$, $y$, $z$ components of velocity, $V$, m/sec</td>
<td>VX, VY, VZ</td>
<td>490 to 492</td>
</tr>
<tr>
<td>Velocity magnitude, $v$, m/sec</td>
<td>VEL</td>
<td>493</td>
</tr>
<tr>
<td>Block name</td>
<td>Variable name</td>
<td>Relative location</td>
</tr>
<tr>
<td>------------</td>
<td>---------------</td>
<td>------------------</td>
</tr>
<tr>
<td>TIME</td>
<td>TB</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>DTOFF</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>TOPFT</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>TABLT</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>TMAX</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>TTEST</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>TOL</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>DEBT</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TO</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>STEP</td>
<td>10</td>
</tr>
<tr>
<td>FIXED</td>
<td>AU</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>SPD</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>G</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>RE</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>RESQRD</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>SQEDK1</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>DEGREE</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>PI</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>TWOP1</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>DUMMY1</td>
<td>10</td>
</tr>
<tr>
<td>ENTER</td>
<td>END</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>T</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>EPHEM</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>OBLATE</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>ROTATE</td>
<td>6</td>
</tr>
<tr>
<td>LAT</td>
<td>LAT</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>LONG</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>AZI</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>ELEV</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>VELO</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>ALTO</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>TKICK</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>DUMMY2</td>
<td>8</td>
</tr>
<tr>
<td>LOOK</td>
<td>XLOOK</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>LOOKSW</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>SWLOOK</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>ENDX</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>LOOKY</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>DUMMY3</td>
<td>6</td>
</tr>
<tr>
<td>CASES</td>
<td>NSWEEP</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>NSAVE</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>RECALL</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>KCASE</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>NCASE</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>NCASES</td>
<td>6</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>STEPS</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>DELMAX</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>ADOUT</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>NOUT</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>NBUG</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>STEP1</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>STEP2</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>DEBUG</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>OUTPUT</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>OUTEND</td>
<td>10</td>
</tr>
<tr>
<td>Block name</td>
<td>Variable name</td>
<td>Relative location</td>
</tr>
<tr>
<td>------------</td>
<td>---------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>LOCATE</td>
<td>LCAPPA</td>
<td>1</td>
</tr>
<tr>
<td>LLAMDA</td>
<td>( \Lambda )</td>
<td>2</td>
</tr>
<tr>
<td>LV</td>
<td>( V )</td>
<td>3</td>
</tr>
<tr>
<td>LVEL</td>
<td>( v )</td>
<td>4</td>
</tr>
<tr>
<td>LPATH</td>
<td>( \gamma )</td>
<td>5</td>
</tr>
<tr>
<td>LSIMP</td>
<td>( I )</td>
<td>6</td>
</tr>
<tr>
<td>LFLOW</td>
<td>( m_p )</td>
<td>7</td>
</tr>
<tr>
<td>LVBI</td>
<td>( v_l )</td>
<td>8</td>
</tr>
<tr>
<td>LVBI2</td>
<td>( v_l^* )</td>
<td>9</td>
</tr>
<tr>
<td>LTC</td>
<td>( T_c )</td>
<td>10</td>
</tr>
<tr>
<td>LTA</td>
<td>( T_m )</td>
<td>11</td>
</tr>
<tr>
<td>LTVB1</td>
<td>( T_{vl} )</td>
<td>12</td>
</tr>
<tr>
<td>LTVB2</td>
<td>( T_{vl}^* )</td>
<td>13</td>
</tr>
<tr>
<td>LALFSW</td>
<td>( \Delta(\Lambda \cdot T_i) )</td>
<td>14</td>
</tr>
<tr>
<td>LPSI</td>
<td>( \psi_0 )</td>
<td>15</td>
</tr>
<tr>
<td>LRMAG</td>
<td>( r )</td>
<td>16</td>
</tr>
<tr>
<td>LLMF</td>
<td>( \Lambda )</td>
<td>17</td>
</tr>
<tr>
<td>LTHETA</td>
<td>( \theta )</td>
<td>18</td>
</tr>
<tr>
<td>JSLOPE</td>
<td>( \xi )</td>
<td>19</td>
</tr>
<tr>
<td>LTV</td>
<td>( T_{vf} )</td>
<td>24</td>
</tr>
<tr>
<td>NDM</td>
<td>( \Delta )</td>
<td>1</td>
</tr>
<tr>
<td>AI</td>
<td>( \Delta )</td>
<td>2</td>
</tr>
<tr>
<td>ADJOIN</td>
<td>( \Delta )</td>
<td>3</td>
</tr>
<tr>
<td>ADJOINNT</td>
<td>( \Delta )</td>
<td>4</td>
</tr>
<tr>
<td>DONE</td>
<td>( \Delta )</td>
<td>5</td>
</tr>
<tr>
<td>ERROR</td>
<td>( \Delta )</td>
<td>6</td>
</tr>
<tr>
<td>EMONE</td>
<td>( \Delta )</td>
<td>7</td>
</tr>
<tr>
<td>ERLG</td>
<td>( \Delta )</td>
<td>8</td>
</tr>
<tr>
<td>H2</td>
<td>( \Delta )</td>
<td>9</td>
</tr>
<tr>
<td>ESTART</td>
<td>( \Delta )</td>
<td>10</td>
</tr>
<tr>
<td>ETOL</td>
<td>( \Delta )</td>
<td>11</td>
</tr>
<tr>
<td>ER</td>
<td>( \Delta )</td>
<td>12</td>
</tr>
<tr>
<td>EXMODE</td>
<td>( \Delta )</td>
<td>13</td>
</tr>
<tr>
<td>KBERROR</td>
<td>( \Delta )</td>
<td>14</td>
</tr>
<tr>
<td>KSUB</td>
<td>( \Delta )</td>
<td>15</td>
</tr>
<tr>
<td>NETAGE</td>
<td>( \Delta )</td>
<td>16</td>
</tr>
<tr>
<td>NSTYPE</td>
<td>( \Delta )</td>
<td>17</td>
</tr>
<tr>
<td>MORDI</td>
<td>( \Delta )</td>
<td>18</td>
</tr>
<tr>
<td>EREF</td>
<td>( \Delta )</td>
<td>19</td>
</tr>
<tr>
<td>ERLIMIT</td>
<td>( \Delta )</td>
<td>20</td>
</tr>
<tr>
<td>STEPMX</td>
<td>( \Delta )</td>
<td>21</td>
</tr>
<tr>
<td>NRQ</td>
<td>( \Delta )</td>
<td>22</td>
</tr>
<tr>
<td>NSTART</td>
<td>( \Delta )</td>
<td>23</td>
</tr>
<tr>
<td>RATIO</td>
<td>( \Delta )</td>
<td>24</td>
</tr>
<tr>
<td>XW整数</td>
<td>( \Delta )</td>
<td>25</td>
</tr>
<tr>
<td>HINC</td>
<td>( \Delta )</td>
<td>26</td>
</tr>
<tr>
<td>ASSO</td>
<td>( \Delta )</td>
<td>27</td>
</tr>
<tr>
<td>ADSLAM</td>
<td>( \Delta )</td>
<td>28</td>
</tr>
<tr>
<td>CAPPA</td>
<td>( \Delta )</td>
<td>29</td>
</tr>
<tr>
<td>CV</td>
<td>( \Delta )</td>
<td>30</td>
</tr>
<tr>
<td>DCAPPA</td>
<td>( \Delta )</td>
<td>31</td>
</tr>
<tr>
<td>FCV</td>
<td>( \Delta )</td>
<td>32</td>
</tr>
<tr>
<td>HAM</td>
<td>( \Delta )</td>
<td>33</td>
</tr>
<tr>
<td>HAMMAX</td>
<td>( \Delta )</td>
<td>34</td>
</tr>
<tr>
<td>ALF</td>
<td>( \Delta )</td>
<td>35</td>
</tr>
<tr>
<td>NALF</td>
<td>( \Delta )</td>
<td>36</td>
</tr>
<tr>
<td>Block name</td>
<td>Variable name</td>
<td>Relative location</td>
</tr>
<tr>
<td>------------</td>
<td>---------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>COFY</td>
<td>NCAPPA</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>NLAMDA</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>NOPT</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>PLXOTL</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>PSI</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>PS</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>DPS</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>KAPPA</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td>DKAFFPA</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>LAMDA</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>IMAX</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>COAST</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>LAM</td>
<td>55</td>
</tr>
<tr>
<td></td>
<td>ALFSWT</td>
<td>56</td>
</tr>
<tr>
<td></td>
<td>TRAC</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>TRAA</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>TRAVB1</td>
<td>59</td>
</tr>
<tr>
<td></td>
<td>TRAVB2</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>LAMDAF</td>
<td>61</td>
</tr>
<tr>
<td></td>
<td>GETDOT</td>
<td>67</td>
</tr>
<tr>
<td></td>
<td>OPTA</td>
<td>73</td>
</tr>
<tr>
<td></td>
<td>OPTC</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>OPTVB1</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>OPTVB2</td>
<td>76</td>
</tr>
<tr>
<td>ROCKET</td>
<td>ROCKETALFPow</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>POWER</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>BE</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>DE</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>BOOSTM</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>FW</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>K1</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>K2</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>KE</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>POWER</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>VMASS</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>TW</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>ISP</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>SOLAR</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>VBI</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>VB2</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>VJET1</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>VJET2</td>
<td>54</td>
</tr>
<tr>
<td></td>
<td>AO</td>
<td>55</td>
</tr>
<tr>
<td></td>
<td>CPAR</td>
<td>56</td>
</tr>
<tr>
<td></td>
<td>ETAPOW</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>FLOW</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>PAY</td>
<td>59</td>
</tr>
<tr>
<td></td>
<td>PUSH</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>PUSH0</td>
<td>61</td>
</tr>
<tr>
<td></td>
<td>RMAS0</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>SIMP</td>
<td>83</td>
</tr>
<tr>
<td></td>
<td>TMAG</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>VJET</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td>FLOWX</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>OXISO</td>
<td>87</td>
</tr>
<tr>
<td></td>
<td>STRUCT</td>
<td>68</td>
</tr>
</tbody>
</table>
### TABLE III. - Continued. GLOSSARY OF COMMON VARIABLES

<table>
<thead>
<tr>
<th>Block name</th>
<th>Variable name</th>
<th>Relative location</th>
<th>Absolute location</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRAJEC</td>
<td>ORBELS</td>
<td>1</td>
<td>447</td>
<td>Six-element array of orbit elements e, ω, ξ, i, M, p (in radians and meters)</td>
</tr>
<tr>
<td></td>
<td>U</td>
<td>7</td>
<td>453</td>
<td>Eccentric anomaly, E, rad</td>
</tr>
<tr>
<td></td>
<td>RAMC</td>
<td>8</td>
<td>454</td>
<td>Three components of relative angular momentum, Hₚ, m²/sec²</td>
</tr>
<tr>
<td></td>
<td>RAM</td>
<td>11</td>
<td>457</td>
<td>Relative angular momentum magnitude,</td>
</tr>
<tr>
<td></td>
<td>RAMSRD</td>
<td>12</td>
<td>458</td>
<td>Square of</td>
</tr>
</tbody>
</table>
|           | ALPHAC        | 13                | 459               | Input option indicating circumferential thrust angle reference (using ALF or ALFcoe) if TRUE.
|           | BETA          | 14                | 460               | Out-of-orbit plane thrust angle, β, deg |
|           | EBC2          | 15                | 461               | Eccentricity of planetary capture orbit, eₓ |
|           | ENERGY        | 16                | 462               | Vehicle energy per unit mass, m²/sec² |
|           | SPIR          | 17                | 463               | Input option indicating an analytic Earth escape spiral if TRUE. |
|           | VC1           | 18                | 464               | Circular orbit speed about departure planet (at radius r₁), Vc₁, m/sec |
|           | VC2           | 19                | 465               | Circular orbit speed about arrival planet (at radius r₂), Vc₂, m/sec |
|           | RRAT1         | 20                | 466               | Radius ratio r₁/r₂ at departure planet |
|           | RRAT2         | 21                | 467               | Radius ratio r₁/r₂ at arrival planet |
|           | ALPHA         | 22                | 468               | Angle between thrust and velocity vectors, α, deg |
|           | AMC           | 23                | 469               | Angular momentum components, m²/sec |
|           | AM            | 26                | 472               | Magnitude of angular momentum, h, m²/sec |
|           | AMSQRD        | 27                | 473               | Square of h, m⁴/sec² |
|           | COSALF        | 28                | 474               | cosine α |
|           | COSBET        | 29                | 475               | cosine β |
|           | COSTRU        | 30                | 476               | cosine γ |
|           | DELV          | 31                | 477               | Change in vehicle velocity, Δv, m/sec |
|           | EPSI          | 32                | 478               | Derivative of thrust angle, ψ, rad/sec |
|           | PATH          | 33                | 479               | Path angle of vehicle, γ, deg |
|           | RADIUS        | 34                | 480               | Vehicle’s position vector magnitude, r, m |
|           | RSQRD         | 35                | 481               | Square of r, m² |
|           | SINALF        | 36                | 482               | sine α |
|           | SINBET        | 37                | 483               | sine β |
|           | SINTRU        | 38                | 484               | sine γ |
|           | THETA         | 39                | 485               | Central travel angle, δ, deg |
|           | TRU           | 40                | 486               | True anomaly, ν, rad |
|           | X             | 41                | 487               | x-component of position vector R, m |
|           | Y             | 42                | 488               | y-component of position vector R, m |
|           | Z             | 43                | 489               | z-component of position vector R, m |
|           | VX             | 44                | 490               | x-component of velocity vector V, m/sec |
|           | VY             | 45                | 491               | y-component of velocity vector V, m/sec |
|           | VZ             | 46                | 492               | z-component of velocity vector V, m/sec |
|           | VEL            | 47                | 493               | Vehicle speed, v, m/sec |
|           | VEL            | 48                | 494               | Speed squared, v², m²/sec² |
| ITERAT    | IA             | 1                 | 495               | Ten-element array of COMMON locations of the level 1 independent variables X |
|           | IAA           | 11                | 505               | Ten-element array of COMMON locations of the level 2 optimization variables Z |
|           | IB             | 21                | 515               | Ten-element array of COMMON locations of the level 1 dependent variables Y |
|           | IBB            | 31                | 525               | COMMON location of the level 2 criterion of merit T |
|           | MAXNUM         | 32                | 526               | Maximum number of trajectories allowed for a particular case |
|           | WEIGHT         | 33                | 527               | Ten-element array of level 1 weighting factors for boundary-value problems w₁ |
|           | NUM            | 43                | 537               | Length of IA array (dimensionality of level 1 boundary-value problem) |
|           | NUM2           | 44                | 538               | Length of IAA array (dimensionality of level optimization problem) |
|           | JNUM           | 45                | 539               | IA+IAA |
|           | DAMP           | 46                | 540               | Inhibitor for level 1 linear correction scheme, χ |
|           | CHANGE         | 47                | 541               | Array increments in the level 1 independent variable vector ΔX |
|           | XIA            | 47                | 561               | Reference value of the level 1 independent variation vector X |
|           | XIB            | 87                | 562               | Reference value of the level 2 optimization variable vector Z |
|           | NKUNB          | 107               | 601               | Run number of best trajectory yet calculated |
|           | TOLER          | 108               | 602               | Level 1 iteration convergence tolerance, γ |
|           | ELEMEM         | 109               | 603               | 10 x 11 Element (double precision) partial derivative matrix G |
|           | PERTW          | 329               | 823               | Ten-element array of perturbation factors for univariate search scheme |
### TABLE III. - Continued. GLOSSARY OF COMMON VARIABLES

<table>
<thead>
<tr>
<th>Block name</th>
<th>Variable name</th>
<th>Relative location</th>
<th>Absolute location</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITERAT</td>
<td>PERTNR</td>
<td>339 833</td>
<td>Ten-element array of perturbation factors for linear correction scheme</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PERT2</td>
<td>349 843</td>
<td>Ten-element array of perturbation factors for level 2 search scheme</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VALUE</td>
<td>359 853</td>
<td>Set of desired sweep variable values, ( a_i )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAXPTS</td>
<td>369 863</td>
<td>Maximum number of points to be used in the sweep extrapolation of ( X )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MORDER</td>
<td>379 874</td>
<td>Order of curve fit to be used in the sweep extrapolation of ( X )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KOUNT</td>
<td>371 855</td>
<td>Iteration counter for level 1 boundary-value problems</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DESIRE</td>
<td>372 866</td>
<td>Ten-element array of desired values of the dependent vector ( \mathbf{y} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TSkip</td>
<td>372 876</td>
<td>Two-element array specifying a time interval in which trajectory interrupts are inhibited, sec</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ERBar</td>
<td>394 878</td>
<td>Preferred value of initial level 1 error for each step of an automatic sweep</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ERStar</td>
<td>395 879</td>
<td>Value of level 1 boundary-value error separating univariate and linear correction schemes, ( \gamma )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NBnp</td>
<td>396 880</td>
<td>Stage number defining beginning of level 1 boundary-value problem</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NRun</td>
<td>387 881</td>
<td>Trajectory counter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RETURN</td>
<td>388 882</td>
<td>Internal control used to indicate boundary-value-problem termination</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ATOE</td>
<td>389 883</td>
<td>Internal control indicating which stage data are saved for boundary-value problems</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XMISL</td>
<td>390 884</td>
<td>Current lowest error ( \gamma_i ) obtained during the level 1 iteration process</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TOL2</td>
<td>391 885</td>
<td>Tolerance criterion used between level 2 loops to indicate convergence</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DUMMYY</td>
<td>392 886</td>
<td>See DUMMYY above</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PNAME</td>
<td>401 887</td>
<td>Alphabetical list of gravitational body names defining permissible body choices</td>
<td></td>
</tr>
<tr>
<td></td>
<td>REFER</td>
<td>410 896</td>
<td>Alphabetical list of reference body names corresponding to PNAME list</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AMASS</td>
<td>411 897</td>
<td>List of body masses corresponding to PNAME list, Sun mass units</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RCRIT</td>
<td>412 898</td>
<td>List of body sphere-of-influence radii corresponding to PNAME list, m</td>
<td></td>
</tr>
<tr>
<td>BODIES</td>
<td>NUMBOD</td>
<td>413 899</td>
<td>Input list of selected body numbers, corresponds to PNAME list</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BODIES</td>
<td>420 900</td>
<td>Alphabetical list of body names corresponding to NUMBOD list of indexes</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BNAME</td>
<td>421 901</td>
<td>Same list as BODIES but reordered for computational purposes</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BMass</td>
<td>422 902</td>
<td>List of body masses corresponding to BNAME list, Sun mass units</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EFMRS</td>
<td>423 903</td>
<td>Alphabetical list of ephemerides needed for a particular case</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GKM</td>
<td>430 904</td>
<td>Gravitational constant of the origin body, ( \mu, \text{m}^3/\text{sec}^2 )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GKM</td>
<td>431 905</td>
<td>Square root of GKM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HBODY</td>
<td>432 906</td>
<td>Index list corresponding to BNAME that indicates position of reference body (also in BNAME)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MBody</td>
<td>433 907</td>
<td>Number of perturbating bodies selected</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NBODY</td>
<td>434 908</td>
<td>Total number of bodies selected</td>
<td></td>
</tr>
<tr>
<td></td>
<td>KBODY</td>
<td>435 909</td>
<td>Number of bodies selected for inclusion in the variational equations</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Xmass</td>
<td>436 910</td>
<td>Mass scaling factor (usually from 0 to 1) that may be varied to smoothly include ( n )-body effects</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NCHAMP</td>
<td>437 911</td>
<td>BNAME index of the dominant gravitational body</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NEFMs</td>
<td>438 912</td>
<td>Index list indicating location of EFMRS bodies in PNAME list</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TRESF</td>
<td>439 913</td>
<td>Control whose value is .TRUE. when an origin shift is required</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lbody</td>
<td>440 914</td>
<td>An unconditional origin shift to LBody will take place at trajectory termination if Lbody is loaded (alphanumeric)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RCRIT</td>
<td>441 915</td>
<td>List of body sphere of influences corresponding to BNAME list, m</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VEFM</td>
<td>442 916</td>
<td>( 3 \times 8 ) Array of velocity components of vehicle relative to all bodies, m/sec</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SQDK</td>
<td>443 917</td>
<td>Gravitational constant of the Sun, ( \mu, \text{m}^3/\text{sec}^2 )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XPBody</td>
<td>444 918</td>
<td>Control indicating an origin transfer is in progress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TDAAT</td>
<td>445 919</td>
<td>( 3 \times 8 ) Array of ephemerides data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XP</td>
<td>446 920</td>
<td>( 3 \times 8 ) Array of perturbing body position components relative to the origin, m</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OBla</td>
<td>447 921</td>
<td>Control whose value is .TRUE. if oblateness effects are being included</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RB</td>
<td>448 922</td>
<td>( 3 \times 8 ) Array of position components of the vehicle to all bodies, m</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NMAG</td>
<td>449 923</td>
<td>List of distances of the vehicle to all bodies, m</td>
<td></td>
</tr>
<tr>
<td>AKEROY</td>
<td>ALT</td>
<td>1 1263</td>
<td>Vehicle altitude above ground, m</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AREA</td>
<td>2 1264</td>
<td>Vehicle's aerodynamic reference area for current stage ( S_{ref} ), ( \text{m}^2 )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>REFa</td>
<td>3 1265</td>
<td>Ten-element array of stage aerodynamic reference areas, ( \text{m}^2 )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CD</td>
<td>4 1266</td>
<td>Total vehicle drag coefficient, ( \text{CD} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cl</td>
<td>5 1267</td>
<td>Vehicle lift coefficient, ( \text{CL} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Density</td>
<td>6 1268</td>
<td>Atmospheric density, ( \rho, \text{kg/m}^3 )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Press</td>
<td>7 1269</td>
<td>Atmospheric pressure, ( \rho, \text{N/m}^2 )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TM</td>
<td>8 1270</td>
<td>Atmospheric molecular scale temperature, ( \text{K} )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ExitA</td>
<td>9 1271</td>
<td>Engine exit area, ( A_{exit} ), ( \text{m}^2 )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>T</td>
<td>10 1272</td>
<td>Dynamic pressure, ( q, \text{N/m}^2 )</td>
<td></td>
</tr>
<tr>
<td>Block name</td>
<td>Variable name</td>
<td>Relative location</td>
<td>Absolute location</td>
<td>Definition</td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
<td>------------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td>AERODY</td>
<td>AEXIT</td>
<td>20</td>
<td>1282</td>
<td>Ten-element array of stage exit areas, ( m^2 )</td>
</tr>
<tr>
<td></td>
<td>REVOLV</td>
<td>30</td>
<td>1292</td>
<td>Earth's rotation rate, ( \omega_e ), rad/sec</td>
</tr>
<tr>
<td></td>
<td>HEATR</td>
<td>31</td>
<td>1293</td>
<td>Vehicle heating rate, ( W/m^2/kg )</td>
</tr>
<tr>
<td></td>
<td>P</td>
<td>32</td>
<td>1294</td>
<td>The vector ( \mathbf{B} = V_e \times \mathbf{H} )</td>
</tr>
<tr>
<td></td>
<td>PMAGN</td>
<td>35</td>
<td>1297</td>
<td>Magnitude of the vector ( \mathbf{B} )</td>
</tr>
<tr>
<td></td>
<td>RATMOS</td>
<td>36</td>
<td>1298</td>
<td>Radius of the outer limit of the sensible atmosphere, ( m )</td>
</tr>
<tr>
<td></td>
<td>TDMAG</td>
<td>37</td>
<td>1299</td>
<td>Magnitude of vehicle drag acceleration, (</td>
</tr>
<tr>
<td></td>
<td>TLIFT</td>
<td>38</td>
<td>1300</td>
<td>Magnitude of vehicle lift acceleration, (</td>
</tr>
<tr>
<td></td>
<td>VATM</td>
<td>39</td>
<td>1301</td>
<td>Components of vehicle velocity relative to planet, ( V_x ), m/sec</td>
</tr>
<tr>
<td></td>
<td>VQ</td>
<td>42</td>
<td>1304</td>
<td>Vehicle relative speed, (</td>
</tr>
<tr>
<td></td>
<td>VQSQRD</td>
<td>43</td>
<td>1305</td>
<td>Square of ( V_e ), ( m^2/sec^2 )</td>
</tr>
<tr>
<td></td>
<td>VMACH</td>
<td>44</td>
<td>1306</td>
<td>Vehicle Mach number, ( M )</td>
</tr>
<tr>
<td></td>
<td>ICDF</td>
<td>45</td>
<td>1307</td>
<td>Index that points at current position in CDIC array</td>
</tr>
<tr>
<td></td>
<td>CDIC</td>
<td>46</td>
<td>1308</td>
<td>Array of parasite drag coefficient data, ( C_{DD} ) vs. ( M )</td>
</tr>
<tr>
<td></td>
<td>ICDI</td>
<td>45</td>
<td>1307</td>
<td>Index that points at current position in CDIC array</td>
</tr>
<tr>
<td></td>
<td>CDIC</td>
<td>46</td>
<td>1308</td>
<td>Array of induced drag coefficient data, ( C_{DI} ) vs. ( M )</td>
</tr>
<tr>
<td></td>
<td>ICDF</td>
<td>45</td>
<td>1307</td>
<td>Index that points at current position in CDIC array</td>
</tr>
<tr>
<td></td>
<td>CDIC</td>
<td>46</td>
<td>1308</td>
<td>Array of induced drag coefficient data, ( C_{DI} ) vs. ( M )</td>
</tr>
<tr>
<td></td>
<td>ICL</td>
<td>145</td>
<td>1407</td>
<td>Index that points at current position in CLC array</td>
</tr>
<tr>
<td></td>
<td>CLC</td>
<td>146</td>
<td>1408</td>
<td>Array of lift coefficient data, ( C_l ) vs. ( M )</td>
</tr>
<tr>
<td></td>
<td>IALF</td>
<td>195</td>
<td>1457</td>
<td>Index that points at current position in ALFCOE array</td>
</tr>
<tr>
<td></td>
<td>ALFCOE</td>
<td>196</td>
<td>1458</td>
<td>Array of angle of attack (thrust angle ( \alpha )) data, ( \alpha ) vs. ( t ), deg and sec</td>
</tr>
<tr>
<td>SAVE</td>
<td>TIME</td>
<td>1</td>
<td>1507</td>
<td>Time (double precision), ( t ), sec</td>
</tr>
<tr>
<td></td>
<td>STEPSGO</td>
<td>3</td>
<td>1509</td>
<td>Number of successful integration steps</td>
</tr>
<tr>
<td></td>
<td>STEPNQ</td>
<td>4</td>
<td>1510</td>
<td>Number of unsuccessful integration steps</td>
</tr>
<tr>
<td></td>
<td>REV</td>
<td>5</td>
<td>1511</td>
<td>Number of complete revolutions around x-axis</td>
</tr>
<tr>
<td></td>
<td>DEL</td>
<td>6</td>
<td>1512</td>
<td>Time increment to next output point, sec</td>
</tr>
<tr>
<td></td>
<td>IMODE</td>
<td>7</td>
<td>1513</td>
<td>Current indicator of the integration mode (see MODE1)</td>
</tr>
<tr>
<td></td>
<td>ASYMP</td>
<td>8</td>
<td>1514</td>
<td>Control set to TRUE when path lies too close to an asymptote to use orbit element integration</td>
</tr>
<tr>
<td></td>
<td>LOOKX</td>
<td>9</td>
<td>1515</td>
<td>Five-element array defining COMMON locations of trajectory interrupt variables</td>
</tr>
<tr>
<td></td>
<td>LOCKX</td>
<td>14</td>
<td>1520</td>
<td>Five-element array of counters for each trajectory interrupt, corresponds to LOCKX</td>
</tr>
<tr>
<td></td>
<td>SAVE1</td>
<td>19</td>
<td>1525</td>
<td>Seventeen-element array of initial-value variables saved during level 2 optimization</td>
</tr>
<tr>
<td></td>
<td>SAVE2</td>
<td>38</td>
<td>1542</td>
<td>Seventeen-element array of initial-value variables saved during level 1 iterations</td>
</tr>
<tr>
<td></td>
<td>HDS1</td>
<td>53</td>
<td>1559</td>
<td>One-hundred-fifty-array element of initial integration values saved during level 2 optimization (double precision)</td>
</tr>
<tr>
<td></td>
<td>HDS2</td>
<td>353</td>
<td>1559</td>
<td>One-hundred-fifty-array element of initial integration values saved during level 1 iterations (double precision)</td>
</tr>
<tr>
<td>HD</td>
<td>RMASS</td>
<td>1</td>
<td>2159</td>
<td>Vehicle mass (double precision), ( m ), kg</td>
</tr>
<tr>
<td></td>
<td>V</td>
<td>3</td>
<td>2161</td>
<td>Vehicle velocity (double precision), ( V ), m/sec</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>9</td>
<td>2167</td>
<td>Vehicle position (double precision), ( R ), m</td>
</tr>
<tr>
<td></td>
<td>L</td>
<td>15</td>
<td>2173</td>
<td>Adjoint variables (double precision), ( L ), ( L_r ), ( L_m )</td>
</tr>
<tr>
<td></td>
<td>PV</td>
<td>51</td>
<td>2189</td>
<td>Velocity partial derivatives (double precision), ( \delta V )</td>
</tr>
<tr>
<td></td>
<td>PR</td>
<td>85</td>
<td>2343</td>
<td>Position partial derivatives (double precision), ( \delta R )</td>
</tr>
<tr>
<td></td>
<td>PL</td>
<td>139</td>
<td>2397</td>
<td>Adjoint partial derivatives (double precision), ( \delta A )</td>
</tr>
<tr>
<td></td>
<td>PJ</td>
<td>193</td>
<td>2351</td>
<td>Adjoint partial derivatives (double precision), ( \delta F )</td>
</tr>
<tr>
<td></td>
<td>PM</td>
<td>247</td>
<td>2405</td>
<td>Adjoint partial derivatives (double precision), ( \delta M )</td>
</tr>
<tr>
<td></td>
<td>P7</td>
<td>265</td>
<td>2423</td>
<td>Adjoint partial derivatives (double precision), ( \delta c )</td>
</tr>
<tr>
<td></td>
<td>PB</td>
<td>293</td>
<td>2441</td>
<td>Adjoint partial derivatives (double precision), ( \delta q )</td>
</tr>
<tr>
<td></td>
<td>H</td>
<td>1</td>
<td>2458</td>
<td>Array of current values of the integration variables ( y_n )</td>
</tr>
<tr>
<td></td>
<td>HDOT</td>
<td>1</td>
<td>2609</td>
<td>Array of current values of the derivatives of the integration variables ( \dot{y}_n )</td>
</tr>
</tbody>
</table>
### TABLE IV. - SUMMARY OF NOPT OPTIONS\(^a,b\)

<table>
<thead>
<tr>
<th>Value of NOPT</th>
<th>Dependent variables (at arrival)</th>
<th>Independent variables (at departure)</th>
<th>Dimensionality of coordinate system</th>
<th>Typical usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Input</td>
<td>Input</td>
<td>2 or 3</td>
<td>Nonvariational problems</td>
</tr>
<tr>
<td>1</td>
<td>(v_x, v_y, v_z, x, y, z)</td>
<td>(\lambda_1, \ldots, \lambda_6)</td>
<td>2 or 3</td>
<td>Cartesian rendezvous</td>
</tr>
<tr>
<td>2</td>
<td>(r, v, \gamma, \theta)</td>
<td>(\lambda_1, \lambda_2, \lambda_4, \lambda_5)</td>
<td>2</td>
<td>Polar rendezvous</td>
</tr>
<tr>
<td>3</td>
<td>(r, v, \gamma)</td>
<td>(\lambda_1, \lambda_2, \lambda_4)</td>
<td></td>
<td>Optimum-angle rendezvous</td>
</tr>
<tr>
<td>4</td>
<td>(r, \theta, \Lambda/\lambda_m)</td>
<td>(\lambda_1, \lambda_2, \lambda_4, \lambda_5)</td>
<td></td>
<td>Flybys</td>
</tr>
<tr>
<td>5</td>
<td>(r, \Lambda/\lambda_m)</td>
<td>(\lambda_1, \lambda_2, \lambda_4)</td>
<td></td>
<td>Optimum-angle flybys</td>
</tr>
<tr>
<td>6</td>
<td>Input</td>
<td>Input</td>
<td>2 or 3</td>
<td>Any variational case</td>
</tr>
<tr>
<td>7</td>
<td>Input</td>
<td>Input</td>
<td>2 or 3</td>
<td>Same as option 5 but with optimum travel angle</td>
</tr>
</tbody>
</table>

\(^a\)By default, the program will generate the partial derivatives by numerical integration if \(1 \leq \text{NOPT} \leq 5\) and by finite differencing otherwise. If the user prefers the finite differ- ence scheme even if \(1 \leq \text{NOPT} \leq 5\), he should attach a minus sign to his NOPT entry.

\(^b\)For all propulsion cases, \(\lambda_1\) in this table is replaced by \(\dot{\tau}_0\) (or by \(a_0/g\) if initial thrust-weight ratio was input).
Figure 1. - Trajectory interrupt situations.

(a) Phases or staging.

(b) Trajectory penetrates a sphere of influence.

(c) Optimal engine on-off times.
Optimum fixed-thrust-angle selection from a set of angles $\alpha_i$.

User-selected interrupt on arbitrary variable.

Third-case data set: $s_3$

Second-case data set: $s_2$ (close to $s_1$)

First-case data set: all usual data entries including $s_1$, plus $\text{NSWEEP} = \text{COMMON}$ location of sweep parameter $s$

Data deck setup for manual sweep.
All usual data entries, plus

IAA = COMMON

location of sweep parameter $s$

SVALUE = $s_1$, $s_2$, ..., $s_n$; $n \leq 10$ (the values of $s$ where a full trajectory printout will occur)

MAXPTS = 2

Figure 3. - Data deck setup for automatic sweep.

---

Figure 4. - NBODY flow diagram.
Figure 5. Subprogram call sequence.