N63-11536
code 1

NASA TN D-1455

copy 2

NASA TN D-1455

TECHNICAL NOTE

D-1455

THE N-BODY CODE - A GENERAL FORTRAN CODE FOR SOLUTION OF PROBLEMS IN SPACE MECHANICS BY NUMERICAL METHODS

By William C. Strack, Wilbur F. Dobson, and Vearl N. Huff

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

WASHINGTON January 1963

CONTENTS

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

TECHNICAL NOTE D-1455

THE N-BODY CODE - A GENERAL FORTRAN CODE FOR SOLUTION OF PROBLEMS IN SPACE MECHANICS BY NUMERICAL METHODS

By William C. Strack, Wilbur F. Dobson, and Vearl N. Huff

SUMMARY

A general astronomical integration code designed for a large class of problems in space mechanics that may be solved by numerical integration is described. The equations of motion provide for the effects of up to eight gravitating celestial bodies, oblateness and aerodynamic forces from the celestial body at the problem origin, propulsion system thrust, and rotation of the body at the origin.

INTRODUCTION

The general problems of space mechanics (i.e., n-bodies plus nonconservative forces such as thrust) cannot be solved analytically. Therefore, numerical integration through the use of computing machinery is usually employed.

Several codes have been written for the numerical solution of problems in orbit mechanics; for example, the Themis Code of reference 1 is a double-precision code intended primarily for close satellites or interplanetary coasting flight. Reference 2 describes a space-trajectory program of considerable merit. A listing of several other trajectory codes may be found in reference 3.

The general purpose code described herein has several distinctive features not all of which are found in any one of the previously available codes. As described herein, this code is designed to operate on an IBM 704 computer that has an 8000 word (8 K) memory and at least 1 K of drum. The fact that the program is written in FORTRAN should make it applicable to installations having other types of equipment that accept the FORTRAN language. An edition of this program (differing primarily in that segmenting of the program is not required) is available for an IBM 7090 computer that has a 32-K core.

The program is compartmented into 25 subroutines to facilitate modifications for specific problems. The integration is carried out in either rectangular coordinates or orbit elements at the option of the user. A compact ephemeris that occupies about one-seventh of a reel of tape is utilized for positions and velocities of the planets (except Mercury) and the moon. An atmosphere is included so that aerodynamic forces may be considered.

STATEMENT OF PROBLEM

The problem to be solved may be stated as follows: Given certain initial conditions, compute, using three degrees of freedom, the path of an object, such as a space vehicle, subject to any or all of the following forces:

origin body gravitational field

other celestial body gravitational fields

propulsive thrust

aerodynamic forces

any other defined forces

or, in equation form, with respect to a noninertial Cartesian coordinate system,

$$\ddot{\vec{r}} = \nabla U + \left[k^2 \sum_{i=1}^{n} m_i \ \nabla\left(\left|\frac{1}{\vec{r} - \vec{r}_i}\right| - \frac{\vec{r} \cdot \vec{r}_i}{r_i^3}\right)\right] + \frac{\vec{F}}{m} + \frac{\vec{L}}{m} + \frac{\vec{D}}{m} + \frac{\vec{X}}{m} \tag{1}$$

where n equals the number of perturbating bodies and ∇ denotes the del operator. (All symbols are defined in appendix A.)

Origin Body Gravitational Field and Oblateness Perturbations

The first term, ∇U, in the equation of motion (eq. (1)) represents the gravitational forces due to the origin body. When the origin body is spherical and made up of homogeneous layers, this term becomes simply $-\mu\vec{r}/r^3$. In the case of the Earth, however, the effect of oblateness may be important, and additional terms must be added to account for the oblateness effects. The expression for the gravitational potential U of an oblate spheroid may be written, according to reference 4, as

$$U = \frac{k^2 m_r}{r}\left\{1 + J\left(\frac{R_r}{r}\right)^2\left(\frac{1}{3} - \frac{z^2}{r^2}\right) + \frac{K}{30}\left(\frac{R_r}{r}\right)^4\left[3 - 30\left(\frac{z}{r}\right)^2 + 35\left(\frac{z}{r}\right)^4\right]\right\} \tag{2}$$

where the x,y plane lies in the equatorial plane. The components of gravitational acceleration are as follows:

$$U_x = +\frac{\partial U}{\partial x} = \frac{k^2 m_r}{r^2}\left\{-1 + 5J\left(\frac{R_r}{r}\right)^2\left[\left(\frac{z}{r}\right)^2 - \frac{1}{5}\right] + \frac{K}{2}\left(\frac{R_r}{r}\right)^4\left[-1 + 14\left(\frac{z}{r}\right)^2 - 21\left(\frac{z}{r}\right)^4\right]\right\}\frac{x}{r}$$

$$U_y = +\frac{\partial U}{\partial y} = \frac{k^2 m_r}{r^2}\left\{-1 + 5J\left(\frac{R_r}{r}\right)^2\left[\left(\frac{z}{r}\right)^2 - \frac{1}{5}\right] + \frac{K}{2}\left(\frac{R_r}{r}\right)^4\left[-1 + 14\left(\frac{z}{r}\right)^2 - 21\left(\frac{z}{r}\right)^4\right]\right\}\frac{y}{r}$$

$$U_z = +\frac{\partial U}{\partial z} = \frac{k^2 m_r}{r^2}\left\{-1 + 5J\left(\frac{R_r}{r}\right)^2\left[\left(\frac{z}{r}\right)^2 - \frac{3}{5}\right] + \frac{K}{6}\left(\frac{R_r}{r}\right)^4\left[-15 + 70\left(\frac{z}{r}\right)^2 - 63\left(\frac{z}{r}\right)^4\right]\right\}\frac{z}{r} \qquad (3)$$

The first terms exist for a spherical planet composed of concentric layers of uniform density. The terms containing J and K are frequently called the second and fourth harmonic terms, while J and K are known as the harmonic coefficients.

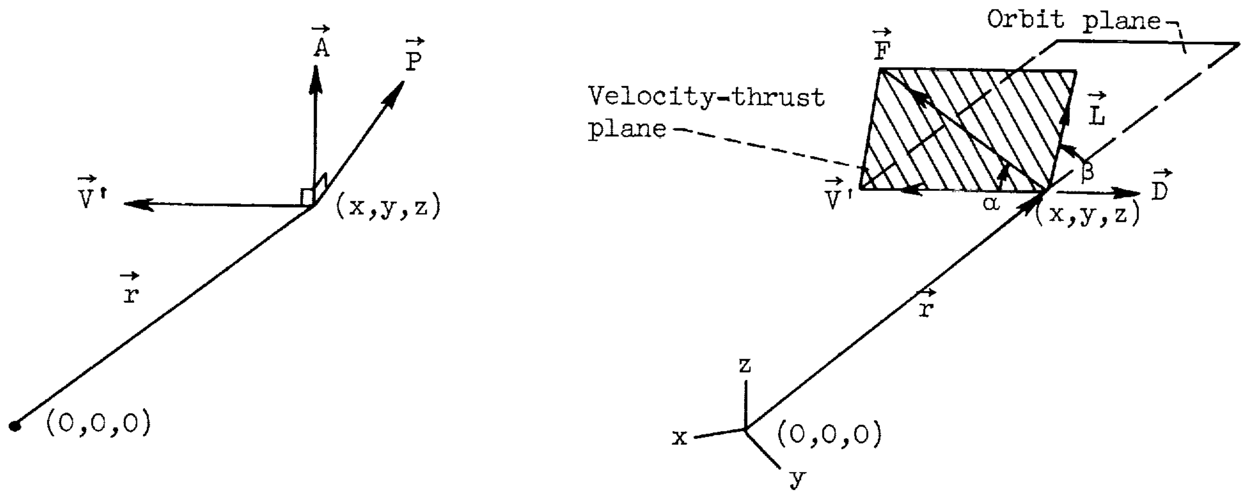
It is expected that oblateness perturbations need to be computed only for the origin body, since at large distances, such as that between celestial bodies, the gravitational field of an oblate body is essentially an inverse-square field. Consideration of oblate bodies other than the Earth requires only different values of J and K if that body's rotation axis is parallel to the z-axis. When the body has triaxial asymmetry or when the z-axis cannot conveniently be alined with the rotation axis of the origin body, the equations must be extended if oblateness is to be included.

Celestial Body Perturbations

The presence of more than one gravitating body in addition to the object results in the inclusion of the second term of equation (1). The evaluation of this term requires a knowledge of the positions of the bodies as a function of time. The degree of precision desired determines the method to be used to obtain the positions such as elements of ellipses or an ephemeris.

Propulsive Thrust

The propulsive acceleration is completely specified by a direction and a magnitude. The thrust direction may be referred to the velocity vector by two angles: α, the angle between the velocity and the thrust vectors, and β, the angle between the orbit plane and the velocity-thrust plane. The sense of each angle is indicated in sketch (a).



(a)

The velocity may be referenced with respect to one of several coordinate systems. If the computation refers to a takeoff of a rocket or winged vehicle, the coordinate system rotating with the Earth may be preferred. In such cases the relative velocity (i.e., the velocity of the object relative to the atmosphere) will serve to orient the thrust vector. Resolution of the thrust-vector components along the x,y,z axes is shown in appendix B.

The thrust magnitude of a rocket engine is

$$F = \dot{m} I g_c - P A_e \tag{4}$$

This relation is sufficient for many space powerplants and is used in the present program.

Aerodynamic Forces

The aerodynamic forces are usually divided into the two components, lift and drag. The drag force is directed opposite to the relative wind vector, and the lift vector is perpendicular to the relative wind vector. The angles α and β, defined in the previous section, serve as the angles of attack and roll, respectively. Yaw effects are not considered. Resolution of the lift and drag vectors into components along the x,y,z axes is given in appendix B.

The magnitudes of the lift and drag forces may be conveniently determined through use of a tabular group of coefficients in relatively simple equations. The lift and drag magnitudes may then be expressed (as is usual in aerodynamics) as

$$L = C_L(\alpha, N_M)qS \tag{5}$$

$$D = C_D(C_L, N_M)qS \tag{6}$$

where

$\alpha \ = \alpha(t)$

$C_L = f_1(N_M)\sin\alpha$

$C_D = C_{D,0} + C_{D,i} = C_{D,0}(N_M) + f_2(N_M)C_L^2$

$q \ = \frac{1}{2}\rho(V')^2$

$\rho \ = \rho(P,T) = \rho(h)$

If $\alpha(t)$, $C_{D,0}(N_M)$, $f_1(N_M)$, and $f_2(N_M)$ are assumed to be quadratic functions and β is assumed to be constant, the expressions for α, β, C_L, $C_{D,0}$ and $C_{D,i}$ become

$$\alpha = a_{11} + a_{12}t + a_{13}t^2$$

$$\beta = \beta_0$$

$$C_L = \left(a_{21} + a_{22}N_M + a_{23}N_M^2\right)\sin\alpha$$

$$C_{D,0} = a_{31} + a_{32}N_M + a_{33}N_M^2$$

$$C_{D,i} = \left(a_{41} + a_{42}N_M + a_{43}N_M^2\right)C_L^2$$

where the quadratic constants $a_{i,j}$ may have different values for different regions of the independent variables t and N_M.

It should be remembered that these choices are arbitrary and are not restrictive because other functions may easily be used by simply changing the equation where it appears in the program. In fact, any propulsion system and aerodynamic configuration can presumably be incorporated by writing proper thrust and aerodynamic subroutines.

The pressure, temperature, and density may be determined as a function of altitude in accordance with the ICAO standard atmosphere. The oblate Earth model is used to determine the altitude.

Other Forces

The $\vec{X}$ forces may be any forces such as electrostatic, magnetic, or solar radiation pressure that affect the trajectory. While these forces are not considered further herein, their inclusion would usually be feasible and would be similar to thrust, lift, and drag.

METHOD OF SOLUTION

The method of solution selected for the stated problem is presented in this section. A later section discusses the FORTRAN coding.

A description of several numerical integration techniques and their relative merits are contained in reference 5. A straightforward method for finding the position of the object as a function of time is to integrate the total acceleration of the object expressed in rectangular components. An example of this method is Cowell's method (ref. 5).

However, when the system under investigation consists of two nonoblate bodies (one of which is the object) with no forces other than gravitational attraction forces, an exact analytical solution for the motion of the body exists. Further, if the conditions of the actual problem are such as to approximate the two-body problem closely, another approach is to use the exact two-body solution as a basis and simply integrate the changes in the two-body parameters, since they should be slowly varying. This technique, sometimes called the "variation of parameters," will be referred to as "integration of orbit elements."

Since problems both remote and near to the exact two-body problem are encountered in orbit mechanics, and since either type of problem is solved more efficiently by using the technique most suitably applicable, it was considered desirable to use either of the previously mentioned integration techniques at will. Accordingly, two methods of integration are provided in the program, namely, rectangular coordinates and orbit elements.

Integration Variables

In order to use either of these integration techniques, it is necessary to select a suitable set of variables for integration. Because a differential equation may determine the mass of the object (i.e., spacecraft), mass has been selected as a variable to be integrated. Selection of the remaining parameters follows in the subsequent paragraphs.

Rectangular coordinates. - In the first technique, the total acceleration components $\ddot{x},\ddot{y}$, and $\ddot{z}$ are integrated to obtain x,y, and z where x,y, and z are the rectangular components of the origin-to-object radius $\vec{r}$. The positive x-axis points in the direction of the mean vernal equinox of 1950.0. The positive y-axis lies in the mean equator of 1950.0 and is perpendicular to and counterclockwise from the positive x-axis. The z-axis points north and completes the righthanded orthogonal set. The integration in rectangular coordinates involves numerical solution of three second-order linear differential equations; that is, a double integration is required for integrating the accelerations to obtain velocities and the velocities to obtain positions. The rectangular variables have advantages of complete generality and a minimum amount of computing per step.

Orbit elements. - In the variation-of-parameters technique, a set of six independent two-body parameters called orbit elements are integrated. These six parameters may be arbitrarily chosen from a host of possibilities. The set selected for this program is composed of the eccentricity e, the argument of pericenter ω, the equatorial longitude of ascending node Ω, the inclination of the orbit plane to the equatorial plane i, the mean anomaly M, and the semilatus rectum p. The transformation equations between the two sets of variables are given in appendix C.

The integration of orbit elements requires the numerical solution of six first-order linear differential equations. The rather involved transformation by which the three second-order linear differential equations in $\ddot{x},\ddot{y},\ddot{z}$ are reduced to six first-order equations in $\dot{e}$, $\dot{\omega}$, $\dot{\Omega}$, $\dot{i}$, $\dot{M}$, and $\dot{p}$ is contained in reference 6. Integration in orbit elements is frequently advantageous because the smaller orbit-element derivatives may permit larger integration intervals that result in fewer steps. In the special case of two-body motion, the derivatives are zero (except $\dot{M}$, which is a constant).

Mathematical difficulties may arise occasionally with most sets of orbit elements. In particular, for the selected set, these occur when e approaches unity (parabolic trajectory), which causes a loss of numerical accuracy in the frequently used quantity $(1 - e^2)$; and when an asymptote is approached too closely, which causes numerical difficulties in the iterative solution for eccentric anomaly from Kepler's equation. The selected solution to these difficulties is to shift temporarily to rectangular-coordinate integration whenever the difficulty arises.

Integration Method

It is clear that regardless of the choice of integration technique, the magnitudes of the derivatives of the variables to be integrated may vary considerably along the trajectory. With fixed step size (constant intervals in time), the integration scheme will take unnecessary steps in the regions where the changes in the derivatives are small and thus will waste computing time and increase roundoff error. When the derivatives are large and change rapidly, a fixed step size will result in large truncation error (error due to excessive step size). Thus, in the interest of computing accuracy and economy, use of variable step size along the trajectory becomes desirable.

One of the integration schemes that allows variable step-size control to be incorporated easily is the Runge-Kutta scheme. For this and other reasons, it was decided to use a fourth-order Runge-Kutta method with variable step-size control.

Truncation error and step size may be controlled by examining the relative errors between the fourth-order Runge-Kutta integration scheme and a lower-order integration procedure. The arbitrarily chosen low-order integration scheme was an unequal-interval Simpson rule method. Details of the fourth-order Runge-Kutta integration method and the step-size control are given in appendix D. Roundoff error may be reduced by accumulating the integration variables in double precision.

Origin Translation

As noted previously, machine computing time and roundoff error may be minimized by maximizing the integration interval. The largest intervals are possible in orbit elements when the celestial body at the problem origin is the one that has the greatest influence on the vehicle motion. For this and sometimes other reasons, it may become desirable to translate the problem origin occasionally as the vehicle moves along its path.

Such translations of the origin may be made when the object enters a body's "sphere of influence," that is, the sphere about a body within which the greatest influence upon the object is due to forces originating from that particular body. In this program, the orientation of the coordinate system is always alined with the system determined by the Earth's mean equator and equinox of 1950.0, as is standard in astronomy.

THE CODE AND ITS USAGE

The stated problem was programmed in FORTRAN routines that are separately designed to accomplish one task but when combined form a complete program. This feature facilitates modifications.

The program is labeled as a general-purpose code, but an efficient general-purpose code cannot be a reality. As a result, this code is not especially general, but an attempt has been made to retain efficiency and to provide for easy modification of the routines to recover generality as needed. For example, the program is an "open system," that is, it solves an initial value problem. There is no link provided to obtain specific end conditions. Provision of this link is left to the user for his specific needs. In particular, when certain end conditions of a trajectory are to be met by determining the correct initial conditions (two-point boundary value problem), the user may program an iteration scheme to compute initial conditions from end conditions of previous runs.

The code is designed to operate on an IBM 704 computer that has an 8-K core and drum and also a number of tape units. To operate the code on an 8-K computer, it is necessary to divide the program into two segments (core loads). The program of segment 1 arranges certain data in the core. The program of segment 2 overwrites the program but not the data of segment 1 when it is called for. Figure 1 is a simplified diagram that shows how the various major subprograms are arranged in the segments. The segmenting was done as efficiently as possible in

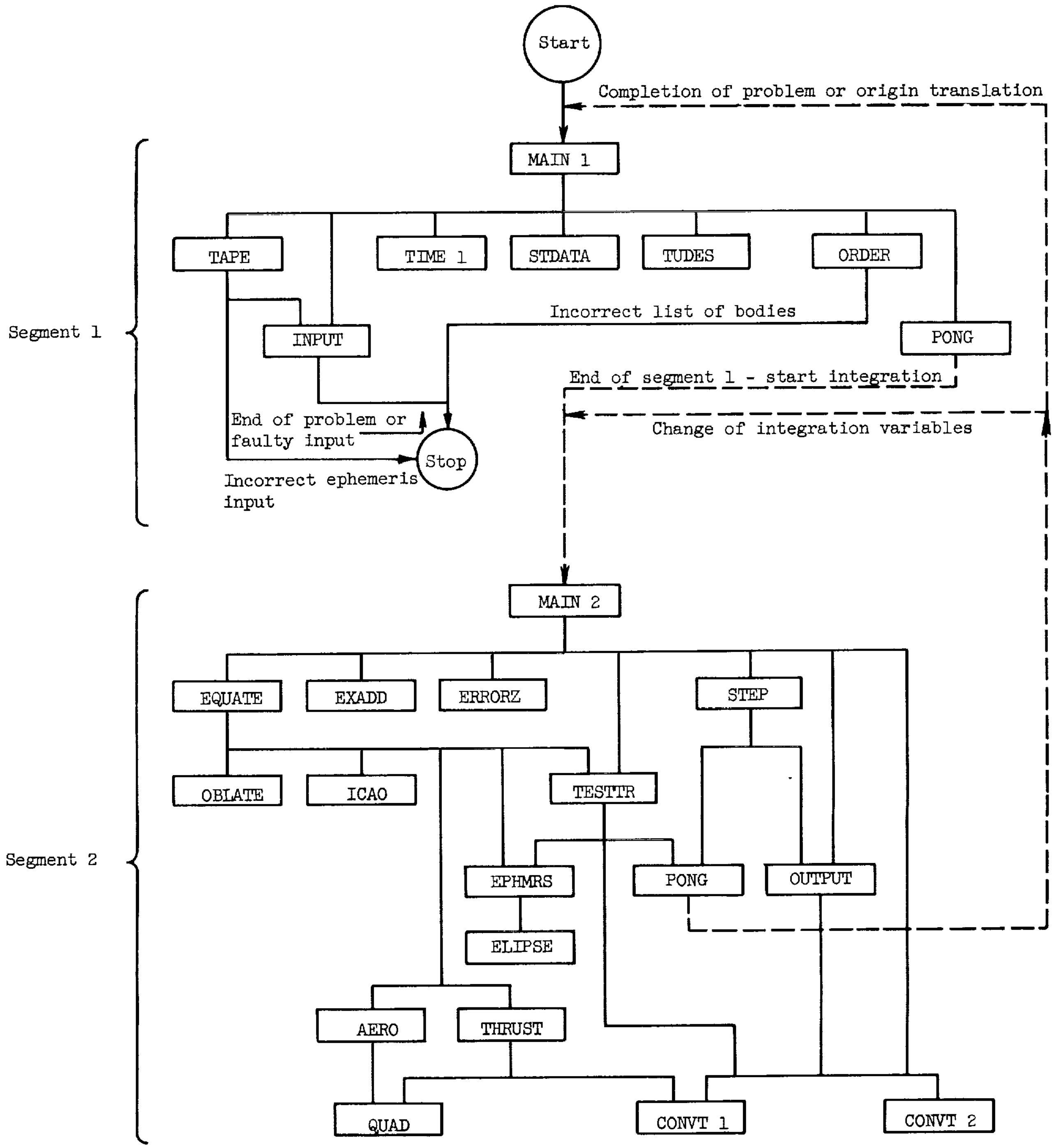


Figure 1. - Block diagram of program segments showing principal subprograms.

terms of execution time, but further gains can be realized by users of larger computers who may wish to modify the code to utilize the increased computer capacity.

In the following sections, discussing the program in terms of the FORTRAN variables and routines is sometimes desirable. A glossary of these variables is given in appendix E.

Ephemerides

To determine the position of each celestial body, there is offered a choice between ellipses and a precision ephemeris. Any appropriate ellipse data may be used, and an example of such data is given in table I.

The precision-ephemeris tape that is used in the program was so made that position and velocity were obtainable through the use of a fifth-order polynomial whose coefficients are stored on tape. The details concerning the making of the tape and its structure are given in appendix F. This master tape is a merged ephemeris containing all the planets (except Mercury), the moon, and the Earth-moon barycenter from October 25, 1960 to about 2000 (except for the moon, which has an ending date of 1970). The Earth ephemeris is called "sun" because it gives sun to Earth distances.

Direct use of the master merged ephemeris tape would, in general, be wasteful of computing time, since excess tape handling would occur in order to bypass data not required for the particular problem. To minimize tape handling during execution, a shorter merged ephemeris containing only that data needed for a specific problem is constructed at execution time. Several of these working ephemerides may be constructed before the integration of the problem. (Several problems may be loaded simultaneously with the same ephemeris, or each problem may require a distinct ephemeris, or several ephemerides may be desired for a single problem.)

Step Size and Output Control

Truncation error and step size are controlled by computing the relative errors between the Runge-Kutta integration and the lower-order integration procedure. If the greatest relative error between the methods is greater than a maximum limit (ERLIMT), the integration step will be repeated after a smaller step size is computed. In either case, a new step size is computed from the relative errors of the previous steps and is intended to result in an error that is close to a reference value (EREF). Further, the step size may then be reduced by the output controls. In any case, a step can be no larger than three times the size of the previous successful step. (See appendix D.)

Output is sometimes desired at specific points along the trajectory, while at other times this is unimportant. This option is provided for the user so that he may choose output to occur at equal intervals in step number or equal time intervals (which places a constraint on the step size). Also, he may choose to change from one mode to another along the trajectory. These choices of output spacing are effected through the use of the FORTRAN variables MODOUT, DELMAX, STEPS, and TMIN, which is explained under the MODOUT entry of table II, a table of program control parameters.

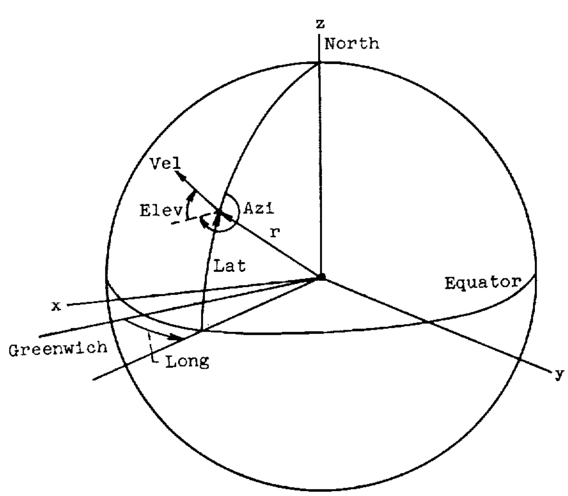
Computer Output

A basic output format was programmed to serve as a basis for modification and is illustrated in table III. It is intended that a user of the code modify the output to suit his purpose. In addition to examining the normal output, it is sometimes desirable to examine the error-control data, such as the relative errors in the integration variables, along the path. These data are printed as a single block after completion of the problem if the sign of the input error reference value EREF is negative. The sign of EREF is irrelevant in the error-control portion of the program since its absolute value is taken.

Computer Input

The user has a choice of three possible sets of input data that specify position and velocity: (1) the six orbital elements, (2) the three Cartesian components of both velocity and position, and (3) the latitude, longitude, azimuth, elevation, velocity, altitude, and time.

The third set mentioned is programmed for the Earth only where the latitude and longitude are the geocentric latitude and longitude measured from the equator and Greenwich, respectively. The azimuth angle is measured in a plane tangent to the sphere of radius r at the point on the sphere determined by the geocentric latitude and longitude, and relative to the local meridian, positive eastward from north. The elevation angle is then measured in a plane normal to the tangent plane, positive outward (sketch (b)). The tangent plane is taken horizontal



Long = longitude measured from Greenwich in earth's equatorial plane, positive east

Lat = latitude, measured positive north, geocentric

Azi = azimuth angle, measured east from north from local meridian

Elev = elevation angle, positive outward

Vel = vehicle's initial velocity

r = radius of vehicle from Earth's center

(b)

with the effects of oblateness and rotation considered if these effects are "on." If oblateness and rotation are "off," the horizontal is perpendicular to the radial direction. This input option ignores the correction between universal time and ephemeris time and between the instantaneous equator and equinox and those of 1950.0.

A list of input instructions is contained in appendix G along with an input check list.

The input routine described in reference 7 was used because of its simplicity; however, another input routine may be used if it is desired.

Sequence of Operations

Before the program begins to integrate a trajectory, it performs an assortment of operations that may be called "initialization." All these operations are expected to be done once or only a few times during the trajectory integration and, for this reason, are contained wholly in segment 1. Likewise, at the end of a problem, a return to the segment 1 causes several concluding operations to be performed. A condensed description of the operations carried out in segment 1 is contained in the flow diagram of figure 2. Other than the normal end of a problem (reaching a maximum number of integration steps or a particular time) there is only one way in which segment 1 may be called by segment 2, namely, a translation of the origin. When the translation occurs, segment 1 is needed to reorder the body list and perhaps to cause input or ephemeris change.

After completion of the initialization, which leaves numerical data stored in the common area, segment 1 is overwritten by segment 2, which may be termed the integration segment.

CODING

General

Appendix H contains the code listing of the program. Although most of the program is coded in basic FORTRAN II, on several occasions it was preferable to use the pseudo-SAP statements of FORTRAN II. Typically, the pseudo-SAP statement LXD (I),I is used whenever the index I was to be transferred from one subroutine to another (since FORTRAN II does not do this automatically). Wherever such a statement appears, the FORTRAN II statement I = I can be used instead to accomplish this initialization but with additional commands.

Some of the FORMAT statements are of the G-type. These statements will print output in I, E, or F format depending on the nature of the variable. Fixed-point variables will take the I format, while floating-point variables will assume the F format unless the magnitude of the variable falls outside the useful F range, in which case the E format is used. FORTRAN facilities that do not accept the G-type format statements may easily substitute E-type formats.

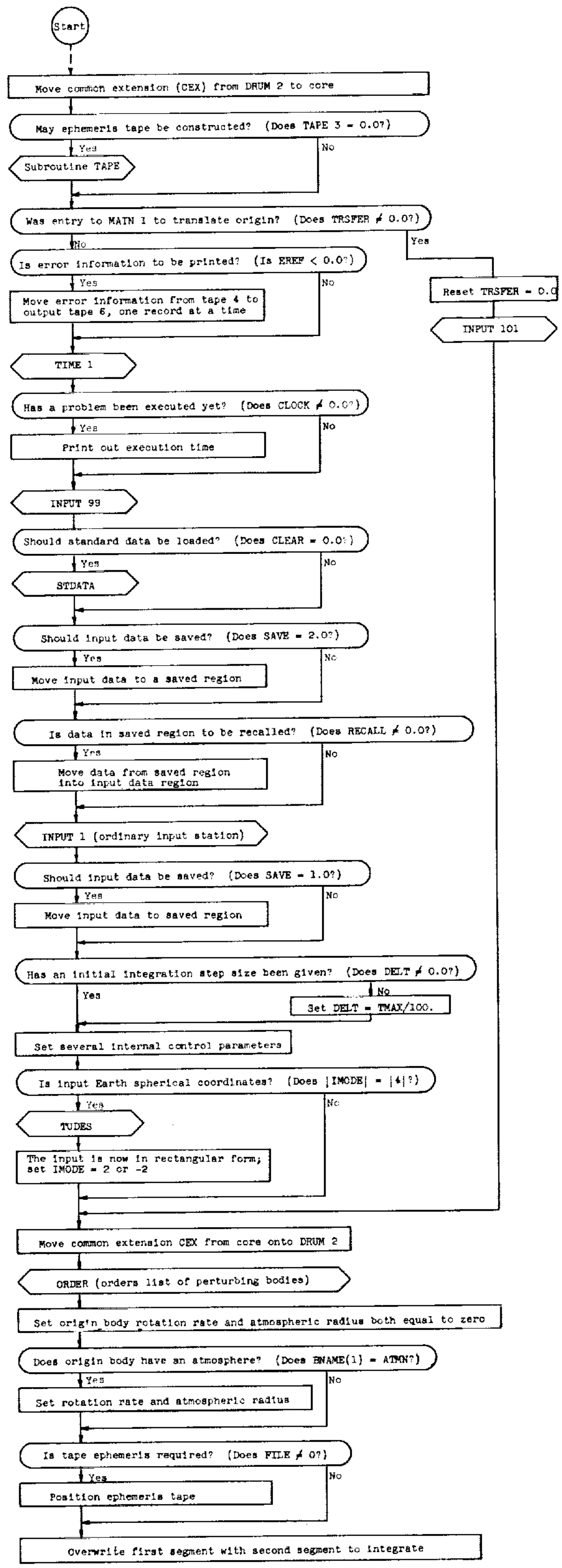


Figure 2. - Flow diagram of segment 1.

Table IV is a map of COMMON allocation (blanks are left for the user) and table II contains a description of the program control parameters. The elements of the integration variable array (XPRIM) are given in table V. The assumed values of the astronomical constants are given in table VI. These values are easy to change to any set desired. A selected set is given in reference 8.

Examples

Two examples of code usage are presented in the following sections. The first example is a problem of raising a low-altitude satellite into a 24-hour orbit by using low-tangential acceleration. The other example is a more complex problem involving a ground-launched lunar probe with a three-stage rocket. Both problems were selected to illustrate the usage of the program rather than to attempt a detailed analysis of the example problem.

Example I: Low-tangential thrust. - The trajectory to be determined is that used to raise a 3850-kilogram package from an initial 300-statute-mile circular equatorial orbit to a 24-hour orbit using a 60,000-watt nuclear electric system with a specific impulse of 2540 seconds and an overall efficiency of 40 percent. The required engine parameters may be calculated as follows:

thrust force:

$$F = \frac{2P_w \eta}{Ig_c} = \frac{2 \times 60{,}000 \times 0.4}{2540 \times 9.80665} = 1.927 \text{ newtons}$$

initial acceleration:

$$\frac{F}{m_0} = \frac{1.927}{3850} = 5.0051948 \times 10^{-4} \text{ m/sec}^2$$

propellant flow rate:

$$-\dot{m} = \frac{F}{Ig_c} = \frac{1.927}{2540 \times 9.80665} = 7.7361935 \times 10^{-5} \text{ kg/sec}$$

A detailed account is given in the following paragraphs for the solution of this problem by the prescribed program. Only those features of the program that have a direct bearing on this particular problem are discussed. Additional program features are discussed in the account of the second example problem. It may prove beneficial to refer to figure 2 during these two discussions.

It is assumed in the program that all memory data stores are cleared (set equal to zero) before operation begins. Control begins when the routine MAIN 1 is entered in segment 1. After several noninfluencing commands, the reading of a "clock" takes place at statement 10 and this value is stored. This value is later subtracted from the subsequent reading in order to yield the computing time. (All references to the "clock" may be deleted without ill effect.) Then a set of so-called "standard data" is initialized by executing subroutine STDATA. Before initializing, STDATA clears most of COMMON C.

The next step is calling for input at statement 21. The following list of parameters constitutes the input:

Parameter	FORTRAN name	Value
Initial mass, m_0, kg	RMASS	3850
Semilatus rectum, p, m	P	6.86×10^{6}
Specific impulse, I, sec	SIMP	2540
Flow rate, $-\dot{m}$, kg/sec	FLOW	7.7361935×10^{-5}
Time limit, sec	TMAX	[a]42605
Initial step size, sec	DELT	[a]1500
Step number limit, steps	STEPMX	[a]2000
Frequency of output, steps/output	STEPS	[a]200

[a]Assumed value.

Variables such as eccentricity and mean anomaly that are initially zero are not included in this list since all memory data stores are initially zero.

In accordance with the input routine of reference 7, the input cards may appear as

```
$DATA=1,$TABLE,41=RMASS,47=P,5=SIMP,33=       $$  IDENTIFICATION AND
FLOW,10=DELT,30=TMAX,20=STEPMX,21=STEPS/      $$  TABLE DEFINITION

RMASS=3850,SIMP=2540,FLOW=7.7361935E-5 $$  VEHICLE MASS, ISP, MASS FLOW
P=6.86E6,TMAX=42605,STEPMX=2000      $$  SEMILATUS-RECTUM, TIME LIMIT, STEP LIMIT
DELT=1500,STEPS=200                 $$  INITIAL STEP SIZE, OUTPUT EVERY 200TH STEP
$DATA=1,    $$  LAST CARD
```

where the entries between the $TABLE and slash (/) reference the subsequent entries to the second argument C of the calling statement. Thus, for example, RMASS is equivalent to C(41), the 41[st] location from the beginning of COMMON C.

Several commands follow the input none of which has an important effect on this particular problem with one exception: subroutine ORDER (part 11) computes the gravitational constants μ and $\sqrt{\mu}$. The initialization process is now completed.

Segment 2 overwrites segment 1, except COMMON C(1) to COMMON C(800), and control begins when the routine MAIN 2 is entered. Immediately, the tape that stores the two segments (tape 2 at Lewis) is rewound to position this tape at the beginning of segment 1.

The next sequence is that of integrating the first two steps. These two steps are of equal size and are integrated before an error check is made. If the first two steps are satisfactory (determined by statement 25), the remaining steps are integrated while the relative error is being checked at the end of each step. Parts 1 and 5 of MAIN 2 are concerned solely with this starting phase. Part 1 sets up the starting sequence and causes the initial conditions to appear on the output sheet. Parts 2 to 4 accomplish the Runge-Kutta integration for a single step.

The derivatives used in the integration are obtained from subroutine EQUATE. The first half of this subroutine finds the Cartesian coordinates and velocities through use of Kepler's equation. The thrust is computed in statement 34, and then subroutine THRUST is called to determine the components of the thrust acceleration in the Cartesian coordinate system. (After control is returned to subroutine EQUATE, the thrust acceleration is resolved into circumferential, radial, and normal components.) Finally the derivatives of the orbit elements are calculated, and a return is made to MAIN 2.

After the Runge-Kutta integration is performed, the error check is made in part 5B (part 6 after the starting sequence) by computing the difference between the Runge-Kutta integration and the low-order integration. Subroutine ERRORZ is called to determine the largest of the relative errors. If the largest of the relative errors is greater than the limit value, ERLIMT (set in STDATA), part 8, which computes a smaller step size for the same interval, is entered and control is returned to part 1. If the greatest relative error is smaller than the limit value, part 7, which advances the variables of integration, is entered and calls subroutine STEP to compute the next step size and print out the variables of the first step. Part 7 also counts the revolutions past the x-axis and adjusts the argument of pericenter and mean anomaly to within $\pm\pi$ to retain accuracy in the sine-cosine routines. If the step size exceeds 1/2 revolution, the revolution count may be short by an integral number. Control is finally transferred to part 1 to begin computation of the next step.

The problem is terminated when the time limit TMAX is reached. This check is done in subroutine STEP. Had the problem exceeded the step number limit STEPMX, it would have terminated at that point. In either case, control is returned to MAIN 1 in segment 1 to print out the computing time and begin the next problem. When no data for another problem are given, the execution is terminated (i.e., control is returned to the monitor by subroutine INPUT as a result of an end of file on tape 7). The output of the last step is:

```
STEP= 821. + 45.    ECCENTRICITY= 2.37578762E-04 OMEGA= 1.57668670
TIME= 42605.000     SEMILATUS R.= 6898571.50     TRU A= 1.57089765
JDAY= 2440000.4927  MEAN ANOMALY= 1.57042252      NODE= 0.
ALFA= 0.              PATH ANGLE= 1.36122511E-02  INCL= 0.

 V= 7599.09540     R= 6898571.62     REFER=EARTH  ORBIT  1
VX= 43.7259269     X=-6898447.56     RMASS= 3846.70401
VY=-7598.96967     Y=-41333.9687     REVS.= 7.50095356
VZ=-0.             Z=-0.              DELT= 263.055664
```

The time histories of several trajectory parameters for this example are shown as solid lines in figure 3. The oscillations of the eccentricity and mean

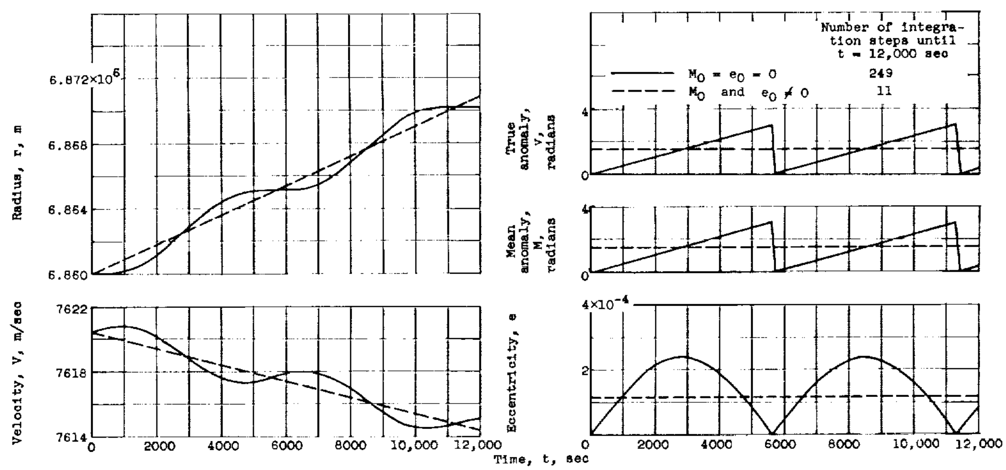


Figure 3. - Time histories of several trajectory parameters for example 1.

anomaly cause a rather small step size, as is noted in the figure. To indicate how exercising care in selecting the input can increase the computational efficiency, the same problem may again be run with the following initial values (according to ref. 9) of eccentricity and mean anomaly:

$$e_0 = \frac{2(F/m_0)p^2}{\mu}; \; M_0 = \frac{\pi}{2} - 3e_0 - \frac{e_0 V_0}{2Ig_c}$$

The input cards for this case make use of the algebraic properties of the input routine to compute the desired value of these parameters. The cards are:

```
$DATA=1,$TABLE,41=RMASS,47=P,5=SIMP,33=      $$  IDENTIFICATION AND
FLOW,10=DELT,30=TMAX,20=STEPMX,21=STEPS/     $$  TABLE DEFINITION

RMASS=3850,SIMP=2540,FLOW=7.7361935E-5 $$  VEHICLE MASS, ISP, MASS FLOW
P=6.86E6,TMAX=42605,STEPMX=2000      $$  SEMILATUS-RECTUM, TIME LIMIT, STEP LIMIT
DELT=1500,STEPS=200                 $$  INITIAL STEP SIZE, OUTPUT EVERY 200TH STEP
$TABLE,42=E,46=MA/ E=2*5.0051948E-4*P*P/3.983667E14   $$  ECCENTRICITY
MA=-7620.429/SIMP/9.80665-6*E+3.1415926/2,STEPS=5 $$ MEAN ANOMALY,OUTPUT CONTROL
$DATA=1,   $$  LAST CARD
```

The dashed lines in figure 3 show the time histories of the same trajectory parameters when initial values of e and M given immediately preceding are used. The increase in average step size is 20 to 1. To compare the accuracy of this approximation with the exact case ($e_0 = M_0 = 0$), the final time was chosen when the corresponding orbit positions were identical (when the true anomalies were equal). At t = 42,605 seconds, the orbit positions are nearly identical, and, at this time, the values of position and velocity may be compared as follows:

	Case A: $e_0 = M_0 = 0$	Case B: e_0 and $M_0 \neq 0$
Radius, m	6898571.62	6898571.56
Velocity, m/sec	7599.09540	7599.09546
Number of steps	821	39

For most purposes the two answers would be accepted as equivalent and case B would be preferred because of the smaller computer time required.

Example II: Lunar impact probe. This example of a lunar impact probe illustrates the use of the ephemeris tape and the control parameters needed to consider the effects of perturbing bodies, atmospheric forces, oblateness, rotating Earth, and thrust. No effort was made to optimize this trajectory but rather to use at least plausible values for illustrative purposes.

Suppose the probe is launched at Cape Canaveral on December 7, 1961 by a three-stage vehicle with stage parameters as shown in the following table:

Parameters	Stage			
	1	2	3	4
Initial mass, m_0, kg	150,000	52,500	23,625	945
Engine exit area, A_e, m^2	3.0	1.0	.5	(Coasting payload)
Vacuum specific impulse, I, sec	300	420	420	
Propellant loading, W_p/W_0	.65	.55	.96	
Propellant fraction, W_{pf}/W_p	.9	.9	.91765873	
Propellant flow rate, $-\dot{m}$, kg/sec	750	125	56.25	
Burning time, $t_b = W_{pf}/\dot{W}$, sec	117	207.9	370	
Aerodynamic reference area, S, m^2	7.5	4.0	2.0	2.0

Figure 4 shows the assumed variation of $C_{D,0}$, $C_{D,i}$, and C_L with Mach number as well as the angle-of-attack schedule.

The vehicle will be flown as follows: First, a short nondrag vertical flight, after which the desired velocity orientation will be set, and then a turn determined by gravity and the angle-of-attack schedule until first-stage burnout. The second and third stages follow the same turn pattern. The final stage consists of the payload. The staging will be accomplished by treating each stage as

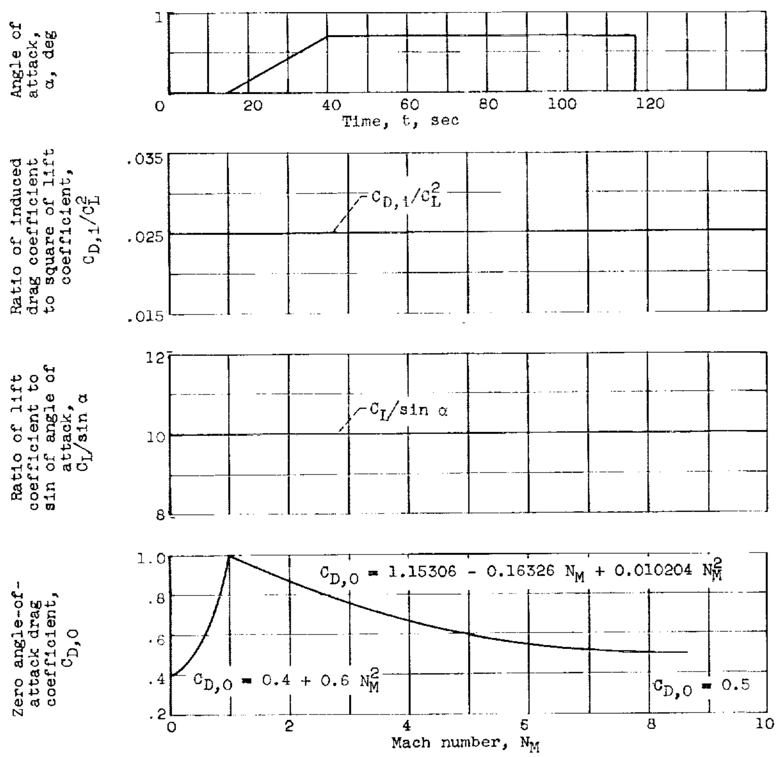


Figure 4. - Angle-of-attack schedule and variation of drag and lift coefficients with Mach number.

a single flight, with the burnout conditions of the previous stage used as initial conditions. The chosen integration mode will be rectangular for the powered flight but the mode of orbit elements will be used for the coast portion. Other bodies considered besides the Earth and the vehicle are the sun, the moon, and Jupiter. Jupiter is included to illustrate the use of ellipse ephemerides. The sun and moon will illustrate the use of the tape ephemeris.

The correct firing direction and launch time remain to be determined. This determination can be made by finding approximate values and then adjusting these values after one or more shots are fired. The adjustments could be made by an iteration scheme programmed internally to make a closed system. For this example, however, they were made by hand by firing several shots at various azimuth angles close to an estimate obtained by using reference 10 and an ephemeris. From a plot of the z-direction cosine of the vehicle-moon distance against vehicle-Earth distance, the azimuth angle that will intersect the moon orbit can be determined. The correct launch time is found by using the previously determined azimuth angle and various times of day to determine the time of day at which the vehicle intersects the correct position in the moon orbit (location of the moon). This type of analysis gives an azimuth angle of about 78.9° and a time of day of about 7.94^{h} E.T. (E.T. is ephemeris time which is approximately equal to Greenwich mean time.) For the present purpose, these values will be used.

The problem begins by constructing the merged ephemeris tape for the sun and moon. This is done by subroutine TAPE in conjunction with the input shown as follows:

```
$DATA=300,$TABLE,2=TAPE3,17=ELIST,29=TBEGIN,30=TEND/ $$ ID. AND TABLE DEFINITION
TAPE3=0                     $$  NECESSARY TO MAKE TAPE
TBEGIN=2437640.5            $$  JULIAN BEGINNING DATE
TEND=TBEGIN+5               $$  JULIAN ENDING DATE
ELIST=(A3)SUN,(A4)MOON      $$  LIST OF DESIRED EPHEMERIS BODIES
```

After the merged ephemeris tape is constructed, the clock is read, the standard data are initialized, and the first-stage input is loaded as shown:

```
$DATA=1,$TAB,104=LAT,105=LONG,106=AZI,107=ELEV,108=ALT,                  $$    STAGE 1
28.=IMODE,31=DTOFFJ,32=TOFFT,811=BODYCD,26=ATMN,29=RATM,459=             $$ ID. AND
ROTATE,41=RMASS,5=SIMP,33=FLOW,35=AREA,24=AEXIT,27=OBLATN,941=           $$ TABLE
ELIPS,601=COEFN,238.=ICC,37=EREF,17=ERLIMT,19=CLEAR,30=TMAX,20=          $$ DEFINITION
STEPMX,7=TKICK,10=DELT,103.=MODOUT,23=DELMAX,22=TMIN,21=STEPS/           $$
                                                                           $$
LAT=28.280,LONG=-80.571,ELEV=89.7            $$ LATITUDE,LONGITUDE,ELEVATION
AZI=78.9,ALT=10,IMODE=4                      $$ AZIMUTH,ALTITUDE,INTEGRATION MODE
DTOFFJ=2437640.5,TOFFT=7.94/24   $$  TAKE-OFF DATE AND FRACTION OF DAY
BODYCD=(A5)EARTH,(A4)MOON,(A6)JUPITE,(A3)SUN   $$ BODY NAMES, 1ST IS ORIGIN
ATMN=(A5)EARTH,RATM=1E11,ROTATE=7.29211585E-5 $$ ATMOSPHERE NAME,RADIUS,ROTATION
RMASS=150000,SIMP=300,FLOW=750  $$ VEHICLE MASS,ISP(VAC),MASS FLOW RATE
AREA=7.5,AEXIT=3.0,OBLATN=(A5)EARTH  $$ DRAG AREA,ENGINE EXIT AREA,OBLATE BODY
ELIPS=(ALF6)JUPITE,(ALF3)SUN,.9547861E-3,4.81E+10,5.1913995,   $$ ELLIPTIC DATA
.0486288,.1765935,.056971884,.40587194,2433964.,.6664,4333.7153$$ FOR JUPITER
COEFN=0,.4,0,.6,1,1.15306,-.16326,.010204,8,.5,,100,,10,,,  $$ AERO. COEFF. AND
100,,.025,,,100,,,,,15,-.6,.04,,40,.7,,,117,,,,1E6,ICC=24,14,19,1  $$ INDICES
EREF=1E-5,ERLIMT=5E-5,CLEAR=1 $$ REFERENCE ERROR,LIMIT ERROR,STDATA BY-PASS SWT
TMAX=117,STEPMX=250 $$ MAXIMUM ALLOWED PROBLEM TIME AND STEP NUMBER
TKICK=10,DELT=2 $$ TIME OF THE VERTICAL NON-DRAG STEP,1ST INTEGRATION STEP SIZE
MODOUT=2,DELMAX=60, $$ MODE OF OUTPUT,TIME INTERVALS OF OUTPUT
```

The value of IMODE is set equal to 4, which causes execution of subroutine TUDES. TUDES transforms the spherical Earth coordinates into rectangular coordinates, which are the variables of integration. In addition, TUDES computes the closed-form solution for the initial vertical nondrag step. From this point on, the trajectory is integrated with the initial orientation specified by the spherical coordinates. The small error introduced by this procedure is offset by avoiding the complications associated with integrating the takeoff. One such difficulty is the thrust-direction specification when the velocity is zero, especially if the origin body is rotating.

Subroutine ORDER reorders the list of bodies putting the sun before Jupiter (i.e., the sun's position relative to the vehicle must be found before Jupiter's relative position can be computed). The elliptic data for finding Jupiter's position are modified somewhat and relocated according to the computed body list. After calculating the gravitational constants, control is returned to MAIN 1.

The atmosphere belongs to the body at the origin (Earth) so that the rotation rate and atmospheric radius are set. The final duty of MAIN 1 is to position the merged ephemerides tape at the beginning of the correct ephemeris. In this case, only one merged ephemeris was constructed; nevertheless, it still must be identified and spaced to the beginning of the data.

Control then passes to MAIN 2, where integration takes place in the same manner described in example I. Additional subroutines called from EQUATE are EPHMRS, ELIPSE, ICAO, AERO, THRUST, and OBLATE. Subroutine EPHMRS is responsible for computing the perturbations that result from bodies other than the origin body. This computation is accomplished by determining the perturbating body position through use of the merged ephemeris tape or subroutine ELIPSE.

The AERO subroutine determines the aerodynamic accelerations through use of quadratic equations for the lift and drag coefficients and subroutine ICAO, which determines density, pressure, and temperature as functions of altitude. Oblateness accelerations are found in subroutine OBLATE. The thrust direction is determined by subroutine THRUST, while the thrust magnitude is computed in EQUATE as $\dot{m}g_cI - PA_e$.

The first vehicle stage integration is terminated by subroutine STEP when $t = 117$ seconds. Control is then transferred to MAIN 1, where the following input initiates the second vehicle stage integration:

```
$D=1,RMASS=52500,SIMP=420,FLOW=125,TMAX=TMAX+207.9,AREA=4,AEXIT=1  $$    STAGE 2
```

Integration of the second stage proceeds in a manner similar to the integration of the first stage and is terminated when $t = 324.9$ seconds. The third-stage data are similar to the second-stage data and are as follows:

```
$D=1,RMASS=23625,FLOW=56.25,DELMAX=100,TMAX=TMAX+370,AREA=2,AEXIT=.5  $$ STAGE 3
```

The fourth stage differs from the preceding stages since the thrust is turned off and integration proceeds in orbit elements rather than in Cartesian coordinates. Output occurs every 6 hours until $t = 1$ day; then it occurs at every tenth step. Also, the error-control data are printed (therefore, make EREF negative). The fourth-stage input is as follows:

```
$D=1,RMASS=945.0,DELT=3600,FLOW=0,TMAX=172800       $$                  STAGE 4
IMODE=-2,EREF=-() $$ INTEGRATE ORBIT ELEMENTS.  RECORD ERROR DATA.
MODOUT=3,DELMAX=DELT*6,STEPS=10,TMIN=86400 $$ OUTPUT EVERY 6 HOURS UNTIL TIME =
                                           $$ 86400,THEN EVERY 10TH STEP
$D=1, $$  LAST CARD
```

About 1/2 day later the vehicle is close enough to the moon that the coordinate system origin is translated to the moon. This translation is accompanied by a shift in integration mode to Cartesian coordinates, since the vehicle is approaching the moon far out on a hyperbolic leg. The last step output is reproduced as follows:

```
STEP= 184. + 17.      ECCENTRICITY= 10.6771772      OMEGA=-3.22087839
TIME= 172800.00       SEMILATUS R.= 3.16835663E 09 TRU A= 1.16945998
JDAY= 2437642.8306    MEAN ANOMALY=-18.8108633       NODE= 0.77242933
ALFA= 0.                PATH ANGLE= 62.2506247       INCL= 0.51408862
MOON    R= 2.5560079E 08 -0.391661  0.734785  0.553798
JUPITE R= 8.4571112E 11  0.581702 -0.741635 -0.334068

         V= 3938.07312       R= 6.12713230E 08 REFER=EARTH  RECTAN 3
        VX= 2403.45856       X= 1.27258404E 08 RMASS= 944.999992
        VY=-2445.76614       Y=-5.36514068E 08 REVS.= 0.78706574
        VZ=-1936.50073       Z=-2.67161870E 08  DELT= 5887.73633
SUN      R= 1.4668335E 11 -0.229169 -0.893118 0.387068
```

At this time the vehicle is again primarily under the Earth's influence after missing the point mass moon by 1.2×10^6 meters.

Lewis Research Center
National Aeronautics and Space Administration
Cleveland, Ohio, September 6, 1962

APPENDIX A

SYMBOLS

$\vec{A}$ relative angular momentum per unit mass, $\vec{r} \times \vec{V}'$ (appendix B)

A_e engine exit area, m^2

$a_{i,j}$ coefficients for quadratic functions

C_D total drag coefficient

$C_{D,0}$ zero angle-of-attack drag coefficient

$C_{D,i}$ induced drag coefficient

C_L lift coefficient

D drag force, newtons

E eccentric anomaly, radians

e eccentricity

F thrust force, newtons

f_1,f_2 functions of Mach number

g_c gravitational conversion factor, 9.80665 m/sec^2 (sometimes referred to as standard Earth gravity)

h altitude above Earth's surface, m

I vacuum specific impulse, sec

i orbit inclination to mean equator of 1950.0, radians

J second harmonic coefficient in oblateness equations

K fourth harmonic coefficient in oblateness equations

k^2 universal gravitational constant, $1.32452139 \times 10^{20}$, $m^3/(sec^2)$(sun mass units)

L lift force, newtons

M mean anomaly, radians

m object mass, kg

m_i	mass of i^{th} perturbating body, sun mass units
m_r	mass of reference body plus m, sun mass units
N_M	Mach number
P	atmospheric pressure, newtons/m^2
$\vec{P}$	$\vec{V}' \times \vec{A}$ (appendix B)
P_w	power, w
p	semilatus rectum, m
q	dynamic pressure, $\frac{1}{2}\rho(V')^2$, newtons/m^2
R_r	radius of reference body, m
r	radius from origin to object, m
r_i	radius from origin to i^{th} perturbating body, m
S	aerodynamic reference area, m^2
T	temperature, ^{o}K
t	time, sec
U	gravitational potential
U_x,U_y,U_z	x,y,z accelerations due to gravity, m/sec^2
u	$\omega + v$
V	absolute velocity, m/sec
V'	relative velocity, m/sec
v	true anomaly, radians
W	object weight, newtons
W_p	propellant loading, fraction of mass that departs during a stage
W_{pf}	propellant fraction, fraction of W_p used for propellant
X	forces acting on object other than gravity, thrust, lift, drag, and perturbations due to perturbating bodies
x,y,z	components of r, m

α angle between thrust and velocity vectors (sketch (a)), deg

β angle of rotation of thrust out of orbit plane (sketch(a)), deg

η power efficiency factor

μ $k^2 m_r$

ρ atmospheric density, kg/m^3

ω argument of pericenter, radians

$\vec{\omega}$ origin body rotation rate, radians/sec

Ω equatorial longitude of ascending node, radians

Subscripts:

0 initial value

1,2,3,4 values at consecutive points along trajectory

APPENDIX B

VECTOR RESOLUTION

Relative Velocity

The relative velocity is defined as the velocity of the object with respect to the origin body. If the origin body is assumed to rotate about the z-axis, this velocity is given by

$$\vec{V}' = \vec{V} - \vec{\omega} \times \vec{r} \tag{B1}$$

In x,y,z component form,

$$V_x' = V_x + \omega y \tag{B2a}$$

$$V_y' = V_y - \omega x \tag{B2b}$$

$$V_z' = V_z \tag{B2c}$$

In the following sections, the atmosphere of the origin body is assumed to rotate as a solid body at the rate $\vec{\omega}$.

Thrust Resolution Along x,y,z Axes

The thrust direction is specified with respect to the relative velocity vector $\vec{V}'$ by the angles α and β, as shown in sketch (a). For resolution of thrust vector into x,y,z components, it is convenient to define vectors $\vec{A}$ and $\vec{P}$ normal to and within the $\vec{r}$, $\vec{V}'$ plane, respectively, such that $\vec{V}'$, $\vec{A}$, and $\vec{P}$ form an orthogonal set. Thus,

$$\vec{A} \equiv \vec{r} \times \vec{V}' = \text{Relative angular momentum per unit mass} \tag{B3}$$

$$\vec{P} \equiv \vec{V}' \times \vec{A} \tag{B4}$$

The thrust vector can then be resolved in the $\vec{V}'$, $\vec{A}$, $\vec{P}$ set as:

$$\vec{F} \cdot \vec{V}' = FV' \cos \alpha \tag{B5a}$$

$$\vec{F} \cdot \vec{A} = FA \sin \alpha \sin \beta \tag{B5b}$$

$$\vec{F} \cdot \vec{P} = FP \sin \alpha \cos \beta \tag{B5c}$$

Solving for $\vec{F}$ yields

$$\vec{F} = \frac{F}{P^2} (V' \cos \alpha \, \vec{A} \times \vec{P} + A \sin \alpha \sin \beta \, \vec{P} \times \vec{V}' + \vec{P} \sin \alpha \cos \beta \, \vec{P}) \tag{B6}$$

or, in x,y,z component form,

$$F_x = \frac{F}{P^2}\Big[V' \cos \alpha(A_y P_z - A_z P_y) + A \sin \alpha \sin \beta(P_y V_z' - P_z V_y') + P \sin \alpha \cos \beta \, P_x\Big] \tag{B7a}$$

$$F_y = \frac{F}{P^2}\Big[V' \cos \alpha(A_z P_x - A_x P_z) + A \sin \alpha \sin \beta(P_z V_x' - P_x V_z') + P \sin \alpha \cos \beta \, P_y\Big] \tag{B7b}$$

$$F_z = \frac{F}{P^2}\Big[V' \cos \alpha(A_x P_y - A_y P_x) + A \sin \alpha \sin \beta(P_x V_y' - P_y V_x') + P \sin \alpha \cos \beta \, P_z\Big] \tag{B7c}$$

Aerodynamic Lift and Drag Resolution Along x,y,z Axes

The drag vector is alined with the relative velocity vector $\vec{V}'$ and is therefore given in x,y,z components as

$$\vec{D} = -D \frac{V_x'}{V'} - D \frac{V_y'}{V'} - D \frac{V_z'}{V'} \tag{B8}$$

The lift vector $\vec{L}$ may be resolved into components along the previously defined orthogonal set $\vec{V}'$, $\vec{A}$, and $\vec{P}$ by the following relations:

$$\vec{L} \cdot \vec{V}' = 0 \tag{B9a}$$

$$\vec{L} \cdot \vec{A} = LA \sin \beta \tag{B9b}$$

$$\vec{L} \cdot \vec{P} = LP \cos \beta \tag{B9c}$$

Solving for $\vec{L}$ yields

$$\vec{L} = \frac{L}{P^2} (A \sin \beta \, \vec{P} \times \vec{V}' + P \cos \beta \, \vec{P}) \tag{B10}$$

or, in x,y,z component form,

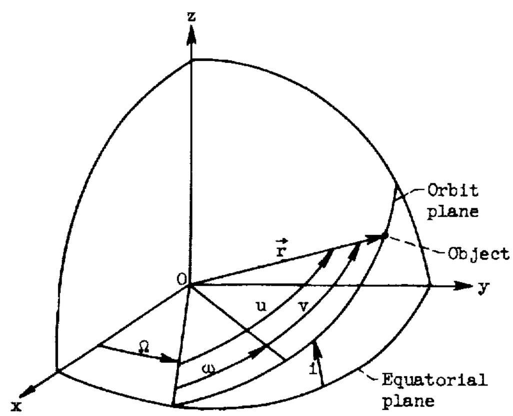
$$L_x = \frac{L}{P^2}\Big[A \sin \beta(P_y V_z' - P_z V_y') + P \cos \beta \, P_x\Big] \tag{B11a}$$

$$L_y = \frac{L}{P^2}\Big[A \sin \beta(P_z V_x' - P_x V_z') + P \cos \beta \, P_y\Big] \tag{B11b}$$

$$L_z = \frac{L}{P^2}\Big[A \sin \beta(P_x V_y' - P_y V_x') + P \cos \beta \, P_z\Big] \tag{B11c}$$

APPENDIX C

TRANSFORMATION EQUATIONS BETWEEN RECTANGULAR COORDINATES AND ORBIT ELEMENTS



(c)

From spherical trigonometry used in reference to the celestial sphere shown in sketch (c) the following relations may be derived for the position coordinates:

$$x = r(\cos \Omega \cos u - \sin \Omega \sin u \cos i) \tag{C1a}$$

$$y = r(\sin \Omega \cos u + \cos \Omega \sin u \cos i) \tag{C1b}$$

$$z = r(\sin u \sin i) \tag{C1c}$$

where

$$r = \frac{p}{1 + e \cos v} \tag{C2a}$$

$$u = \omega + v \tag{C2b}$$

and v is found from the relations

$$\cos v = \frac{\cos E - e}{1 - e \cos E} \tag{C2c}$$

and

$$M = E - e \sin E \tag{C2d}$$

The velocity components may be obtained by differentiating the position equations using the two-body relations $\dot{u} = \dot{v} = \frac{\sqrt{\mu p}}{r^2}$ and $\dot{r} = \sqrt{\frac{\mu}{p}}\, e \sin v$:

$$\dot{x} = -\sqrt{\frac{\mu}{p}}\,(N \cos i \sin \Omega + Q \cos \Omega) \tag{C3a}$$

$$\dot{y} = \sqrt{\frac{\mu}{p}}\,(N \cos i \cos \Omega - Q \sin \Omega) \tag{C3b}$$

$$\dot{z} = \sqrt{\frac{\mu}{p}}\,(N \sin i) \tag{C3c}$$

where

$$N = e \cos \omega + \cos u \tag{C4a}$$

$$Q = e \sin \omega + \sin u \tag{C4b}$$

APPENDIX D

RUNGE-KUTTA AND LOW-ORDER INTEGRATION SCHEMES WITH ERROR CONTROL

The Runge-Kutta formula used is of fourth-order accuracy in step size h. It is of the form

$$X\Big]_1^2 \equiv X_2 - X_1 = \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{D1}$$

where

$$X = \text{a dependent variable}$$

$$X\Big]_1^2 = \text{increment in the dependent variable}$$

$$h_2 = \text{increment in the independent variable } t$$

$$k_1 = h_2\dot{X}_2(t_1, X_1)$$

$$k_2 = h_2\dot{X}_2\left(t_1 + \frac{h_2}{2}, X_1 + \frac{k_1}{2}\right)$$

$$k_3 = h_2\dot{X}_2\left(t_1 + \frac{h_2}{2}, X_1 + \frac{k_2}{2}\right)$$

$$k_4 = h_2\dot{X}_2\left(t_1 + h_2, X_1 + k_3\right)$$

A lower-order formula may be found by utilizing the three derivatives at $t = t_0$, t_1, and t_2. If $h_1 = t_1 - t_0$ and $h_2 = t_2 - t_1$, the following Lagrangian interpolation formula gives the derivative at any time $t_0 \leq t \leq t_2$:

$$\dot{X} \equiv \dot{X}_0 \frac{(t - t_1)(t - t_2)}{h_1(h_1 + h_2)} - \dot{X}_1 \frac{(t - t_0)(t - t_2)}{h_1h_2} + \dot{X}_2 \frac{(t - t_0)(t - t_1)}{h_2(h_1 + h_2)} \tag{D2}$$

Integration of this equation from t_1 to t_2 yields

$$X'\Big]_1^2 = \frac{1}{6}\left[\left(\frac{h_2}{h_1}\right)^2\left(\frac{-h_2}{1 + \frac{h_2}{h_1}}\right)\dot{X}_0 + \frac{h_2}{h_1}\left(h_2 + 3h_1\right)\dot{X}_1 + \left(2h_2 + \frac{h_2}{1 + \frac{h_2}{h_1}}\right)\dot{X}_2\right] \tag{D3}$$

The difference in the increments over the interval h_2 between the Runge-Kutta scheme and the low-order scheme may be divided by a nominal value of the dependent variable $\overline{X}$ to obtain the relative error δ_2. Thus,

$$\delta_2 = \left| \frac{X'\Big]_1^2 - X\Big]_1^2}{\overline{X}} \right| \tag{D4}$$

The error is expected to vary as approximately the fifth power of h, which leads to

$$\delta = Ah^5 \tag{D5a}$$

(where A is a suitable coefficient) or in the logarithmic form

$$\log \delta = A' + 5 \log h \tag{D5b}$$

where

$$A' = \log A \tag{D6a}$$

Let it be assumed that A' will vary linearly with t, the variable of integration. Then A' at a time corresponding to t_3 can be found from A' at two previous points t_1 and t_2 as

$$A_3' = A_2' + \frac{A_2' - A_1'}{t_2 - t_1} (t_3 - t_2) \tag{D6b}$$

and if $h_3 = (t_3 - t_2)$ and $h_2 = (t_2 - t_1)$

$$A_3' = A_2' + (A_2' - A_1') \frac{h_3}{h_2} \tag{D6c}$$

and on this basis δ_3 would be predicted to be

$$\log \delta_3 = A_3' + 5 \log h_3 \tag{D7}$$

It is desired that δ_3 should approximate $\overline{\delta}$, the reference error; therefore,

$$\log h_3 = \frac{1}{5} (\log \overline{\delta} - A_3') \tag{D8}$$

Each dependent variable has an associated relative error and would lead to computation of a different step size for each variable; however, the maximum relative error of all variables may be selected for δ. Obviously, inaccurate predictions of step size can occur when the maximum relative error shifts from one variable to another or when any sudden change occurs. When a step size produces an excessively large error ($\delta > \delta_{limit}$), a reduced step size must be used. It may be obtained from the reference error $\bar{\delta}$ as

$$h_3 = \exp\left[\frac{1}{5}\left(\log \bar{\delta} - A_2'\right)\right] \tag{D9}$$

Starting the integration. - The Runge-Kutta scheme is simple to start, since integration from X_n to X_{n+1} requires no knowledge of X less than X_n. Since the error control coefficient A has no value at $t = 0$, however, a prediction of the second step size is difficult. To overcome this difficulty, two equal size first steps may be made before checking the error. The A for the first step may be arbitrarily set equal to the A for the second step so that h_3 may be predicted. The low-order integration scheme equation in this case becomes, with $h_2 = h_1$,

$$X'\Big]_1^2 = \frac{h_1}{3}\left(\dot{X}_0 + 4\dot{X}_1 + \dot{X}_2\right) \tag{D10}$$

Failures. - Should two consecutive predictions of the same step fail to produce an error δ less than δ_{limit}, a return to the starting procedure will be made with a third prediction on step size, which is no larger than one-half of the second estimate. The step-size control described here will operate stably with nearly constant error per step only for a well-behaved function. For most problems it will repeat a step occasionally to reduce a large error, and on sharp corners it will restart. This action is not regarded as objectionable. The objective is to attain a desired level of accuracy with a minimum total number of steps.

APPENDIX E

GLOSSARY OF VARIABLES

Variable	COMMON location	Definition
A	562	Total angular momentum per unit mass, m^2/sec
A (3)	559-561	x,y,z components of angular momentum per unit mass, m^2/sec
A1	236	Error control parameter defined by eq. (D6a) at t_1
A2	237	Error control parameter defined by eq. (D6a) at t_2
ACOEF 1	265	Interpolation polynomial coefficients for variable step size (coefficient of $\dot{X}_0$, $\dot{X}_1$, $\dot{X}_2$ in eq. (D3))
ACOEF 2	266	
ACOEF 3	267	
AK (3)	233-235	Runge-Kutta coefficients; set in STDATA
ALPHA	564	Angle between velocity and thrust vectors, positive when thrust vector is outward (sketch (a))
ALT	463 or 108	Vehicle altitude above an elliptic Earth, m
AMASS (30)	881-910	Permanent list of body masses (sun mass units) in order of PNAME list; set in STDATA; masses from ELIPS data begin at AMASS(21)
ANGLES (4)	104-107	Same as LAT, LONG, AZI, and ELEV, respectively
AREA	35	Effective area used to compute lift and drag forces in AERO, m^2
ASQRD	563	Square of total angular momentum, A^2, m^4/sec^2
ASYMPT	543	See table II
ATMN	26	See table II
AW (4)	261-264	Runge-Kutta coefficients; set in STDATA
AZI	106	Azimuth angle, measured east from north at local meridian, input in deg
BETA	565	Angle between velocity-thrust plane and orbit plane (sketch (a))

Variable	COMMON location	Definition
BEX (14)	801-813	List of error data
BMASS (8)	417-424	Body masses selected from AMASS list in sequence corresponding to BNAME list
BNAME (8)	402-409	Ordered list of BCD body names
BODY CD (8)	811-818	Original unordered list of BCD body names read from cards
BODY L (8)	801-808	Auxiliary ordered list of BCD body names
CD	797	Total drag coefficient per unit area, sec^2/m
CDI	795	Induced drag coefficient per unit area, sec^2/m
CEX (800)	801-1600	Common extension; common used in segment 1 but not needed in segment 2 and therefore saved on drum 2 during execution of segment 2
CF (126)	276-401	Coefficients from ephemerides tape used to determine positions of perturbing bodies
CINCL	495	cos i
CIRCUM	541	Circumferential component of total perturbative acceleration
CHAMP	246	Smallest critical radius within which object lies
CL	796	Lift coefficient per unit area, sec^2/m
CLEAR	19	See table II
CLOCK	3	Contains reading of clock (to compute time used for particular problem)
COEFN (190)	601-790	Storage array for coefficients used to compute ALPHA, CL, CDI, CD, or other parameters
COMPA (3)	537-539	Components of total perturbative acceleration in x,y,z coordinate system
CON (9)	576-581	Constants in the oblateness equations; set in STDATA
CONSTU	18	See table II

Variable	COMMON location	Definition
CONSU	36	See table II
COS ALF	575	cos α
COS BET	599	cos β
COSTRU	493	cos v
COSV	497	cos u
DE	162	$\dot{e}$
DEL	255	Used to control output in STEP
DELMAX	23	See table II
DELT	10	Step size, sec
DINCL	165	$\dot{i}$
DM	161	$\dot{m}$
DMA	166	$\dot{M}$
DNODE	164	$\dot{\Omega}$
DNSITY	460	Atmospheric density, kg/m^3
DOMEGA	163	$\dot{\omega}$
DRAG (3)	531-533	x,y,z components of the drag acceleration
DTOFF J	31	Julian date of takeoff
E	42	e
E2	260	Largest of relative errors between Runge-Kutta and Simpson rule integration methods defined by eq. (D4)
EFMRS (7)	410-416	List BCD body names whose positions are to be determined from ephemerides-tape data
ELEV	107	Elevation angle, measured outward, deg

Variable	COMMON location	Definition
ELIPS (120)	941-1060	Ellipse data for perturbing bodies, read from cards; for each body there are 15 pieces of data [NOTE: SUBROUTINE ORDER then converts 15 pieces of data into working set of 15]
EPAR	245	$\sqrt{e^2 - 1}$
EREF	37	See table II
ERLIMT	17	See table II
ERLOG	259	Natural logarithm of EREF
ETOL	25	See table II
EXMODE	244	Eccentricity (used when IMODE = 3)
EMONE	243	e - 1
FILE	249	See table II
FLOW	33	Rate of propellant flow, kg/sec
FORCE (3)	525-527	x,y,z components of acceleration due to thrust
GASFAC	458	Defined in AERO; set in STDATA
GEOH	465	Geopotential, m
GK2M	469	Gravitational constant, μ, m^3/sec^2
GKM	470	Square root of GK2M
H2	256	Value of DELT for previous step
IBODY (8)	425-432	Defined in SUBROUTINE ORDER
ICC (5)	238-242	See table II
IMODE	28	See table II
INCL	45	i, radians
IND (3)	791-793	Index set in STDATA

Variable	COMMON location	Definition
INDERR	491	Number of sets of error data; set in ERRORZ for use in MAIN 1
KSUB	254	Index of Runge-Kutta subintervals
LENGTH	257	See table II
LAT	104	Geocentric latitude, positive northward, deg
LONG	105	Longitude relative to Greenwich, positive eastward, deg
MA	46	M
MBODYS	441	Number of perturbing bodies
MODOUT	103	See table II
NBODYS	489	Total number of bodies, excluding vehicle
NDUMP (4)	268-271	See table II
NEFMRS (8)	433-440	Defined in SUBROUTINE ORDER
NODE	44	Ω, radians
NPONG (5)	11-15	See table II
NSKIP (4)	272-275	See table II
NSTART	247	Internal control in MAIN 2 and EQUATE
OBLAT (3)	534-536	x,y,z components of oblateness acceleration
OBLAT J	38	Oblateness coefficient of 2^{nd} harmonic
OBLAT K	39	Oblateness coefficient of 4^{th} harmonic
OBLAT N	27	See table II
OMEGA	43	ω, radians
OLDDEL	225	Value of DELT for previous good step
ORBELS (6)	227-232	Array of output variables, either rectangular or orbit elements

Variable	COMMON location	Definition
P	47	p, m
P (3)	571-573	Defined in eq. (B4)
PAR (3)	798-800	Defined by equations in SUBROUTINE THRUST
PMAGN	574	Defined in equation form by SUBROUTINE THRUST
PRESS	466	Atmospheric pressure, mb
PUSH	34	Thrust force, newtons
PNAA	---	ALF list of body names
PNAME (30)	821-850	Permanent list of body names made from PNAA list in SUBROUTINE ORDER; ELIPS names begin at PNAME(21)
PSI	462	Path angle, angle between path and local horizontal, deg
QVAL	794	Defined in SUBROUTINE AERO
QX (3)	522-524	x,y,z perturbative acceleration components due to perturbing bodies, m/sec^2
R	442	Origin to object radius, m
RADIAL	540	Radial component of total perturbative acceleration, positive outward, m/sec^2
RATIO	600	Ratio of adjacent step sizes
RATM	29	Radius of atmosphere, m
RATMOS	248	Set equal to RATM when ATMN equals reference body name (BNAME (1))
RB (3,8)	200-223	x,y,z components of distance from all bodies to object, m
RBCRIT (8)	450-457	List of sphere-of-influence radii of all bodies in BNAME list, m
RCRIT (30)	911-940	Permanent list of sphere-of-influence radii corresponding to PNAME list of body names, m; radii from ELIPS data begin at RCRIT(21)
RECALL	9	See table II

Variable	COMMON location	Definition
REFER (30)	851-880	List of reference bodies corresponding to PNAME list; reference bodies from ELIPS data begin at REFER(21)
REVOLV	250	Rotation rate (rad/sec) of reference body when ATMN = BNAME (1)
RESQRD	40	Square of Earth's equatorial radius, m^2; used in SUBROUTINE OBLATE; set in STDATA
REVS	490	Revolution counter, used only for output
RMASS	41	m, kg
ROTATE	459	Rotation rate of a reference body, radians/sec
RREL (8)	442-449	Distances between bodies and object in order of BNAME list, m
RSQRD	567	Radius squared of object to origin, m^2
SAVE	8	See table II
SIMP	5	Specific impulse, I, sec
SINALF	569	sin α
SINBET	568	sin β
SINCL	494	sin i
SINTRU	492	sin v
SINV	496	sin u
SPD	253	Seconds per day; set in STDATA
SQRDK	468	Gravitational constant k^2, $m^3/(sec^2)$(sun mass units); set in STDATA; value of 1.495×10^{11} m/AU (equivalent to solar parallax of 8.80008445 sec of arc) was used to convert units from $2.959122083 \times 10^{-4}$ $(AU)^3/(\text{mean solar day})^2$(sun mass units) to $1.32452139 \times 10^{20}$ $m^3/(sec^2)$(sun mass units)
STEPGO	101	See table II
STEPNO	102	See table II

Variable	COMMON location	Definition
STEPMX	20	See table II
STEPS	21	See table II
TAB (189)	1301-1489	Table array of input variables and their common storage assignment; used by SUBROUTINE INPUT; room for 94 variables
TABLT	252	Time measured relative to DTOFFT, days
TAPE 3	2	See table II
TDATA (126)	276-401	Same as CF
TDEL (7)	592-598	One-half of time spacing between two particular adjacent entries of like body name on ephemerides tape; read from tape for each body
TEST	1	See table II
TFILE	16	See table II
TIM (7)	585-591	Time for set of ephemeris data; read from ephemerides tape; one for each body
TIME	48	Time, t, independent variable, sec
TM	467	Temperature, $^{\circ}$K, times ratio of molecular to actual molecular weight
TMAX	30	See table II
TMIN	22	See table II
TOFFT	32	Fractional part of takeoff day (Julian), days
TRSFER	224	See table II
TRU	483	v, radians
TTEST	251	See table II
TTOL	226	Time tolerance within which problem time minus TMAX must lie to end problem
V	475	Velocity of object relative to origin V, m/sec

Variable	COMMON location	Definition
VATM (3)	477-479	x,y,z components of VQ
VEFM (3,8)	498-521	x,y,z components of object velocity relative to all various bodies, m/sec
VEL	109	Initial velocity at input
VQ	480	Velocity of object relative to atmosphere, m/sec
VQSQRD	481	$(VQ)^2$, m^2/sec^2
VMACH	471	Mach number of object, N_M
VSQRD	476	V^2, m^2/sec^2
VX	42	x-component of V; also in COMMON location C(472), m/sec
VY	43	y-component of V; also in COMMON location C(473), m/sec
VZ	44	z-component of V; also in COMMON location C(474), m/sec
X	45	x-component of R, m
X (15)	131-145	Working set of integration variables
XDOT (15)	161-175	Array of integration derivatives
XIFT (3)	528-530	x,y,z components of lift acceleration, m/sec^2
XINC (15)	146-160	Increments of integration variables per step
XP (3,8)	176-199	x,y,z components of perturbing body positions relative to origin
XPRIM (15,2)	41-70	Two 15-variable arrays; second is integrated and first contains values of integration variables for last good step; see table V
XPRIMB (15,2)	71-100	Least significant half of double precision integration variables corresponding to XPRIM
XWHOLE (15)	544-558	Temporary storage for integration variables
Y	46	y-component of R

Variable	COMMON location	Definition
Z	47	z-component of R
ZN	487	Mean angular motion of object, radians/sec
ZMA	46	M
ZORMAL	542	z-component of total perturbative acceleration, m/sec^2

APPENDIX F

LEWIS RESEARCH CENTER EPHEMERIS

General Description

The ephemeris data initially available on magnetic tape for use on the IBM 704 computer were from the Themis code prepared by the Livermore Laboratory, evidently from U. S. Naval Observatory data. Later, an ephemeris was obtained from the Jet Propulsion Laboratory assembled as a joint project of the Jet Propulsion Laboratory and the Space Technology Laboratory. These data are given relative to the mean vernal equinox and equator of 1950.0 and are tabulated with ephemeris time as the argument.

An ephemeris was desired for certain uses in connection with the IBM 704 computer that would be shorter than the original ephemeris tapes mentioned and would be as accurate as possible consistent with the length. A short investigation of the various possibilities led to adoption of fitted equations. In particular, fifth-order polynomials were simultaneously fitted to the position and velocities of a body at three points. This procedure provides continuity of position and velocity from one fit to the next, because the exterior points are common to adjacent fits. Polynomials were selected rather than another type of function, because they are easy to evaluate. Three separate polynomials are used for the x,y, and z coordinates, respectively.

Procedure Used to Fit Data

The process of computing the fitting equations is as follows:

(1) A group of 50 sets of the components of planetary position was read into the machine memory for a single planet together with differences as they existed on the original magnetic tape. The differences were verified by computation (in double precision because some data required it); and any errors were investigated, corrected, and verified. Published ephemeris data were adequate to correct all errors found.

(2) The components of velocity v_x, v_y, and v_z were computed and stored in the memory for each of the 50 positions by means of a numerical differentiation formula using ninth differences, namely,

$$\dot{X} = (T_1 - T_{-1}) \left[\frac{\Delta I_{-1} + \Delta I_{+1}}{2} - \frac{\Delta III_{-1} + \Delta III_{+1}}{12} + \frac{\Delta V_{-1} + \Delta V_{+1}}{60} - \frac{\Delta VII_{-1} + \Delta VII_{+1}}{280} + \frac{\Delta IX_{-1} + \Delta IX_{+1}}{1260} \right] \tag{F1}$$

(See ref. 11 pp. 42 and 99 for notation.) Double-precision arithmetic was used for differences, but velocities were tabulated with single precision.

(3) Coefficients C, D, E, and F in the fifth-order polynomial

$$X = X_0 + \dot{X}_0(T - T_0) + C(T - T_0)^2 + D(T - T_0)^3 + E(T - T_0)^4 + F(T - T_0)^5 \qquad \text{(F2)}$$

and its derivative

$$\dot{X} = \dot{X}_0 + 2C(T - T_0) + 3D(T - T_0)^2 + 4E(T - T_0)^3 + 5F(T - T_0)^4 \qquad \text{(F3)}$$

were found to fit a first point (which was far enough from the beginning point to have all differences computed) and two equally spaced points for each component of position and velocity. (The initial spacing is not important, as will be seen later.) Spacing is defined as the number of original data points fitted by one equation. Single-precision arithmetic was used.

(4) The coefficients C, D, E, and F in step (3) were then used in equations (F2) and (F3) to calculate components of all positions and velocities given in the original data and lying within the interval fitted. These values were checked with the original data. Radius R and velocity V were computed at the times tabulated in the original data. If any component of the position differed from the original data by more than $R \times 10^{-7}$ or if any velocity differed from the original by more than $V \times 10^{-6}$, the fit was considered unsatisfactory.

(5) If the fit were considered unsatisfactory, this fact was recorded; and the spacing was reduced by two data points. Steps 2 to 4 were then repeated. If the fit were considered satisfactory, this fact was recorded; and the spacing was increased by two spaces. Steps 2 to 4 were repeated. The largest satisfactory fit was identified when a certain spacing was satisfactory and the next larger fit was not satisfactory.

(6) The coefficients that corresponded to the largest satisfactory fit were recorded on tape in binary mode as follows:

Word number	Data	Mode	Definitions and/or units
1	Planet name	BCD	Six characters (first six)
2	Julian date	Floating point	Date of midpoint of fit, Julian date
3	Delta T		Number of days on each side of midpoint
4	F_x		[a]AU/day^5
5	E_x		[a]AU/day^4
6	D_x		[a]AU/day^3
7	C_x		[a]AU/day^2
8	$\dot{x}$		[a]AU/day
9	x		[a]AU
10	F_y		[a]AU/day^5
11	E_y		[a]AU/day^4
12	D_y		[a]AU/day^3
13	C_y		[a]AU/day^2
14	$\dot{y}$		[a]AU/day
15	y		[a]AU
16	F_z		[a]AU/day^5
17	E_z		[a]AU/day^4
18	D_z		[a]AU/day^3
19	C_z		[a]AU/day^2
20	$\dot{z}$		[a]AU/day
21	z		[a]AU

[a]Except for moon data, which are in Earth radii and days.

(7) As soon as a set of coefficients was selected for an interval, additional data were read from the source ephemeris tape and used to replace the points already fitted (except the last point). These data were processed as described in steps 1 and 2 so that the next 50 points were ready to be fitted. Steps 3 to 6 were then used to find the next set of coefficients, and steps 1 to 6 were repeated until all data for all planets and so forth, were fitted.

Data Treated

The preceding process was applied to all data available at the time. For the moon, the technique usually led to the use of every point in the fitted interval (i.e., only three points were fitted). Thus, a check of accuracy was not available. The error on the attempt to fit the next greater interval (five points) was not excessive, however, and it is judged that the accuracy obtained from these fits is about equal to that held on the other bodies.

Merged Ephemeris Tape

Once all the positions and velocities of all the bodies then available were fitted, the coefficients were merged in order of the starting date of each fit. The resulting tape was written in binary mode with 12 sets of fits per record.

The detail of this record is as follows:

	1st word:	FORTRAN compatible
	2nd word:	file number, fixed point in decrement
Set 1	3rd word:	planet name, code in BCD, first six characters
	4th word:	Julian date, floating point
	-	etc., according to list in paragraph 6
	-	21 words
	23rd word:	z
Set 2	24th word:	planet name, code in BCD, first six characters
	25th word:	Julian date, floating point
	-	
	-	
	44th word:	z

Successive sets follow one another with a total of 12 sets.

Set 12 (last set)	234th word:	planet name
	235th word:	Julian date, floating point
	254th word:	z
	255th word:	zero
	256th word:	zero
	End-of-record gap	

One record contains 256 words, the first is for FORTRAN compatibility, the second is a file number used for identification in the system. It is a fixed point 2. The third is the beginning of the first set of data, and 12 sets follow, each with 21 words. The last word is the 256th word (counting the FORTRAN compatible word) followed by an end-of-record gap. The remaining records are compiled in the same manner with an end-of-file recorded as a terminating mark.

Because of the merging operation, all bodies are given in one list in a random order according to the starting date of the interval. The starting date is the Julian day (word 2) minus the half interval (word 3) (see procedure, paragraph 6). The entire ephemeris occupies about one-seventh reel of tape. A summary of data is given in table VII.

APPENDIX G

INPUT-DATA REQUIREMENTS

The procedure needed to run actual problems with the aid of this routine is described herein. It is intended to permit a person with a specific problem in mind to make a complete list of data required and to select desirable operating alternatives from those available to him. The details of this procedure are contained in the following instructions:

(1) Provision has been made for two types of ephemeris data to specify the locations of celestial bodies that perturb the vehicle. They are ellipse data and ephemeris-tape data. If the problem does not involve perturbing bodies (except a reference body) or if elliptic data are used for all the perturbing bodies, skip to instruction 5.

(2) If the perturbing-body data are to be taken from an ephemeris tape, list the names of the ephemerides and Julian dates to be covered along with the following auxiliary information:

1st card: $DATA = 300, $TABLE, 2 = TAPE 3, 17 = ELIST, 29 = TBEGIN, 30 = TEND/

Other cards: TAPE 3 = 0

TBEGIN = ephemeris beginning Julian date

TEND = ephemeris ending Julian date

ELIST = (names of perturbing bodies in "ALF" format, see example in text)

The ephemerides of all planets except Earth bear the name of the planet. The ephemeris giving the distance from Earth to the sun is called "sun," as is astronomical practice.

(3) If successive files on the ephemeris tape are to be made, punch the corresponding sets as follows:

$DATA = 300, TAPE 3 = 0, TBEGIN = , TEND = , ELIST =

As many similar sets as are needed may be appended.

(4) If ellipse data are to be loaded from cards, they are prepared later under instruction 12.

(5) On the first execution after loading the routine, the common area is cleared whether an ephemeris tape is constructed or not. It is now necessary

to load a table of variable names. Once loaded, this table will not be cleared again (except if the control variable TAPE 3 is set to zero). These names are for use on the input cards. If a different name is desirable for any variable, it may be changed in the table and where it appears on the input card (ref. 7). The cards are:

```
$DATA=1,$TABLE,104=LAT,105=LONG,106=AZI,107=ELEV,108=ALT,109=VEL,7=TKICK
,28.=IMODE,45=X,46=Y,47=Z,42=VX,43=VY,44=VZ,42=E,43=OMEGA,44=NODES,45=
INCL,46=MA,47=P,41=RMASS,31=DTOFFJ,32=TOFFT,48=TIME,811=BODYCD,16=TFILE,
941=ELIPS,27=OBLATN,38=OBLATJ,39=OBLATK,34=PUSH,5=SIMP,33=FLOW,24=AEXIT,
565=BETA,601=COEFN,238.=ICC,26=ATMN,29=RATM,459=ROTATE,35=AREA,37=EREF,
17=ERLIMT,103.=MODOUT,30=TMAX,20=STEPMX,23=DELMAX,21=STEPS,22=TMIN,1=
TEST,268.=NDUMP,272.=NSKIP,257.=LENGTH,19=CLEAR,8=SAVE,9=RECALL,10=DELT/
```

(6) The initial position and velocity of the vehicle may be given in any one of three coordinate systems. If the initial data are given in orbit elements, skip to instruction 8. If the initial data are given in rectangular coordinates, skip to instruction 7. If the initial data are given in Earth-centered spherical coordinates, the following variables should be punched:

LAT = latitude, deg, positive north of equator

LONG = longitude, relative to Greenwich, deg

ALT = altitude above sea level, m

AZI = azimuth angle, east from north, deg

ELEV = elevation angle, horizontal to path, deg

VEL = initial velocity, m/sec

TKICK = size of initial vertical, nondrag step to facilitate starting, sec

IMODE = 4

These geocentric coordinates are converted by subroutine TUDES to rectangular coordinates and IMODE will be changed to 2 with its original sign. Skip to instruction 9.

(7) If the initial data are in rectangular coordinates, set the following variables:

X = x-component of position in x,y,z coordinate system, m

Y = y-component of position in x,y,z coordinate system, m

Z = z-component of position in x,y,z coordinate system, m

VX = x-component of velocity in x,y,z coordinate system, m/sec

VY = y-component of velocity in x,y,z coordinate system, m/sec

VZ = z-component of velocity in x,y,z coordinate system, m/sec

IMODE = 2

Skip to instruction 9.

(8) If the initial data are in orbit elements, set the following variables:

E = eccentricity

OMEGA = argument of pericenter, radians

NODES = longitude of ascending node (to mean vernal equinox of 1950.0), radians

INCL = orbit inclination to mean equator of 1950.0, radians

MA = mean anomaly, radians

P = semilatus rectum, m

IMODE = 1

(9) Integration is performed on either rectangular variables or orbit elements. If the initial data are of the same type as the desired integration variables, the positive sign on IMODE, as given in instruction 8, will signal a matching condition; but if the desired integration variables are of the opposite type to the input variables, a minus sign should be affixed to the value of IMODE. Note that in the case of geocentric coordinates, an automatic conversion to rectangular coordinates is effected. To convert geocentric coordinates to orbit elements requires IMODE = - 4, whereupon subroutine TUDES will convert the geocentric coordinates to rectangular coordinates, IMODE will be set to -2, and then in MAIN 2 the further conversion to orbit elements will be sensed with IMODE finally being set to +1 by the program.

(10) To specify vehicle mass and takeoff time, set the following variables:

RMASS = mass of vehicle, kg

DTOFFJ = Julian day number

TOFFT = fraction of day

TIME = time from previously set Julian date, sec

Takeoff occurs at the instant corresponding to the sum of the last three quantities. If a specific date or time is not required, these variables may be skipped. In that case, the STDATA subroutine sets DTOFFJ to 2440 000.

(11) To specify the origin and any perturbing bodies, list them as BODYCD = (list of body names in "AIF" format, see text example). The first body in the list is taken to be the reference body. The distances between the bodies in this list must be computable from either ellipse data (instruction 12) or ephemeris-tape data (instruction 2). There may be no more than eight names in the list. Also, if the ephemeris tape is being used, the correct file must be found on it. For this purpose, set TFILE = desired ephemeris tape file. The ephemeris files were numbered in sequence when written in instruction 2. If TFILE is not given, it will be set equal to 1.0 by the STDATA subroutine.

(12) For each body whose path is represented by an ellipse, a 15-element set of data must be loaded. A 15-element set consists of:

1. body name in "AIF" format (maximum of six characters)
2. reference body name in "AIF" format (maximum of six characters)
3. mass of body, sun mass units
4. radius of sphere of influence, m
5. semilatus rectum, AU
6. eccentricity
7. argument of pericenter, radians
8. longitude of ascending node (to mean vernal equinox of 1950.0), radians
9. orbit inclination (to mean equator of 1950.0), radians
10. Julian day at perihelion
11. fraction of day at perihelion
12. period, mean solar days

$$\left.\begin{array}{l}13.\\14.\\15.\end{array}\right\}\text{zero}$$

It is convenient to punch a 15-element set in sequence and to separate the elements by commas on as many cards as are required. Several sets may then be loaded consecutively. The order of the sets is immaterial. Ellipse data, if present, take precedence over ephemeris-tape data. The sets are loaded consecutively, in any order, as follows:

ELIPS = set 1, set 2, set 3, . . ., set n; $n \leq 8$ (see example in text)

(13) To specify the initial integration step size, set

DELT = initial integration step size

If no value of DELT is given, it will be set to TMAX/100 by MAIN 1.

(14) If oblateness effects of the Earth are to be included, set

OBLATN = (ALF5)EARTH

(15) If thrust forces are present, set either

(1) PUSH = thrust magnitude, newtons (for $\dot{m} = 0$)

or

(2) SIMP = specific impulse (vacuum), sec

FLOW = mass-flow rate, $\dot{m}$, kg/sec

For either choice, set

AEXIT = engine exit area, m^2

Also, the thrust orientation must be specified by setting

BETA = angle β, deg (see sketch (a))

COEFN (I) = angle-of-attack schedule, $\alpha = \alpha(t)$ (see instruction 17)

ICC = fixed-point integer (see instruction 17)

For the special case of tangential thrust, none of the last three variables need be set.

(16) If aerodynamic forces are present, set

ATMN = name of body that has atmosphere, in "ALF" format

RATM = radius above which atmospheric forces are not to be considered, m

ROTATE = atmospheric-rotation rate, radians/sec ($7.29211585 \times 10^{-5}$ for Earth)

AREA = reference area, m^2

BETA = angle β, deg (see sketch (a))

COEFN (I) = angle-of attack schedule, $\alpha = \alpha(t)$, $C_L/\sin\alpha$, $C_{D,O}$, and $C_{D,i}/C_L^2$ curves (see instruction 17)

ICC = fixed-point integers (see instruction 17)

(17) If neither thrust nor aerodynamic forces are present, skip to instruction 18. The relations $\alpha(t)$, $C_L/\sin\alpha$, $C_{D,O}$, and $C_{D,i}/C_L^2$ are assumed to be quadratic functions that involve coefficients which are located in the COEFN(J) array. The arrangement of these coefficients is best explained by an example. Suppose the functions $\alpha(t)$ is as follows:

$$\alpha = \begin{cases} a_{11} + a_{12}t + a_{13}t^2 & (t_1 \leq t \leq t_2) \\ a_{21} + a_{22}t + a_{23}t^2 & (t_2 < t \leq t_3) \\ a_{31} + a_{32}t + a_{33}t^2 & (t_3 < t \leq t_4) \\ . \quad . \quad . & . \quad . \quad . \\ \text{etc.} & \text{etc.} \end{cases}$$

The coefficients $a_{i,j}$ should then be loaded into the COEFN(J) array as:

$$\text{COEFN(J)} = t_1, a_{11}, a_{12}, a_{13}, t_2, a_{21}, a_{22}, a_{23}, t_3, a_{31}, a_{32}, a_{33}, t_4, \ldots, t_n$$

Furthermore, additional sets of coefficients for the other functions may simply be added to the COEFN(J) array, which results in a string of sets of coefficients, and can be represented, for example, as:

COEFN(J) = α coefficients, $C_L/\sin\alpha$ coefficients, $C_{D,O}$ coefficients, etc.

$$= t_1, a_{11}, a_{12} \ldots, t_n, N_{M,1}, b_{11}, b_{12}, \ldots, N_{M,k}, \text{etc.}$$

The starting point in the COEFN(J) array of each function must also be loaded to identify the correct region of coefficients. To this end, the following array must also be loaded:

ICC(1) = fixed-point value of J where α coefficients begin

ICC(2) = fixed-point value of J where $C_L/\sin\alpha$ coefficients begin

ICC(3) = fixed-point value of J where $C_{D,i}/C_L^2$ coefficients begin

ICC(4) = fixed-point value of J where $C_{D,0}$ coefficients begin

For this purpose, all values in the COEFN(J) array are called coefficients (i.e., the t's and the N_M's are coefficients). The sequence of the sets is arbitrary, since changing the sequence requires only a change in the ICC(I) array. See Example II - Lunar impact probe section.

(18) The size of the integration steps is determined primarily by the error control variables. These are loaded as:

EREF = error reference value; $\bar{\delta}$ in appendix D

ERLIMT = maximum value of δ that is acceptable on any particular step

EREF is always treated as a positive number; however, if it is loaded with a minus sign, this will cause error information to be printed at the completion of the problem. If no error control data is loaded, STDATA subroutine will set EREF = 1×10^{-6}, ERLIMT = 3×10^{-6}.

(19) The output control offers a choice on the frequency of output data as follows:

If MODOUT = 1, output will occur every n^{th} step (n = STEPS) until t = TMIN, and then MODOUT is set equal to 2 by the program

If MODOUT = 2, output occurs at equal time intervals of DELMAX until t = TMAX

If MODOUT = 3, output occurs at equal time intervals of DELMAX until t = TMIN, then MODOUT is set equal to 4 by the program

If MODOUT = 4, output occurs every n^{th} step (n = STEPS) until t = TMAX

TMAX = maximum time limit before problem is completed

STEPMX = maximum step limit before problem is completed

DELMAX = time interval between outputs

STEPS = number of steps between outputs

TMIN = time when MODOUT changes

Note that output control may, at times, strongly influence the integration step size especially if MODOUT is 2 or 3 and DELMAX is small. TMAX must be loaded. All others may be skipped; if so, STDATA will put MODOUT = 4, and STEPS = 1.

(20) For debugging operations and for occasional supplementary output, it may be desirable to obtain G-type format dumps. These may be obtained through strategic positioning of the FORTRAN calling statements CALL DUMP (ID, DATA, LENGTH) where ID is the identification number to appear in the output, DATA is the starting location of the dump area, and LENGTH is the number of consecutive words to be dumped. To actually obtain dumps at execution time, set

TEST = total desired number of dumps

NDUMP(J) = identification numbers of desired dumps, corresponding to ID's of calling statement, $J \leq 4$

NSKIP(J) = number of skips to occur between dumps, NSKIP(J) acts upon NDUMP(J), $J \leq 4$

LENGTH = number of consecutive words to be dumped

Note NDUMP(J) will occur the NSKIP(J)th time control passes through the calling statement and will occur every NSKIP(J)th time thereafter. If NSKIP(J) is omitted, it is taken to be 1. DATA may be a common location or the name of a relative variable. If the value of a word to be dumped is zero, it is skipped.

(21) For certain problems, it is desirable to save the initial data read in on cards or the data generated at the completion of a part of a problem. The saved data may then be recalled at a later time to be used as intial conditions for another problem. To prevent the "standard data" set from being loaded (and the accompanying common clearing loop), set

$DATA = 99, CLEAR = any nonzero number

To save the initial data before the input is read in (i.e., the result of a previous calculation), set

SAVE = 2

To save the initial data after the input is read in, set

SAVE = 1

To recall the saved data, set

RECALL = any nonzero number

CLEAR = any nonzero number

By taking advantage of the place in the program where each discrimination is made, several useful combinations of these controls are possible (see fig. 2).

(22) When a transfer of origin occurs, provision has been made to read input into the program. This is done with the aid of $DATA = 101, followed by the data statements desired.

(23) Following is an input check list that may be helpful at execution time:

INPUT CHECK LIST[a]

<table>
<tr><td rowspan="2">Time and Mass</td><td colspan="3">Position and velocity
(completely fill in one and only one block)</td><td colspan="2">Reference and perturbing bodies</td></tr>
<tr><td>Rectangular</td><td>Orbit elements</td><td>Spherical</td><td colspan="2">BODYCD =
Tape   Elliptic</td></tr>
<tr><td>DTOFFJ =
TOFFT =
DELT =
TIME =
RMASS =</td><td>X =
Y =
Z =
VX =
VY =
VZ =
IMODE = 2</td><td>E =
OMEGA =
NODES =
INCL =
MA =
P =
IMODE = 1</td><td>LAT =
LONG =
AZI =
ELEV =
ALT =
VEL =
IMODE = 4</td><td>TAPE 3 = 0 (b)
TBEGIN =
TEND =
ELIST =
TFILE =</td><td>ELIPS =</td></tr>
</table>

<table>
<tr><td>Output control</td><td>Error control</td><td>Restart</td><td>Thrust (d)</td><td>Atmosphere</td><td>Oblateness</td><td>Dump</td></tr>
<tr><td rowspan="2">TMAX = (c)
TMIN =
MODOUT =
STEPS =
DELMAX =
STEPMX =</td><td rowspan="2">EREF =
ERLIMT =</td><td rowspan="2">SAVE =
RECALL =
CLEAR =</td><td>SIMP =
FLOW =
PUSH =</td><td>ATMN =
RATM =
ROTATE =
AREA =</td><td rowspan="2">OBLATN =</td><td rowspan="2">NDUMP =
NSKIP =
LENGTH =
TEST =</td></tr>
<tr><td colspan="2">COEF =
ICC =
BETA =</td></tr>
</table>

[a]The following standard data are loaded by subroutine STDATA:

DTOFFJ = 2440 000.0	MODOUT = 4	EREF = 1×10^{-6}
IMODE = 1	STEPS = 1.0	ERLIMT = 3×10^{-6}
BODYCD(1) = (ALF5)EARTH	STEPMX = 100.0	TFILE = 1.0
RMASS = 1.0		

(b)At input 300, setting TAPE 3 = 0 is necessary to make an ephemeris tape.

(c)A value for TMAX is always required.

(d)Use either SIMP = and FLOW = or PUSH = .

APPENDIX H

PROGRAM LISTING

```
C     MAIN 1 -- FLOW CONTROL PROGRAM FOR SEGMENT 1.  THIS ROUTINE IS ENTERED FOR
C     EITHER (1) THE START OF A PROBLEM OR (2) AN IN-FLIGHT ORIGIN BODY CHANGE.
C     THE REGION OF COMMON FROM 801 TO 1600, CEX, IS NOT NEEDED IN SEGMENT 2 AND
C     IS THEREFORE SAVED ON DRUM 2 DURING EXECUTION OF THAT SEGMENT TO OPEN UP
C     THE ADDITIONAL 800 STORES.
C
      COMMON C
C
      DIMENSION
     1        C (1600),         BNAME (8),            NPONG (5),
     2      CEX (800),         BODYCD (8),             BEX (14)
C
      EQUIVALENCE
     1( CLEAR,C( 19)),( CLOCK,C(  3)),(   BEX,C(801)),(INDERR,C(491)),
     2(  TTOL,C(226)),( IMODE,C( 28)),(RECALL,C(  9)),(  SAVE,C(  8)),
     3(   DEL,C(255)),(  EREF,C( 37)),(LENGTH,C(257)),(  TMAX,C( 30)),
     4( TAB ,C(1301)),(DELMAX,C( 23)),(  DELT,C( 10)),( ERLOG,C(259)),
     5(REVOLV,C(250)),(ATM N ,C( 26)),(R ATM ,C( 29)),(RATMOS,C(248)),
     6(ROTATE,C(459)),( NPONG,C( 11)),(BODYCD,C(811)),( BNAME,C(402)),
     7( TFILE,C( 16)),(  FILE,C(249)),( TAPE3,C(  2)),(TRSFER,C(224)),
     8(   CEX,C(801))
C
C     THE COMMON EXTENSION, CEX, IS RESTORED (JUNK IS BROUGHT IN UPON THE FIRST
C     ENTRY).  WHEN TAPE3=0.0, SUBROUTINE TAPE IS CALLED TO COMPILE THE
C     EPHEMERIDES.  SUBROUTINE TAPE ALWAYS SETS TAPE3=3.
      READ DRUM 2,0,CEX
      IF (TAPE3) 2,1,2
    1 CALL TAPE
C
C     WHEN AN IN-FLIGHT ORIGIN TRANSFER OCCURS, SEGMENT 1 IS CALLED WITH TRSFER
C     =1.0.  HERE, AN INPUT IS ALLOWED AND THEN CONTROL IS SENT TO REORDER THE
C     BODY LIST.
    2 IF (TRSFER) 4,4,3
    3 TRSFER = 0.
      CALL INPUT(101,C,TAB)
      GO TO 28
C
C     PRINT OUT THE ERROR INFORMATION IF EREF HAS A - SIGN.
    4 IF (EREF) 5,10,10
    5 WRITE OUTPUT TAPE 6,8
      REWIND 4
      DO 6  I=1,INDERR
      READ TAPE 4, BEX
    6 WRITE OUTPUT TAPE 6,9,(BEX(J),J=1,14)
    7 REWIND 4
      INDERR = 0
      READ DRUM 2,786,BEX
    8 FORMAT(7H1  STEP,6X,4HTIME,6X,4HDELT,7X,2HA2,8X,2HE2,7X,4HMASS,6X,
     14HE,VX,4X,8HOMEGA,VY,2X,8HNODES,VZ,3X,6HINCL,X,5X,4HMA,Y,6X,3HP,Z,
     24X,1HK//)
    9 FORMAT(F5.,1H+F3.,1P11G10.2,I2)
C
C     PRINT OUT THE COMPUTATION TIME ELAPSED SINCE THE LAST ENTRY TO MAIN 1.
   10 CALL TIME1 (CLOCK1)
      IF (CLOCK)  11,13,11
   11 TUSED = CLOCK1 - CLOCK
      WRITE OUTPUT TAPE 6 ,12,TUSED
   12 FORMAT( 15HOMINUTES USED =F7.1/1H1)
   13 CLOCK = CLOCK1
C
C     CALL IN THE STANDARD DATA IF CLEAR=0.  INPUT 99 IS BASICALLY AN AUXILLARY
C     INPUT TO ALLOW A CHANGE IN CLEAR.  IF SAVE=2.0, THE DATA FROM COMMON
C     5 TO 115 IS SAVED.
      CALL INPUT (99,C,TAB)
      IF (CLEAR) 15,14,15
   14 CALL STDATA
   15 IF (SAVE-2.) 18,16,18
   16 DO 17 J=5,115
   17 C(J+1485) = C(J)
C
C     WHEN RECALL DOES NOT EQUAL ZERO, THE INITIAL DATA PREVIOUSLY STORED BY A
C     SAVE STATEMENT WILL BE RECALLED IN ORDER TO RESTART WITH THE SAME DATA.
   18 IF (RECALL ) 19,21,19
   19 DO 20 J=5,115
   20 C(J) = C(J+1485)
C
C     INPUT 1 IS THE MAIN INPUT STATEMENT, DATA READ IN HERE OVERWRITES ANY
C     STANDARD VALUES SET BY STDATA.  IF SAVE=1.0, THE INITIAL SET OF DATA FROM
C     COMMON 5 TO 115 WILL BE SAVED FOR LATER USE.
   21 CALL INPUT (1,C,TAB)
      IF (SAVE-1.) 24,22,24
   22 DO 23 J=5,115
   23 C(J+1485) = C(J)
   24 IF (DELT) 26,25,26
   25 DELT = TMAX/100.
   26 ERLOG = LOGF(ABSF(EREF))
      DEL = DELMAX
      TTOL = 5E-8*TMAX
      BNAME(1) = BODYCD(1)
C
```

```
C       IF IMODE IS 4, THE INITIAL DATA IS EARTH CENTERED SPHERICAL
C       COORDINATES AND IS TO BE TRANSFORMED INTO RECTANGULAR COORDINATES.  THIS
C       IS DONE BY SUBROUTINE TUDES WHERE, ALSO, AN INITIAL STEP MAY BE TAKEN TO
C       FACILITATE STARTING.  THE COMMON EXTENSION, CEX, IS SAVED.  SUBROUTINE
C       ORDER IS CALLED TO ORDER THE LIST OF BODIES, COMPUTE THE GRAVITATIONAL
C       CONSTANT, AND MODIFY ANY ELLIPTIC EPHEMERIS DATA.
        IF (XABSF(IMODE)-4) 28,27,28
     27 CALL TUDES
        IMODE = XSIGNF(2,IMODE)
     28 WRITE DRUM 2,0,CEX
        CALL ORDER
        CALL DUMP (1,C,LENGTH)
C
C       IF ORIGIN BODY HAS AN ATMOSPHERE, SET ROTATION RATE AND ATMOSPHERE RADIUS.
        REVOLV = 0.
        RATMOS = 0.
        IF (ATMN - BNAME(1)) 31,29,31
     29 REVOLV = ROTATE
        RATMOS  = RATM
C
C       POSITION THE EPHEMERIDES TAPE AT THE BEGINNING OF THE CORRECT EPHEMERIS
C       BY MATCHING THE EPHEMERIS NUMBER READ FROM TAPE (FILE) WITH THE DESIRED
C       EPHEMERIS NUMBER (TFILE).  THEN CALL IN SEGMENT 2.
     31 IF (FILE) 36,36,32
     32 CALL BKFILE(3)
     33 READ TAPE 3, FILE
        IF (FILE-TFILE) 34,36,32
S    34 RTB 3
S    35 CPY
S       TRA *35
S       TRA *33
S       TRA *34
     36 CALL PONG(NPONG(1))
C
C       END OF THE FORTRAN STATEMENTS.                                  ********


        SUBROUTINE TAPE
C
C       SUBROUTINE TAPE USES THE MASTER MERGED EPHEMERIDES TAPE (TAPE 8 AT LEWIS)
C       TO COMPILE A WORKING EPHEMERIS TAPE (TAPE 3 AT LEWIS) WHICH CONTAINS ONLY
C       THAT DATA NEEDED AT EXECUTION TIME.  THIS MINIMIZES TAPE HANDLING DURING
C       EXECUTION.  THERE ARE 2 FILES ON TAPE 8, FIRST FILE HAS THE DATA AND IS
C       IDENTIFIED BY THE SECOND WORD OF EACH 256 WORD RECORD (FIRST WORD IS THE
C       DUMMY FORTRAN COMPATIBLE WORD, SECOND WORD=2).  THE SECOND FILE IS ONLY 2
C       WORDS LONG, FIRST WORD IS FORTRAN COMPATIBLE, SECOND WORD=3).
C       MASTER FILE 1 -- PLANETS (EXCEPT MERCURY AND EARTH), SUN, MOON, AND
C                        EARTH-MOON BARYCENTER FROM SEPT.25, 1960 TO ABOUT 2000.
C       EACH EPHEMERIS COMPILED REQUIRES A SET OF INPUT 300 DATA.  THE FIRST PIECE
C       OF DATA WRITTEN ON A FILE IS THE FILE IDENTIFICATION NUMBER, FILE.  EACH
C       FILE IS NUMBERED CONSECUTIVELY STARTING WITH FILE=1. SINCE MOON DATA IS IN
C       TERMS OF EARTH RADII, THE CONVERSION OF MOON DATA TO A.U. IS MADE BEFORE
C       WRITING ON TAPE 3.  THE COMMON USED IN SUBROUTINE TAPE IS LOCAL AND ALL
C       BUT TAPE3 IS CLEARED BY A FINAL CLEARING LOOP.
C       FUNCTION COMPARF(A,B) IS EQUIVALENT TO (A-B) BUT WILL NOT OVERFLOW.
C       NORMAL INPUT - ELIST, TBEGIN, TEND, TAPE3
C
C       ELIST-  THE BCD LIST OF EPHEMERIS DATA NAMES TO BE PLACED ON
C               TAPE 3 .  THE NAMES ARE READ FROM CARDS,  AND IS USED TO
C               MAKE THE TMAKE LIST. ELIST IS NOT CHANGED IN STORAGE UNTIL
C               THE FINAL CLEAR FOR THIS SUBROUTINE.
C       TMAKE-  THE LIST OF EPHEMERIS NAMES WITH DUPLICATES DROPPED AND
C               ZERO SPACES CLOSED IN. AS THE EPHEMERIDES ARE FINISHED THE
C               NAMES ARE ERRASED FROM THIS LIST.
C       TMADE-  LIKE TMAKE BUT IS HELD FOR OUTPUT.
C       TBEGIN- THE BEGINNING DATE  EXPRESSED AS A JULIAN DAY.
C       TEND-   ENDING DATE EXPRESSED AS A JULIAN DAY.
C       INTVAL- THE APPROX. NUMBER OF DAYS COVERED BY ONE SET OF COEFF. IT
C               IS USED TO DECIDE WHICH DATA ARE TO BE ENTERED DOUBLE. THE
C               DOUBLE ENTRIES PERMIT FASTER OPERATION IF REVERSAL OF
C               INTEGRATION IS REQUIRED FOR ANY REASON.
C       EDATE-  JULIAN ENDING DATE FOR THE MASTER EPHEMERIS.
C       ERTOAU- EARTH RADII PER A.U.
C
        COMMON   C
C
        DIMENSION
       1        C (1600),        TMAKE (12),          LIST (30),
       2    EDATE (12),         INTVAL (30),          KTAG (12),
       3    ELIST (12),          TMADE (12),         INTVA (2),
       4    PNAME (30),         TDATUM (1100),      DATUMT (21,12)
C
```

```
      EQUIVALENCE
     1( TAPE3,C(  2)),(ERTOAU,C(  3)),(  KTAG,C(  4)),(  FILE,C( 16)),
     2( ELIST,C( 17)),(TBEGIN,C( 29)),(  TEND,C( 30)),( PNAME,C( 31)),
     3( KHAMP,C( 61)),( TMADE,C( 73)),( TMAKE,C( 85)),(TDATUM,C(441)),
     4( EDATE,C(127)),(INTVAL,C(157)),( INTVA,C(156)),(DATUMT,C(189))
C
C     PART 1. REWIND 3 AND CLEAR COMMON.
B     COMPARF(A,B) = (A+B)*(-(A*B))
      REWIND 3
      DO 1  K=1,1600
    1 C(K) = 0.0
C
C     THE FOLLOWING NH STATEMENTS LOAD THE BODY NAMES INTO THE MACHINE.
C     NOTE.  THE EARTH IS NOT IN THIS LIST (NO EPHEMERIS FOR EARTH.)
      PNAME(1) = 3HSUN
      PNAME(2) = 6HMERCUR
      PNAME(3) = 5HVENUS
      PNAME(4) = 4HMARS
      PNAME(5) = 6HJUPITE
      PNAME(6) = 6HSATURN
      PNAME(7) = 6HURANUS
      PNAME(8) = 6HNEPTUN
      PNAME(9) = 5HPLUTO
      PNAME(10)= 4HMOON
      PNAME(11)= 6HEARTHM
C
C     PART 2.  SET UP JULIAN DATES ENDING EACH EPHEMERIS.
      EDATE(1) = 2451872.5                                            11/24/00
      EDATE(3) = 2451848.5                                            10/31/00
      EDATE(4) = 2451020.5                                             7/26/98
      EDATE(5) = 2473520.5                                                2060
      EDATE(6) = 2473520.5                                                2060
      EDATE(7) = 2473520.5                                                2060
      EDATE(8) = 2473520.5                                                2060
      EDATE(9) = 2473520.5                                                2060
      EDATE(10)= 2440916.5                                            11/26/70
      EDATE(11)= 2451848.5                                            10/31/00
      INTVA  = 30000
      INTVAL(1) = 8
      INTVAL(2) = 5
      INTVAL(3) = 15
      INTVAL(4) = 44
      INTVAL(5) = 330
      INTVAL(6) = 825
      INTVAL(7) = 1211
      INTVAL(8) = 1172
      INTVAL(9) = 1101
      INTVAL(10) = 2
      INTVAL(11) = 15
      FILE = 1.
      ERTOAU = 4.26546512 E-5
    2 END FILE 3
      MOON = 0
      LI = 1
C
C     PART 2B. CALL INPUT AND SEE IF TAPE IS TO BE MADE.  INPUT  MUST ALWAYS
C              MAKE TAPE3=0.0 IF TAPE IS TO BE MADE.
      TAPE3 = 3.
      CALL INPUT(300,C,LIST)
      IF (TAPE3) 63,3,63
C
C     PART 3. TAPE IS TO BE MADE SO MOVE EPHEMERIS LIST TO TMAKE AND
C         TO TMADE (FOR OUTPUT), CANCEL ANY ZERO OR DUPLICATE NAMES.
    3 KOUNT = 1
      DO 6 K=1,12
      TMAKE(K) = 0.
      TMADE(K) = 0.
    4 DO 5 J=1,KOUNT
      IF (COMPARF(ELIST(K),TMAKE(J-1))) 5,6,5
    5 CONTINUE
      TMAKE(KOUNT) = ELIST(K)
      TMADE(KOUNT) = ELIST(K)
      KOUNT = KOUNT+1
    6 CONTINUE
      KOUNT = KOUNT - 1
C
C     PART 4. FIND INPUT ERRORS.
    7 IF(TBEGIN-2437202.5) 66,9,9
    9 KM = 2
   11 ERROR = 0.
      WRITE TAPE 3,FILE
      DO 21 J=1,KOUNT
      KTAG(J) = 0
   12 DO 13 K=1,20
      IF (COMPARF(PNAME(K),TMAKE(J))) 13,16,13
   13 CONTINUE
C
```

```
C     PART 5. PRINTS OUT THE  MISSPELLED NAMES AND OTHER ERRORS.
   14 PRINT   15,  TMAKE(J), TBEGIN, TEND
      WRITE OUTPUT TAPE 6 , 15,  TMAKE(J), TBEGIN, TEND,(PNAME(K),
     1EDATE(K),K=1,20)
   15 FORMAT( 23H TROUBLE ON TAPE 3 MAKE  / 2X,A6,10H  T BEGIN= F10.1,8H
     1  T END= F10.1//2(2X,A6,F20.1))
      ERROR = 1.
      GO TO 21
C
C     PART 4B. CHECKS DATES AND STORES INDEX FOR MOON SO THAT EARTH
C     RADII CAN BE CONVERTED TO A.U.
   16 IF (10-K) 18,17,18
   17 MOON = J
   18 KTAG(J) = K
   19 IF (EDATE(K)- TEND)  14,21,21
   21 CONTINUE
      ASSIGN 36  TO NS1
      IF (ERROR) 22,22,68
C
C     PART 6. FIX UP A TAG (KTAG) TO INDICATE WHETHER TO ENTER DATA DOUBLE OR
C             NOT.  KHAMP WILL BE SHORTEST INTERVAL.  KTAG WILL BE NON-ZERO IF
C             ANY DATA ENTERS MORE THAN ONCE FOR 10 ENTRIES OF THE MOST
C             FREQUENT DATA.
   22 KHAMP = INTVAL(0)
      DO 23  J=1,KOUNT
      K = KTAG(J)
      KHAMP = XMINOF(KHAMP,INTVAL(K))
   23 CONTINUE
      KHAMP = KHAMP *10
      DO 24  J=1,KOUNT
      K = KTAG(J)
   24 KTAG(J) = INTVAL(K) / KHAMP
C
C     PART 7.  LOCATE FILE 2 ON TAPE 8.
S  25 RTB 8
S     STZ J1
S     CPY DUD
S     CPY  KFILE
S     TRA  *26
S     TRA  *25
S     TRA  *25
   26 IF (KM-KFILE) 27,32,29
   27 IF (KFILE - 3)  28,28,29
   28 CALL BKFILE(8)
      GO TO 25
C     BY PASS A FILE.
S29   RTB  8
S30   CPY  DUD
S     TRA  *29
S     TRA  *25
S     TRA  *29
C
C     PART 8. THIS IS CORRECT FILE ON TAPE 8, READ DATA. THERE CAN BE UP
C         TO 12 SETS OF DATA PER RECORD. A SET OF DATA IS 21 WORDS.
   31 J1 = -1
S     RTB 8
S     CPY DUD
S     TRA *32
S     TRA *34
S     TRA *34
   32 J1 = J1 +1
S     CPY TDATUM(J1)
S     TRA * 32
S     TRA *34
S     TRA *33
   33 J1 = J1 - 1
      GO TO NS1,(36,46)
   34 WRITE OUTPUT TAPE 6,35, KFILE,(TMAKE(K),K=1,KOUNT)
   35 FORMAT (13H END OF FILE I3,67H ENCOUNTERED ON TAPE 8 BEFORE END TI
     1ME SATISFIED FOR THE FOLLOWING  /12(3X,A6))
      GO TO 68
C
C     PART 9. IS THIS A SATISFACTORY STARTING POINT, QUESTION MARK.
C     THE 1ST SET OF DATA FOR EACH PLANET MUST PRE DATE TBEGIN.
C     PART 9 IS EXECUTED ONLY ONCE.
   36 DO 42 J=LI,KOUNT
      DO  37 K=1,J1,21
      IF (COMPARF(TDATUM(K),TMAKE(J))) 37,39,37
   37 CONTINUE
   38 LI = J
      BACKSPACE 8
      BACKSPACE 8
      GO TO 31
   39 IF (TDATUM(K+1)-TDATUM(K+2)-TBEGIN) 40,40,38
   40 DO 41 KJ=1,21
      K1 = K + KJ - 1
   41 DATUMT(KJ,J) = TDATUM(K1)
   42 CONTINUE
      IF (MOON) 43,45,43
   43 DO 44 KJ=4,21
   44 DATUMT(KJ,MOON) = DATUMT(KJ,MOON)*ERTOAU
   45 ASSIGN 46 TO NS1
C
```

```
C        PART 10. PUT AWAY NEEDED DATA.  TEST NAME, TIME OF BEGIN AND END.  DO NOT
C                 WRITE TAPE 3 UNTIL TBEGIN PREDATES THE END OF THE FITTED
C                 INTERVAL.  50 REPEATS OLD DATA, 57 WRITES NEW DATA.  THE NAMES
C                 ARE ERASED FROM TMAKE AS SOON AS THE DATA POST DATES TEND.  WHEN
C                 ALL NAMES ARE GONE, RETURN TO INPUT 300 TO SEE IF ANOTHER
C                 EPHEMERIS IS TO BE CONSTRUCTED.
   46 DO 65 K=1,J1,21
      DO 47 J=1,KOUNT
      IF (COMPARF(TDATUM(K),TMAKE(J))) 47,48,47
   47 CONTINUE
      GO TO 65
   48 SWT = TBEGIN-TDATUM(K+1)-TDATUM(K+2)
      IF (SWT) 49,49,52
   49 IF(KTAG(J)) 50,52,50
   50 WRITE TAPE 3,(DATUMT(KJ,J) , KJ=1,21)
   51 FORMAT (1X,A6,F10.1)
   52 DO  53 KJ=1,21
      K1 = K + KJ
   53 DATUMT(KJ,J) = TDATUM(K1-1)
      IF (J-MOON) 56,54,56
   54 DO  55 KJ = 4,21
   55 DATUMT(KJ,J) = DATUMT(KJ,J)*ERTOAU
   56 IF (SWT) 57,57,58
   57 WRITE TAPE 3,(DATUMT(KJ,J),KJ=1,21)
   58 IF(TEND-DATUMT(2,J)-DATUMT(3,J)) 59,59,65
   59 TMAKE(J) = 0
      DO 60 KK=1,KOUNT
      IF (TMAKE(KK)) 65,60,65
   60 CONTINUE
      WRITE OUTPUT TAPE 6, 61, FILE,TBEGIN,TEND, KOUNT,(TMADE(KK),
     1KK=1,KOUNT)
   61 FORMAT(28H0EPHEMERIS COMPLETED,  FILE=F3.,6H, FROM F10.1,3H TO
     1 F10.1, 4H FOR I2, 18H BODIES AS FOLLOWS / 12(2X,A6))
   62 FORMAT(1X,A6,7E16.8/(1X,7E16.8))
      FILE = FILE + 1.
      GO TO 2
   63 WRITE TAPE 3, FILE
      REWIND 3
      REWIND 8
      TAPE3 = 3.
      DO 64 J=3,1600
   64 C(J) = 0
      RETURN
C
   65 CONTINUE
      GO TO 31
   66 PRINT 67, TBEGIN
      WRITE OUTPUT TAPE 6,67,TBEGIN
   67 FORMAT(33H TBEGIN PREDATES 2437202.5,IT IS F10.1)
   68 CONTINUE
      REWIND 8
C
C        END OF THE FORTRAN STATEMENTS.                                 ********


      SUBROUTINE STDATA
C
C     THIS ROUTINE CLEARS COMMON 4 TO 1300 AND LOADS A SET OF STANDARD DATA INTO
C     THE MACHINE.  ANY VALUES SET HERE MAY BE OVERWRITTEN BY INPUT 1 IN MAIN 1.
C
      COMMON C
C
      DIMENSION
     1    PNAME (12),            AMASS (30),           NPONG (5),
     2      CON (9),             COEFN (190),            ICC (4),
     3       AK (3),              XDOT (15),             IND (3),
     4    REFER (12),            RCRIT (30),              AW (4),
     5        C (1)
C
      EQUIVALENCE
     1(STEPMX,C( 20)),(CONSTU,C( 18)),(   ICC,C(238)),( IMODE,C( 28)),
     2(  ETOL,C( 25)),(ERLIMT,C( 17)),(  EREF,C( 37)),( SQRDK,C(468)),
     3( TFILE,C( 16)),( NPONG,C( 11)),( RCRIT,C(911)),( AMASS,C(881)),
     4(BODYCD,C(811)),(MODOUT,C(103)),(   IND,C(791)),( STEPS,C( 21)),
     5(  XDOT,C(161)),(   SPD,C(253)),( CONSU,C( 36)),( COEFN,C(601)),
     6(OBLATK,C(39 )),(RESQRD,C( 40)),(PNAME ,C(821)),(REFER ,C(851)),
     7( RMASS,C( 41)),(GASFAC,C(458)),(OBLATJ,C( 38)),(    AW,C(261)),
     8(   CON,C(576)),(    AK,C(233)),(DTOFFJ,C( 31)),(    AU,C(461))
C
C     CLEAR COMMON FROM 4 TO 1300.
      DO 1    J = 4,1300
    1 C(J) = 0.0
C
```

```
C     THE FOLLOWING NH STATEMENTS LOAD THE BODY NAMES INTO THE MACHINE.
      PNAME(1) = 3HSUN
      PNAME(2) = 6HMERCUR
      PNAME(3) = 5HVENUS
      PNAME(4) = 5HEARTH
      PNAME(5) = 4HMARS
      PNAME(6) = 6HJUPITE
      PNAME(7) = 6HSATURN
      PNAME(8) = 6HURANUS
      PNAME(9) = 6HNEPTUN
      PNAME(10)= 5HPLUTO
      PNAME(11)= 4HMOON
      PNAME(12)= 6HEARTHM
C
C     FILL OUT SUN REFERENCE LIST.
      DO  2    K = 2,12
    2 REFER(K) = PNAME(1)
C
C     FILL OUT EARTH REFERENCE LIST.
      REFER(1) = PNAME(4)
      REFER(4) = 5HZERO+
      REFER(11) = PNAME(4)
C
C     LOAD THE REMAINING STANDARD DATA.
      AK(1) = 0.5
      AK(2) = 0.5
      AK(3) = 1.0
      AMASS(1) = 1.0
      AMASS(2) = 1.0/6120000.0
      AMASS(3) = 1.0/406645.0
      AMASS(4) = 1.0/332488.0
      AMASS(5) = 1.0/3088000.0
      AMASS(6) = 1.0/1047.39
      AMASS(7) = 1.0/3500.0
      AMASS(8) = 1.0/22869.0
      AMASS(9) = 1.0/18889.0
      AMASS(10) = 1.0/400000.0
      AMASS(11) =AMASS(4)/81.375
      AMASS(12) =AMASS(4)+ AMASS(11)
      AU = 1.495 E11
      AW(1)=1./6.
      AW(2)=AW(1)+AW(1)
      AW(4)=AW(1)
      AW(3)=1.-(AW(2)+(AW(1)+AW(4)))
      BODYCD = PNAME(4)
      COEFN(83) = 1E20
      CON(1) = 0.2
      CON(2) = 0.2
      CON(3) = 0.6
      CON(4) = 1.4
      CON(5) = 1.4
      CON(6) = 2.33333333
      CON(7) = 0.1
      CON(8) = 0.1
      CON(9) = 0.5
      CONSTU = 1.0 E-6
      CONSU = 1E-6
      ETOL = 0.01
      DTOFFJ = 244.E4
      EREF = 1E-6
      ERLIMT = 3E-6
      GASFAC = 20.064881
      ICC(1) = 79
      ICC(2) = 79
      ICC(3) = 79
      ICC(4) = 79
      IMODE = 1
      IND(1)=2
      IND(2)=3
      IND(3)=1
      MODOUT = 4
      NPONG(1) = 2
      NPONG(2) = 1
      NPONG(3) = 3
      NPONG(5) = 1
      OBLATJ=1.6238E-3
      OBLATK = 6.4E-6
      RCRIT(1) = 1.0 E+20
      RCRIT(2) = 1.0 E+8
      RCRIT(3) = 6.14 E+8
      RCRIT(4) = 9.25 E+8
      RCRIT(5) = 5.78 E+8
      RCRIT(6) = 4.81 E+10
      RCRIT(7) = 5.46 E+10
      RCRIT(8) = 5.17 E+10
      RCRIT(9) = 8.61 E+10
      RCRIT(10) =3.81 E+10
      RCRIT(11) =1.60 E+8
```

```
      RESQRD =4.068098877 E+13
      RMASS = 1.
      SPD = 86400.0
      SQRDK  = 1.32452139 E+20
      STEPMX= 100.0
      STEPS = 1.
      TFILE = 1.
      XDOT(8) = 1.0
      WRITE OUTPUT TAPE 6,3
    3 FORMAT (7HOSTDATA)
      RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                    ********


      SUBROUTINE TUDES
C
C     THIS ROUTINE COMPUTES THE RECTANGULAR POSITION AND VELOCITY COMPONENTS
C     WITH RESPECT TO THE EARTH MEAN EQUINOX AND EQUATOR OF 1950.0 FROM THE
C     LATITUDE, LONGITUDE, AZIMUTH, ELEVATION, ALTITUDE, TOTAL VELOCITY, AND
C     TIME.  ALSO, WHEN TKICK DOES NOT EQUAL ZERO, A NON-DRAG VERTICAL STEP OF
C     SIZE TKICK IS MADE IN CLOSED FORM (STATEMENTS 2 TO 4).  THE INTEGRATION
C     WILL THEN BEGIN AT TIME EQUAL TO TIME+TKICK WITH THE ORIENTATION SPECIFIED
C     BY THE ABOVE FOUR ANGLES AND THE COMPUTED VALUES OF ALTITUDE AND VELOCITY.
C     FOR THE CLOSED FORM APPROXIMATION, A CONSTANT FLOW RATE (FLOW), VACUUM
C     SPECIFIC IMPULSE (SIMP) AND ENGINE EXIT AREA (AEXIT) ARE ASSUMED KNOWN.
C     THE ATMOSPHERIC PRESSURE IS TAKEN TO BE THE SEA LEVEL VALUE.
C
      COMMON C
C
      DIMENSION
     1   AMASS (30),          ANGLES (4),            SINA (4),
     2     COSA (4),          ANGLEB (4)
C
      EQUIVALENCE
     1(     X,C( 45)),(      Y,C( 46)),(      Z,C( 47)),(     VX,C( 42)),
     2(    VY,C( 43)),(     VZ,C( 44)),(DTOFFJ,C( 31)),( TOFFT,C( 32)),
     3(ANGLES,C(104)),(    ALT,C(108)),(    VEL,C(109)),(ROTATE,C(459)),
     4(  TIME,C( 48)),(   SIMP,C(  5)),( RMASS,C( 41)),( TKICK,C(  7)),
     5(  FLOW,C( 33)),(STEPGO,C(101)),(STEPNO,C(102)),( AEXIT,C( 24)),
     6(OBLATN,C( 27)),( BNAME,C(402)),(RESQRD,C( 40)),(OBLATJ,C( 38)),
     7( AMASS,C(881)),( SQRDK,C(468)),(    SPD,C(253))
C
      ALT1 = 0.
      VEL1 = VEL
      DEL1 = 0.
      DEL = 0.
      ASSIGN 1 TO NGO
      GREEN = 360.0*(MODF((DTOFFJ-2437665.5)/.997269566,1.)+
     1 MODF((TOFFT+TIME/SPD-.71979301)/.997269566,1.))
      SINA(1) = SINF(ANGLES(1)/57.2957795)
      RADIUS=6356783.28/SQRTF(.9933065783+.006693421685*SINA(1)**2)+ALT
      GO TO 8
    1 X = COSA(2)*COSA(1)*RADIUS
      Y = SINA(2)*COSA(1)*RADIUS
      Z = SINA(1)*RADIUS
      IF (TKICK) 2,4,2
    2 RMASSO = RMASS
      RMASS = RMASS-FLOW*TKICK
      WRITE OUTPUT TAPE 6,3,STEPGO,STEPNO,(ANGLES(I),I=1,4)ALT,TIME,VEL,
     1 RMASSO,X,Y,Z
    3 FORMAT(6HOSTEP=F5.,2H +F4.,4X,6H LAT.=1PG15.8,7H LONG.=G15.8,6H AZ
     1I.=G15.8,7H ELEV.=G15.8,6H ALT.=G15.8/6H TIME=G15.8,6H VEL.=G15.8,
     67H RMASS=G15.8,4X,2HX=G15.8,5X,2HY=G15.8,4X,2HZ=G15.8)
      TIME = TIME+TKICK
      B1 = LOGF(RMASSO/RMASS)
      SIMPSL = SIMP-AEXIT/FLOW*10332.275
      VEL1 = VEL+SIMPSL*9.80665*B1-G*TKICK
      ALT1 = TKICK*(VEL-G*TKICK/2.+9.80665*SIMPSL*(1.-B1*RMASS/
     1      (RMASSO-RMASS)))
    4 RADIUS = RADIUS + ALT1
      GREEN = GREEN + ROTATE*TKICK*57.2957795
      ASSIGN 5 TO NGO
      GO TO 8
    5 X = COSA(2)*COSA(1)*RADIUS
      Y = SINA(2)*COSA(1)*RADIUS
      Z = SINA(1)*RADIUS
      IF (OBLATN-BNAME) 7,6,7
    6 DEL1 = ATANF((C2-1.)/(C3-1.)*SINA(1)/COSA(1))*57.2957795-ANGLES(1)
    7 DEL2 = RADIUS/G*SINA(1)*COSA(1)*ROTATE*ROTATE*57.29577951
      DEL = DEL1 + DEL2
      ASSIGN 10 TO NGO
    8 ANGLEB(1) = ANGLES(1) + DEL
      ANGLEB(2) = ANGLES(2) + GREEN
      ANGLEB(3) = ANGLES(3)
      ANGLEB(4) = ANGLES(4)
```

```
      DO 9 I=1,4
      SINA(I) = SINF(ANGLEB(I)/57.2957795)
    9 COSA(I) = COSF(ANGLEB(I)/57.2957795)
      C1 = 5.*RESQRD/RADIUS/RADIUS*OBLATJ
      C2 = C1*(SINA(1)*SINA(1)-.6)
      C3 = C1*(SINA(1)*SINA(1)-.2)
      G = SQRDK*AMASS(4)/RADIUS/RADIUS
      GO TO NGO, (1,5,10)
   10 COS1 = COSA(1)*SINA(4)-COSA(4)*COSA(3)*SINA(1)
      COS2 = COSA(4)*SINA(3)
      VX = VEL1*(COS1*COSA(2)-COS2*SINA(2))-Y*ROTATE
      VY = VEL1*(COS1*SINA(2)+COS2*COSA(2))+X*ROTATE
      VZ = VEL1*(SINA(1)*SINA(4)+COSA(1)*COSA(3)*COSA(4))
      RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                    ********


      SUBROUTINE ORDER
C
C     THIS ROUTINE TAKES THE BODY LIST READ FROM CARDS AND SORTS THEM IN
C     ORDER SO THAT THE DISTANCE FROM THE REFERENCE TO EACH BODY IS
C     DEPENDENT UPON ALREADY COMPUTED DISTANCES ONLY.
C
C     ELLIPSE DATA ARE READ INTO A BLOCK OF 120 STORES RESERVED FOR
C     EIGHT   ELLIPSES. ONE ELLIPSE IS READ INTO A 15 STORE BLOCK.
C     THE SINES OF THE 3 ANGLES ARE COMPUTED AND REPLACE THE 3 ANGLES.
C     THE COSINES ARE COMPUTED AND STORED LAST IN A BLOCK.
C     A BLOCK IS ARRANGED AS FOLLOWS-
C
C     (1) = NAME OF BODY IN BCD,ONLY 6 CHARACTERS.
C     (2) = NAME OF REFERENCE BODY IN BCD,SAME RESTRICTION.
C     (3) = MASS OF THE BODY IN SUN MASS UNITS.
C     (4) = RADUIS INSIDE OF WHICH COORDINATES WILL BE TRANSLATED TO THIS BODY.
C     (5) = SEMILATUS RECTUM IN ASTRONOMICAL UNITS.
C     (6) = ECCENTRICITY OF THE ORBIT.
C     (7) = SINE OF ARGUMENT OF PERIGEE.
C     (8) = SINE OF NODES.
C     (9) = SINE OF INCLINATION OF THE ORBIT.
C     (10)= PERIGEE PASSAGE JULIAN DAY.
C     (11)= PERIGEE PASSAGE FRACTION OF DAY.
C     (12)= PERIOD OF THE ELLIPSE IN MEAN SOLAR DAYS.
C     (13)= COSINE OF ARGUMENT OF PERIGEE.
C     (14)= COSINE OF NODES.
C     (15)= COSINE OF INCLINATION OF THE ORBIT.
C
C     DEFINITIONS-- NOTE. COMMON EXTENSION IS TRANSFERRED TO DRUM 2 DURING SEG2.
C     AMASS  = MASS OF EACH BODY, SUN MASSES.  ORDER OF PNAME. COMMON EXTENSION.
C     BMASS  = SELECTED FROM AMASS. CORRESPONDS TO BNAME LIST. COMMON EXTENSION.
C     BNAME  = THE ORDERED LIST OF BCD BODY NAMES. CAN BE USED IN OUTPUT.COMMON.
C     BODYCD = THE ORIGINAL BCD NAMES READ FROM CARDS.  COMMON EXTENSION.
C     BODY L = THE LIST OF BCD BODY NAMES WITH THE REFERENCE BODY AT TOP.
C              INITIALLY EQUAL TO BODY CARD LIST (BODYCD). COMMON EXTENSION.
C     IBODY  = ARRAY OF SUBSCRIPTS. WHEN A DISTANCE IS FOUND FROM EPHEMERIS, IT
C              MAY BE ADDED (OR SUBTRACTED) FROM THE BODY POSITION GIVEN BY
C              XP(IBODY) TO OBTAIN THE POSITION OF THE PRESENT BODY.  COMMON.
C     KZERO  = COUNT OF ZERO REFERENCES. THERE MUST BE ONE AND ONLY ONE ZERO.
C     NAME   = ARRAY OF SUBSCRIPTS.  GIVES OLD LOCATION OF NAMES IN BODYL
C              FROM LOCATION IN BNAME LIST. NOT IN COMMON.
C     MANE   = ARRAY OF SUBSCRIPTS.  INVERSE OF NAME.  GIVES NEW LOCATION OF
C              BNAME LIST IN TERMS OF BODYL.  NOT IN COMMON.
C     NBODYS = COUNTED INTERNALY. TOTAL NUMBER OF  BODYS.
C     MBODYS = COMPUTED INTERNALY. TOTAL NUMBER OF EPHEMERIDES (NBODYS-1).
C     NEFMRS = ARRAY OF SUBSCRIPTS. GIVES LOCATION OF BODY IN PNAME LIST
C              IN TERMS OF THE EFMRS LIST.  STORED IN COMMON.
C     NREFER = ARRAY OF SUBSCRIPTS. LOCATES THE REFERENCE BODY IN BODYL.
C              ORDER OF THE ARRAY CORRESPONDS TO BODYL.  NOT IN COMMON.
C     NNREFR = ARRAY OF SUBSCRIPTS. LIKE NREFER BUT REFERS AND CORRESPONDS TO
C              BNAME LIST.  NOT IN COMMON.
C     PNAME  = A PERMANENT LIST OF BCD BODY NAMES.  1 WORD EACH (6 CHARACTERS
C              MAX).  USED TO IDENTIFY MASS, REFERENCE NAMES, ETC.  THE LIST IS
C              A MAXIMUM OF 30 NAMES.  PRECISION TAPE NAMES ARE FROM 1 TO 20,
C              ELLIPTIC NAMES ARE FROM 21 TO 30.  COMMON EXTENSION.
C     REFER  = A PERMANENT LIST OF BCD BODYS THAT ARE THE REFERENCES OF
C              DISTANCES GIVEN IN EPHERMERIDES (TAPES OR ELLIPSE). CORRESPONDS
C              TO PNAME LIST. STORED IN COMMON EXTENSION.
C
      COMMON C
C
      DIMENSION
     1    AMASS (30),          BMASS (8),          BNAME (8),
     2    BODYL (8),           EFMRS (7),          IBODY (8),
     3     MANE (8),            NAME (8),         NEFMRS (8),
     3   NEFMRT (8),          NNREFR (8),         BODYCD (8),
     4   NREFER (8),           PNAME (30),        RBCRIT (7),
     5    RCRIT (30),          REFER (30),         TDATA (18,7),
     6     TDEL (7),             TIM (7),          ELIPS (120),
     7     NDUD (9)
C
```

```
      EQUIVALENCE
     1( AMASS,C(881)),(MBODYS,C(441)),(  GK2M,C(469)),( SQRDK,C(468)),
     2( BMASS,C(417)),(NBODYS,C(489)),(   GKM,C(470)),( TDATA,C(276)),
     3(BNAME ,C(402)),(NEFMRS,C(433)),(P NAME,C(821)),(  TDEL,C(592)),
     4(BODY L,C(801)),(BODYCD,C(811)),(RBCRIT,C(450)),(   TIM,C(585)),
     5( EFMRS,C(410)),( RMASS,C( 41)),( RCRIT,C(911)),( ELIPS,C(941)),
     6(IBODY ,C(425)),(  FILE,C(249)),( REFER,C(851)),(MANE(1),NDUD(2))
C
C     THIS SECTION SEES WHAT ELLIPSE DATA WAS READ FROM CARDS AND PUTS THE
C     NAMES IN PLACE SO THAT DATA  WILL BE USED IF NEEDED. ELLIPSE DATA HAS
C     PRIORITY OVER TAPE DATA BECAUSE LAST DATA IN LIST IS THAT ACTUALLY USED.
C     FUNCTION COMPARF(A,B) IS EQUIVALENT TO (A-B) BUT WILL NOT OVERFLOW.
C
B     COMPARF(A,B) = (A+B)*(-(A*B))
      DO 3 K=1,120,15
      IF(ELIPS(K)) 1,3,1
    1 KOUNT = (K-1)/15+21
      PNAME(KOUNT)  = ELIPS(K)
      REFER(KOUNT)  = ELIPS(K+1)
      AMASS(KOUNT)  = ELIPS(K+2)
      RCRIT(KOUNT)  = ELIPS(K+3)
      DO 2 J=6,8
      I=K+J
      ELIPS(I+6) = COSF(ELIPS(I))
    2 ELIPS(I)    = SINF(ELIPS(I))
    3 CONTINUE
C
C     PART 0. THROW AWAY BLANKS AND DUPLICATES IN BNAME LIST.
C             ALSO COUNT THE BODIES.
    4 DO 5 K=1,8
    5 BNAME(K+1)= BODYCD(K)
      L = 1
      BODYL(0) = 0.
      DO 8 I=1,9
      BODYL(I) = 0.
      DO 6 K=1,L
      IF (COMPARF (BNAME(I), BODYL(K-1))) 6,7,6
    6 CONTINUE
      BODYL(L) = BNAME(I)
      L = L+1
    7 BNAME(I) = 0.
    8 CONTINUE
      NBODYS = L-1
      MBODYS = NBODYS-1
C
C     PART 1. FIND THE REFERENCE BODY FOR EACH BODY IN THE LIST OF BODYS
C             READ FROM CARDS.  CLEAR NREFER AND BNAME.
      DO 13 KL=1,NBODYS
      NREFER(KL) = 0
      NEFMRT(KL) =0
      BNAME (KL) = 0.
      DO 12   KP= 1,30
      IF (COMPARF(BODYL(KL),PNAME(KP))) 12,9,12
    9 NEFMRT(KL) = KP
      DO    11   KR = 1,8
      IF (COMPARF(REFER(KP),BODYL(KR))) 11,10,11
   10 NREFER(KL) = KR
   11 CONTINUE
   12 CONTINUE
   13 CONTINUE
C
C     PART 2 . COUNTS 0  REFERENCES AND SAVES TEMPORARY SET OF INDEXS.
   14 IF (NBODYS) 24,24,15
   15 KZEROS = 0
      MISPEL = 0
      DO  20  K = 1,NBODYS
      NNREFR(K) = NREFER(K)
   16 IF (NEFMRT(K)) 18,17,18
   17 MISPEL = MISPEL + 1
   18 IF(NREFER(K)) 20,19,20
   19 KZEROS = KZEROS + 1
   20 CONTINUE
   21 IF  (KZEROS- 1)  24,22,24
   22 IF (MISPEL) 24,23,24
   23 IF (NBODYS-8) 28,28,24
C
C     PART 3 . REPORTS ERRORS IN BODY LIST.
   24 WRITE OUTPUT TAPE 6,25 ,NBODYS,MISPEL,KZEROS,(BODYL(K),K=1,NBODYS)
      WRITE OUTPUT TAPE 6,26  ,(NREFER(K),K=1,NBODYS)
      WRITE OUTPUT TAPE 6,27  ,(K,PNAME(K),REFER(K),K=1,30)
   25 FORMAT (26HOGOOFY BODY LIST (NBODYS =I2,13H,  MISSPELL =I2,
     1 11H,  KZEROS =I2,1H)/11HOBODYLIST =8(3X,A6))
   26 FORMAT (11H   NREFER =I6,7I9)
   27 FORMAT (/5(3H  K3X,4HBODY4X,5HREFER5X,)/5(I3,2X,A6,2X,A6,5X))
      GO TO 50
C
```

```
C     PART 4.  TRACES OUT ..REFERENCE TO BODY.. RELATIONSHIPS
   28 KK = 2
      KN = 1
      NAME(1) = 1
   29 IF (NREFER(KN))  24,31,30
   30 NAME(KK) = NNREFR(KN)
      NNREFR(KN) = 0
      KN = NAME(KK)
      KK = KK + 1
      GO TO 29
C
C     PART 5.  TRACES OUT  ..BODY TO REFERENCE..  RELATIONSHIP
   31 DO  34  KN = 1,NBODYS
      DO  34 K = 1,NBODYS
   32 IF (NNREFR(K) - NAME(KN)) 34,33,34
   33 NAME(KK) = K
      KK = KK + 1
   34 CONTINUE
C
C     PART 6. INVERTS NAME TO MANE,STORES BNAME, BMASS, RBCRIT,  AND A
C             TEMPORARY NEFMRS.
      DO 35 K = 1,NBODYS
      N = NAME(K)
      MANE(N) = K
      NEF = NEFMRT(N)
      BNAME(K) = PNAME(NEF)
      BMASS(K) = AMASS(NEF)
      RBCRIT(K) = RCRIT(NEF)
      NEFMRS(K) = NEF
   35 CONTINUE
C
C     PART 7. FINDS NNREFR REFERENCE FOR BNAME LIST , ALSO TEMP. IBODY
      DO 36  K = 1, NBODYS
      N = NAME(K)
      NRF = NREFER(N)
      NNREFR(K) = MANE(NRF)
   36 IBODY(K) = MANE(NRF)
C
C     PART 8 . FINDS IBODY FOR BACKWARD REFERENCE.
      DO 39 K=1,8
   37 IF(NNREFR(K)) 24,40,38
   38 N = NNREFR(K)
      IBODY(N) = -K
   39 CONTINUE
C     IBODY LIST IS COMPLETE.
C
C     PART 9 . WRITES OUT EPHEMERIS LIST TO BE USED IN STORING DATA AND
C              MAKES FINAL NEFMRS LIST.
   40 KK = 1
      DO 43 K=1,NBODYS
   41 IF(NNREFR(K)) 42,43,42
   42 EFMRS(KK) = BNAME(K)
      NEFMRS(KK) = NEFMRS(K)
      KK = KK + 1
   43 CONTINUE
      NEFMRS(NBODYS) = 0
C
C     PART 10. SAVES ELLIPSE DATA
      FILE = 0.
      DO 48 K=1,MBODYS
   44 IF(NEFMRS(K)-20) 47,47,45
   45 DO 46 J=5,15
      L= (NEFMRS(K) - 21) * 15 +J
      TDATA(J-4,K) = ELIPS(L)
   46 CONTINUE
      GO TO 48
C
C     PART 10A.  LOADS A FALSE (VERY EARLY) TAPE TIME TO FORCE TAPE
C                READING BY THE EPHMRS ROUTINE. FILE = 0 UNLESS TAPE IS USED.
   47 TDEL(K) = 0.
      TIM(K) = 2400000.5
      FILE = 10.
   48 CONTINUE
C
C     PART 11. COMPUTE GRAVITATIONAL CONSTANTS. 1.9866 E+30 = KILOGRAMS/SUN MASS
      GK2M = SQRDK*(BMASS(1)+RMASS/ 1.9866 E+30 )
      GKM = SQRTF(GK2M)
C
C     PART 12.    WRITES THE BNAME LIST ON TAPE 6 .
      WRITE OUTPUT TAPE 6,49,BNAME(1),(BNAME(K),K=2,NBODYS)
   49 FORMAT (19HOREFERENCE BODY IS A6,5X,23H  PERTURBING BODIES ARE
     1  7(2X,A6))
      RETURN
   50 CONTINUE
C
C     END OF THE FORTRAN STATEMENTS.                              ********
```

```
C     MAIN 2
C     MAIN 2 CONTROLS THE PROGRAM SEQUENCING FOR THE SECOND SEGMENT.  IT ALSO
C     CONTAINS THE INTEGRATION SCHEMES.  THE SET OF INTEGRATION VARIABLES IS
C     IDENTIFIED BY IMODE ACCORDING TO THE FOLLOWING
C
C       IMODE      VARIABLES
C         1    ORBIT ELEMENTS
C         2    RECTANGULAR
C         3    RECTANGULAR TEMPORARY
C        -1    ORBIT ELEMENTS--CHANGE TO RECTANGULAR
C        -2    RECTANGULAR--CHANGE TO ORBIT ELEMENTS
C        -3    ORBIT ELEMENTS--CHANGE TO TEMPORARY RECTANGULAR
C
      COMMON C
C
      DIMENSION
     1     XPRIM (15,2),       XPRIMB (15,2),       XDOTPM (15,2),
     2         X (15),           XINC (15),         OLDINC (15),
     3      XDOT (15),             RB (3),              XK (15),
     4         C (1),              AK (3),              AW (4),
     5    XWHOLE (15),             VX (3)
C
      EQUIVALENCE
     1( IMODE,C( 28)),(TRSFER,C(224)),(XWHOLE,C(544)),( XPRIM,C( 41)),
     2(XPRIMB,C( 71)),( RATIO,C(600)),(  XDOT,C(161)),(  DELT,C( 10)),
     3(    AW,C(261)),(    AK,C(233)),(OLDDEL,C(225)),(ACOEF1,C(265)),
     4(ACOEF2,C(266)),(ACOEF3,C(267)),(  XINC,C(146)),(    E2,C(260)),
     5(ERLIMT,C( 17)),(  KSUB,C(254)),(   DEL,C(255)),(STEPGO,C(101)),
     6(STEPNO,C(102)),( ASQRD,C(563)),(  GK2M,C(469)),(  REVS,C(490)),
     7(  ETOL,C( 25)),( TTEST,C(251)),(CONSTU,C( 18)),(ASYMPT,C(543)),
     8(   TRU,C(483)),( VSQRD,C(476)),(    RB,C(200)),(    VX,C(472)),
     9( ERLOG,C(259)),(    A1,C(236)),(     X,C(131)),(STEPMX,C( 20))
      EQUIVALENCE
     1(NSTART,C(247)),(     R,C(442)),(MBODYS,C(441)),(  TIME,C(138)),
     2(LENGTH,C(257)),(    H2,C(256)),(    A2,C(237)),(    ZN,C(487)),
     3( EMONE,C(243))
C
C     PART 1. SET UP THE STARTING SEQUENCE FOR ERROR CONTROL AND DELAY CHECKING
C     THE ERROR UNTIL TWO STEPS ARE COMPLETED.  THE ASSIGNED GO TOS NSTART AND
C     IBEGIN CONTROL STARTING. REWINDING 2 USUALLY SAVES TIME ON PING-PONG TAPE.
      REWIND 2
    1 DO 2 J=1,8
      XPRIM(J,2) = XPRIM(J,1)
      XPRIMB(J,2) = XPRIMB(J,1)
    2 X(J) = XPRIM(J,1)
      NSTART = 0
      H2 = DELT
      DELT = DELT/2.
      CALL EQUATE
      CALL OUTPUT
      DO 3 J=1,3
      XWHOLE(J)=VX(J)
    3 XWHOLE(J+3) = RB(J)
C     CHANGE INTEGRATION VARIABLES IF IMODE IS -.  RETURN FROM TESTTR IS AT
C     BEGINNING OF MAIN 2.
      IF (IMODE) 4,5,5
    4 CALL TESTTR
    5 ASSIGN 21 TO NSTART
C     STATEMENTS 7 TO 9 INITIALIZE NREV1 AND NREV2 FOR USE IN PART 7A.
      IF (RB(2)) 7,6,8
    6 IF(VX(2)) 7,8,8
    7 ASSIGN 37 TO NREV1
      ASSIGN 35 TO NREV2
      GO TO 9
    8 ASSIGN 33 TO NREV1
      ASSIGN 37 TO NREV2
    9 DO 10 J=1,8
      XDOTPM(J,1) = XDOT(J)
      XINC(J) = 0.
   10 CONTINUE
   11 KSUB = 1
      ASSIGN 16 TO N
C
C     PART 2.  RUNGE-KUTTA SUBINTERVAL SCHEME.  EQUATE PRODUCES THE NECCESSARY
C     DERIVATIVES XDOT(J).
   12 DO 13 J=1,8
      XK(J) = XDOT(J) * DELT
      XINC(J) = XINC(J) + AW(KSUB)*XK(J)
   13 X(J) = XPRIM(J,2) + AK(KSUB)*XK(J)
   14 CALL EQUATE
      CALL DUMP (3,C,LENGTH)
   15 GO TO N,(16,17,18,20)
C
```

```
C     PART 3. SUBINTERVALS 2, 3, AND 4, TO STATEMENT 19 FINISH A
C     RUNGE-KUTTA STEP AND INCREMENT XPRIM(J,2) IN DOUBLE PRECISION.
   16 KSUB = 2
      ASSIGN 17 TO N
      GO TO 12
   17 KSUB = 3
      ASSIGN 18 TO N
      GO TO 12
   18 DO 19 J=1,8
      XINC(J) = XINC(J) + AW(4) *XDOT(J) * DELT
      CALL EXADD(XPRIM(J,2), XPRIMB(J,2), XINC(J))
      X(J) = XPRIM(J,2)
   19 CONTINUE
C
C     PART 4. BEGIN A NEW RUNGA-KUTTA STEP.  THIS ALSO GIVES DERIVATIVES
C     FOR THE LOWER ORDER INTEGRATION CHECK.
      ASSIGN 20 TO N
      GO TO 14
   20 GO TO NSTART,(27,23,21)
C
C     PART 5.  STARTING PHASE PROGRAM.
C     PART 5A. THIS SECTION COMPLETES THE FIRST STEP OF STARTING PHASE.
   21 ASSIGN 23 TO NSTART
      DO 22 J=1,8
      OLDINC(J)=XINC(J)
      XINC(J)=0.
      XDOTPM(J,2) = XDOT(J)
   22 CONTINUE
      GO TO 11
C
C     PART 5B. MAX ERROR TEST--STARTING ONLY--CHECK THE MAX ERROR AND
C     EITHER ENTER RUNNING MODE OR REPEAT START WITH SMALLER STEP.
   23 DO 24 J=1,7
   24 XINC(J) =(XINC(J)+OLDINC(J))*3.-(XDOTPM(J,1)+XDOTPM(J,2)*4.
     1+XDOT(J))*DELT
      CALL ERRORZ
   25 IF(E2-ERLIMT) 26,26,56
   26 ASSIGN 27 TO NSTART
      ASSIGN 11 TO IBEGIN
      A1 = A2
      GO TO 31
C
C     PART 6.  RUNNING PHASE PROGRAM.
C     PART 6A. CHECK THE INTEGRATION BY INTEGRATING OVER THE LAST
C     RUNGE KUTTA STEP BUT USE DOTS FOR LAST TWO INTERVALS, OLDDEL
C     AND DELT RESPECTIVELY.  STATEMENT 28 IS THE LOWER INTEGRATION
C     MINUS RUNGE-KUTTA INCREMENTS.  ERRORZ COMPUTES THE MAXIMUM RELATIVE
C     ERROR AND STATEMENT 29 TESTS THIS ERROR AGAINST THE LIMIT VALUE.
   27 RATIO = DELT/OLDDEL
      HFACT=DELT/(1.+RATIO)
      ACOEF1=-RATIO*RATIO*HFACT
      ACOEF2=RATIO*(DELT+3.*OLDDEL)
      ACOEF3=DELT+DELT+HFACT
      DO 28 J=1,8
   28 XINC(J) = ACOEF1*XDOTPM(J,1)+ACOEF2*XDOTPM(J,2)-6.*XINC(J)
     1+ACOEF3*XDOT(J)
      CALL ERRORZ
   29 IF (E2-ERLIMT) 30,30,57
C
C     PART 7A. LAST POINT OKAY. COUNT THE REVOLUTIONS PAST THE X-AXIS.
C     A STEP GREATER THAN 1/2 REV. MAY FAIL TO ADD IN.
   30 H2 = DELT
   31 IF(RB(2)) 32,34,34
   32 GO TO NREV1, (37,33)
   33 ASSIGN 37 TO NREV1
      ASSIGN 35 TO NREV2
      GO TO 37
   34 GO TO NREV2, (37,35)
   35 ASSIGN 33 TO NREV1
      ASSIGN 37 TO NREV2
   36 REVS = REVS + 1.
S  37 LXD IMODE,(IMODE)
      GO TO (38,42,42), IMODE
C
C     PART 7B. IN ORBIT ELEMENTS. ADJUST ARGUMENT OF PERICENTER AND MEAN ANOMALY
C     TO + OR - PI TO MAINTAIN ACCURACY IN SIN-COS ROUTINES.
   38 IF (EMONE) 39,42,42
   39 DO 41 J=3,6,3
      ADJ2=INTF(XPRIM(J,2)/6.28318532+SIGNF(.5,XPRIM(J,2)))
      IF (ADJ2) 40,41,40
   40 ADJ3 = -ADJ2*6.28125
      CALL EXADD(XPRIM(J,2),XPRIMB(J,2),ADJ3)
      ADJ3=-ADJ2*.0019353072
      CALL EXADD(XPRIM(J,2),XPRIMB(J,2),ADJ3)
   41 CONTINUE
C
```

```
C     PART 7C. ADVANCE THE REMAINING PARAMETERS, FIND NEW STEP SIZE,
C     AND TEST FOR AN ORIGIN TRANSLATION.
   42 DO 43 K=1,3
      XWHOLE(K)=VX(K)
   43 XWHOLE(K+3) = RB(K)
      DO 44 J=1,8
      XDOTPM(J,1) = XDOTPM(J,2)
      XDOTPM(J,2) = XDOT(J)
      XPRIM(J,1) = XPRIM(J,2)
      XPRIMB(J,1) = XPRIMB(J,2)
      XINC(J) = 0.
   44 CONTINUE
      OLDDEL = DELT
   45 CALL STEP
      IF (MBODYS) 46,47,46
   46 CALL TESTTR
   47 GO TO (11,11,48) , IMODE
C
C     PART 7D. IF IN TEMPORARY RECTANGULAR COORDINATES, TEST FOR RETURN
C     TO ORBIT ELEMENTS.  FIRST, E IS FOUND.  IF TIME HAS NOT ADVANCED
C     SUFFICIENTLY, INTEGRATION CONTINUES IN RECTANGULAR VARIABLES (STATE. 48).
C     STATEMENT 49 DETERMINES IF KEPLERS EQUATION CAUSED IMODE = 3.  IF NOT,
C     AN E CLOSE TO 1 CHECK IS MADE IN STATEMENT 50.  IF IT DID, RECTANGULAR
C     VARIABLES WILL BE USED IF THE LIMIT IS TOO SMALL (STATEMENT 52), OR
C     IF E IS 5 OR GREATER (STATEMENT 53) OR IF THE PATH LIES CLOSE TO AN
C     ASYMPTOTE (STATEMENT 55).
   48 CALL CONVT1 (VX,C(559))
      EXMODE=SQRTF(1.+ASQRD/GK2M*(VSQRD/GK2M-2./R))
      EMONE=EXMODE-1.
      IF ((TIME-TTEST)*DELT) 11,11,49
   49 IF (ASYMPT) 51,50,51
   50 IF (ETOL-ABSF(EMONE)) 55,11,11
   51 IF(EMONE) 55,55,52
   52 IF(CONSTU-1.E-7) 11,53,53
   53 IF (EXMODE-5.) 54,11,11
   54 CALL CONVT2
      IF (ABSF(TRU)-2.2/SQRTF(EXMODE)) 55,55,11
   55 ASYMPT = 0.0
      IMODE=-2
      GO TO 46
C
C     PART 8. COMES HERE WHEN ERROR TEST FAILED--BOTH STARTING AND RUN.
C     RETRIEVE OLD POINT AND RECOMPUTE WITH SMALLER INTERVAL.
C     IF TWO CONSECUTIVE TRYS FAIL (STATEMENT 59) THE STARTING SEQUENCE OCCURS.
   56 ASSIGN 1 TO IBEGIN
   57 DO 58 J=1,8
      XPRIM(J,2) = XPRIM(J,1)
      XPRIMB(J,2) = XPRIMB(J,1)
      XDOT(J)=XDOTPM(J,2)
      XINC(J)= 0.
   58 CONTINUE
      STEPNO=STEPNO+1.
      H2 = DELT
      DELT=SIGNF(EXPF((ERLOG-A2)/5.),DELT)
      A2 =A1
   59 IF (FAIL-STEPGO) 60,61,60
   60 FAIL = STEPGO
      GO TO IBEGIN, (11,1)
   61 ASSIGN 1 TO IBEGIN
      IF (STEPNO + STEPGO - STEPMX) 62,62,45
   62 GO TO IBEGIN, (11,1)
C
C     END OF THE FORTRAN STATEMENTS.                                 ********


      SUBROUTINE EQUATE
C
C     THIS SUBROUTINE IS CALLED FROM MAIN 2 TO EVALUATE THE DERIVATIVES OF THE
C     VARIABLES OF INTEGRATION.  EITHER RECTANGULAR COORDINATES OR ORBIT ELE-
C     MENTS MAY BE USED AS THE VARIABLES OF INTEGRATION, BUT IN THE CASE OF THE
C     LATTER, THE CORRESPONDING RECTANGULAR COORDINATES MUST FIRST BE FOUND.
C     THIS IS DONE AT THE BEGINNING THRU THE USE OF KEPLERS EQUATION.  THE
C     PERTURBATING ACCELERATIONS ARE FOUND BY CALLING VARIOUS OTHER SUBROUTINES
C     AND THEIR SUM RESOLVED ALONG THE X,Y,Z AXIS.  FINALLY, THE DERIVATIVES
C     ARE CALCULATED.  IN THE CASE OF ORBIT ELEMENTS, THE X,Y,Z PERTURBATING
C     ACCELERATION COMPONENTS MUST FIRST BE RESOLVED INTO CIRCUMFERENTIAL,RADIAL
C     AND NORMAL COMPONENTS.  THIS ROUTINE ALSO CHANGES THE INTEGRATION VARI-
C     ABLES FROM ORBIT ELEMENTS TO RECTANGULAR VARIABLES IF THE ECCENTRICITY
C     APPROACHES UNITY.
C
      COMMON C
C
      DIMENSION
     1        C (1),              VX (3),              QX (3),
     2       RB (3),          NEFMRS (8),               X (3),
     3   XPRIMB (15,2),        FORCE (3),            XIFT (3),
     4     DRAG (3),           OBLAT (3),           COMPA (3),
     5      XDD (6),          XDOTTR (6),          XPRIM(15,2)
C
```

```
      EQUIVALENCE
     1(    DM,C(161)),(   DMA,C(166)),(     P,C(137)),(  DRAG,C(531)),
     2( ASQRD,C(563)),(     E,C(132)),(   PHI,C(485)),(TRSFER,C(224)),
     3(NSTART,C(247)),( PRESS,C(466)),(    QX,C(522)),(RATMOS,C(248)),
     4( CINCL,C(495)),( EMONE,C(243)),(     R,C(442)),( TTEST,C(251)),
     5(CIRCUM,C(541)),(  EPAR,C(245)),(RADIAL,C(540)),( ZNODE,C(134)),
     6(  SIMP,C(  5)),(  ETOL,C( 25)),(OBLATN,C( 27)),(     V,C(475)),
     7( COMPA,C(537)),(EXMODE,C(244)),(    RB,C(200)),( VSQRD,C(476)),
     8( BNAME,C(402)),( FORCE,C(525)),( TOFFT,C( 32)),(    VX,C(472)),
     9(ZORMAL,C(542)),(   GKM,C(470)),( RMASS,C(131)),(     X,C(135))
      EQUIVALENCE
     1(ASYMPT,C(543)),(  GK2M,C(469)),( RSQRD,C(567)),(   XDD,C(162)),
     2(CONSTU,C( 18)),( IMODE,C( 28)),( SINCL,C(494)),(XDOTTR,C(132)),
     3(COSTRU,C(493)),(  KSUB,C(254)),(  SINV,C(496)),(  XIFT,C(528)),
     4(  COSV,C(497)),(SINTRU,C(492)),(   SPD,C(253)),( XPRIM,C( 41)),
     5(    DE,C(162)),(MBODYS,C(441)),(    DP,C(167)),(XPRIMB,C( 71)),
     6(    ZN,C(487)),(OMEGA ,C(133)),( TABLT,C(252)),(XWHOLE,C(544)),
     7( DINCL,C(165)),(NEFMRS,C(433)),(  PUSH,C( 34)),( ZINCL,C(135)),
     8( DNODE,C(164)),( OBLAT,C(534)),(  FLOW,C( 33)),(    ZM,C(136)),
     9(DOMEGA,C(163)),(OBLATJ,C( 38)),(  TIME,C(138)),( AEXIT,C( 24))
C
      TABLT=TIME/SPD+TOFFT
S     LXD IMODE,(IMODE)
    1 GO TO (2,16,16),IMODE
C
C     STATEMENTS 2 TO 16 FIND THE RECTANGULAR POSITION AND VELOCITY FROM ORBIT
C     ELEMENTS AND TRUE ANOMALY.  THE TRUE ANOMALY IS FOUND FROM ITERATIVE
C     SOLUTION OF KEPLERS EQUATION.
    2 E2 = E*E
      E2M1=1.-E2
      EMONE=E-1.
      EPAR=SQRTF(ABSF(E2M1))
      VCIRCL=GKM/SQRTF(P)
C
C     COMPUTE SINE AND COSINE OF TRUE ANOMALY.
C     PART A. E=1
    3 IF (EMONE) 10,4,5
    4 SINTRU = 0.
      COSTRU = 1.
      GO TO 14
C
C     PART B. E IS GREATER THAN 1
    5 DO 7 J=1,100
      DELM=ZM-U+E*SINHF(U)
      ECOSU=E*COSHF(U)
      DELU = DELM/(1.0-ECOSU)
      U = U+DELU
    6 IF (ABSF(DELM)-CONSTU) 9,9,7
    7 CONTINUE
      ASYMPT = 1.0
      IF (MBODYS) 8,23,8
    8 CALL EPHMRS
      GO TO 23
    9 COSU = COSHF(U)
      DEM1 = 1.0-E*COSU
      COSTRU = (COSU-E)/DEM1
      SINTRU =-EPAR*SINHF(U)/DEM1
      GO TO 14
C
C     PART C. E IS LESS THAN 1
   10 DO 12 J=1,5
      DELM=ZM-U+E*SINF(U)
      ECOSU = E*COSF(U)
      DELU = DELM/(1.0-ECOSU+0.01*ECOSU**3)
      U = U+DELU
   11 IF (ABSF(DELM)-CONSTU) 13,13,12
   12 CONTINUE
   13 COSU = COSF(U)
      DEM1 = 1.0-E*COSU
      COSTRU = (COSU-E)/DEM1
      SINTRU = EPAR*SINF(U)/DEM1
   14 PDVR = 1.+E*COSTRU
C
C     COMPUTE POSITION AND VELOCITY FROM ORBIT ELEMENTS AND TRUE ANOMALY.
C     ALSO, CLEAR THE PERTURBATING ACCELERATIONS.
   15 SOMEGA=SINF(OMEGA)
      COMEGA=COSF(OMEGA)
      SNODE=SINF(ZNODE)
      CNODE=COSF(ZNODE)
      SINCL=SINF(ZINCL)
      CINCL=COSF(ZINCL)
      SINV=SINTRU*COMEGA+COSTRU*SOMEGA
      COSV=COSTRU*COMEGA-SINTRU*SOMEGA
      AR=COSV*CNODE-SINV*SNODE*CINCL
```

```
      B1=SINV*CNODE+COSV*SNODE*CINCL
      C1=COSV*SNODE+SINV*CNODE*CINCL
      D1=SINV*SNODE-COSV*CNODE*CINCL
      E1=E*SOMEGA+SINV
      F1=E*COMEGA+COSV
      AS=E1*CNODE+F1*SNODE*CINCL
      B2=F1*CNODE*CINCL-E1*SNODE
      R = P/PDVR
      RSQRD = R*R
      SINVY=SINV*SINCL
      RB(1)    = R*AR
      RB(2)    = R*C1
      RB(3)    = R*SINVY
      VX(1)=-VCIRCL*AS
      VX(2)=VCIRCL*B2
      VX(3)=VCIRCL*F1*SINCL
      GO TO 18
C
   16 DO 17 K=1,3
      VX(K)=XDOTTR(K)
   17 RB(K) = X(K)
      RSQRD = RB(1)*RB(1) + RB(2)*RB(2) + RB(3)*RB(3)
      R=SQRTF(RSQRD)
   18 VSQRD=VX(1)*VX(1)+VX(2)*VX(2)+VX(3)*VX(3)
      V = SQRTF(VSQRD)
      DO 19 I=1,15
   19 C(I+521) = 0.
C
C     TEST FOR PRESENCE OF PERTURBING BODIES.
      IF (MBODYS) 20,21,20
   20 CALL EPHMRS
   21 IF (XABSF(IMODE)-1) 26,22,26
C
C     TEST FOR CHANGE FROM ORBIT ELEMENTS TO TEMPORARY RECTANGULAR
C     COORDINATES IF E IS TOO NEAR TO UNITY.
   22 IF (ETOL-ABSF(EMONE)) 26,23,23
   23 IF (IMODE) 54,24,24
   24 IMODE=-3
      IF (NSTART) 25,54,25
   25 TTEST=TIME
      CALL TESTTR
C
C     TEST FOR OBLATENESS PERTURBATION COMPUTATION.
S  26  CLA OBLATN
S       CAS BNAME
S       TRA *30
S       TRA *29
S       TRA *30
   29 CALL OBLATE
C
C     TEST FOR PRESENCE OF THRUST.  COMPUTE THRUST MAGNITUDE IF NOT SPECIFIED.
   30 DM = -FLOW
      IF (R-RATMOS) 31,31,32
   31 CALL ICAO
      GO TO 33
   32 PRESS=0.
   33 IF(SIMP) 34,35,34
   34 PUSH = SIMP*FLOW*9.80665 - AEXIT*PRESS*100.
   35 IF(PUSH) 37,36,37
   36 ASSIGN 40 TO NDONE
      GO TO 38
   37 CALL THRUST
      ASSIGN 41 TO NDONE
C
C     TEST FOR EXISTENCE OF ATMOSPHERE.  FIND AERODYNAMIC FORCES.
   38 IF (PRESS ) 39,42,39
   39 GO TO NDONE, (40,41)
   40 CALL THRUST
   41 CALL AERO
C
C     SUM COMPONENTS OF THE PERTURBING ACCELERATION.
   42 DO 43 J=1,3
   43 COMPA(J) = -QX(J)+OBLAT(J)+FORCE(J)+XIFT(J)+DRAG(J)
   44 GO TO (47,45,45),IMODE
C
C     COMPUTE DERIVATIVES FOR THE RECTANGULAR VARIABLES OF INTEGRATION.
   45 DO 46 K=1,3
      XDD(K) = COMPA(K)-GK2M*X(K)/R/RSQRD
   46 XDD(K+3) = XDOTTR(K)
      GO TO 54
C
```

```
C     COMPUTE THE DERIVATIVES OF THE ORBIT ELEMENTS.  (AFTER RESOLVING
C     PERTURBATING ACCELERATION INTO CIRCUMFERENTIAL, RADIAL, NORMAL COMPONENTS)
   47 CIRCUM=COMPA(3)*COSV*SINCL-COMPA(1)*B1-COMPA(2)*D1
      RADIAL=COMPA(1)*AR+COMPA(2)*C1+COMPA(3)*SINVY
      ZORMAL=COMPA(1)*SNODE*SINCL-COMPA(2)*CNODE*SINCL+COMPA(3)*CINCL
      ZN=VCIRCL*E2M1*EPAR/P
      RDVPP1 = 1./PDVR + 1.
      RDVA = E2M1/PDVR
      DP=2.*R/VCIRCL*CIRCUM
      IF(E) 48,48,49
   48 CSQRD  = CIRCUM*CIRCUM
      RASQRD = RADIAL*RADIAL
      DEM1 = (4.*CSQRD+RASQRD)*VCIRCL
      VDV2R=VCIRCL/R/2.
      DE = SQRTF(4.*CSQRD+RASQRD)/VCIRCL
      DOMEGA = VDV2R+(2.*CSQRD+RASQRD)/DEM1*RADIAL
      DMA = ZN-VDV2R+(6.*CSQRD+RASQRD)/DEM1*RADIAL
      GO TO 50
   49 DE = (SINTRU*RADIAL+(PDVR-RDVA)/E*CIRCUM)/VCIRCL
      DOMEGA=(SINTRU/E*RDVPP1*CIRCUM-COSTRU*RADIAL/E)/VCIRCL
      DMA=ZN+EPAR/VCIRCL*((COSTRU/E-2./PDVR)*RADIAL-(SINTRU/E*RDVPP1*CIR
     1 CUM))
   50 IF(SINCL) 51,52,51
   51 DNODE = SINV/SINCL*ZORMAL/VCIRCL/PDVR
      GO TO 53
   52 DNODE = 0.0
   53 DINCL = COSV*ZORMAL/PDVR/VCIRCL
   54 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                   ********


      SUBROUTINE ERRORZ
C
C    THIS SUBROUTINE COMPUTES THE RELATIVE ERRORS BETWEEN THE R-K AND LOW-ORDER
C    INTEGRATION SCHEMES.  IT ALSO COMPUTES THE ERROR COEFFICIENT, A, AND SAVES
C    THE ERROR DATA WHEN EREF HAS A - SIGN.  THE BRANCH ON IMODE DETERMINES
C    WHICH SET OF NORMALIZING FACTORS ARE TO BE USED.
C
      COMMON C
C
      DIMENSION RELERR(7)
C
      EQUIVALENCE
     1( RMASS,C( 56)),(      E,C( 57)),(     AS,C(151)),(OMEGAS,C(148)),
     2(RMASSS,C(146)),(      P,C( 62)),(     ES,C(147)),(ZNODES,C(149)),
     3(      R,C(442)),(     PS,C(152)),(ZINCLS,C(150)),(XINC  ,C(146)),
     4(      V,C(475)),( IMODE,C( 28)),(  TIME,C(138)),(     E2,C(260)),
     5(     VX,C(147)),(     VY,C(148)),(     VZ,C(149)),(      X,C(150)),
     6(      Y,C(151)),(      Z,C(152)),(RELERR,C(146)),(     A2,C(237)),
     7(  DELT,C( 10)),(     A1,C(236)),(  EREF,C( 37)),(STEPGO,C(101)),
     8(STEPNO,C(102)),(INDERR,C(491))
C
      E2 = 0.
      RELERR(1)=RMASSS/RMASS
      IF (IMODE-1) 2,1,2
C
C     COMPUTE THE NORMALIZED INTEGRATION ERRORS FOR THE ORBIT ELEMENTS.
    1 RELERR(2)=ES/(E+1.0)/10.0
      RELERR(3)=OMEGAS/62.831853
      RELERR(4)=ZNODES/62.831853
      RELERR(5)=ZINCLS/62.831853
      RELERR(6)= AS/62.831853
      RELERR(7)=PS/P/10.0
      GO TO 3
C
C     COMPUTE THE NORMALIZED INTEGRATION ERRORS IN RECTANGULAR VARIABLES.
    2 V1 = V+100.
      RELERR(2)=VX/V1
      RELERR(3)=VY/V1
      RELERR(4)=VZ/V1
      RELERR(5)=X/R
      RELERR(6)=Y/R
      RELERR(7)=Z/R
C
C     SELECT MAXIMUM ERROR, COMPUTE ERROR COEFFICIENT, POSSIBLY SAVE ERROR DATA.
    3 DO 5 J=1,7
      IF (ABSF(RELERR(J))-E2) 5,5,4
    4 K=J
      E2 = ABSF(RELERR(J))
    5 CONTINUE
      E2 = E2 + 2E-8
      A1 = A2
      A2 = LOGF(E2)-5.*LOGF(ABSF(DELT))
      IF (EREF) 6,7,7
    6 WRITE TAPE 4,K,RELERR,E2,A2,DELT,TIME,STEPNO,STEPGO
      INDERR = INDERR + 1
    7 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                   ********
```

```
      SUBROUTINE STEP
C
C   SUBROUTINE STEP TESTS FOR THE END OF THE PROBLEM, COMPUTES STEP SIZE, AND
C   CONTROLS QUANTITY OF OUTPUT DATA.  WHEN END OF PROBLEM IS DETECTED, OUTPUT
C   OCCURS, THE ERROR DATA TAPE IS REWOUND, AND THE FIRST SEGMENT IS CALLED TO
C   ALLOW INPUT.  FOLLOWING IS AN EXPLANATION OF CONTROL ON QUANITY OF OUTPUT.
C
C     MODOUT=1  OUTPUT EVERY NTH STEP(N=STEPS) UNTIL TIME = TMIN, THEN
C                   GO TO MODE 2 .
C            2  OUTPUT AT INTERVALS OF DELMAX UNTIL TIME = TMAX.
C            3  OUTPUT AT INTERVALS OF DELMAX UNTIL TIME = TMIN, THEN
C                   GO TO MODE 4 .
C            4  OUTPUT EVERY NTH STEP UNTIL TIME = TMAX.
C
      COMMON C
C
      DIMENSION      NPONG(5)
C
      EQUIVALENCE
     1(  DELT,C( 10)),(    E2,C(260)),( NPONG,C( 11)),(    A1,C(236)),
     2(   DEL,C(255)),( ERLOG,C(259)),(  TIME,C(138)),(  TMIN,C( 22)),
     3(DELMAX,C( 23)),(STEPNO,C(102)),( STEPS,C( 21)),(SPACES,C(258)),
     4(STEPMX,C( 20)),(STEPGO,C(101)),(  TMAX,C( 30)),(    H2,C(256)),
     5(MODOUT,C(103)),(    A2,C(237)),( RATIO,C(600)),(  TTOL,C(226))
C
C     PART 1.  TEST FOR END OF THE PROBLEM (MAXIMUM PROBLEM TIME OR MAXIMUM
C              NUMBER OF STEPS).
      STEPGO = STEPGO + 1.
      IF (ABSF(TMAX-TIME)-TTOL) 1,1,3
    1 CALL OUTPUT
      WRITE OUTPUT TAPE 6,2
      PRINT 2
    2 FORMAT(25HOCASE COMPLETED,TIME=TMAX)
      GO TO 6
    3 IF (STEPGO+STEPNO-STEPMX) 7,4,4
    4 CALL OUTPUT
      WRITE OUTPUT TAPE 6,5,STEPMX
    5 FORMAT (22HOSTEPGO+STEPNO=STEPMX=F6.)
    6 REWIND 4
      CALL PONG(NPONG(5))
C
C     PART 2.  COMPUTE STEP SIZE (DELT) AND CONTROL OUTPUT.
    7 A3 = (A2-A1)*RATIO+A2
    8 DELT = SIGNF(EXPF((ERLOG-A3)/5.),DELT)
      IF (DELT/H2-3.) 10,10,9
    9 DELT = 3.*H2
S  10  LXD MODOUT,(MODOUT)
      GO TO (11,15,13,21),MODOUT
   11 IF(DELT*(TIME + 3.*DELT-TMIN)) 21,12,12
   12 MODOUT = 2
      DEL = TMIN - TIME
      GO TO 16
   13 IF(DELT * (TIME - TMIN)) 15,15,14
   14 MODOUT = 4
      GO TO 21
   15 DEL = DEL-H2
   16 SPACES = INTF((DEL/DELT)+SIGNF(.9,(DEL/DELT)))
   17 IF(SPACES) 20, 18,20
   18 CALL OUTPUT
      DEL = DELMAX
      IF (ABSF(DEL) - ABSF(DELT)) 19,16,16
   19 DELT = SIGNF(DEL,DELT)
      GO TO 16
   20 DELT = DEL/SPACES
      GO TO 23
   21 IF (MODF(STEPGO,STEPS)) 23,22,23
   22 CALL OUTPUT
   23 GO TO (26,24,26,24),MODOUT
   24 IF((TIME + DELT - TMAX)*DELT) 26,25,25
   25 DELT = TMAX-TIME
   26 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                 ********


      SUBROUTINE TESTTR
C
C   SUBROUTINE TESTTR MAY BE CALLED FOR ONE OF TWO REASONS, (1) TO TEST FOR AND
C   POSSIBLY TRANSLATE THE ORIGIN (WHEN IMODE IS +) OR (2) TO CHANGE THE
C   VARIABLES OF INTEGRATION (WHEN IMODE IS -).  A TRANSLATION OF THE ORIGIN
C   OCCURS WHEN THE OBJECT MOVES INTO A SPHERE OF INFLUENCE WHICH IS SMALLER
C   THAN ANY OTHERS IT MAY ALSO BE IN.  WHEN THIS HAPPENS, THE NAME OF THE NEW
C   ORIGIN IS MOVED TO THE BEGINNING OF THE BNAME LIST AND THE FIRST SEGMENT
C   CALLED TO REORDER THE BNAME LIST.
C
      COMMON C
C
```

```
      DIMENSION   BMASS(8), BNAME(8), RB(3,8), RBCRIT(8), RREL(8), C(1),
     1X(3),XPRIM(15,2),XPRIMB(15,2),XWHOLE(6),VEFM(3,8), NPONG(5),
     2VX(3),ORBELS(6)
C
      EQUIVALENCE
     1( BMASS,C(417)),( BNAME,C(402)),( CHAMP,C(246)),( NPONG,C( 11)),
     2(  GK2M,C(469)),( IMODE,C( 28)),(NBODYS,C(489)),(STEPNO,C(257)),
     3(    RB,C(200)),(RBCRIT,C(450)),(  RREL,C(442)),( SQRDK,C(468)),
     4(ORBELS,C(227)),(TRSFER,C(224)),(     X,C(200)),( XPRIM,C( 41)),
     5(XPRIMB,C( 71)),(XWHOLE,C(544)),( TTEST,C(251)),(  VEFM,C(498)),
     6(    VX,C(472)),(  REVS,C(490)),(  DELT,C( 10)),(  TMAX,C( 30)),
     7(  TIME,C(138)),(     E,C(227)),(   TRU,C(483)),(ASYMPT,C(543))
C
S     LXD IMODE,(IMODE)
      IF (IMODE) 12,12,1
C
C     IF IMODE IS +, TEST FOR TRANSLATIuN OF THE ORIGIN.
    1 ASSIGN 27 TO N
      CHAMP= 1.E+30
      DO 4 JB=1,NBODYS
      IF (RREL(JB)-RBCRIT(JB)) 2,4,4
    2 IF (CHAMP-RBCRIT(JB)) 4,4,3
    3 CHAMP = RBCRIT(JB)
      NCHAMP = JB
    4 CONTINUE
      IF (NCHAMP-1) 26,26,5
    5 TRSFER = 1.0
      ASSIGN 29 TO N
    8 BTEMP = BNAME(1)
      BNAME(1) = BNAME(NCHAMP)
      BNAME(NCHAMP) = BTEMP
      TTEST = 0.
      REVS = 0.
    9 PRINT 10, BNAME(NCHAMP),BNAME(1)
      WRITE OUTPUT TAPE 6,10,BNAME(NCHAMP),BNAME(1)
   10 FORMAT (28HOORIGIN IS TRANSLATING FROM A6,4H TO A6)
      CALL EPHMRS
      DO 11 K=1,3
      VX(K) = VX(K)-VEFM(K,NCHAMP)
      X(K) = RB(K,NCHAMP)
      XPRIM(K+1,1)=VX(K)
      XPRIM(K+4,1)=X(K)
      XPRIMB(K+1,1) = 0.
      XPRIMB(K+4,1) = 0.
      XWHOLE(K)= VX(K)
   11 XWHOLE(K+3) = X(K)
      GO TO 20
C
C     IF IMODE IS -, CHANGE THE VARIABLES OF INTEGRATION.
   12 ASSIGN 28 TO N
      DO 13 K=1,3
      XPRIM(K+1,1)=XWHOLE(K)
      XPRIM(K+4,1)=XWHOLE(K+3)
      XPRIMB(K+1,1) = 0.
      XPRIMB(K+4,1) = 0.
      VX(K) = XWHOLE(K)
   13 X(K) = XWHOLE(K+3)
      GO TO (16,14,15),IMODE
   14 CODE = 5HORBIT
      IMODE = 1
      GO TO 18
   15 IMODE = 3
      GO TO 17
   16 IMODE = 2
   17 CODE = 6HRECTAN
   18 NCHAMP = 1
      PRINT 19, CODE
      WRITE OUTPUT TAPE 6,19,CODE
   19 FORMAT (33HOINTEGRATION MODE IS CHANGING TO A6)
   20 GO TO (21,26,26),IMODE
   21 CALL CONVT1(VX,C(559))
      GK2M= SQRDK*(BMASS(NCHAMP)+XPRIM(1,1)/1.9866 E+30)
      CALL CONVT2
C     IF ORIGIN TRANSLATION CAUSES PATH TO LIE NEAR AN ASYMPTOTE, CHANGE
C     INTEGRATION VARIABLES TO RECTANGULAR IF THEY ARE ORBIT ELEMENTS.
      IF (E-1.) 24,24,22
   22 IF (ABSF(TRU)-2.3/SQRTF(E)) 24,24,23
   23 ASYMPT = 1.0
      GO TO 15
   24 DO 25 J=1,6
   25 XPRIM(J+1,1) = ORBELS(J)
   26 GO TO N,(27,28,29)
   27 RETURN
   28 CALL PONG (NPONG(1))
   29 CALL PONG (NPONG(5))
C
C     END OF THE FORTRAN STATEMENTS.                              ********
```

```
      SUBROUTINE ICAO
C
C   SUBROUTINE ICAO DETERMINES THE ATMOSPHERIC TEMPERATURE, PRESSURE, AND
C   DENSITY AS A FUNCTION OF ALTITUDE ABOVE AN OBLATE EARTH IN ACCORDANCE WITH
C   NACA 1235 AND U.S. EXTENSION TO THE ICAO STANDARD ATMOSPHERE.  A SHORT SAP
C   PROGRAM FOLLOWS ICAO WHICH PROVIDES A MEANS OF LOADING DATA INTO MACHINE.
C   IT MUST BE LOADED DIRECTLY AFTER ICAO.  IF THE LENGTH OF ICAO IS CHANGED,
C   THE DATA MUST BE RELOCATED.
C
C            R IS DISTANCE TO CENTER OF EARTH IN METERS.
C          ALT IS VEHICLE ALTITUDE ABOVE AN ELLIPTIC EARTH IN METERS.
C        GEO H IS THE GRAVITATIONAL POTENTIAL IN METERS.
C      TABLE H IS METERS OF ALTITUDE FROM THE EARTHS SURFACE AND IS
C                THE ARGUMENT OF ATMOSPHERE PROPERTY TABLE.
C          ALM IS THE MEAN SLOPE OF THE TABLE H VS. TM CURVE AT TABLE H.
C          TMR IS TM AT TABLE H.
C        REF P IS THE PRESSURE IN MILLIBARS AT TABLE H.
C           TM IS THE TEMPERATURE TIMES STD. MOLECULAR WEIGHT / ACTUAL
C                MOLECULAR WEIGHT.  DEGREES KELVIN.
C        PRESS IS PRESSURE IN  MILLIBARS.
C       DNSITY IS DENSITY IN  KILOGRAMS PER CUBIC METER.
C
      COMMON C
C
      DIMENSION  TABLE H(11),TMR(11), REF P(11),ALM(11)
C
      EQUIVALENCE
     2( GEO H,C(465)),( PRESS,C(466)),(    TM,C(467)),(DNSITY,C(460)),
     3( TABLT,C(252)),(   ALT,C(463)),(     R,C(442)),(     Z,C(137)),
     4(TABLE H(12),TMR),(TABLE H(23),ALM),(TABLE H(34),REF P)
C
      ALT = R-6356783.28/SQRTF(.9933065783+.006693421685(Z/R)**2)
      GEO H = ALT/(1.0 + ALT/6356766.0)
C
C     FIND THE GEOPOTENTIAL HEIGHT IN A TABLE OF BASE DATA. DATA ARE
C     ARRANGED IN DECENDING GEO H WITH TEN REGIONS, AN 11TH IS GIVEN
C     FOR EXTRAPOLATION.   ABOVE THAT, PRESSURE AND DENSITY ARE SET =0.
S     LXD K,(K)
    1 IF (K-11) 2,6,6
    2 IF (GEO H - TABLE H(K+1)) 5,3,3
    3 K = K+1
      GO TO 1
    4 K = K-1
    5 IF (K) 7,7,6
    6 H INC = GEO H -TABLE H(K)
      IF (H INC) 4,8,8
    7 K = 1
    8 GO TO (9,11,9,11,9,11,9,9,9,9,12),K
C
C     CONTROL COMES HERE FOR NONISOTHERMAL LAYERS
    9 TM = TMR(K) + ALM(K)*H INC
      PRESS= REF P(K)*(EXPF((.03416475/ALM(K))*LOGF(TMR(K)/TM)))
   10 DNSITY = PRESS/(2.8704*TM)
      GO TO 13
C
C     CONTROL COMES HERE FOR ISOTHERMAL LAYERS
   11 TM = TMR(K)
      PRESS= REF P(K)*EXPF(-0.03416475*H INC/TMR(K))
      GO TO 10
C
C     CONTROL COMES HERE FOR EXTREME ALTITUDES
   12 PRESS = 0.0
      DNSITY = 0.0
      TM = 2000.
   13 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                               ********


        REM   THIS IS THE SAP PROGRAM WHICH LOADS ICAO DATA INTO MACHINE.
        REM THE 170 IN ORG 170 WAS FOUND BY SUBTRACTING 10 FROM THE DEC LOCATION
        REM OF REF P (FROM SAP LISTING OF ICAO, THIS WAS FOUND TO BE 180).
        REM THUS, 180-10=170.
        REM
        REM   A1 IS REF P(11)
        REM   A2 IS ALM(11)
        REM   A3 IS TMR(11)
        REM   A4 IS TABLE H(11)
        REM
        ORG 170
        REL
A1      DEC 1.01E-8,1.477E-8,6.19E-7,1.451E-5,1.815E-3,2.452E-2,5.832E-1
        DEC 1.2044,24.886,226.32,1013.25
A2      DEC 0.0,0.0005,0.0058,0.01,0.035,0.0,-0.0039,0.0,0.003,0.0
        DEC -0.0065
A3      DEC 0.0,1537.86,812.86,322.86,196.86,196.86,282.66,282.66,216.66
        DEC 216.66,288.16
A4      DEC 3000000.0,300000.0,175000.0,126000.0,90000.0,75000.0,53000.0
        DEC 47000.0,25000.0,11000.0,0.0
        REM END OF THE SAP STATEMENTS.                             ********
        END                                                               1
```

```
      SUBROUTINE THRUST
C
C     THIS ROUTINE COMPUTES X,Y,Z THRUST ACCELERATIONS.  THE THRUST VECTOR IS
C     ASSUMED COINCIDENT WITH THE LONGITUNDINAL AXIS OF THE VEHICLE, WHICH IS
C     ORIENTED TO THE RELATIVE WIND VELOCITY BY THE ANGLE OF ATTACK (ALPHA) AND
C     THE ROLL ANGLE (BETA). ALPHA IS ASSUMED TO BE A QUADRATIC FUNCTION OF TIME
C     WHEREAS BETA IS ASSUMED TO BE CONSTANT.
C     REVOLV IS THE EARTHS ROTATION RATE IN RADIANS/SEC (7.29211585E-5) AND THE
C     FACTOR 8589934592.= 2**33 IS REMOVED TO PREVENT OVERFLOW.
C
      COMMON C
C
      DIMENSION  FORCE (3), PAR(3), C(30),VATM(3),P(3),AQ(5),IND(3)
C
      EQUIVALENCE      (  SIMP,C(  5)),(  FLOW,C( 33)),( FORCE,C(525)),
     2( RMASS,C(131)),(   PAR,C(798)),( RSQRD,C(567)),(COSBET,C(599)),
     3(    VX,C(472)),(   IND,C(791)),(     X,C(200)),(SINALF,C(569)),
     4(    VY,C(473)),(  TIME,C(138)),(     Y,C(201)),(SINBET,C(568)),
     5(    VZ,C(474)),(COSALF,C(575)),(     Z,C(202)),(REVOLV,C(250)),
     6(ALPHA ,C(564)),( PMAGN,C(574)),(    P ,C(571)),(RATMOS,C(248)),
     7(  BETA,C(565)),(VQSQRD,C(481)),(     R,C(442)),(  VATM,C(477)),
     8(    VQ,C(480)),(  PUSH,C( 34))
C
      SINBET = SINF(BETA)
      COSBET = COSF(BETA)
      VATM(1)=VX+REVOLV*Y
      VATM(2)=VY-REVOLV*X
      VATM(3)=VZ
      CALL CONVT1(VATM,AQ)
      ALPHA = QUAD(TIME,1)/57.29577951
      SINALF=SINF(ALPHA)
      COSALF=COSF(ALPHA)
      DO 1 J1=1,3
      J2=IND(J1)
      J3=IND(J2)
    1 P(J1) = (VATM(J2)*AQ(J3)-VATM(J3)*AQ(J2))/8589934592.
      PMAGN= SQRTF(P(1)*P(1)+P(2)*P(2)+P(3)*P(3))
      TDPMAG = PUSH/RMASS/PMAGN
      R4   =  SINBET/VQ
      R5   =  COSALF/AQ(4)
      DO 2 J1=1,3
      J2=IND(J1)
      J3=IND(J2)
      PAR(J1)=P(J2)*VATM(J3)-P(J3)*VATM(J2)
    2 FORCE(J1) = TDPMAG*(SINALF*(COSBET*P(J1)+R4*PAR(J1))-R5*(P(J2)*AQ
     1             (J3)-P(J3)*AQ(J2)))
      RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                   ********


      SUBROUTINE AERO
C
C     SUBROUTINE AERO COMPUTES THE LIFT AND DRAG ACCELERATIONS.  AS IN SUBROUT-
C     INE THRUST, THESE VECTORS ARE REFERENCED TO THE RELATIVE WIND VELOCITY.
C     COEFFICIENTS OF LIFT, INDUCED DRAG, AND DRAG AT ZERO ANGLE OF ATTACK ARE
C     ASSUMED TO BE FUNCTIONS OF MACH NUMBER AND ANGLE OF ATTACK.  TABLES OF
C     CDI/CL**2, CL/SIN(ALPHA), AND CDO ARE ASSUMED AS FITTED QUADRATIC EQUAT-
C     IONS IN THE COEFN ARRAY.  GASFAC IS THE SQRTF(SPECIFIC HEAT RATIO * STAND-
C     ARD ACCELERATION OF GRAVITY * UNIVERSAL GAS CONSTANT).  FOR EARTH, GASFAC=
C     20.064881 (METERS / SEC / KELVIN DEGREE).
C
      COMMON C
C
      DIMENSION C(1),VATM(3),P(3),XIFT(3),DRAG(3),PAR(3)
C
      EQUIVALENCE
     1(  QVAL,C(794)),(  AREA,C( 35)),(  TIME,C(138)),(DNSITY,C(460)),
     2(  BETA,C(565)),( PMAGN,C(574)),(    TM,C(467)),(SINALF,C(569)),
     3(  PHIP,C(462)),(VQSQRD,C(481)),(    VQ,C(480)),(SINBET,C(568)),
     4(  XIFT,C(528)),( RMASS,C(131)),(     R,C(442)),(   CDI,C(795)),
     5(  DRAG,C(531)),(  VATM,C(477)),(     P,C(571)),(GASFAC,C(458)),
     6(COSALF,C(575)),( VMACH,C(471)),( ALPHA,C(564)),(COSBET,C(599)),
     7(   PAR,C(798)),(    CD,C(797)),(    CL,C(796))
C
      QVAL=0.5*DNSITY*VQSQRD*AREA/RMASS
      VMACH=SQRTF(VQSQRD/TM)/GASFAC
C
C     COMPUTE THE X,Y,Z COMPONENTS OF LIFT.
      IF (ALPHA) 2,1,2
    1 CL = 0.0
      CDI=0.0
      XIFT(1) = 0.
      XIFT(2) = 0.
      XIFT(3) = 0.
      GO TO 4
    2 CL = QUAD(VMACH,2)*SINALF
      DO 3 K=1,3
    3 XIFT(K) = QVAL*CL/PMAGN*(SINBET*PAR(K)/VQ+COSBET*P(K))
      CDI=QUAD(VMACH,3)*CL*CL
C
```

```
C       COMPUTE THE X,Y,Z COMPONENTS OF DRAG.
      4 CD = CDI+QUAD(VMACH,4)
        DO 5 K=1,3
      5 DRAG(K) = -CD*QVAL*VATM(K)/VQ
        RETURN
C
C       END OF THE FORTRAN STATEMENTS.                                  ********


        SUBROUTINE OBLATE
C
C     THIS SUBROUTINE COMPUTES THE OBLATENESS ACCELERATIONS (OBLAT) DUE TO AN
C     AXIALLY SYMMETRIC EARTH.  THE 2ND AND 4TH SPHERICAL HARMONIC COEFF. ARE
C     OBLATJ AND OBLATK, RESPECTIVELY.  OBLATJ, OBLATK, RESQRD, AND THE CONSTANTS
C     CON ARE LOADED BY STDATA.
C
        COMMON C
C
        DIMENSION RB(3), OBLAT(3), CON(9)
C
        EQUIVALENCE
       1(    CON,C(576)),(      R,C(442)),(  GK2M,C(469)),( RSQRD,C(567)),
       2(RESQRD,C( 40)),(OBLATJ,C( 38)),(OBLATK,C( 39)),(     RB,C(200)),
       3( OBLAT,C(534))
C
        Z2DVR2=RB(3)*RB(3)/RSQRD
        REDVR=RESQRD/RSQRD
        DO 1 K=1,3
      1 OBLAT(K)=RB(K)*REDVR*GK2M*5.0/R/RSQRD*(OBLATJ*(Z2DVR2-CON(K))+
       1            OBLATK*REDVR*(Z2DVR2*(CON(K+3)-2.1*Z2DVR2)-CON(K+6)))
        RETURN
C
C       END OF THE FORTRAN STATEMENTS.                                  ********


        SUBROUTINE EPHMRS
C
C     SUBROUTINE EPHMRS IS CALLED TO COMPUTE THE POSITIONS OF THE PERTURBING
C     BODIES RELATIVE TO THE VEHICLE AND, FROM THESE, THEIR PERTURBING ACCELERA-
C     TIONS UPON THE VEHICLE.  OCCASIONALLY THIS ROUTINE IS CALLED FOR THE PURPOSE
C     OF TRANSLATING THE ORIGIN IN WHICH CASE (TRSFER=1) THE RELATIVE VELOCITIES
C     ARE ALSO CALCULATED.  IF A BODYS POSITION IS TO BE COMPUTED FROM AN ELLIPTIC
C     APPROXIMATION SUBROUTINE ELIPSE IS CALLED. OTHERWISE, THE POSITION WILL BE
C     CALCULATED IN EPHMRS FROM THE PRECISION TAPE EPHEMERIS.  THE DO 19 LOOP
C     ENCOMPASSES ALMOST THE ENTIRE EPHMRS SUBROUTINE AND ,IN EFFECT, ELIPSE TOO.
C
        COMMON  C
C
        DIMENSION  QX(3),IBODY(8),EFMRS(7),XP(3,8),RB(3,8),RREL(8),NEFMRS
       1 (8),TDATA(18,7),CF(6,3,7),TIM(7),TDEL(7),BMASS(8),XDOT(3,8),C(1)
C
        EQUIVALENCE      (QX     ,C(522)),( IBODY,C(425)),(MBODYS,C(441)),
       1(EFMRS ,C(410)),(XP     ,C(176)),(RB     ,C(200)),(RREL   ,C(442)),
       2(NEFMRS,C(433)),(TRSFER,C(224)),(TABL T,C(252)),(DTOFFJ,C( 31)),
       3(TDATA ,C(276)),(CF     ,C(276)),(TIM    ,C(585)),(TDEL   ,C(592)),
       4(BMASS ,C(417)),(SQRDK ,C(468)),(XDOT   ,C(498)),(LENGTH,C(257)),
       5(    AU,C(461)),(   IBF,FIB)
C
C       PART 2.  SET INDEXS, FIND POSITION IF ELLIPSE IS USED (NEFMRS = 20 OR UP).
        DO 19 JB=1,MBODYS
        JB1 = JB+1
        IBF = IBODY(JB1)
        IB = XABSF(IBF)
        IF (NEFMRS(JB)-20) 2,2,1
      1 CALL ELIPSE (JB1)
        IF (TRSFER) 12,12,17
C
C       PART 3.  TAPE EPHEMERIS IS TO BE USED.  FIND DIFFERENCE (DT) BETWEEN
C       CURRENT PROBLEM TIME (DTOFFJ+TABLT) AND MIDPOINT TIME (TIM) OF CURRENTLY
C       STORED TAPE DATA.  THEN SEE IF CURRENT DATA IS OKAY.  TDEL = TIME INTERVAL
C       ON EITHER SIDE OF TIM FOR WHICH CURRENT DATA IS GOOD.
      2 DT = TABL T - (TIM(JB) -DTOFFJ)
        IF (ABSF(DT)-TDEL(JB)) 10,10,3
C
C       PART 4A.  CURRENT DATA NOT OKAY.  READ IN NEXT DATA SET.  IF DT IS -,
C       BACK UP THE TAPE 2 RECORDS BEFORE READING.
      3 IF (DT) 4,5,5
      4 BACKSPACE 3
        BACKSPACE 3
      5 READ TAPE 3, (C(J), J=8051,8071)
        LYE = 8051
C
```

```
C      PART 4B. IF THIS DATA IS FOR A BODY IN THE BNAME LIST, STORE IT.
C      (IF NOT STORED, WE MIGHT HAVE TO RETURN FOR IT.)  IF ELLIPSE DATA IS
C      PROVIDED FOR THE BODY FOUND, BY-PASS THE TAPE DATA AND READ IN NEXT SET.
       DO 7  J = 1,MBODYS
S      CLA C(LYE)
S      CAS EFMRS(J)
S       TRA *7
S       TRA *6
S       TRA *7
     6 IF (NEFMRS(J)-20) 8,8,3
     7 CONTINUE
       GO TO 3
C
C      PART 4C.  MOVE THE DATA INTO PLACE AND THEN GO BACK AND SEE IF IT IS OKAY.
     8 TIM(J) = C(LYE+1)
       TDEL(J) = C(LYE+2)
       DO 9 JJ=1,18
       TDATA(JJ,J) = C(JJ+8053)
     9 CONTINUE
       GO TO 2
C
C
C      PART 5.  CURRENT DATA IS OKAY.  GET POSITION FROM THE POLONOMIAL
C      P = A + BX + CX**2 + DX**3 + EX**4 + FX**5.
    10 DO 11 K=1,3
       XP(K,JB1) = CF(1,K,JB)
       DO 11 KT=2,6
       XP(K,JB1) = XP(K,JB1)* DT +CF(KT,K,JB)
    11 CONTINUE
       IF (TRSFER) 12,12,15
C
C      PART 6. COMPUTE DISTANCE FROM REFERENCE AND FROM ROCKET .
    12 DO 13 K=1,3
       XP(K,JB1) = XP(K,IB) +XP(K,JB1)*SIGNF(AU,FIB)
    13 RB(K,JB1)= RB(K,1) - XP(K,JB1)
C
C      PART 7. COMPUTE PERTJRBING ACCELERATIONS (QX).  4194304=2**22 IS REMOVED
C      TO PREVENT OVERFLOW.  2048=2**11 AND 8589934592=2**33 RESTORE THE SCALE.
       PRSQRD = (RB(1,JB1)**2 + RB(2,JB1)**2 + RB(3,JB1)**2)/4194304.
       RRELL = SQRTF(PRSQRD)
       RSQRD = ( XP(1,JB1)**2 + XP(2,JB1)**2 + XP(3,JB1)**2)/4194304.
       RCUBE = RSQRD * SQRTF(R SQRD)
       PRCUBE = PRSQRD * RRELL
       RREL(JB1) = RRELL* 2048.
       DO 14 K=1,3
    14 QX(K)=SQRDK * BMASS(JB1) * ((XP(K,JB1)/RCUBE) + RB(K,JB1)/PRCUBE)/
      1  8589934592. + QX(K)
       GO TO 19
C
C      PART 8.  COMPUTE VELOCITY FROM V = B + 2CX + 3DX**2 + 4EX**3 + 5FX**4
C      AND FROM REFERENCE BODY VELOCITY (XDOT(IB)).
    15 DO 16 K=1,3
       XDOT(K,JB1) = 0.
       DO 16 KT=1,5
    16 XDOT(K,JB1) = (XDOT(K,JB1) * DT + CF(KT,K,JB) *FLOATF(-KT+6) )
    17 DO 18 K=1,3
    18 XDOT(K,JB1) = XDOT(K,IB) + XDOT(K,JB1)*SIGNF(AU/86400.0,FIB)
       GO TO 12
    19 CONTINUE
       CALL DUMP (4,C,LENGTH)
       RETURN
C
C      END OF THE FORTRAN STATEMENTS.                                  ********


       SUBROUTINE ELIPSE (JB1)
C
C    THIS SUBROUTINE IS CALLED FROM EPHMRS TO COMPUTE THE POSITION OF A BODY
C    USING APPROXIMATE ELLIPTIC DATA.  THE VELOCITY IS ALSO COMPUTED IF THE
C    ORIGIN IS BEING TRANSLATED (TRSFER=1.0).  THE ELLIPSE DATA IS READ FROM
C    INPUT CARDS AND ORGANIZED IN SUBROUTINE ORDER. TPD IS TIME SINCE PERIHELION
C    PASSAGE, ZM IS MEAN ANOMALY, U IS ECCENTRIC ANOMALY, E IS ECCENTRICITY.
C
       COMMON C
C
       DIMENSION
      1        XP (3,8),          XDOT (3,8),              P (1),
      2         E (1),           SINCL (1),            SNODE (1),
      3   SOMEGA (1),           PPJD (1),            PPFRAC (1),
      4   PERIOD (1),           CINCL (1),            CNODE (1),
      5   COMEGA (1)
C
```

```
      EQUIVALENCE
     1(  XDOT,C(498)),(DTOFFJ,C( 31)),(COMEGA,C(284)),( CNODE,C(285)),
     2(     P,C(276)),(     E,C(277)),(SOMEGA,C(278)),( SNODE,C(279)),
     3( SINCL,C(280)),(  PPJD,C(281)),(PPFRAC,C(282)),(PERIOD,C(283)),
     4( CONSU,C( 36)),( TABLT,C(252)),(    XP,C(176)),(TRSFER,C(224)),
     5( CINCL,C(286))
C
      K = 18*(JB1-2)+1
      TPD = (DTOFFJ-PPJD(K))+(TABLT-PPFRAC(K))
      ZN = 6.28318533/PERIOD(K)
      ZM = ZN*MODF(TPD,PERIOD(K))
C
C     GET THE SINE(SINTRU) AND THE COSINE (COSTRU) OF THE TRUE ANOMALY
C     BY ITERATING KEPLERS EQUATION. THEN COMPUTE X,Y,Z (XP).
      U = ZM+E(K)*SINF(ZM)+0.5*E(K)**2*SINF(2.0*ZM)
      DO 1 J=1,10
      DELM = ZM-U+E(K)*SINF(U)
      DELU = DELM/(1.-E(K)*COSF(U))
      U = U+DELU
      IF (ABSF(DELM)-CONSU) 2,2,1
    1 CONTINUE
    2 COSU = COSF(U)
      DENOM = 1.-E(K)*COSU
      COSTRU = (COSU-E(K))/DENOM
      R = P(K)/(1.+E(K)*COSTRU)
      SINTRU=SQRTF(1.-E(K)**2)*SINF(U)/DENOM
      SINV = SINTRU*COMEGA(K)+COSTRU*SOMEGA(K)
      COSV = COSTRU*COMEGA(K)-SINTRU*SOMEGA(K)
      XP(1,JB1) = R*(COSV*CNODE(K)-SINV*SNODE(K)*CINCL(K))
      XP(2,JB1) = R*(COSV*SNODE(K)+SINV*CNODE(K)*CINCL(K))
      XP(3,JB1) = R*SINV*SINCL(K)
      IF (TRSFER) 3,4,3
C
C     COMPUTE THE VELOCITIES FOR TRANSFER OF ORIGIN.
    3 EX = E(K)*SOMEGA(K)+SINV
      FX = E(K)*COMEGA(K)+COSV
      CFACT = ZN*P(K)/(SQRTF((1.0-E(K)**2)**3))
      AX = EX*CNODE(K)+FX*SNODE(K)*CINCL(K)
      BX = FX*CNODE(K)*CINCL(K)-EX*SNODE(K)
      XDOT(1,JB1) = -AX*CFACT
      XDOT(2,JB1) = BX*CFACT
      XDOT(3,JB1) = FX*CFACT*SINCL(K)
    4 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                 ********


      SUBROUTINE CONVT1(VX,A)
C
C     THIS ROUTINE COMPUTES -- (1) ANGULAR MOMENTUM, A(4)
C                              (2) ANGULAR MOMENTUM SQUARED, A(5)
C                              (3) X,Y,Z COMPONENTS OF ANG. MOM., A(1),A(2),A(3)
C                              (4) VELOCITY, VX(4)
C                              (5) VELOCITY SQUARED, VX(5)
C
      COMMON C
C
      DIMENSION A(5),VX(5),X(3),IND(3)
C
      EQUIVALENCE (X,C(200)),(IND,C(791))
C
      DO 1 J1=1,3
      J2=IND(J1)
      J3=IND(J2)
    1 A(J3)=X(J1)*VX(J2)-X(J2)*VX(J1)
      A(5)=A(1)*A(1)+A(2)*A(2)+A(3)*A(3)
      A(4)=SQRTF(A(5))
      VX(5)=VX(1)*VX(1)+VX(2)*VX(2)+VX(3)*VX(3)
      VX(4)=SQRTF(VX(5))
      RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                 ********
```

```
      SUBROUTINE CONVT2
C
C     THIS ROUTINE CONVERTS RECTANGULAR COORDINATES INTO ORBIT ELEMENTS.
C     RECTANGULAR COORDINATES- POSITION COMPONENTS,X,AND VELOCITY COMPONENTS,VX.
C     ORBIT ELEMENTS - (1) ECCENTRICITY,E                (4) INCLINATION, ZINCL
C                      (2) ARG. OF PERICENTER,OMEGA      (5) MEAN AMOMALY,ZMA
C                      (3)LONG. OF ASCENDING NODE,ZNODE  (6) SEMILATUS RECTUM,P
C
      COMMON C
C
      DIMENSION C(1),VX(3),X(3)
C
      EQUIVALENCE
     1(     A2,C(559)),( OMEGA,C(228)),( ASQRD,C(563)),(     VX,C(472)),
     2(     A3,C(560)),( ZNODE,C(229)),(      V,C(475)),(  GK2M,C(469)),
     3(     A1,C(561)),( ZINCL,C(230)),( VSQRD,C(476)),(  EPAR,C(245)),
     4(      P,C(232)),(    ZMA,C(231)),(   TRU,C(483)),(TRSFER,C(224)),
     5(      R,C(442)),(SINTRU,C(492)),(COSTRU,C(493)),(      X,C(200)),
     6(      E,C(227)),(      A,C(562))
C
      P=ASQRD/GK2M
      R = SQRTF(X(1)**2+X(2)**2+X(3)**2)
      TRU=ARCTAN(A/GK2M*(X(1)*VX(1)+X(2)*VX(2)+X(3)*VX(3)),P-R)
      IF (A2) 2,1,2
    1 ZNODE = 0.0
      GO TO 3
    2 ZNODE = ARCTAN(A2,-A3)
    3 ZINCL = ARCTAN(SQRTF(A2**2+A3**2),A1)
      SNODE = SINF(ZNODE)
      CNODE = COSF(ZNODE)
      XTWOD = X(1)*CNODE+X(2)*SNODE
      YTWOD = X(3)*SINF(ZINCL) + COSF(ZINCL) *(X(2)*CNODE-X(1)*SNODE)
      OMEGA=ARCTAN(YTWOD,XTWOD)-TRU
      E = SQRTF(ABSF(1.+P*(VSQRD/GK2M-2./R)))
      EPONE = SQRTF(1.+E)
      E2M1 = 1.-E*E
      EPAR = SQRTF(ABSF(E2M1))
      SINTRU=SINF(TRU)
      COSTRU=COSF(TRU)
      EPAS = SQRTF(ABSF(1.-E))*SINTRU/(1.0+COSTRU)
      ETHETA=E*SINTRU/(1.0+E*COSTRU)*EPAR
    4 IF (E2M1) 5,6,6
    5 ZMA = LOGF((EPONE+EPAS)/(EPONE-EPAS)) - ETHETA
      GO TO 7
    6 ZMA = 2.0*ARCTAN(EPAS,EPONE) - ETHETA
    7 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                ********


      FUNCTION ARCTAN (Y,X)
C
C     THE FORTRAN II LIBRARY ATANF(+ OR - Z=TAN(THETA)) USES A SINGLE
C     ARGUMENT WITH ITS SIGN TO GIVE THETA IN THE FIRST (+Z) OR FOURTH
C     (-Z) QUADRANT.                  \
C
C     THE ARCTAN FUNCTION MAY BE USED IF + OR - Z IS DERIVED FROM A
C     FRACTION SO THAT ARCTAN (Y,X) = TAN-1 ((+OR-Y=SIN(THETA))/(+OR-X=
C     COS(THETA))). THUS THE ARCTAN (Y,X) GIVES THETA IN ITS PROPER
C     QUADRANT FROM -180 DEGREES TO +180 DEGREES.
C
      IF (X) 2,1,2
    1 ARCTAN=SIGNF(1.57079632,Y)
      GO TO 4
    2 ARCTAN=ATANF(Y/X)
      IF(X) 3,1,4
    3 ARCTAN=ARCTAN+SIGNF(3.14159265,Y)
    4 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                ********
```

```
      SUBROUTINE OUTPUT
C
C     THIS IS THE ROUTINE WHICH FORMS THE BASIC DATA OUTPUT.  BOTH ORBIT ELEM-
C     ENTS AND RECTANGULAR COORDINATES ARE OUTPUTTED.  IF THE OBJECT IS NOT WITH
C     IN AN ATMOSPHERE (PRESS=0.), ONE LINE OF DATA IS DELETED.  LIKEWISE,
C     ONLY THOSE PERTURBING BODIES PRESENT HAVE THEIR DISTANCES OUTPUTTED.
C
      COMMON C
C
      DIMENSION
     1     RREL (8),          ORBELS (6),               C (1),
     2    BNAME (8),             RB(3,8),            DIRCOS(3,8),
     3      VAR (4)
C
      EQUIVALENCE
     1( TABLT,C(252)),(  TIME,C(138)),(STEPNO,C(102)),(BNAME ,C(402)),
     2(     E,C(227)),( OMEGA,C(228)),( ZNODE,C(229)),( ZINCL,C(230)),
     3(   ZMA,C(231)),(     P,C(232)),(    RB,C(200)),(   TRU,C(483)),
     4(     V,C(475)),(    VX,C(472)),(    VY,C(473)),(    VZ,C(474)),
     5(  RREL,C(442)),(     X,C(200)),(     Y,C(201)),(     Z,C(202)),
     6(STEPGO,C(101)),(  DELT,C(256)),( RMASS,C(131)),( ALPHA,C(564)),
     7(DIRCOS,C(176)),(ORBELS,C(227)),( IMODE,C( 28)),( PRESS,C(466)),
     8(MBODYS,C(441)),(NBODYS,C(489)),(DTOFFJ,C( 31)),(     A,C(562)),
     9(SINTRU,C(492)),(COSTRU,C(493)),(  REVS,C(490)),(LENGTH,C(257))
      EQUIVALENCE
     1(   ALT,C(463)),( VATM1,C(477)),( VATM2,C(478)),( VATM3,C(479)),
     2(    VQ,C(480)),(   PSI,C(462))
C
      PATHANF(VX,VY,VZ) = ATANF((X*VX+Y*VY+Z*VZ)/A)*57.29577951
C
      DAYJ=(DTOFFJ-2.4E6)+TABLT
      ALPHA1 = ALPHA*57.29577951
      REV = REVS + ARCTAN(-Y,-X)/6.28318532 + .5
      CALL CONVT1(VX,C(559))
S     LXD IMODE, (IMODE)
      GO TO (2,1,1),IMODE
    1 CODE=6HRECTAN
      CALL CONVT 2
      GO TO 4
    2 DO 3 K=1,6
    3 ORBELS(K) = C(K+131)
      CODE=5HORBIT
      TRU=ARCTAN(SINTRU,COSTRU)
    4 PSI = PATHANF(VX,VY,VZ)
      WRITE OUTPUT TAPE 6, 11,STEPGO,STEPNO,E,OMEGA,V,RREL(1),BNAME(1),
     1CODE,IMODE,TIME,P,TRU,VX,X,RMASS,DAYJ,ZMA,ZNODE,VY,Y,REV,ALPHA1,
     2PSI,ZINCL,VZ,Z,DELT
C
C     IF WITHIN AN ATMOSPHERE COMPUTE DRAG, LIFT, G, ETC., AND PRINT EXTRA LINE.
      IF (PRESS) 5,7,5
    5 J=0
      DO 6 I=1,4
      J = J+3
    6 VAR(I) = SQRTF(C(J+525)**2+C(J+526)**2+C(J+527)**2)*RMASS/9.80665
      G = VAR(4)/RMASS
      CALL CONVT1(VATM1,C(559))
      PSI = PATHANF(VATM1,VATM2,VATM3)
      WRITE OUTPUT TAPE 6,12,ALT,PSI,VAR(2),VQ,G,VAR(1)
C
C     IF PERTURBATING BODIES ARE PRESENT, FIND THEIR DISTANCES AND PRINT THEM.
    7 IF(MBODYS) 8,10,8
    8 DO 9 J=2,NBODYS
      DO 9 K=1,3
    9 DIRCOS(K,J) = -RB(K,J)/RREL(J)
      WRITE OUTPUT TAPE 6,13,
     1(BNAME(J),RREL(J),DIRCOS(1,J),DIRCOS(2,J),DIRCOS(3,J),J=2,NBODYS)
   10 CALL DUMP(2,C,LENGTH)
      RETURN
   11 FORMAT(6H0STEP=F5.,2H +F4.,4X,13HECCENTRICITY=1PG15.8,7H OMEGA=G15
     1.8,4H  V=G15.8,3H R=G15.8,7H REFER=A6,1X,A6,I2/6H TIME=1PG14.7,14
     2H SEMILATUS R.=G15.8,7H TRU A=G15.8,4H VX=G15.8,3H X=G15.8,7H RMAS
     3S=G15.8/9H JDAY= 240PF10.4,15H  MEAN ANOMALY=1PG15.8,7H  NODE=G15.
     48,4H VY=G15.8,3H Y=G15.8,7H REVS.=G15.8/6H ALFA=G14.7,14H   PATH A
     5NGLE=G15.8,7H  INCL=G15.8,4H VZ=G15.8,3H Z=G15.8,7H  DELT=G15.8)
   12 FORMAT(6H ALT.=1PG14.7,14H R PATH ANGLE=G15.8,7H  DRAG=G15.8,4H VR
     1=G15.8,3H G=G15.8,7H  LIFT=G15.8)
   13 FORMAT(2(1X,A6,3H R=1PG14.7,0P3F10.6,11X))
C
C     END OF THE FORTRAN STATEMENTS.                                ********
```

```
      SUBROUTINE DUMP (IDENT,DATA,LENGTH)
C
C    THIS SUBROUTINE WILL DUMP IN G TYPE FORMAT A VARIABLE NUMBER OF CONSECUTIVE
C    WORDS, BEGINNING AT A SPECIFIED LOCATION.  DUMP OCCURS WHEN THE FOLLOWING
C    CONDITIONS ARE SATISFIED
C
C       A) IDENTIFICATION NUMBER (IDENT) = AN INPUT DUMP NUMBER (NDUMP).
C       B) DUMP NUMBER IDENT HAS BEEN SKIPED NSKIP TIMES.
C       C) TOTAL NUMBER (TEST) OF DESIRED DUMPS HAS NOT BEEN EXCEEDED.  (IF TEST
C          IS NEGATIVE, DUMP ALWAYS OCCURS).
C
C    NOTE-   IDENT = IDENTIFICATION NUMBER OF DUMP
C             DATA = STARTING LOCATION OF DUMP
C           LENGTH = NUMBER OF CONSECUTIVE WORDS TO BE DUMPED.  (ZEROES COUNT BUT
C                    ARE NOT DUMPED)
C
      COMMON C
C
      DIMENSION
     1     DATA (1),              I6 (6),          DATA6 (6),
     2   NSKIPN (4),           NDUMP (4),          NSKIP (4)
C
      EQUIVALENCE
     1(   TEST,C(   1)),( NDUMP,C(268)),( NSKIP,C(272))
C
C     PART 1.  TEST FOR OVERFLOW AND DIVIDE CHECK.
      IF DIVIDE CHECK 1,2
    1 ASSIGN 2 TO N
      WORD1 = 6HDIVIDE
      WORD2 = 6H CHECK
      GO TO 6
    2 IF ACCUMULATOR OVERFLOW 3,4
    3 ASSIGN 4 TO N
      WORD1 = 6HACC OV
      WORD2 = 6HER FLO
      GO TO 6
    4 IF QUOTIENT OVERFLOW 5,8
    5 ASSIGN 8 TO N
      WORD1 = 6HMQ OVE
      WORD2 = 6HR FLOW
    6 WRITE OUTPUT TAPE 6,7,WORD1,WORD2,IDENT
    7 FORMAT(1H02A6,18H    IDENTIFICATION=I4)
      GO TO N,(2,4,8)
C
C     PART 2.  DETERMINE IF DUMP MAY OCCUR.
    8 IF (TEST) 15,26,9
    9 DO 12 I=1,4
      IF (IDENT-NDUMP(I)) 12,10,12
   10 IF (XABSF(NSKIP(I))-NSKIPN(I)) 13,13,11
   11 NSKIPN(I) = NSKIPN(I)+1
   12 CONTINUE
      GO TO 26
   13 NSKIPN(I) = 0
      IF (NSKIP(I)) 14,15,15
   14 NSKIP(I) = 0
C
C     PART 3.  DUMP OCCURS.  DUMP NON-ZERO WORDS AND THEN REDUCE TEST BY 1.
   15 WRITE OUTPUT TAPE 6,23,TEST,IDENT,LENGTH
      K2=6
      J=0
   16 DO 21 K=1,6
   17 J = J+1
      IF (J-LENGTH) 18,18,19
   18 IF (DATA(J)) 20,17,20
   19 K2=K-1
      IF(K2) 22,25,22
   20 DATA6(K)=DATA(J)
   21 I6(K) = J
   22 WRITE OUTPUT TAPE 6,24,(I6(K1),DATA6(K1),K1=1,K2)
   23 FORMAT (12HODUMP, TEST=F6.1,18H IDENTIFICATION IS,I4, 20H, NUMBER
     1OF WORDS IS, I5)
   24 FORMAT (1X,I4,1PG15.8,5(I7,1PG15.8))
      GO TO 16
   25 TEST = TEST-1.
   26 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                    ********
```

```
      FUNCTION QUAD  (X,IC)
C
C     THIS ROUTINE COMPUTES ANY VARIABLE, QUAD, AS A QUADRATIC FUNCTION OF X.
C     QUAD = A + BX + CXX.  THERE MAY BE SEVERAL SETS OF COEFFIENTS, EACH SET
C     BELONGING TO A PARTICULAR REGION OF X.  THE COEFN ARRAY IS ARRANGED AS --
C        X1,A1,B1,C1,X2,A2,B2,C2,X3,A3,B3,C3,X4, .............
C     WHERE  A1,B1,C1 ARE THE COEFFIENTS TO BE USED FOR X BETWEEN X1 AND X2,ETC.
C     AND X1 IS LESS THAN X2, X2 IS LESS THAN X3, X3 IS LESS THAN X4, ETC.
C     IC IDENTIFIES WHICH DEPENDENT VARIABLE, QUAD, IS BEING SOUGHT.
C     ICC(IC) DEFINE THE STARTING LOCATIONS IN THE COEFN ARRAY FOR VARIABLES X.

      COMMON C
C
      DIMENSION C(10),COEFN(190),ICC(5)
C
      EQUIVALENCE     (    ICC,C(238)),( COEFN,C(601))
C
      I=ICC(IC)
    1 IF (X-COEFN(I)) 2,3,3
    2 I = I-4
      GO TO 1
    3 IF(X-COEFN(I+4)) 5,5,4
    4 I = I+4
      GO TO 3
    5 QUAD = COEFN(I+1)+X*(COEFN(I+2)+X*COEFN(I+3))
      ICC(IC)=I
      RETURN
C
C     END OF THE FORTRAN STATEMENTS.                                   ********
```

```
       REM    SUBROUTINE EXADD (A,B,C)
       REM THIS ROUTINE WILL ADD IN DOUBLE PRECISION A QUANTITY C TO THE DOUBLE
       REM PRECISION VARIABLE A+B WHERE A IS THE MOST SIGNIFICANT PART AND B IS
       REM THE LEAST SIGNIFICIANT PART.
       ORG 0
       PGM
       PZE END+1,0,0
       PZE
       BCD 1EXADD
       PZE EXADD
       ORG 0
       REL
Q1     SYN 32700
Q2     SYN 32701
TEMP1  SYN 32702
TEMP2  SYN 32703
       BCD 1EXADD
EXADD  CLA 1,4
       STA TOP1
       STA TOP2
       CLA 2,4
       STA BOT1
       STA BOT2
       CLA 3,4
       STA ARG1
TOP1   CLA **
ARG1   FAD **
       STQ Q1
BOT1   FAD **
       STQ Q2
       FAD Q1
       STQ Q1
       STO TEMP1
       CLA Q1
       FAD Q2
       STO TEMP2
       FAD TEMP2
       FAD TEMP1
       STQ Q1
       FSB TEMP2
TOP2   STO **
       STQ Q2
       CLA Q1
       FAD Q2
BOT2   STO **
END    TRA 4,4
       REM END OF THE SAP STATEMENTS.                                  ********
       END
```

```
      SUBROUTINE BKFILE(N)
C     THIS ROUTINE SIMPLY BACKSPACES TAPE N ONE FILE.
C
S   1  CAL *1
S      STP *2
      M = 10-N
S     BST 10,(M)
S     BST 10,(M)
S   2  BST 10,(M)
S     NOP
S     RTB 10,(M)
S     CPY DUD
S      TRA *3
S      TRA *4
S      TRA *3
S   3  BST 10,(M)
    4 RETURN
C
C     END OF THE FORTRAN STATEMENTS.                              ********


       REM     SUBROUTINE PONG(N)
       REM THIS ROUTINE FINDS THE SEGMENT N ON TAPE AND LOADS IT IN THE CORE.
       REM IF SEGMENT N IS ALREADY IN THE CORE, CONTROL IS SIMPLY SWITCHED TO
       REM THE BEGINNING OF SEGMENT N.
       ORG
       PGM
       PZE END+1,0,0
       PZE -1
       BCD 1SELPGM
       PZE SELPGM
       BCD 1PING
       PZE PING
       BCD 1PONG
       PZE PONG
       REL
       ORG
SPC0   PZE  DEC IS TOTAL RECORDS, ADDRS IS THIS RECORD, SET BY PING-PONG.
PING   TSX SPC4,1
SELPGM CLA 1,4
PONG   CLA 1,4
       STA **1
       CLA **
       SSP
       TZE PING            PROGRAM NUMBER TOO SMALL.
       SUB SPC0            COMPARE WITH TOTAL
       TPL ERR             PROGRAM NUMBER TOO LARGE.
       ADD SPC0
       ARS 18
       STO COMMON          DESIRED NUMBER IN ADDRESS.
       CLA SPC0            PRESENT POSITION IN ADDRESS.
       ADM SPC7            ADD ONE IN ADDRESS.
       ANA SPC8            SAVE ADDRESS ONLY.
       SUB COMMON
       PAX ,1              NUMBER OF RECORDS TO MOVE.
       TXL SPC4,1,0        PROPERLY POSITIONED IF ZERO.
       TXH SPC1,1,1        CORE LOAD OK IF ONE.
       PXD
       TRA SELPGM          GO TO THE TRANSFER TO BEGINNING OF PROGRAM.
SPC1   TMI SPC2            ADVANCE TAPE.
       BST T               BACKSPACE TAPE.
       TRA SPC3
SPC2   RTB T
SPC3   TIX SPC1,1,1        KEEP MOVING TAPE.
SPC4   RTB T
       CPY 0               RIGHT POSITION NOW SO LOAD IT.
       TRA SPC5
       REW T               EOF, NEXT IN SEQUENCE IS FIRST RECORD.
       TRA SPC4            FALSE EOR.
SPC5   CAL 0
       ANA SPC8            ADDRESS OF FIRST WORD IS REQUIRED NUMBER.
       SUB COMMON          DESIRED NUMBER.
       TXH SPC7,1,1        BYPASS CHECK ON SELPGM ENTRY.
       TZE SPC7            PROPERLY FOUND.
SPC6   HPR 1,5             IMPROPER POSITIONING DUE TO MACHINE ERROR.
       TRA SPC6
SPC7   CPY 1
       TRA 0               GO TO LOADER.
SPC8   PZE -1              IS ONEA.
ERR1   BCD 10PONG
ERR2   BCD 1FAIL.
ERR    WTD 6               THIS IS THE ERROR PRINTOUT ROUTINE.
       CPY ERR1
       CPY ERR2
       IOD
       RDR 1               CALL MONITOR.
       CPY 0
       CPY 1
END    TRA 0
T      EQU 2
COMMON SYN -1
       REM END OF THE SAP STATEMENTS.                             ********
       END                                                               I
```

REFERENCES

1. Vienop, Edna, and Brady, Joseph L.: The Themis Code: An Astronomical Numerical Integration Program for the IBM 704. UCRL 5242, Radiation Lab., Univ. Calif., May 20, 1958.

2. Holdridge, D. B.: Space Trajectories Program for the IBM 7090 Computer. TR 32-223, Jet Prop. Lab., C.I.T., Mar 2, 1962.

3. Technical Staff of Radiation, Inc., ed.: Space Trajectories. Academic Press, Inc., 1960.

4. O'Keefe, John A., Eckels, Ann, and Squires, R. Kenneth: The Gravitational Field of the Earth. Astronomical Jour., vol. 64, no. 7, Sept. 1959, pp. 245-253.

5. Baker, R. M. L., Jr., et al.: Efficient Precision Orbit Computation Techniques. ARS Jour., vol. 30, no. 8, Aug. 1960, pp. 740-747.

6. Dobson, Wilbur F., Huff, Vearl N., and Zimmerman, Arthur V.: Elements and Parameters of the Osculating Orbit and Their Derivatives. NASA TN D-1106, 1962.

7. Turner, Don N., and Huff, Vearl N.: An Input Routine Using Arithmetic Statements for the IBM 704 Digital Computer. NASA TN D-1092, 1961.

8. Clarke, V. C., Jr.: Constants and Related Data Used in Trajectory Calculations at the Jet Propulsion Laboratory. TR 32-273, Jet Prop. Lab., C.I.T., May 1962.

9. Dobson, Wilbur F., Mackay, John S., and Huff, Vearl N.: Starting Conditions for Nonoscillatory Low-Thrust Planet-Escape Trajectories. NASA TN D-1410, 1962.

10. Weber, Richard J., Pauson, Werner M., and Burley, Richard R.: Lunar Trajectories. NASA TN D-866, 1961.

11. Kopal, Zdeněk: Numerical Analysis. John Wiley & Sons, Inc., 1955.

TABLE I. - ORBIT ELEMENTS AND OTHER DATA FOR ELLIPSE EPHEMERIDES

[Epoch: Sept. 23, 1960; Julian day, 2437 200.5; mean equinox and equator, 1950.0.]

Name	Reference	Mass, sun mass units	Radius of influence sphere, m	Semilatus rectum, AU	Eccentricity	Argument of pericenter, radians	Longitude of ascending node, radians	Inclination, radians	Julian day of perihelion passage	Fractional day of perihelion passage	Period, mean solar days
Mercury	Sun	1/6,120,000	10^8	0.3707315	0.205627	1.1679154	0.1896133	0.49924366	2437 163	0.283386	87.969252
Venus	Sun	1/466,645	6.14×10^8	.72329863	.006792	2.1567353	.13931743	.42703751	2437 132	.682782	224.70087
Mars	Sun	1/3,088,000	5.78×10^8	1.5104078	.093369	5.7966845	.058500499	.4310002	2437 081	.09531	686.97964
Jupiter	Sun	1/1047.39	4.81×10^{10}	5.1913995	.0486288	.1765935	.056971884	.40587194	2433 964	.6664	4333.7153
Saturn	Sun	1/3500	5.46×10^{10}	9.5554288	.0509895	1.4938359	.10416467	.39404007	2431 246	.5163	10,829.478
Uranus	Sun	1/22,869	5.17×10^{10}	19.100903	.0457866	2.9848628	.032257032	.41321621	2409 019	.272	30,587.016
Neptune	Sun	1/18,889	8.61×10^{10}	30.197622	.0045616	.3302296	.061416599	.38947933	2404 118	.842	60,612.183
Pluto	Sun	1/400,000	3.81×10^{10}	36.969138	.2502358	3.1999771	.76630286	.41231716	1639 376	.44	904,658.99
Sun	Earth	1.0	10^{20}	.99972025	.016716	4.923277	0	.4092062	2436 937.1	0	365.256

TABLE II. - PROGRAM CONTROL PARAMETERS

Control variables	COMMON location	Possible values	Setting	Description of use
ASYMPT	543	0.0 or 1.0	Internal	Normally equal to 0.0, set equal to 1.0 in SUBROUTINE EQUATE when Kepler's equation fails to converge for $e > 1$, and then used to control branching in MAIN 2 for IMODE = 3.
ATMN	26	Any ALF coded body name	Input	Contains name of body which is to have an atmosphere. Causes SUBROUTINE AERO to be called in SUBROUTINE EQUATE if object is within that atmosphere.
CLEAR	19	Any value	Input	If CLEAR = 0, SUBROUTINE STDATA is called from MAIN 1; if CLEAR ≠ 0, SUBROUTINE STDATA is bypassed. STDATA clears C(4) to C(1300).
CONSTU	18	>0, $\sim 10^{-8}$ to $\sim 10^{-2}$ radian	STDATA: 10^{-6} Input	Controls branching in SUBROUTINE EQUATE, which determines how accurate eccentric anomaly will be computed by Kepler's equation.
CONSU	36	>0, $\sim 10^{-8}$ to $\sim 10^{-2}$ radian	STDATA: 10^{-6} Input	Similar to CONSTU except that it is used in SUBROUTINE ELIPSE for perturbing bodies instead of object.
DELMAX	23	Any number of seconds	Input	If MODOUT = 2 or 3, output is given only at intervals of DELMAX.
EREF	37	Any number	STDATA: 10^{-6} Input	Desired error value. Error control predicts step size such that E2 ~ EREF. If EREF < 0, it will be treated as +EREF; however, error data will be recorded and printed.
ERLIMT	17	Any plus number	STDATA: 3×10^{-6} Input	Maximum error value that allows step in question to be passed as good step. If E2 > ERLIMT, step is recomputed with smaller step size.
ETOL	25	Positive number of order 0.01	STDATA: 0.01 Input	If eccentricity falls in region 1 ± ETOL and integration is in orbit elements, integration mode is switched to temporary rectangular until eccentricity falls outside this region.
FILE	249	Any plus integer	Internal	Set equal to 10.0 in SUBROUTINE ORDER if tape data is used to determine positions, velocities, and attractions of perturbing bodies. Then read as file number of tape 3 in MAIN 1. See TFILE.
ICC(5)	238-242	Any fixed-point integer	Input Internal	Index of independent variable in COEFN array used in FUNCTION QUAD. For each set of coefficients there is an ICC. They are set at input time and are reset each time QUAD is called.
IMODE	28	1,2,3,4,-1,-2,-3,-4 (fixed point)	STDATA: 1 Input Internal	Indicates integration mode. Must agree with input data (if input data is rectangular, IMODE should equal 2 or -2). Values indicate: 1 = orbit elements; -1 = orbit elements, change to rectangular 2 = rectangular variables; -2 = rectangular, change to orbit elements 3 = temporary rectangular; -3 = orbit, change to temporary rectangular 4 = Earth spherical change to rectangular; -4 = Earth spherical, change to orbit element
LENGTH	257	Any fixed-point integer	Input	Length of dump (i.e., number of words to be dumped).
MODOUT	103	1,2,3,4 (fixed point)	STDATA: 4 Input Internal	MODOUT = 1 Output every n^{th} step (n = STEPS) until TIME = TMIN, then shift to mode 2. = 2 Output at time intervals of DELMAX until TIME = TMAX. = 3 Output at time intervals of DELMAX until TIME = TMIN, then shift to mode 4. = 4 Output every n^{th} step until TIME = TMAX.
NDUMP(4)	268-271	Any fixed-point integer	Input	If i in CALL DUMP (i, C, LENGTH) command equals any number in NDUMP array, dump will be executed conditionally (see NSKIP).
NSKIP(4)	272-275	Any fixed-point integer	Input	Causes skipping of NSKIP(i) dumps where NSKIP(i) corresponds to NDUMP(i). See SUBROUTINE DUMP.
NPONG(5)	11-15	Any fixed-point integer	STDATA: 2,1, , ,1 Input	NPONG(i) refers to segment that is being called in statements CALL PONG (NPONG(i)). Control is to beginning of segment.
OBLATN	27	Any ALF coded body name	Input	If oblateness effects are to be considered, loading a body name will cause SUBROUTINE OBLATE to be called from SUBROUTINE EQUATE when OBLATN matches reference body.
RECALL	9	Any value	Input	If RECALL ≠ 0.0, "starting" data will be restored from C(5) to C(115) in MAIN 1. See SAVE.
SAVE	8	1.0, 2.0, or any other value	Input	If SAVE = 1.0, "starting" data from C(5) to C(115) will be saved to be used later for another start requiring same data. If SAVE = 2.0, same thing happens, only before CALL INPUT (1) statement in MAIN 1. This saves result of previous integration for future use.
STEPGO	101	Any plus number	Internal	Total number of good steps.
STEPNO	102	Any plus number	Internal	Total number of bad steps. Bad step does not pass error control test.
STEPMX	20	Any plus number	STDATA: 100.0 Input	If (STEPGO + STEPNO) ≥ STEPMX, problem terminates.
STEPS	21	Any plus number	STDATA: 1.0 Input	Used when MODOUT = 1 or 4. Output will occur at every n^{th} step where n = STEPS.
TAPE 3	2	0.0 or 3.0	Internal Input	If "working" ephemeris tape is to be made, TAPE 3 must be set equal to zero through input contained in SUBROUTINE TAPE. If no tape is to be made, or after tape is made, TAPE 3 is set to 3.0.
TEST	1	Any integer	Input Internal	Total number of dumps. Initially set through input and thereafter decreased by one each time a dump occurs until TEST = 0. When TEST = 0.0 no more dumps will occur. If negative value of TEST is loaded, there is no limit on number of dumps.
TFILE	16	Any plus integer	STDATA: 1.0 Input	Selects which file of "working" ephemeris tape is to be used. MAIN 1 positions tape in correct position by matching desired file number (TFILE) with code word (FILE) written at beginning of each file on tape.
TMAX	30	Any number in seconds	Input	When TIME = TMAX control is switched to MAIN 1 to either read new input or end problem.
TMIN	22	Any number in seconds	Input	When TIME = TMIN output mode is changed. See MODOUT.
TRSFER	224	0.0 or 1.0	Internal	Normally TRSFER = 0.0, but when origin is being translated TRSFER = 1.0 which causes SUBROUTINES EPHMRS and ELIPSE to compute velocities as well as positions.
TTEST	251	Any number in seconds	Internal	When integration mode is changed to temporary rectangular, TTEST is set as time at which program will begin checking for return to orbit elements. See MAIN 2, part 7D.

TABLE III. - BASIC OUTPUT FORMAT

(a) Sample output

```
STEP=    0. +  0.     ECCENTRICITY= 1.00000000      OMEGA=-2.64801353
TIME= 0.              SEMILATUS R.= 1.93844640E-09 TRU A= 3.14159262
JDAY= 2437640.8350  MEAN ANOMALY= 0.                 NODE= 2.02516600
ALFA= 0.                PATH ANGLE= 89.9209976       INCL= 1.57079409
ALT.=-0.1875000       R PATH ANGLE= 89.9209976       DRAG= 4.99665982E-03
SUN     R= 1.4728028E 11 -0.261730 -0.885466 -0.383989

        V= 9.99999976E-02 R= 6373346.50      REFER=EARTH   RECTAN 2
       VX=-3.86224359E-02 X=-2463371.37      RMASS= 150000.000
       VY= 7.90702742E-02 Y= 5043168.50      REVS.= 0.32231534
       VZ= 4.74994606E-02 Z= 3019569.50       DELT= 6.00000000
       VR= 9.99999976E-02 G= 1.49946962       LIFT= 0.
MOON    R= 3.8293912E 08 -0.387660 -0.874846 -0.290456
```

(b) Parameter identification

FORTRAN code name		Identification
Output format mnemonic	Internal	
STEP	STEPGO, STEPNO	Count of total number of successful integration steps to left of plus sign and count of failures on right
TIME	TIME	Time since beginning of integration process, t, sec
JDAY	DAYJ	Current Julian date
ECCENTRICITY	E	Osculating orbit eccentricity, e
SEMILATUS R.	P	Semilatus rectum of osculating orbit, p, m
MEAN ANOMALY	ZMA	Mean anomaly of osculating orbit, M
OMEGA	OMEGA	Argument of pericenter, ω, radians
TRU A	TRU	True anomaly of osculating orbit, v, radians
NODE	ZNODE	Equatorial longitude of ascending node of osculating orbit, Ω, radians
INCL	ZINCL	Orbit inclination referred to mean equator and equinox of 1950.0, i, radians
ALFA	ALPHA	Angle between thrust and velocity, α, deg
PATH ANGLE	PSI	Angle between path and local horizontal, deg
V,VX,VY,VZ	V,VX,VY,VZ	Velocity and its x,y,z components, V, m/sec
R,X,Y,Z	RREL(1), X,Y,Z	Radius and its x,y,z components, r, m
REFER	BNAME(1)	Name of reference body, followed by integration mode, IMODE
RMASS	RMASS	Vehicle mass, m, kg
REVS.	REV	Revolutions past x-axis
DELT	DELT	Step size for current step, h, sec
ALT.	ALT	Altitude above oblate Earth, m
R PATH ANGLE	PSI	Relative path angle, relative to Earth, deg
DRAG	VAR(2)	Total drag force, D, kg
VR	VQ	Velocity relative to rotating reference body
G	G	Total Earth g's acting on missile
LIFT	VAR(1)	Total lift force, L, kg
BNAME(i)R	BNAME(i), DIR COS	Vehicle to perturbing body distance, r_i, plus direction cosines

TABLE IV. - COMMON ALLOCATION

Index	0	100	200	300	400	500	600	700	800	900	1000
0		XPRIMB(15,2)	RB(1,1)	(1,2,2)	–	–	RATIO	–	PAR(3)	–	–
1	TEST	STEPGO	–	–	TDATA(6,3,7)	(1,2)	COEFN(1)	–	BODYL(1)	–	(1,5)
2	TAPE3	STEPNO	–	–	BNAME(1)	–	–	–	–	–	–
3	CLOCK	MODOUT	(1,2)	–	–	–	–	–	–	–	–
4		ANGLES(1)	–	–	–	(1,3)	–	–	–	–	–
5	SIMP	–	–	–	–	–	–	–	–	–	–
6		–	(1,3)	(1,3,2)	–	–	–	–	–	–	–
7	TKICK	ANGLES(4)	–	–	–	(1,4)	–	–	–	–	–
8	SAVE	ALT	–	–	–	–	–	–	–	–	–
9	RECALL	VEL	(1,4)	–	BNAME(8)	–	–	–	BODYL(8)	–	–
10	DELT		–	–	EFMRS(1)	(1,5)	–	–	–	AMASS(30)	–
11	NPONG(1)		–	–	–	–	–	–	BODYCD(1)	RCRIT(1)	–
12	–		(1,5)	(1,1,3)	–	–	–	–	–	–	–
13	–		–	–	–	(1,6)	–	–	–	–	–
14	–		–	–	–	–	–	–	–	–	–
15	NPONG(5)		(1,6)	–	–	–	–	–	–	–	–
16	TFILE		–	–	EFMRS(7)	(1,7)	–	–	–	–	(1,6)
17	ERLIMT		–	–	BMASS(1)	–	–	–	–	–	–
18	CONSTU		(1,7)	(1,2,3)	–	–	–	–	BODYCD(8)	–	–
19	CLEAR		–	–	–	(1,8)	–	–	–	–	–
20	STEPMX		–	–	–	–	–	–	–	–	–
21	STEPS		(1,8)	–	–	VEFM(3,8)	–	–	PNAME(1)	–	–
22	TMIN		–	–	–	QX(1)	–	–	–	–	–
23	DELMAX		RB(3,8)	–	–	QX(2)	–	–	–	–	–
24	AEXIT		TRSFER	(1,3,3)	BMASS(8)	QX(3)	–	–	–	–	–
25	ETOL		OLDDEL	–	IBODY(1)	FORCE(1)	–	–	–	–	–
26	ATMN		TTOL	–	–	FORCE(2)	–	–	–	–	–
27	OBLATN		ORBELS(1)	–	–	FORCE(3)	–	–	–	–	–
28	IMODE		–	–	–	XIFT(1)	–	–	–	–	–
29	RATM		–	–	–	XIFT(2)	–	–	–	–	–
30	TMAX		–	(1,1,4)	–	XIFT(3)	–	–	–	–	–
31	DTOFFJ	X(1)	–	–	–	DRAG(1)	–	–	–	–	(1,7)
32	TOFFT	–	ORBELS(6)	–	IBODY(8)	DRAG(2)	–	–	–	–	–
33	FLOW	–	AK(1)	–	NEFMRS(1)	DRAG(3)	–	–	–	–	–
34	PUSH	–	AK(2)	–	–	OBLAT(1)	–	–	–	–	–
35	AREA	–	AK(3)	–	–	OBLAT(2)	–	–	–	–	–
36	CONSU	–	A1	(1,2,4)	–	OBLAT(3)	–	–	–	–	–
37	FREE	–	A2	–	–	COMPA(1)	–	–	–	–	–
38	OBLATJ	–	ICC(1)	–	–	COMPA(2)	–	–	–	–	–
39	OBLATK	–	–	–	–	COMPA(3)	–	–	–	–	–
40	RESORD	–	–	–	NEFMRS(8)	RADIAL	–	–	–	RCRIT(30)	–
41	XPRIM(1,1)	–	–	–	MBODYS	CIRCUM	–	–	–	ELIPS(1,1)	–
42	–	–	ICC(5)	(1,3,4)	R	ZORMAL	–	–	–	–	–
43	–	–	EMONE	–	RREL(2)	ASYMPT	–	–	–	–	–
44	–	–	EXMODE	–	–	XWHOLE(1)	–	–	–	–	–
45	–	X(15)	EPAR	–	–	–	–	–	–	–	–
46	–	XINC(1)	CHAMP	–	–	–	–	–	–	–	(1,8)
47	–	–	NSTART	–	–	–	–	–	–	–	–
48	–	–	RATMOS	(1,1,5)	–	–	–	–	–	–	–
49	–	–	FILE	–	RREL(8)	–	–	–	–	–	–

TABLE IV. - Concluded. COMMON ALLOCATION

Index	0	100	200	300	400	500	600	700	800	900	1000
50	-	-	REVOLV	-	RBCRIT(1)	-	-	-	PNAME(30)	-	-
51	-	-	TTEST	-	-	-	-	-	REFER(1)	-	-
52	-	-	TABLT	-	-	-	-	-	-	-	-
53	-	-	SPD	-	-	-	-	-	-	-	-
54	-	-	KSUB	(1,2,5)	-	-	-	-	-	-	-
55	-	-	DEL	-	-	-	-	-	-	-	-
56	(1,2)	-	H2	-	-	-	-	-	-	(1,2)	-
57	-	-	LENGTH	-	RBCRIT(8)	-	-	-	-	-	-
58	-	-	SPACES	-	GASFAC	XWHOLE(15)	-	-	-	-	-
59	-	-	ERLOG	-	ROTATE	A(1)	-	-	-	-	-
60	-	XINC(15)	E2	(1,3,5)	DNSITY	A(2)	-	-	-	-	ELIPS(15,8)
61	-	XDOT(1)	AW(1)	-	AU	A(3)	-	-	-	-	1061
62	-	-	-	-	PSI	A	-	-	-	-	
63	-	-	-	-	ALT	ASQRD	-	-	-	-	
64	-	-	AW(4)	-		ALPHA	-	-	-	-	
65	-	-	ACOEF1	-	GEOH	BETA	-	-	-	-	
66	-	-	ACOEF2	(1,1,6)	PRESS		-	-	-	-	
67	-	-	ACOEF3	-	TM	RSQRD	-	-	-	-	
68	-	-	NDUMP(1)	-	SQRDK	SINBET	-	-	-	-	
69	-	-	-	-	GK2M	SINALF	-	-	-	-	UNUSED
70	XPRIM(15,2)	-	-	-	GKM		-	-	-	-	COMMON
71	XPRIMB(1,1)	-	NDUMP(4)	-	VMACH	P(1)	-	-	-	(1,3)	
72	-	-	NSKIP(1)	(1,2,6)	VX	P(2)	-	-	-	-	
73	-	-	-	-	VY	P(3)	-	-	-	-	
74	-	-	-	-	VZ	PMAGN	-	-	-	-	
75	-	XDOT(15)	NSKIP(4)	-	V	COSALF	-	-	-	-	
76	-	XP(1,1)	TDATA(1,1,1)	-	VSQRD	CON(1)	-	-	-	-	
77	-	-	-	-	VATM(1)	-	-	-	-	-	
78	-	-	-	(1,3,6)	VATM(2)	-	-	-	-	-	1300
79	-	(1,2)	-	-	VATM(3)	-	-	-	-	-	1301
80	-	-	-	-	VQ	-	-	-	REFER(30)	-	
81	-	-	-	-	VQSQRD	-	-	-	AMASS(1)	-	
82	-	(1,3)	(1,2,1)	-		-	-	-	-	-	
83	-	-	-	-	TRU	-	-	-	-	-	
84	-	-	-	(1,1,7)		CON(9)	-	-	-	-	TAB(189)
85	-	(1,4)	-	-		TIM(1)	-	-	-	-	
86	(1,2)	-	-	-		-	-	-	-	(1,4)	
87	-	-	-	-	ZN	-	-	-	-	-	
88	-	(1,5)	(1,3,1)	-	ZM	-	-	-	-	-	
89	-	-	-	-	NBODYS	-	-	-	-	-	1489
90	-	-	-	(1,2,7)	REVS	-	-	COEFN(190)	-	-	1490
91	-	(1,6)	-	-	INDERR	TIM(7)	-	IND(1)	-	-	
92	-	-	-	-	SINTRU	TDEL(1)	-	IND(2)	-	-	
93	-	-	-	-	COSTRU	-	-	IND(3)	-	-	
94	-	(1,7)	(1,1,2)	-	SINCL	-	-	QVAL	-	-	SAVE
95	-	-	-	-	CINCL	-	-	CDI	-	-	STORAGE
96	-	-	-	(1,3,7)	SINV	-	-	CL	-	-	
97	-	(1,8)	-	-	COSV	-	-	CD	-	-	
98	-	-	-	-	VEFM(1,1)	TDEL(7)	-	PAR(1)	-	-	
99	-	XP(3,8)	-	-	-	COSBET	-	PAR(2)	-	-	1600

TABLE V. - ELEMENTS OF INTEGRATION VARIABLE ARRAY XPRIM

[XPRIM 9 to 15 are left for expansion.]

Integration variables	XPRIM							
	1	2	3	4	5	6	7	8
Rectangular variables	RMASS (mass)	VX (x-component of velocity)	VY (y-component of velocity)	VZ (z-component of velocity)	X (x-component of position)	Y (y-component of position)	Z (z-component of position)	TIME (time)
Orbit elements	RMASS (mass)	E (eccentricity)	OMEGA (argument of pericenter)	NODES (longitude of ascending nodes)	INCL (orbit inclination)	ZMA (mean anomaly)	P (semilatus rectum)	TIME (time)

TABLE VI. - ASSUMED VALUES OF ASTRONOMICAL CONSTANTS

Constant	Assumed value	FORTRAN name	COMMON location
Astronomical unit, m	1.495×10^{11}	AU	461
Gravitational constant, k^2 $m^3/(sec^2)$(sun mass units)	1.32452139×10^{20}	SQRDK	468
Equatorial Earth radius squared, m^2	4.068098877×10^{13}	RESQRD	40
Earth oblateness coefficient, J	1.6238×10^{-3}	OBLATJ	38
Earth oblateness coefficient, K	6.4×10^{-6}	OBLATK	39
Earth radii per AU	4.26546512×10^{-5}	ERTOAU	[a]3
Day, sec	86400	SPD	253
Mass, reciprocal sun mass units:			
Sun	1.0	AMASS(1)	881
Mercury	6,120,000	AMASS(2)	882
Venus	406,645	AMASS(3)	883
Earth	332,488	AMASS(4)	884
Mars	3,088,000	AMASS(5)	885
Jupiter	1047.39	AMASS(6)	886
Saturn	3500.0	AMASS(7)	887
Uranus	22,869	AMASS(8)	888
Neptune	18,889	AMASS(9)	889
Pluto	400,000	AMASS(10)	890
Moon	AMASS(4)/81.375	AMASS(11)	891
Earth-moon	AMASS(4) + AMASS(11)	AMASS(12)	892
Sphere-of-influence radii, m:			
Sun	1.0×10^{20}	RCRIT(1)	911
Mercury	1.0×10^{8}	RCRIT(2)	912
Venus	6.14×10^{8}	RCRIT(3)	913
Earth	9.25×10^{8}	RCRIT(4)	914
Mars	5.78×10^{8}	RCRIT(5)	915
Jupiter	4.81×10^{10}	RCRIT(6)	916
Saturn	5.46×10^{10}	RCRIT(7)	917
Uranus	5.17×10^{10}	RCRIT(8)	918
Neptune	8.61×10^{10}	RCRIT(9)	919
Pluto	3.81×10^{10}	RCRIT(10)	920
Moon	1.60×10^{8}	RCRIT(11)	921

[a]Location relative to COMMON of subroutine TAPE (TAPE has a COMMON that is independent of all other subroutines).

TABLE VII. - LEWIS RESEARCH CENTER EPHEMERIS TAPE DATA

[The beginning date of all the bodies except Mars is 2437 200.5 or Oct. 23, 1960. The beginning date for Mars is 2437 202.5 or Oct. 25, 1960.]

Body	Source	End date		Number of fits	Average, days/fit	Average, deg/fit	Source checked against	Average error	Maximum error
		Gregorian	Julian						
Venus	Themis	Oct. 31, 2000	2451 848.5	968	15	24	JPL	1.7	7.3
Earth-moon barycenter		Oct. 31, 2000	2451 848.5	962	15	15	JPL	1.8	9.5
Sun		Nov. 24, 2000	2451 872.5	1821	8	8	JPL Themis	5.0 .06	21.0 3.0
Moon	JPL	Nov. 26, 1970	2440 916.5	1851	2	26	JPL	.14	9.5
Mars	JPL	July 26, 1998	2451 020.5	315	44	23		1.1	7.2
Jupiter	Themis	March 2, 2060	2473 520.5	110	330	27		1.6	9.5
Saturn				44	825	27		1.5	8.6
Uranus				30	1211	14		.95	6.5
Neptune				31	1172	7		.52	3.2
Pluto				33	1101	4	Themis	.41	3.2

The error in the x-component of position, with similar equations for the y- and z-components, is given by $e_x = \left[(x' - x)/R\right]10^8$ where x' = merged ephemeris position component; x = check source position component; $R^2 = x^2 + y^2 + z^2$.

NASA TN D-1455
National Aeronautics and Space Administration.
THE N-BODY CODE - A GENERAL FORTRAN CODE FOR SOLUTION OF PROBLEMS IN SPACE MECHANICS BY NUMERICAL METHODS.
William C. Strack, Wilbur F. Dobson, and Vearl N. Huff. January 1963. i, 92p. OTS price, $2.25.
(NASA TECHNICAL NOTE D-1455)

A general astronomical integration code designed for a large class of problems in space mechanics that may be solved by numerical integration is described. The equations of motion provide for the effects of up to eight gravitating celestial bodies, oblateness and aerodynamic forces from the celestial body at the problem origin, propulsion system thrust, and rotation of the body at the origin.

I. Strack, William C.
II. Dobson, Wilbur F.
III. Huff, Vearl N.
IV. NASA TN D-1455

NASA

NASA TN D-1455
National Aeronautics and Space Administration.
THE N-BODY CODE - A GENERAL FORTRAN CODE FOR SOLUTION OF PROBLEMS IN SPACE MECHANICS BY NUMERICAL METHODS.
William C. Strack, Wilbur F. Dobson, and Vearl N. Huff. January 1963. i, 92p. OTS price, $2.25.
(NASA TECHNICAL NOTE D-1455)

A general astronomical integration code designed for a large class of problems in space mechanics that may be solved by numerical integration is described. The equations of motion provide for the effects of up to eight gravitating celestial bodies, oblateness and aerodynamic forces from the celestial body at the problem origin, propulsion system thrust, and rotation of the body at the origin.

I. Strack, William C.
II. Dobson, Wilbur F.
III. Huff, Vearl N.
IV. NASA TN D-1455

NASA

NASA TN D-1455
National Aeronautics and Space Administration.
THE N-BODY CODE - A GENERAL FORTRAN CODE FOR SOLUTION OF PROBLEMS IN SPACE MECHANICS BY NUMERICAL METHODS.
William C. Strack, Wilbur F. Dobson, and Vearl N. Huff. January 1963. i, 92p. OTS price, $2.25.
(NASA TECHNICAL NOTE D-1455)

A general astronomical integration code designed for a large class of problems in space mechanics that may be solved by numerical integration is described. The equations of motion provide for the effects of up to eight gravitating celestial bodies, oblateness and aerodynamic forces from the celestial body at the problem origin, propulsion system thrust, and rotation of the body at the origin.

I. Strack, William C.
II. Dobson, Wilbur F.
III. Huff, Vearl N.
IV. NASA TN D-1455

NASA

NASA TN D-1455
National Aeronautics and Space Administration.
THE N-BODY CODE - A GENERAL FORTRAN CODE FOR SOLUTION OF PROBLEMS IN SPACE MECHANICS BY NUMERICAL METHODS.
William C. Strack, Wilbur F. Dobson, and Vearl N. Huff. January 1963. i, 92p. OTS price, $2.25.
(NASA TECHNICAL NOTE D-1455)

A general astronomical integration code designed for a large class of problems in space mechanics that may be solved by numerical integration is described. The equations of motion provide for the effects of up to eight gravitating celestial bodies, oblateness and aerodynamic forces from the celestial body at the problem origin, propulsion system thrust, and rotation of the body at the origin.

I. Strack, William C.
II. Dobson, Wilbur F.
III. Huff, Vearl N.
IV. NASA TN D-1455

NASA