NASA CR- 139038

R-827

A USERS MANUAL FOR A COMPUTER PROGRAM WHICH CALCULATES TIME OPTIMAL GEOCENTRIC TRANSFERS USING SOLAR OR NUCLEAR ELECTRIC AND HIGH THRUST PROPULSION

by

Lester L. Sackett, Theodore N. Edelbaum,
Harvey L. Malchow

June 1974

(NASA-CR-139038) A USERS MANUAL FOR A COMPUTER PROGRAM WHICH CALCULATES TIME OPTICAL GEOCENTRIC TRANSFERS USING SOLAR OR NUCLEAR (Draper (Charles Stark) Lab., Inc.) 169 p HC $11.50 CSCL 22A G3/30 N74-30284 Unclas 54632

The Charles Stark Draper Laboratory, Inc.
Cambridge, Massachusetts 02139

R-827

A USERS MANUAL FOR A COMPUTER PROGRAM WHICH CALCULATES TIME OPTIMAL GEOCENTRIC TRANSFERS USING SOLAR OR NUCLEAR ELECTRIC AND HIGH THRUST PROPULSION

by

Lester L. Sackett, Theodore N. Edelbaum,
Harvey L. Malchow

June 1974

Approved: Norman E. Sears
Norman E. Sears

The Charles Stark Draper Laboratory, Inc.
Cambridge, Massachusetts 02129

ACKNOWLEDGEMENT

This report was prepared under Project 55-51300 sponsored by the Goddard Space Flight Center, National Aeronautics and Space Administration through Contract NAS 5-21791. The Technical Monitor was Mr. Kenneth I. Duck.

The publications of this report does not constitute approval by the National Aeronautics and Space Administration of the findin, or the conclusions contained therein. It is published only for the exchange and stimulation of ideas.

A number of people have contributed to the success of this program. We would particularly like to thank Kenneth Duck of GSFC for his active support and encouragement, Huntington Small of Lockheed for his high thrust optimization program, Saul Serben of CSDL for several subroutines and for advice and assistance in programming. Alan Stanley of the Lincoln Laboratory and James Cake of NASA Lewis for their advice on radiation degradation and Michael Teague of GSFC for providing models of the radiation belts.

Abstract

This manual is a guide for using a computer program which calculates time optimal trajectories for high-and low-thrust geocentric transfers. Either SEP or NEP may be assumed and a one or two impulse, fixed total ΔV, initial high thrust phase may be included. Also a single impulse of specified ΔV may be included after the low thrust state. The low thrust phase utilizes equinoctial orbital elements to avoid the classical singularities and Kryloff-Boguliuboff averaging to help insure more rapid computation time.

The program is written in Fortran IV in double precision for use on an IBM 360 computer. The manual includes a description of the problem treated, input/output information, examples of runs, and source code listings.

The NEP computer program is called NECKSPOT (Nuclear Electric Control Knob Setting Program for Optimal Trajectories) and the SEP program is called SECKSPOT.

TABLE OF CONTENTS

ABSTRACT

SECTION I

DISCUSSION

I. 1 Objective

This computer program is designed to rapidly compute minimum time geocentric transfers for combinations of high thrust and low thrust stages. The most general configuration treated by the program contains three stages. The first stage is a high thrust stage such as a space tug, the second stage is a nuclear electric (NEP) or solar electric propulsion (SEP) low thrust stage and the final stage is a high thrust stage which accelerates a fixed payload. Either or both of the high thrust stages may be eliminated.

For a three stage case, the program operates in the following manner. The vehicle is assumed to start in a low altitude circular orbit. This orbit has a specified altitude and inclination and an optimized line of nodes. The first stage then performs a minimum-fuel time-open transfer to an optimized changeover orbit. This transfer is assumed to use a fixed ΔV and may involve either one or two impulses. All perturbations to the inverse square field are neglected in this phase. The second stage then performs a minimum-time low thrust transfer to a second optimized changeover orbit. This transfer is calculated by Kryloff-Bogoliuboff averaging and includes the effects of oblateness. If solar electric propulsion is used, the program may also calculate the effects of shadowing, radiation degradation, and the varying solar distance. The third stage then makes a minimum fuel, single impulse transfer through a fixed ΔV to the final orbit. This final orbit has a specified major axis, eccentricity and inclination. Its line of nodes and line of apsides are optimized. The overall trajectory is a minimum time trajectory for specified ΔV increments in the high thrust phases.

In addition to the three stage case, the program can be run using purely low thrust or usi' 'ow thrust with either initial or final impulses. For pure low thrust, the final line of nodes and line of apsides may also be specified as well as major axis, eccentricity and inclination.

I.2 Solution Method

A more thorough discussion will be found in the final report (Ref. 1). In addition the low thrust optimization technique has been previously described (Ref. 2). The basic technique is Kryloff-Bogoliuboff averaging (Ref. 3) of both the state and the costate. The averaged rates of change of the mean values of the state and costate are found by numerical quadrature. The differential equations for the mean state and costate may then be integrated in large time steps (typically days). A set of nonsingular orbit elements, the equinoctial elements (Ref. 4), is used to avoid numerical difficulties.

The effect of oblateness is included by analytically adding its rate of change of the mean state and costate to that due to thrust. The effects of shadowing are calculated by assuming that thrust is turned off in shadow. The shadow entrance and exit times are calculated analytically by solving a quartic equation. The effects of radiation degradation are calculated by fitting an equivalent 1 MEV electron flux as a function of radius and geomagnetic latitude. The power is then expressed as a function of the total accumulated particle fluence. As for all perturbations, the effect of radiation degradation on the costate as well as the state is calculated.

The first stage high thrust optimization is based on a very efficient computer program developed by Huntington Small (Ref. 5, 6). This program uses a special set of variables and form of the switching conditions developed by Small. Because the initial orbit is circular, it was possible to use an existing analysis (Ref. 7) to constrain the initial costate to the region that yields solutions. This program rapidly calculates either one or two impulse minimum-fuel time-open trajectories.

Because these transfers always require less than a full revolution, their time is negligible compared to the low thrust phase and is not considered.

The third stage high thrust program applies a single impulse in the direction of the primer vector at the location of its maximum value on the final orbit of the low thrust phase. This produces an optimum single impulse transfer to the terminal state.

The overall trajectory is optimized by a shooting method. Initial values of the unspecified states and costates or functions thereof are chosen at the initial time. An optimum high and low thrust trajectory is then generated by integrating the state and costate through all three stages. This will generate an optimal trajectory to the wrong terminal state. A sensitivity matrix is then generated by varying the initial conditions and running a set of neighboring trajectories. A Newton iteration on the initial conditions is then used to drive the terminal errors to within specified bounds. The final converged trajectory is then a minimum time trajectory for the specified velocity increments in the high thrust phases.

The NEP program and the SEP program have separate blocks of certain subprograms, while also sharing several others. The state for NEP includes the orbital elements while for SEP, mass and fluence are also included in the state.

I. 3 Equinoctal Orbital Elements

The low thrust trajectory calculations are done in equinoctial coordinates (Ref. 4). When trajectory information is printed both classical and equinoctial elements are included. The costate includes the adjoints to the equinoctial orbital elements. Adjoints to the classical elements are generally not calculated or printed.

The equinoctial orbital elements are defined in terms of the classical elements by the following equations.

$$a = a$$
$$h = e \sin(\omega + \Omega)$$
$$k = e \cos(\omega + \Omega)$$
$$p = \tan\frac{i}{2} \sin\Omega$$
$$q = \tan\frac{i}{2} \cos\Omega$$

where a is the semimajor axis (in the program output the equinoctial a is usually given in earth radii and the classical a in kilometers), e is the eccentricity, i is the inclination, Ω is the longitude of the ascending node, and ω is the argument of perigee. The classical elements are in terms of an earth equatorial coordinate system with the x axis toward the vernal equinox and the z axis through the N pole. The equinoctial coordinate frame is defined by unit vectors $\underline{\hat{f}}$, $\underline{\hat{g}}$, $\underline{\hat{w}}$ illustrated in Fig. 1. and defined by

$$\underline{\hat{f}} = \frac{1}{1 + p^2 + q^2} \begin{bmatrix} 1 - p^2 + q^2 \\ 2pq \\ -2p \end{bmatrix}$$

$$\underline{\hat{g}} = \frac{1}{1 + p^2 + q^2} \begin{bmatrix} 2pq \\ 1 + p^2 - q^2 \\ 2q \end{bmatrix}$$

$$\underline{\hat{w}} = \frac{1}{1 + p^2 + q^2} \begin{bmatrix} 2p \\ -2q \\ 1 - p^2 - q^2 \end{bmatrix}$$

The position in an orbit can be indicated by the eccentric longitude, F, where

$$F = E + \omega + \Omega,$$

and where E is the classical eccentric anomaly.

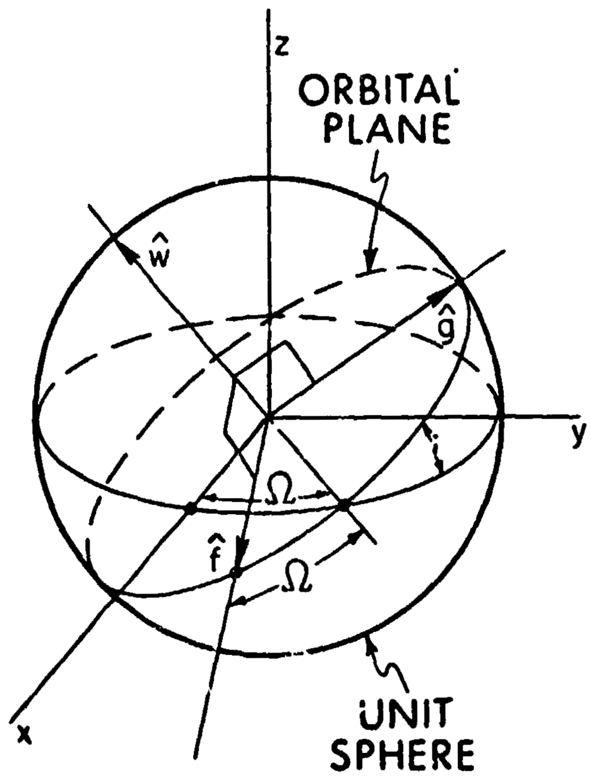


Figure 1 Direct Equinoctial Coordinate Frame

I. 4 Options

The options listed here involve those associated with the physical problem considered, i.e. the nature of the trajectory and orbits, the propulsion system, and physical effects which may be included, and not the options involving variations in the numerical computation techniques. The latter are indicated by the listing of the input variables (Sec. IV) and the comments on their selection in Sec. VI.

1) Low thrust may be SEP or NEP. The decks are slightly different.

2) Initial high thrust may be used with SEP or NEP. Total ΔV for the initial high thrust is an input.

3) A final impulse may be included; the ΔV is an input.

4) Either a, e, i, Ω, ω or just a, e, i may be specified at the final time. If high thrust is included the final Ω and ω are free.

5) If there is no initial high thrust, the a, e, i, Ω, ω for the initial orbit are input. If initial high thrust is included, the initial orbit must be circular; the initial Ω may be selected (with SEP, NEP) or left free (for SEP).

6) Initial power may be selected.

7) Specific impulse may be selected.

8) Initial mass may be selected.

9) Oblateness effect may be included or not included.

10) For NEP, power may be constant or may degrade exponentially with a user supplied time constant.

11) For SEP, the effect of Van Allen radiation on power output may be included or power may be constant (the latter was included mainly for test purposes).

12) A value for housekeeping power may be included.

13) For SEP, the shadow effect may be included or not.

14) For SEP, power may vary with spacecraft distance from the sun, or else the distance may be assumed to be constant at 1 A. U.

SECTION II
THE PROGRAM DECK

There are three blocks of subprograms, one for NEP, one for SEP and a block of subprograms which both NEP and SEP have in common. High thrust may be used in combination with either NEP or SEP. The SEP program includes additional state and costate variables and has several more subprograms than the NEP deck, because of the shadowing and degradation effects. Thus size and run time is considerably greater for the SEP deck.

The program is coded in Fortran IV in double precision for use on an IBM 360 computer and has been compiled using a G compiler. Following is a list of subprograms with a brief description for which the SEP and the NEP decks have separate versions, but which perform essentially the same function.

INPUT, reads and prints input data, sets initial conditions

OUTPC, prints summary of converged trajectory characteristics

ITER, either the Newton-Raphson or Modified Newton-Raphson iterator

PRTN, prints information at each iteration of ITER

DCROUT, essentially inverts a matrix

TRAJ, calculates a single trajectory by calling the high thrust subroutines and the low thrust differential equation integrator, then calculates error vector

OUTP, prints information at each time step of low thrust integrator

FUNCT, calculates low thrust averaged derivative for integrator

The following subprograms are shared by the NEP and SEP decks.

OBLATE, calculates single averaged effect of oblateness on state and costate

QUAD, either 4, 8, 16 or 32 point vector gaussian quadrature

FCT, EVALMP, calculate the derivative due to thrust for state and costate before averaging

MAINE, the main calling program for the initial high thrust calculation

START, called by MAINE, sets initial values of S array

TIME, iterates on ΔV to satisfy Small's optimality condition

SWITCH, computes the coast angle

DTDU, updates the S array after an impulse or a coast

OUTHI, prints initial high thrust orbit characteristics, also an interface with following low thrust phase

IMPLS, calculates effect of a single final impulse

YF, calculates primer vector for IMPLS

CONTL, main controlling and calling program for SEP and NEP decks

MAINE, START, TIME, DTDU are taken directly from Small's thesis (Ref. 5). MAINE was altered slightly, the other subprograms have not been changed at all. For further information and discussion concerning these subprograms see Ref. 5.

The following subprograms are only part of the SEP deck.

EARTH, sets certain constants associated with the earth rotation, revolution and magnetic field

SUN, calculates sun's direction and magnetic field orientation

SHADOW, calculates shadow entrance and exit angles (if any) and certain associated partial derivatives

DQRTIC, solves a quartic equation

DCUBIC, solves a cubic equation

FLUX, calculates flux effect on state and costate

In addition to the above subprograms, the IBM Scientific Subroutine Package Runga-Kutta (DRKGS) or Predictor-Corrector (DHPCG) integrators are required (Ref. 8) A chart showing the relation between the various subprograms is shown in Fig. II-1 for the NEP deck and in Fig. II-2 for the SEP decks. An overall flowchart and flowcharts of some of the main calculations are given in Appendix A.

Additional detail on the above subprograms is given in the final report.

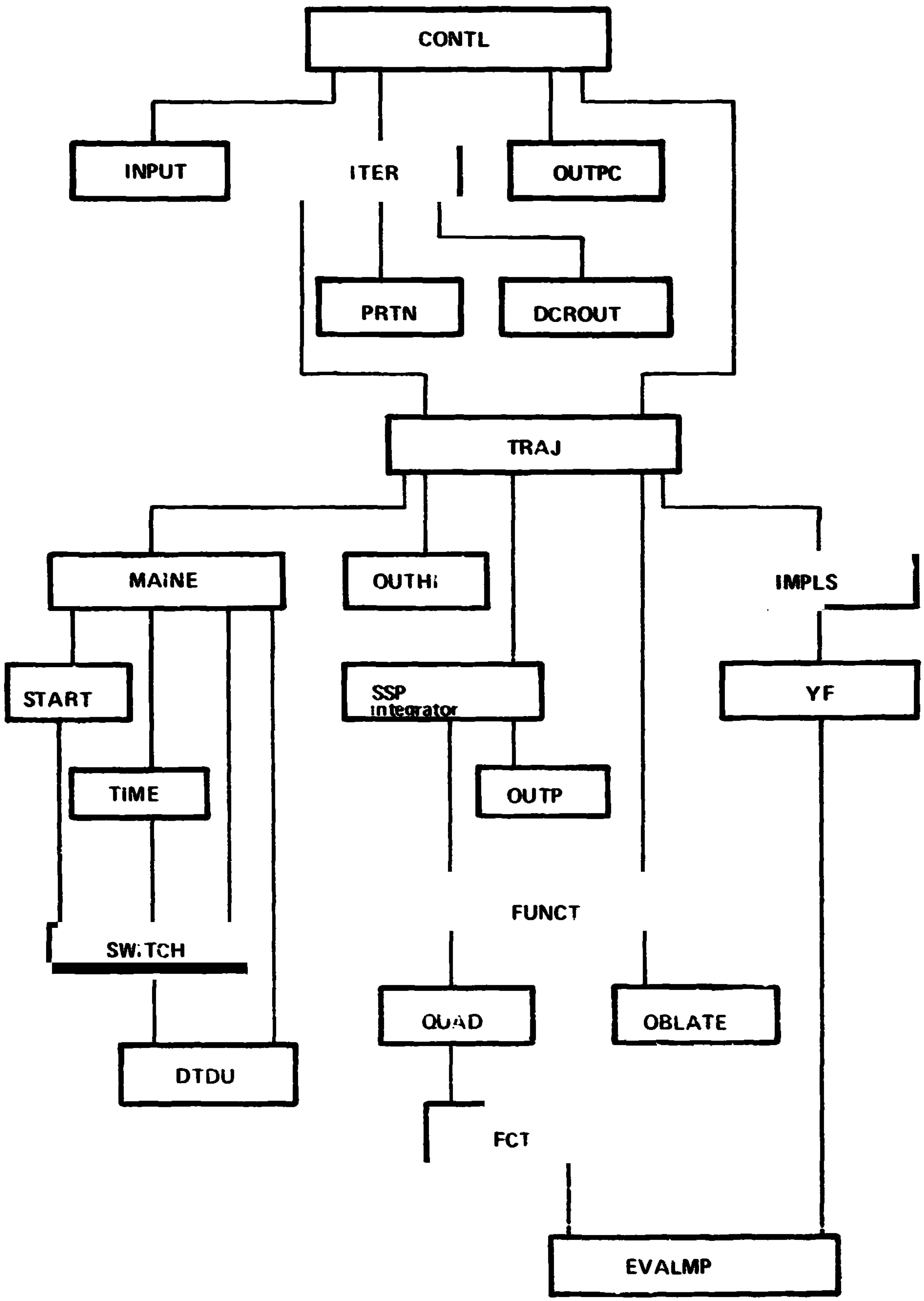


Fig. II-1 NEP System Diagram

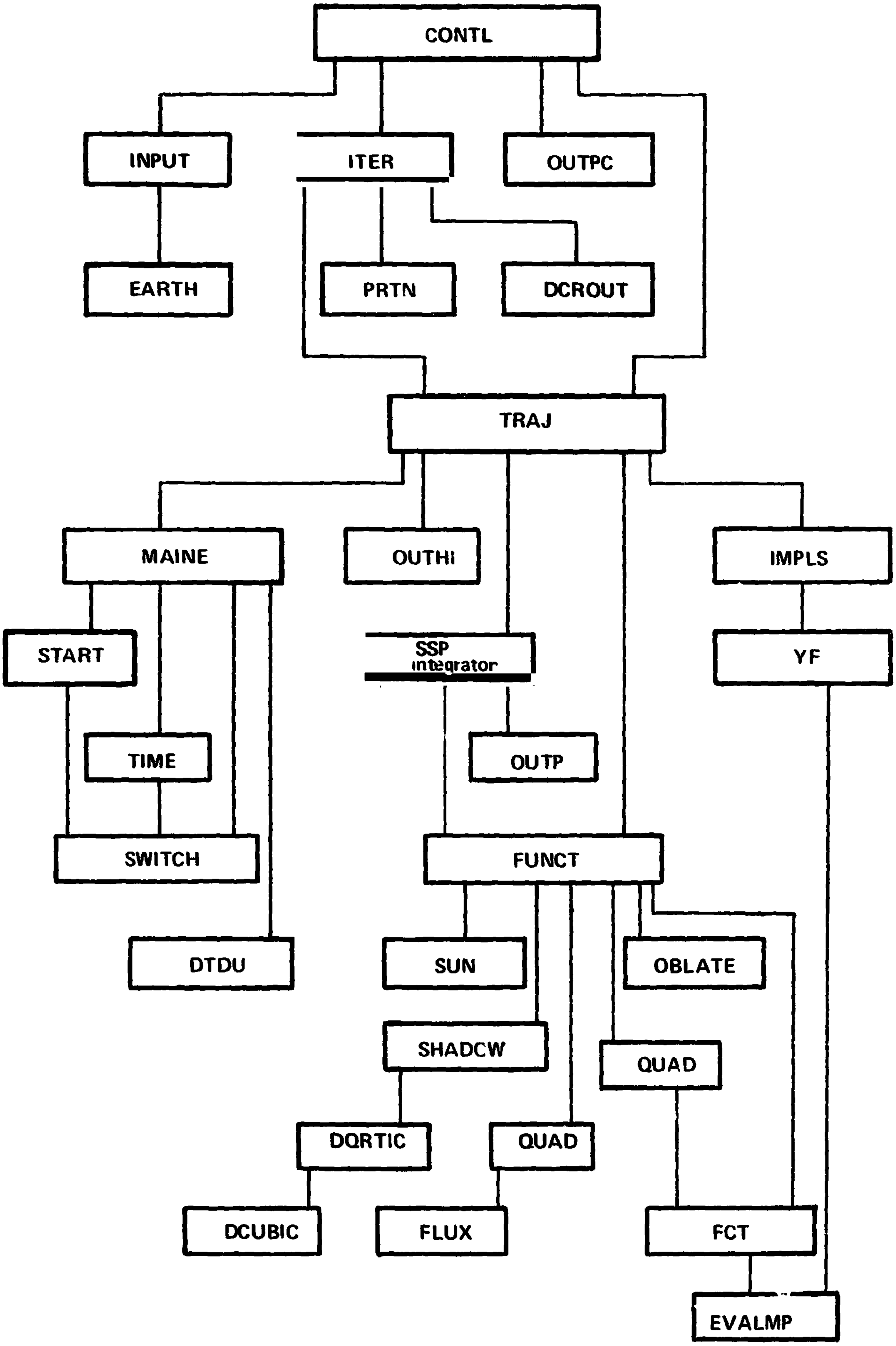


Fig. II-2 SEP System Diagram

SECTION III

CONSTANTS AND CONVERSIONS

The following are some constants and initial values which are assumed.

equatorial earth radius = 6378.16 km (Ref. 9)
earth gravitation coefficient = 398601.2km^3/sec^2 (Ref. 9)
oblateness J_2 = .0010827 (Ref. 9)

Earth's orbital elements (epoch JD=2436935.0) (Ref. 10)

a = 1 A.U.
e = .016726
ω = 102°25253
mean orbital motion = .985609°/day (Ref. 10)
obliquity of ecliptic = 23°45 (Ref. 10)
earth rotational frequency = 359°0170416/day (Ref. 10)
Latitude of north magnetic pole = 78°6 (Ref. 11)
Longitude of north magnetic pole = 289°9 (Ref. 11)

Internal units are in equatorial earth radii, 10^3 kg., and internal time units calculated such that a circular orbit at 1 earth radii would have a period of 2 π internal time units. In this system of units the gravitational coefficient, μ = 1. Other conversions can be derived from these basic equivalences. For example,

units to seconds = 806.8147206095579
units to days = .009338133340388 4
units to kilowatts = 77458.55283702227

SECTION IV

INPUT

IV.1 Introduction

The quantities discussed in this section are all read by the subprogram INPUT. Unless otherwise indicated each value is read on a separate line, real variables in fixed format (F25.15), integer variables, beginning with i, j, k, 1, m, n are read in format, I2. The input for the NEP deck and the SEP deck are listed in this section along with a brief description and nominal values, if any.

IV.2 Input for NEP Deck

Initial orbit characteristics,

W(1) km, semimajor axis

W(2) eccentricity

W(3) (degrees), inclination

W(4) (degrees), longitude of ascending node, not used if initial high thrust

W(5) (degrees), argument of perigee, not used if initial high thrust

Initial guesses for iteration parameters

ZL0(1) λ_a, adjoint to semimajor axis or if initial high thrust the transformed T (see Ref. 6)

ZL0(2) λ_h, adjoint to orbital element h or if initial high thrust the transformed Small variable k(see Eq. 6 of Ref. 6)

ZL0(3) λ_k, adjoint to orbital element k, or if initial high thrust, the transformed Small variable j (see Eq. 6 of Ref. 6)

ZL0(4) λ_p, adjoint to orbital element p, or if initial high thrust the scale factor relating high and low thrust costates

ZL0(5) λ_q, adjoint to orbital element q, if initial high thrust, not used

The desired final orbit

WF(1)	(km), semimajor axis
WF(2)	eccentricity
WF(3)	(degrees), inclination
WF(4)	(degrees), longitude of ascending node, not used if NOP=2
WF(5)	(degrees), argument of perigee, not used if NOP=2
TF2	(days), guess for final time
PKW	(kw), electrical power including efficiency factor
SPIM	(sec), specific impulse of NEP
AM0	(kg), initial mass (NEP)

The following input may be read or, optionally, left at nominal values. IRDFLG is read followed by the additional input or operations and then IRDFLG is again read, until IRDFLG = 01 and input is ended.

IRDFLG		NOMINAL
1	End of input	
2	IPR, print flag	0
3	NIMAX, Max. no. of iterations (if 0, by pass iteration to print time history)	20
4	TFMAX2, (days), max. TF	190.
5	DT2, (days), time step for integrator	1.
6	UEB, upper error bound for integrator	1.D10
7	EW, Error weights (Format 5D6.1)	1., 1., 1., 1., 1., 0., . . .
8	UTKM, Equatorial earth radius	6378.16
9	GM, (km^3/sec^2), earth grav. coefficient	398601.2
10	NOP = 1, five orbital elements specified at TF, use only if IHI=1	1
	= 2, three orbital elements specified at TF	

11	Sets oblateness coeff, AJ2, = 1.0827D-3	0.
12	STEP, Step size for numerical differentiation in ITER	1.D-6
	KSTEP = 0,STEP as fraction in ITER	
	= 1,STEP as constant in ITER	
13	IPOW = 0, constant power	0
14	= 1,exponential degradation	
	BB, (sec), time constant for degradation	
	PH, (kw), housekeeping power if IPOW = 1	
14	EMPTY	
15	IHI = 1, low thrust only	1
	= 2, high/low	
	= 3, high/low/high	
	= 4, low/high	
	DVI1, (m/s), total initial high thrust ΔV	0.
	DVI2, (m/s), ΔV for final impulse	0.
16	EMPTY	
17	FLIM, Norm limit in ITER routine	1.D-6
18	SGN = -1, if initial λ_i is negative	SIGN (WF(3) - W(3))
	= +1, if initial λ_i is positive	

Additional information concerning the meaning of these variables and suggestions in setting their values is given in Sec. VI.

IV.3 Input for SEP Deck

Initial orbit

W(1)	(km) semi major axis
W(2)	eccentricity
W(3)	(degrees) inclination
W(4)	(degrees) longitude of ascending node, not used if any high thrust and NODE = 1
W(5)	(degrees) argument of perigee, not used if any high thrust
W(6)	(kg) mass at beginning of low thrust stage
W(7)	(10^{14} equivalent 1MEV electrons/cm^2) initial fluence

Initial guesses

ZL0(1)	λ_a, adjoint to semi major axis or if initial high thrust the transformed Υ (see Ref. 6)
ZL0(2)	λ_h, adjoint to orbital element h or if initial high thrust the transformed Small variable k (see Eq. 6 of Ref. 6)
ZL0(3)	λ_k, adjoint to orbital element k, or if initial high thrust the transformed Small variable j (see Eq. 6 and Ref. 6)
ZL0(4)	λ_p, adjoint to orbital element p, or if initial high thrust the scale factor relating high and low thrust costates
ZL0(5)	λ_q, adjoint to orbital element q, or if initial high thrust, the long. of node (radians) if NODE= 0, or adjoint to long. of node if NODE= 1
ZL0(6)	λ_m, adjoint to mass
ZL0(7)	λ_N, adjoint to fluence

The desired final orbit

WF(1)	(km) semimajor axis
WF(2)	eccentricity
WF(3)	(deg) inclination
WF(4)	(deg), longitude of ascending node, not used if NOP= 2
WF(5)	(deg), argument of perigee, not used if NOP= 2
TF2	(days), guess for final time
PKW	(kw), electrical power at 1 A. U. including efficiency factor
SPIM	(sec), specific impulse of SEP
TL	Julian date at initial time

The following input may be read or, optionally, left at nominal values. IRDFLG is read followed by the addition input or operations and then IRDFLG is read again until IRDFLG = 01 and input is ended.

IRDFLG			NOMINAL
1	End of Input		
2	IPR	print flag	0
3	NIMAX	max. no. of iterations (if 0, bypass iteration to print time history)	20
4	TFMAX2	(days), max. TF	190.

5	DT2	(days), time step for integrator	1.
6	UEB	upper error bound for integrator	1.D10
7	EW	error weights for integrator (7D6.1)	1., 1, 1, 1, 1, 0, ...
8	UTKM	equatorial earth radius (km)	6378.16
9	GM	(km^3/sec^2) earth grav. const.	398601.2
10	NOP	= 1, five orbital elements specified at TF	1
		use only if IHI = 1	
		= 2, three orbital elements specified at TF	
11	Sets oblateness, AJ2, = 1.0827D-3		0.
12	STEP	step size for numerical differentiation in ITER, 8 dim., eighth element for time variation of Hamiltonian	1.D-6
	KSTEP	= 0, step as fraction in ITER	0
		= 1, step as constant in ITER (except STEP (8))	
13	ISON	= 0, shadow effect off	0
		= 1, shadow effect on	
14	ISUN	= 0, sun distance effect on power off	0
		= 1, effect on	
	PH	(kw) housekeeping power	0.
15	IHI	= 1, low thrust only	1
		= 2, high/low	
		= 3, high/low/high	
		= 4, low/high	
	DVI1	(m/s) total initial high thrust ΔV	0.
	DVI2	(m/s) ΔV for final impulse	1.
	NODE	= 0, initial line of nodes free, λ_Ω fixed	
		= 1, initial line of nodes fixed, λ_Ω free	
16	IPOW	= 0, constant power	1
		= 1, degradation effect	
17	FLIM	norm limit in iteration routine	1.D-6
18	SGN	= -1, if initial λ_i is negative	SIGN (WF(3)-W(3))
		= +1, if initial λ_i is positive	

IV.4 Comments

As coded at the time of this writing the following limitations exist. For NEP and NOP=2 (final ω and Ω free), the final eccentricity and inclination should not be set to zero; they may be set to small numbers (e.g. $e_f = .0001$, $i_f = .01°$). For NEP and initial high thrust, the final inclination should not be set equal to the initial inclination. This may also cause difficulties for SEP and initial high thrust. When initial high thrust is included, ZL0(1) and ZL0(2) (i.e. τ and j) should not both be set to zero. Final inclination should not be set to 180^{o}. At this writing there has been little experience with inclination greater than 90^{o}. Also the coded equations are not valid for eccentricities greater than or equal to one.

SECTION V
OUTPUT

Most of the output is self-explanatory and a look at an example will familiarize the user with it. There are certain basic groups of output. The first is the printing of the read-in initial data and a few internally set constants. Normally this will be followed by output from the iterator. After convergence, a summary of characteristics of the converged trajectory is printed. Finally, a time history of the converged trajectory will be printed. Usually, even if convergence was unsuccessful, a time history of the last trajectory to be calculated will be printed.

The printing of the initial data should be understandable. There are a few abbreviations used.

A,	semi-major axis
E,	eccentricity
I,	inclination
LON ASC NODE,	longitude of ascending node
ARG PERIG,	argument of perigee
SPEC IMP,	specific impulse
EXH VEL,	exhaust velocity
M/S,	meters/second
E. R. /T. U.,	earth radii/time unit
UTKM,	internal units to kilometers
UTS,	internal units to seconds
UTD,	internal units to days
UTKG,	internal units to kilograms
UTKW,,	internal units to kilowatts
UTMS2,	internal units to meters/sec^2

After the initial input print, the iteration begins. The iteration number (ITER NO.) and the total number of calls to TRAJ are printed followed by X, the iteration parameters (ZL0), then Y, the error in the final conditions. The final conditions are the final values of a, h, k, p, q if

NOP = 1, or a, e, $\tan\frac{i}{2}$, λ_Ω , λ_ω if NOP = 2, λ_m, λ_n for SEP, and finally the Hamiltonian. Then the final time (TF) is printed in internal units, followed by, F0, the sum of the squares of the errors in the final conditions. For convergence this value must be less than FLIM, the "norm limit in ITER". In order to calculate the partial derivative matrix or sensitivity matrix the nominal valu "X" are changed slightly by inputed amounts; these perturbed v of X (X(I) + DX(I)) are next printed followed by the corresponding Y. the partial derivative matrix is printed as well as its determinent. This matrix is inverted and premultiplies the error vector to obtain the changes in the X's, DELX:S, which are next printed.

A new trajectory is calculated and the sum of the squares of the errors in the final conditions is printed (F1). If this is smaller than F0, a new iteration begins; if it is larger than F0, the DELX:S are halved and printed. This continues until F1 < F0 or until a certain number of halvings. What follows depends on how well the method converges and on whether the Newton-Raphson or modified Newton-Raphson subprogram is used. Further output is basically permutations of the above, terminating with convergence or a message indicating lack of success.

After exit from the iteration, a summary of characteristics of the last trajectory (the optimal, if convergence was successful) is printed. Included are the actual final orbital elements, the error in the final orbital elements, the values for the iteration parameters, the final time, the equivalent particles (fluence) in units of 10^{14} (for SEP), the final mass, the ratio of final to initial mass, the final power, the ratio of final to initial power and the total low thrust ΔV (DELV).

Next is printed a time history of the final (optimal if convergence was successful) trajectory. If NIMAX = 00, then a time history is printed immediately following printing of the input data, bypassing the iteration routine, and summary print. If the trajectory includes initial high thrust impulses, the orbit number is printed (ORBIT =) followed by "EQUINOCTIAL O. E. AND COSTATE/S. F. 1000", after which the equinoctial orbital elements (a, h, k, p, q) and the equinoctial costate divided by the scale factor x 1000, which relates the high and low thrust costates and is an iteration parameter. Following "CLASSICAL O. E." are printed the classical orbital elements (a(km), e, i (°), Ω(°), ω(°)). Also printed is the true anomaly at which the last impulse occurred, and ϕ, τ and Δ V where the thrust direction is given by

$$\bar{\beta} = \sin \phi \, \underline{e}_R + \cos \phi \cos \tau \, \underline{e}_L + \cos \phi \sin \tau \, \underline{e}_h$$

where $\underline{e}_R$, $\underline{e}_L$, $\underline{e}_h$ are unit vectors, $\underline{e}_R$ along the radius vector, $\underline{e}_h$ perpendicular to the orbit and $\underline{e}_L = \underline{e}_h \times \underline{e}_R$ (Ref. 1, 5 and 6).

Next is printed the low thrust trajectory time history at each time step. First is printed TIME in various units. ΔV (DV(K/S)) in kilometers/sec and the time step number are also printed. Next is printed the equinoctial orbital elements (a, h, k, p, q) and mass (10^3 kg) and fluence (10^{14} particles) if SEP. Then classical orbital elements (a, e, i, Ω, ω) and mass (kg), power (kw), thrust (newtons), and thrust acceleration (meters/sec^2). Next is the costate, then the state derivative, then the costate derivative and then the value of the Hamiltonian, the period (hours), perigee and apogee (km) and the divisions of the time step performed by the integrator. For SEP with shadowing the time spent in shadow is printed in hours and as a fraction of the period (if the orbit passed through shadow). This print is repeated at each time step.

Finally, if a final impulse is included, equinoctial and classical orbital elements are printed for the final orbit, as well as the impulse direction and location in the equinoctial coordinate frame.

A number of error messages are scattered through the code. A few will be mentioned here. Several, in INPUT, call attention to bad input data. For bad input data following an IRDFLG value, a message, IRDFLG = (number), is printed. In some cases additional information is given. When shadowing is included, a message, ISHAD= 1, indicates that only one shadow crossing was found. This arises from small numerical inaccuracies in solving the quartic equation and can usually be ignored.

SECTION VI

COMMENTS ON INITIAL GUESSES AND OTHER PARAMETERS

Picking initial conditions is very important for running a program such as this. There are no built in values and the actual choices of the user can greatly influence the rapidity of convergence or, in some cases, if convergence occurs at all. As additional perturbations are added to the basic most simple problem, solution becomes more and more sensitive to the initial parameter choices, including those parameters of subprograms which affect numerical accuracy such as the integrator and its parameters (error bound, error weights, time step) and quadrature formula (8, 16, 32 point), and step size in the calculation of the numerically derived sensitivity matrix in the 2PBVP solutions.

The usual difficulty when looking at a new case will be picking the initial values for the costate (when initial high thrust impulses are included, some of the iteration parameters are functions of the actual costate) and the guess for the time of flight. Frequently ball-park values for these parameters will be known from previous similar cases. If nothing is known about the likely values it may be less costly to run a simpler case (e.g. without shadowing or oblateness) and with less accuracy (a lower point quadrature formula or a larger time step). However, if numerical accuracy is too poor, convergence will be affected. The converged values for such an example would then be input guesses for the more complex and more accurate case.

A less accurate solution might utilize a 8-point quadrature and time steps of from 3 to 6 days for low thrust accelerations of 10^{-4} g's with smaller steps for larger accelerations and vice/versa.

Previous experience has shown that convergence can be particularly difficult when shadowing is included. One useful technique in this case is to get convergence for a nominal case without shadowing (or without other perturbations which may be causing trouble) and then to add shadowing, and using the iteration parameter values from the nominal converged case as input to the shadow case, try to converge to a point along the nominal trajectory (using a corresponding final time). These new values for the iteration parameters can be used (again with a corresponding final time) to converge to a point further down the nominal trajectory. This process is continued until the desired final conditions are met. Three to six steps might be used. This procedure helps insure that the guessed initial trajectory is not too far from the desired extremal.

If the approximate ΔV is known, a final time estimate can be calculated assuming constant thrust acceleration.

$$\Delta V = a \cdot t_f$$

The initial guesses for the iteration parameters must be specified. These are ZL0(I), I = 1, 5 for NEP and ZL0(I), I = 1, 7 for SEP. For SEP (with or without high thrust) ZL0(6) is the initial adjoint of mass and ZL0(7) is the initial adjoint of fluence. Typical values are

ZL0(6) = - 5000.
ZL0(7) = - 100.

For SEP or NEP, without initial high thrust, ZL0(I), I = 1, 5 are the initial adjoints to the equinoctial orbital elements (a, h, k, p, q). ZL0(1) should always be non zero and positive for orbit raising. The others may have positive, zero or negative values, ZL0(2) and ZL0(3) with magnitudes usually less than 10^3. ZL0(4) and ZL0(5) with magnitudes usually less than 10^4. The signs depend on the values of h, k, p, q. Typical values are:

ZL0(1) = 3000.
ZL0(2) = -100.
ZL0(3) = 500.
ZL0(4) = -100.
ZL0(5) = -5000.

When initial high thrust is combined with SEP or NEP, ZLO(I), I = 1, 5 are no longer the adjoints to the orbital elements. Instead the first three elements are related to Small's variables $\mathcal{T}$, k, j (Ref. 1, 5, 6) by

$$\mathcal{T} = \frac{\pi}{2} \frac{ZL0(1)}{\sqrt{1 + ZL0(1)^2}}$$

$$k = \cos \mathcal{T} \left(.75 + .25 \frac{ZL0(2)}{\sqrt{1 + ZL0(2)^2}}\right)$$

$$j = (1 + k \cos \mathcal{T}) \sqrt{\frac{\cos \mathcal{T} - k}{\cos \mathcal{T} + k}} \cdot \frac{ZL0(3)}{\sqrt{1 + ZL0(3)^2}}$$

The above transformation insures that $\mathcal{T}$, k, j are maintained within valid bounds for the initial circular parking orbit for all values of ZL0(I), I = 1, 3. ZL0(4) is a scale factor, actually relating the adjoints for the initial high thrust and the low thrust phases of a trajectory. This value will almost always be around 1.

For high thrust with NEP, ZL0(5) is not used but should be set to some arbitrary value such as 1. Two options exist for high thrust and SEP. If the initial Ω is fixed (NODE= 1) then ZL0(5) is proportional to the adjoint to Ω. Theoretically this should be zero if non-thrusting perturbations are axially symmetric, and in any case will typically have a magnitude less than 1.0. The other option is to let Ω be free (and therefore the initial adjoint to Ω is zero). ZL0(5) is then Ω. This option may be used if the non-thrust perturbations are not axial symmetric (about the geographic poles). This includes the effect of a tilted radiation field or shadowing. Ω is measured in radians and so ZL0(5) will typically be between $-\pi$ and $+\pi$. Convergence will be facilitated if the correct quadrant is known.

Generally, τ will have a magnitude less than 20° so that ZL0(1) will have a magnitude less than .3 or so. ZL0(2) is typically positive around 1. ZL0(3) typically has a magnitude around 1.

Typically values for the NEP program

- .1
1.0
.2
1.5
1.0 (not used)

for the SEP program

- .2
.4
- .1
.9
0.0

The above discussion simply gives an order of magnitude feeling of values for initial guesses. More information can be gained by looking at the definitions of the iteration variables and then looking at particular cases of interest. Also helpful is looking at special cases, e.g., zero eccentricity, constant thrust, for which analytic results are known. The values for similar cases or simplified cases are always useful.

Now a few words will be given on values for which built in nominals exist. There is nothing special about some of these nominals except that the constructors of the program used those values a lot. Comments on a few of these follow.

IPR is a print flag that would normally only be used if there was trouble. It causes all trajectories to be printed (the number of steps in the low thrust portion printed is equal to about IPR + 1). Normally, the final (converged) trajectory will be printed anyway.

NIMAX, the number of iterations in the N - R or modified N - R procedure is set to 20 and that is usually enough to get convergence. It can be set to zero, which will cause a printing of the trajectory determined by the values of the input data, bypassing the iterator entirely

TFMAX simply prevents integrations past that time of flight and can be left alone unless you expect the time of flight to be near or greater than 190 days.

DT2, the time step in days with a nominal of 1. Frequently, a different value will be needed depending on the problem.

UEB, the upper error bound for the integrator, nominally set to a high value so that it is usually never reached, allowing the user to determine accuracy by picking the time step. This value is supplied to the SSP integrator and tuning it seemed difficult.

EW, error weights supplied to the SSP integrator, nominally set to 1 for the five orbital elements and zero for the other variables. This is somewhat arbitrary, as the writer has had little experience adjusting these values.

UTKM, the earth's equatorial radius. If you don't believe 6378.16 km, you can change it.

GM, the earth's gravitational constant set to 398601.2 km^3/sec^2, likewise.

NOP, a flag, if set to 01 then the final conditions include all five orbital elements, specified by the input, if set to 02, then a, e, i are specified only. If any high thrust is included this is automatically set to 02. In this case, this option should not be set by the user.

Oblateness effect is off nominally. If included, J_2 is set to .0010827. As coded, a different value for J_2 would require a recompilation of INPUT.

KSTEP is a flag which indicates whether STEP, which is used to calculate the numerically determined sensitivity matrix in the iterator, is a fraction of the nominal value of the iteration parameters (KSTEP=0) or a fixed step size. Nominally, STEP as a fraction is specified with all elements of the array set to 10^{-6}. This was found to work pretty well for low thrust only cases. (If the magnitude of the nominal iteration parameter is near zero, the corresponding STEP is considered a step size rather than as a fraction.) When initial high thrust impulses are included a constant step size was more frequently used, especially when the iteration parameters were small in magnitude. Typical values were:

$$10^{-8},\ 10^{-4},\ 10^{-6},\ 10^{-5},\ 10^{-5},\ 10^{-3},\ 10^{-3},\ 10^{-6}$$

(The first five being relevant to the NEP program.) STEP (8) remains the fractional variation in the final time used to calculate $\frac{\partial H}{\partial t_f}$.
Changing these values by a couple orders of magnitude had little effect.

Nominally, shadow effect is not included.

Nominally, the sun distance affect on power is not included, i.e., a 1 A.U. distance is assumed. Housekeeping power is nominally zero.

Nominally, low thrust only is assumed. If initial or final high thrust impulses are required, the ΔV's must be given. For the SEP program, if initial high thrust is included, NODE must be set to 0 if Ω is free, or to 1 if Ω is fixed. Note that if the non-thrust perturbations are symmetric about the equator changing Ω should have no effect on the trajectory, so that if Ω is free, the sensitivity matrix will be singular. This would occur if a radiation model was used which was symmetric about the geographic axis and there was no shadow effect. IPOW is nominally set to 0 for the NEP program. This specifies constant power. The exponential degradation is specified by IPOW= 1. For the SEP program, IPOW is nominally set to 01, so the degradation due to radiation is included. A constant power option is available although porbably would not generally be used. If shadowing is not included, this is equivalent to a corresponding NEP case, but more expensive to run.

FLIM determines the accuracy of the actual final conditions which determine convergence. The nominal value of 10^{-6} would yield an error in each component of the final conditions of about 10^{-4}. (The "cost" which is compared to FLIM is just the sum of the squares of the errors in the final conditions in internal units, scaled so that typical values have an order of magnitude of zero.) This seems consistent with the general accuracy of the model used.

SGN is used only if there are initial high thrust impulses. It sets the sign of the adjoint to inclination. Nominally this is automatically set negative if the final desired inclination is less than the initial inclination or positive if the opposite is the case.

Because of the functional form of the power versous fluence relation, to avoid having to use a very small time step (at least at the beginning of an integration) or else having numerical difficulties, the initial fluence (W(7)) for the SEP input should be nonzero. If the user inputs 0., then a nonzero value will be set by the code. This value is equal to $\dot{N} \cdot T/2$ where $\dot{N}$ is the initial fluence derivative and T is the initial orbital period (after any initial impulses). In order to avoid numerical difficulties, especially when initial impulses are included, it is useful for the user to input some nonzero value for fluence which overides the internal calculated value. A typical value might be .01 (in units of 10^{14} particles).

At the time of this writing the experience using the final version of the code is limited. Few runs have been performed with the final impulse included. The version of the SEP code used for most runs contained a geographically axial symmetric radiation field which corresponds to modeling the geomagnetic axis as lining up with the geographic axis. This allows a larger time step to be used and effectively averages the spatial dependence of the field over one day. Convergence was sometimes adversely affected by including shadowing. Also when initial impulses were included, convergence was more difficult if the initial Ω was free. Most runs were performed with a fixed initial Ω.

As coded, when the predictor-corrector integrator is used the time step is forced to be an integer division of the final time and thus varies somewhat from one trajectory to another as the time of flight varies. The varying time step can cause varying numerical errors for neighboring trajectories, thus most runs, especially with the more complex SEP program have been performed with the Runga-Kutta integrator for which the time step was fixed. Poor numerical accuracy will adversely affect convergence.

REFERENCES

1. Final Technical Report on NASA Contract NAS 5-21791, to be published, C. S. Draper Laboratory, 1974.

2. T. N. Edelbaum, L. L. Sackett, H. L. Malchow, "Optimal Low Thrust Geocentric Transfer," AIAA Paper No. 73-1074, Lake Tahoe, 1973.

3. N. Kryloff and N. Bogoliuboff, Introduction to Non-Linear Mechanics, Princeton University Press, 1947.

4. P. J. Cefola, "Equinoctial Orbit Elements—Application to Artificial Satellite Orbits," AIAA Paper No. 72-937, Palo Alto, 1972.

5. H. W. Small, Minimum Fuel Time-Free Transfer Between Elliptic Orbits, Ph.D. Dissertation, Stanford University, 1972.

6. H. W. Small, "Minimum N-Impulse Time-Free Transfers Between Elliptic Orbits," AIAA J, Vol. 9 No. 4, April, 1971.

7. T. N. Edelbaum, "Minimum Impulse Transfers in the Near Vicinity of a Circular Orbit," J. Astronautical Sciences, Vol. 14, No. 2, pp. 66-73, March-April, 1967.

8. System/360 Scientific Subroutine Package Version IV, Programmers Manual, GH20-0205-4, IBM Corp., 5th Ed., 1970.

9. W. G. Melbourne et. al., Constants and Related Information for Astrodynamic Calculations, 1968, NASA Technical Report 32-1306, JPL, Pasadena, 1968.

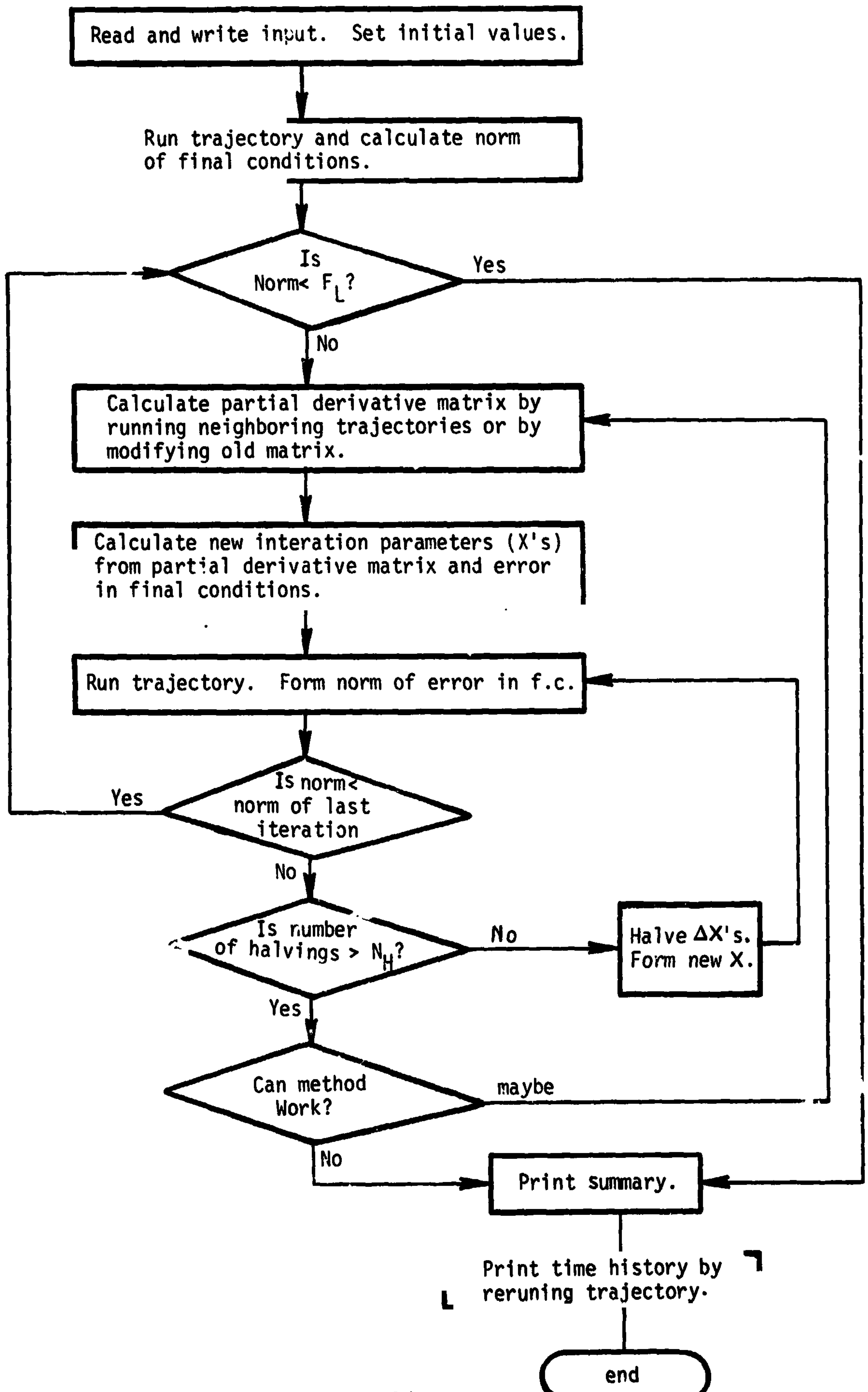
10. R. H. Battin, Astronautical Guidance, McGraw-Hill, New York, 1964.

11. V. M. Blanco and S. W. McCuskey, Basic Physics of the Solar System, Addison-Wesley, Reading, Massachusetts, 1961.

APPENDIX A

FLOW CHARTS

Three flow charts are included on the following pages. The first is an overall flow chart showing the general flow of the code. The chart is indicative of either the Newton-Raphson or modified Newton-Raphson iterator. The second chart shows the flow of the trajectory calculations, and the third shows the operations taken in calculating the averaged derivative which is called by the integrator routine which extrapolates the low thrust portion of the trajectory.


Read and write input. Set initial values.
Run trajectory and calculate norm of final conditions.
Is Norm< F_L?
Yes
No
Calculate partial derivative matrix by running neighboring trajectories or by modifying old matrix.
Calculate new interation parameters (X's) from partial derivative matrix and error in final conditions.
Run trajectory. Form norm of error in f.c.
Is norm< norm of last iteration
Yes
No
Is number of halvings > N_H?
No
Halve ΔX's. Form new X.
Yes
Can method Work?
maybe
No
Print summary.
Print time history by reruning trajectory.
end

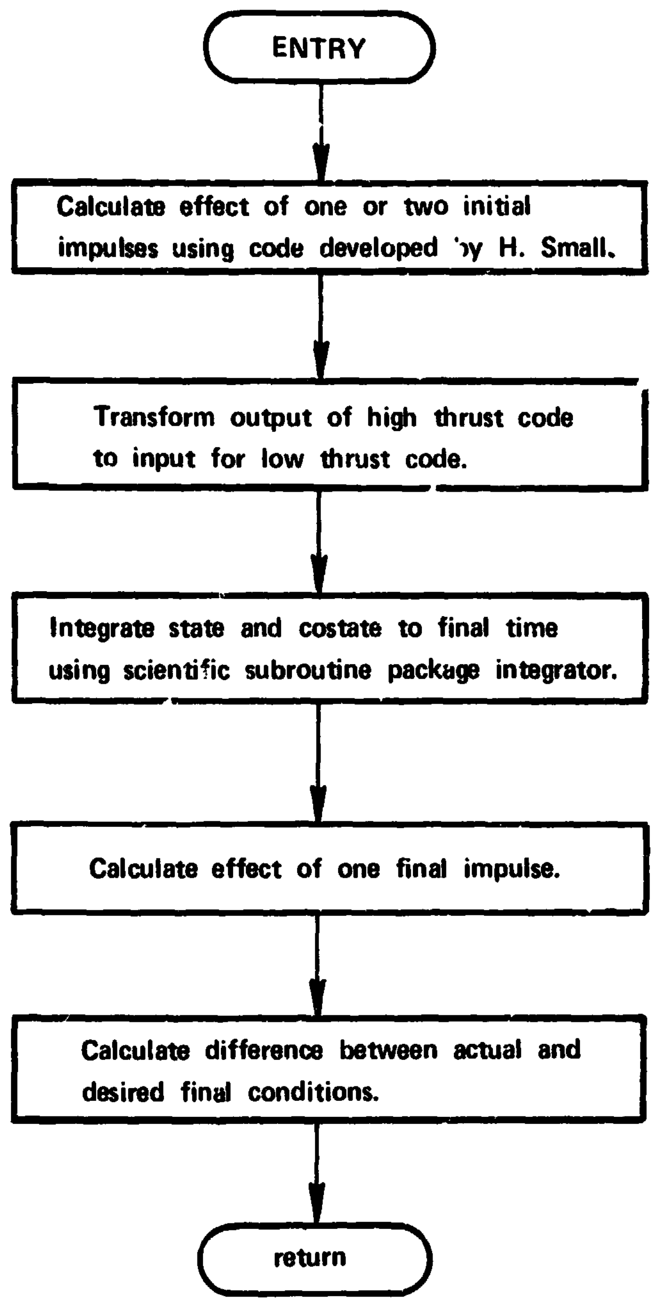


Fig. A-2. Trajectory Flowchart

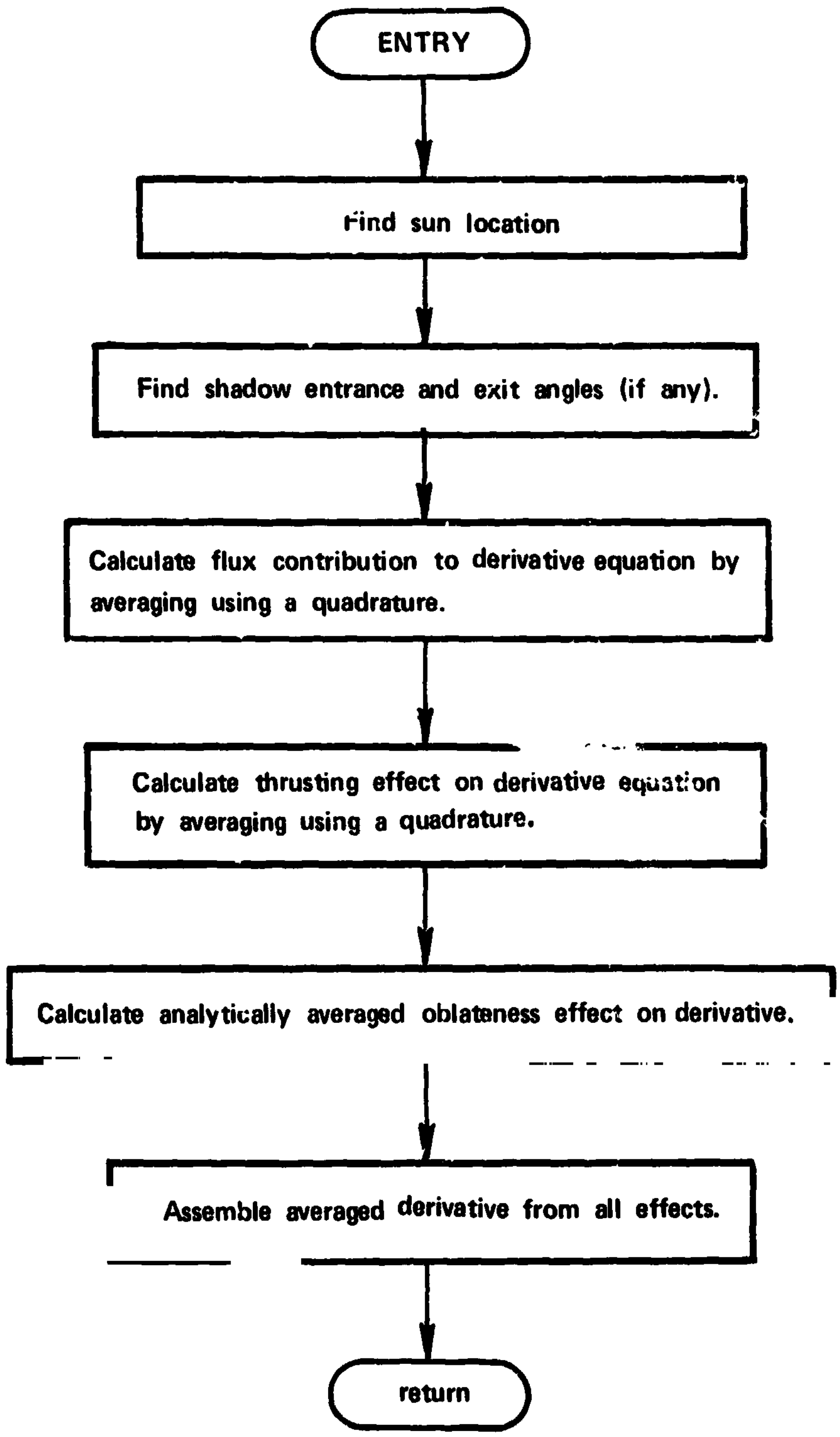


Fig. A-3. Averaged Derivative Calculation Flowchart

APPENDIX B

THRUST DIRECTION ON AN ORBIT

There is no provision in the main deck for printing the thrust direction at points on an individual orbit, prior to the averaging process. This print can be obtained for particular orbits, however, by using a separate main, calling, program, ORBIT, which uses a special version of FCT and EVALMP which contain print statements. The input includes:

Z (I), I = 1, 10, The 5 equinoctial orbital elements, a, h, k, p, q, and their adjoints λ_a, λ_h, λ_k, λ_p, λ_q which can be taken from the time history output of the optimization program.

A Thrust acceleration, may be set to 1.

N One half the number of points on the orbit for which print is desired.

NCONT, Flag, if 0, stop, if greater than 0 read data for another orbit.

Z and A are in format (F25. 15) and N, NCONT in format (I2).

The output is as follows. Beginning at F (the eccentric longitude) = 0°, at points 180°/N apart, the following is printed:

the eccentric longitude, F, the equinoctial coordinates, X1, Y1 (the Z component is zero) and the unit vector indicating the thrust direction in the equinoctial coordinate system.

APPENDIX C

EXAMPLE OF A SEP RUN

This appendix shows an example of actual output for a case including SEP and initial and final impulses. Only the beginning and the end of the iteration print and the time history print is included. The run utilizes modified Newton-Raphson iterator and a 16 point quadrature.

[illegible] TRAJECTORY PROGRAM FOR SATELLITE RAISING USING SEP AND HIGH THRUST

MINIMUM TIME

```
FINAL [illegible] OPTIONS
1. [illegible] SPECIFIED
2. [illegible] SPECIFIED, [illegible] AND ARG PER FREE
[illegible]

[illegible] EFFECT INCLUDED

[illegible] (M/S) = [illegible]     FINAL IMPULSE (M/S) =     [illegible]

THE INITIAL ORBITAL ELEMENTS ARE

     A (KM)            E                       I (DEG)         LON ASC NODE (DEG)      ARG PERIG (DEG)
 6.[illegible] 03   0.0                 2.[illegible] 01   0.0

INITIAL [illegible] =     [illegible] 00

INITIAL MASS (KG) =  [illegible]

INITIAL POWER (KW) =   1.[illegible] 01

[illegible] POWER (KW) =    C.C

INITIAL [illegible] =    1.[illegible]-03

[illegible] EFFECT ON POWER INCLUDED

THE DESIRED FINAL ORBITAL ELEMENTS ARE

     A (KM)              E                      I (DEG)
 [illegible] 04   [illegible]   [illegible] 01

     [illegible]          [illegible]         [illegible]
 [illegible]   [illegible]-01   1.[illegible]-01

INITIAL [illegible] PARAMETERS ARE
 [illegible]-02   [illegible]-01   -5.[illegible]-01   1.[illegible] 00   1.[illegible]-01
-1.[illegible] 0[illegible]   -2.[illegible] 0[illegible]

[illegible] =   -1.C

FINAL TIME ESTIMATE
      1.[illegible] 01 DAYS = 1.[illegible] 06 SECONDS = 1.[illegible] 03 UNITS

[illegible] =    [illegible] SEC, [illegible] VEL =    2.[illegible] 06 M/S =    [illegible] 00 [illegible]

INITIAL [illegible] =   5.[illegible] M/S**2

[illegible] DATE AT INITIAL TIME IS     [illegible]

[illegible] =   [illegible]

INITIAL TIME IS
      C.C                 DAYS = 0.0                  SECONDS = 0.C                  UNITS

[illegible] IS
      C.C                 DAYS = 0.0                  SECONDS = 0.C                  UNITS
[illegible] IS
      1.[illegible] 0[illegible] DAYS = 1.[illegible] 0[illegible] SECONDS = 2.[illegible] 0[illegible] UNITS

[illegible] KSTEP = 1
 1.[illegible]-0[illegible]   1.[illegible]-05   1.[illegible]-0[illegible]   1.[illegible]-05   1.[illegible]-05
 1.[illegible]-0[illegible]   1.[illegible]-06   1.[illegible]-05

TIME STEP FOR INTEGRATION
      2.[illegible] 0[illegible] DAYS = 1.[illegible] 05 SECONDS = 2.[illegible] 02 UNITS

[illegible] INTEGRATION =     1.[illegible] 10

[illegible]
1.[illegible] 1.[illegible] 1.[illegible] 1.[illegible] 1.[illegible] 0.0   0.0   0.0   0.0   0.C   0.0   0.0   0.C   0.C

[illegible] =    14

[illegible] =    [illegible]

[illegible] =   [illegible]

[illegible] RADIUS =        [illegible] KM

[illegible] =        [illegible] KM**3/SEC**2
[illegible]
[illegible] =  [illegible]     [illegible] =  [illegible]     [illegible] =  [illegible]     [illegible] =  [illegible]     [illegible]
 [illegible]     [illegible] =  [illegible]
[illegible]
```

[illegible]

```
ITER NO.   TRAJECTORY CALLS

    4          12

X
  9.4439237BC5529B30D-02  2.2747544572927990D-01 -5.7754897070097160D-01  1.2387222706547460 00  1.4847634035164540-01
 -1.C365C0552549607D 02 -2.51C6448560345670 02

Y
  4.0682919411589850D-03 -3.94247C2560696130D-04 -4.2408000251658320D-04 -2.9362340892589390D-04  8.9850628436607830-04
 -1.3790823270403790D-04  1.2381548570615660-05  1.7264350056304100-03

 TF=    1.17C7155114835830D C3

FO=  2.0989873742110390D-05

PARTIAL DERIV MATRIX
 -2.7532254269917500 01 -3.6624550980464600 00  4.0489858309914850 00  9.8030807731482400-02 -4.5850097942719710-02
 -1.8553287501755670D-04  4.5625352098461150D-04  1.3809103801306260-03
 -8.9075981397676570D-01 -9.1499587408539790-01 -7.8419542202831100-02  9.8195391204238680-03  3.3241144782358130-04
  1.5471544828823870D-06  4.87C4C589749888830-05 -1.2691685480302030-04
 -5.8424443571011750D-01  6.9338852199030300-02  6.8416466305731200-02 -1.7129004448181800-04  1.0300684497946560-02
  8.1194278874643881D-C6 -6.6655225608881830-06 -4.8395770334783150-05
  3.6008808753067610 00 -1.8345275512648330-01  4.4444006442138240-01  5.0074853474798960-05 -3.8396544977397670-01
 -3.2135573165277760-C6  4.1C72958C23607440-06  4.0637850126098830-05
 -5.0899124110448820 C0  2.18C8472671991670-01 -1.8140626010052360-01 -5.6983303579214460-04  1.0666244347488280 00
 -4.5573C9036513664D-06 -1.9437IC1549394110-05 -2.3903974426955940-04
 -7.8627622884725420 00  4.7405275776376960-01  8.2310619479981600-02  8.8374872816779080-01 -5.4722492202985460-02
  1.0390089242C629270-03  8.1499887345454680-06  1.0685858995544700-03
 -1.5007045940692760 C0 -6.6741FE5794154900-02  1.0052718136077070-01  1.9646221794961420-01 -7.8124528742C99490-03
 -1.2904731675245260-C5  9.9161857479258750-04  6.2140255245650090-05
 -2.0799578523638310 00 -4.4911682406077010-02 -9.8231142258814500-02  8.9261170715655240-01 -2.3190911149406470-01
 -1.5054815318799490-05  5.1454526641720340-04  3.7955746994414310-04

DETERMINANT = -9.2286759332896437E-10

DELTAS ARE
 -2.6096856661641220-C4
  1.2458120885454360-C3
  8.6752882211378659D-C4
 -4.3647410010350590-04
 -7.6864741445807160-C3
  5.7330854746075530 C0
  1.4399468182109270-C1

 CEN TF = -6.7895047223332970 CC

F1=  1.1575447133146045E-C8

ITER NO.   TRAJECTORY CALLS

    5          13

X
  5.4714760742913610-C2  2.2473586779482940-01 -5.7666143187779370-01  1.2382858365546420 C0  1.4478986620706450-01
 -1.C3076860207479SC 02 -2.5C9224747223357D 02

Y
  4.1480547158215410-C8  3.2815527881805564D-07  3.1655991386937360-06  9.6735463439046030-07  1.4556912626203160-05
 -7.1520544715776420-05 -2.0876058759375160-06 -6.7077055131148360-05

 TF=    1.1634264067604590 C3

F0=  1.1575447133146045E-C8

F0= 1.1575447133146045E-C8CASE CONVERGED...PERIG
```

```
CONVERGED VALUES FOR OPTIMIZED TRAJECTORY

ACTUAL FINAL ORBITAL ELEMENTS ARE

        A (KM)                   E                        I (DEG)            LON ASC NODE (DEG)        ARG PERIG (DEG)
3.5740244569846900 04  5.6000372815548840-01  1.8210353644688650 01 -1.1659153134752080 01 -8.5815116548318580 01
   P (EARTH RAD)                 H                          K                         P                         Q
5.6037876412872C710 CC -5.8874271403763880-01 -7.2845369960001300-02 -3.2388164400220000-02  1.5695995856C15570-01

THE ERROR IN THE FINAL O.E. IS (ACTUAL - DESIRED)

        A (KM)                   E                        I (DEG)            LON ASC NODE (DEG)        ARG PERIG (DEG)
2.6485954690C435620-01  3.2815527881805560-07  3.5364468865139990-04

   P (EARTH RAD)           SQRT(H**2+K**2)           SQRT(P**2+Q**2)
4.1480547196215410-C8  3.2815527881805560-07  3.1655991386937360-06

THE CONVERGED INITIAL GUESSED PARAMETERS ARE
5.4714765242513610-02  2.2473586779482940-01 -5.7666143187779370-01  1.2382858365546420 00  1.4478986620706450-01
-1.0307686020747590 C2 -2.5092247472233570 C2

THE MINIMIZED FINAL TIME IS
          1.0864230912058618D C1 DAYS = 9.3866955134005470 05 SECONDS = 1.1634264067604490 03 UNITS

EQUIVALENT PARTICLES (*1.D-14) = 8.0252362891183220-01

FINAL MASS =   972.9943823C871C7CC KG, FRACTION OF INITIAL MASS =      0.972994382306

FINAL POWER =    13.72 7341748515S0 KW, FRACTION OF INITIAL POWER =      0.933982049050279

LOW THRUST DELV=    8.0473864848888860-01 KM/SEC
```

```
TIME HISTORY OF OPTIMAL TRAJECTORY

**************************************************

HIGH THRUST

**************************************************

ORBIT = 1

EQUINOCTIAL O.E. AND COSTATE/S.F.=1000.
  [illegible] 00   0.0                  0.0                  0.0                  2.5356[illegible]-01
  2.90[illegible]-01   -1.[illegible]-01   -5.2965608[illegible]-02   0.0                  -6.059854268[illegible]-01

CLASSICAL O.E.
  6.[illegible] 03   0.0                  2.8500000000000 01   0.0                  -1.0603943548[illegible] 02

      TRUE ANOMALY          PSI (DEG)            UPSILON (DEG)        DELTA V (KM/S)
  0.0                  -7.[illegible]-01   [illegible] 00   0.0

ORBIT = 2

EQUINOCTIAL O.E. AND COSTATE/S.F.=1000.
  [illegible] 00   -7.0277[illegible]-01   -1.99786281471030D-01   -1.043453717921[illegible]-02   2.50939536112016D-01
  [illegible]-02   1.[illegible]-01   4.33291[illegible]-02   2.5174592[illegible]-02   -6.05399C10995934D-01

CLASSICAL O.E.
  2.4364874112550[illegible] 04   [illegible]-01   2.4157172025662[illegible] 01   -2.3011855731110[illegible] 00   -1.0348631145[illegible] 02

      TRUE ANOMALY          PSI (DEG)            UPSILON (DEG)        DELTA V (KM/S)
  1.[illegible] 02   1.[illegible] 01   -5.3293210375061[illegible] 01   2.4658605703791[illegible] 00

ORBIT = 3

EQUINOCTIAL O.E. AND COSTATE/S.F.=1000.
  4.1406414215267[illegible] 00   -6.[illegible]-01   -1.2517839[illegible]-01   -5.34201[illegible]-02   2.07531373296113D-01
  [illegible]-02   1.[illegible]-01   2.7653467267844[illegible]-02   1.5038121964[illegible]-01   -5.8421436342[illegible]-01

CLASSICAL O.E.
  2.[illegible] 04   6.[illegible]-01   2.41906942572147D 01   -1.4435CC0312665[illegible] 01   -8.60C462379514[illegible] 01

      TRUE ANOMALY          PSI (DEG)            UPSILON (DEG)        DELTA V (KM/S)
  1.[illegible] 02   1.[illegible] 01   -4.665[illegible] 01   3.45747225620850D-01

**************************************************

**************************************************

LOW THRUST

*********************************************************************
TIME      TIME UNITS              SECONDS      HOURS        DAYS         DV (K/S)        N
          0.0                     0.0          0.0          0.0          0.0             1


THE EQUINOCTIAL ORBITAL ELEMENTS, MASS, AND FLUX ARE
  4.1406[illegible] 00   -6.[illegible]-01   -1.2517[illegible]-01   -5.34201536013[illegible]-02   2.0753137329[illegible]-01

  [illegible] 00   1.[illegible]-03

THE CLASSICAL O.E., MASS (KG), POWER (KW), THRUST (N), THRUST ACC (M/SEC**2)
  [illegible] 04   6.[illegible]-01   2.4190694257220 01   -1.44350C0312670 01   -8.6004823755[illegible] 01

  1.[illegible] 03   [illegible] 01   1.0225882581090 03   1.022588258109[illegible]-03

THE COSTATE IS
  [illegible] 02   1.[illegible] 03   3.4491C24565330 02   2.2342345174580 03   -7.1304660613410 03

  -1.0[illegible] 03   -2.[illegible] 02

THE DERIVATIVE OF THE STATE IS
  1.[illegible]-03   [illegible]-05   6.8955411516210-05   8.2629352258650-07   -3.6735923988850-05

  -2.[illegible]-05   4.[illegible]-04

THE DERIVATIVE OF THE COSTATE IS
  -1.[illegible]-01   1.[illegible] 00   2.797[illegible]-01   1.4090142911490-01   3.0340539337190-01

  [illegible]-01   [illegible] 00

     [illegible]           PERIOD (HRS)       PERIGEE (KM)         APOGEE (KM)       DIV. TIME STEP
  [illegible]        [illegible]      8.1653549729D 03     4.4653992005C 04        0


TIME SPENT IN SHADOW =     [illegible] HOURS,  FRACTION OF PERIOD =      0.16329752


ENTRY ANGLE =       101.[illegible] EXIT ANGLE =       141.0585776986

*********************************************************************
TIME      TIME UNITS              SECONDS      HOURS        DAYS         DV (K/S)        N
          [illegible] 02          1.7280000 05   4.8000000D 01   [illegible] 00   1.44704810-0[illegible]   4


THE EQUINOCTIAL ORBITAL ELEMENTS, MASS, AND FLUX ARE
  4.[illegible] 00   -6.[illegible]-01   -1.1111636850060-01   -5.2622604776870-02   1.9990454160070-01

  5.[illegible]-01   1.[illegible]-0[illegible]

THE CLASSICAL O.E., MASS (KG), POWER (KW), THRUST (N), THRUST ACC (M/SEC**2)
  [illegible] 04   6.[illegible]-01   2.3358753591[illegible] 01   -1.9747850849700 01   -8.4774117546[illegible] 01

  [illegible] 03   1.[illegible] 01   9.7[illegible]-01   9.756779726057D-04

THE COSTATE IS
  3.[illegible] 02   [illegible] 03   3.4131328432500 02   2.2738701311880 03   -7.1714578[illegible] 03

  -[illegible] 02   -7.[illegible] 01

THE DERIVATIVE OF THE STATE IS
  1.0[illegible]-03   8.2100[illegible]-05   6.2985155227570-05   6.4224905404330-06   -3.5361021024600-05

  -2.27[illegible]-05   5.[illegible]-04
```

```
TIME SPENT IN SHADOW =     1.4[illegible]257 HOURS,  FRACTION OF PERIOD =     0.08926182

ENTRY ANGLE =        117.0[illegible] EXIT ANGLE =       139.6517966244

*****************************************************************
TIME        TIME UNITS                SECONDS      HOURS          DAYS         DV (K/S)        N
          1.0100643D 03          8.1493478D 05  2.2637077D 02  9.4321155D 00  6.9359422D-01    6

THE EQUINOCTIAL ORBITAL ELEMENTS, MASS, AND FLUX ARE
  [illegible]D 00   -5.8417[illegible]2290-01   -6.8505473235140-02   -4.2628500292230-02    1.6663996711520-01

  9.[illegible]D-01    4.[illegible]27410-01

THE CLASSICAL O.E., MASS (KG), POWER (KW), THRUST (N), THRUST ACC (M/SEC**2)
  3.[illegible]D 04    5.[illegible]23960-01    1.5756659968710 01   -1.4169831860730 01   -6.2518605744160 01

  [illegible]D 02    1.[illegible]52380 01    9.3611470650660-01    9.5246583024110-04

THE COSTATE IS
  2.[illegible]1220 02    1.[illegible]6850 03    2.[illegible]180 02    2.1969876466780 03   -8.0510532946320 03

  -[illegible]D 02   -9.[illegible]14610 03

THE DERIVATIVE OF THE STATE IS
  [illegible]D-03    [illegible]9080-04    4.3071866833260-05    1.6124793814710-05   -4.5153743632460-05

  -2.[illegible]D-05    [illegible]04930-04

THE DERIVATIVE OF THE COSTATE IS
  -1.1171[illegible]2770-01   -6.[illegible]85930-02   -8.7738989478600-02   -5.0889850574930-01   -2.0643269962050 00

  5.[illegible]2240D-01    [illegible]5401D-02

     HAMILTONIAN        PERIOD (HRS)       PERIGEE (KM)        APOGEE (KM)     DIV. TIME STEP
   [illegible]52124      17.[illegible]47923    1.39918794290 04    5.39595042560 04        0

TIME SPENT IN SHADOW =     1.2[illegible]1839 HOURS,  FRACTION OF PERIOD =     0.07287314

ENTRY ANGLE =        120.2[illegible] EXIT ANGLE =       138.6725875256

*****************************************************************
TIME        TIME UNITS                SECONDS      HOURS          DAYS         DV (K/S)        N
          1.1[illegible]342646 03          9.3866955D 05  2.6074154D 02  1.0864231D 01  8.0473845D-01    7

THE EQUINOCTIAL ORBITAL ELEMENTS, MASS, AND FLUX ARE
  [illegible]D 00   -5.[illegible]6950-01   -6.2274740956200-02   -4.0131[illegible]90130-02    1.6164743567000-01

  [illegible]D-01    5.[illegible]91180-01

THE CLASSICAL O.E., MASS (KG), POWER (KW), THRUST (N), THRUST ACC (M/SEC**2)
  3.[illegible]D 04    5.[illegible]84340-01    1.8912135630860 01   -1.3942523866500 01   -8.2342055255000 01

  [illegible]D 02    1.[illegible]42590 01    9.3345980419910-01    9.5988203054430-04

THE COSTATE IS
  2.12[illegible]3230 02    1.[illegible]71760 03    2.1862024050600 02    2.0923117383640 03   -8.4274387420560 03

  -7.15[illegible]D-02   -2.0[illegible]D-03

THE DERIVATIVE OF THE STATE IS
  [illegible]D-03    1.2[illegible]1127420-04    3.8046595843040-05    1.6354612872430-05   -4.8776091987040-05

  -[illegible]D-05    [illegible]77360-04

THE DERIVATIVE OF THE COSTATE IS
  -5.[illegible]D-02   -7.[illegible]D-02   -1.7419451538160-02   -9.0[illegible]8970524730-01   -2.9606356132230 0[illegible]

  1.0[illegible]D 00    5.41[illegible]46520-02

     HAMILTONIAN        PERIOD (HRS)       PERIGEE (KM)        APOGEE (KM)     DIV. TIME STEP
   [illegible]       1[illegible]5400    1.52215241320 04    5.53942457580 04        0

TIME SPENT IN SHADOW =     0.97[illegible]2576 HOURS,  FRACTION OF PERIOD =     0.05326571

ENTRY ANGLE =        123.771[illegible] EXIT ANGLE =       137.5940384435
************************************************

HIGH THRUST     FINAL ORBIT

************************************************

EQUINOCTIAL O.E.
  5.603[illegible]413232030 00  -5.[illegible]40376390-01  -7.2845369960001000-02  -3.2388164402200D-02   1.5699950580156D-01

CLASSICAL O.E.
  3.[illegible]D 04   5.[illegible]1554280-01   1.8[illegible]35364468870 01  -1.1659153134752 10 01  -8.5815116540318 6D 01

ECCENTRIC LONG. (DEG) =      145.4605393963

RADIUS VECTOR =   -[illegible]D 03    5.8467768642960[illegible]0 03    0.0

IMPULSE DIRECTION =   -[illegible]1962530-01  -3.0883509257587 0D-01   8.0037456 3500886 0D-01

************************************************

PROGRAM HAS RUN SUCCESSFULLY
```

APPENDIX D

EXAMPLE OF A NEP RUN

This appendix shows an example of actual output for a case including NEP and initial high thrust. Only the beginning and the end of the iteration print and the time history print is included. This run utilized the modified Newton-Raphson iterator, the predictor-corrector integrator, and the 16 point quadrature.

```
OPTIMAL TRAJECTORY PROGRAM FOR NEP AND HIGH THRUST SATELLITE RAISING

                         MINIMUM TIME                          NEP

FINAL CONDITION OPTIONS
1. ALL 5 FINAL ORBITAL ELEMENTS SPECIFIED
2. A,E,I SPECIFIED, LON ASC NODE AND ARG PER FREE
FOR THIS RUN, OPTION =   2

TOTAL DELV FOR INITIAL IMPULSES (M/S) =   1605.80390000000    FINAL IMPULSE (M/S) =    0.0

THE INITIAL ORBITAL ELEMENTS ARE

       A (KM)                  E                  I (DEG)            LON ASC NODE (DEG)     ARG PERIG (DEG)
 6.56347000000000 03   0.0                 2.85000000000000 01   0.0

INITIAL A (EARTH RADII) =     1.02905383370753D 00

THE DESIRED FINAL ORBITAL ELEMENTS ARE

       A (KM)                  E                  I (DEG)
 4.21581600000000 04   1.00000000000000D-04   1.00000000000000D-01

     A (EARTH RAD)          SQRT(H**2+K**2)        SQRT(P**2+Q**2)
 6.60976431444128D 00   1.00000000000000D-04   8.72664847521271D-04

INITIAL GUESS FOR HIGH THRUST, S.F., ONE
-1.61300000000000D-01   4.71300000000000D-01   -2.18400000000000D-01   1.18300000000000 00   1.00000000000000 00

SIGN OF INITIAL LAMBDA I IS  -1.0

FINAL TIME ESTIMATE
        4.30000000000000 01 DAYS = 3.71519999999999D 06 SECONDS = 4.60477448824711D 03 UNITS

SPEC IMP =   3.00000000000000 03 SEC, EXH VEL =   2.94197003521499D 01 KM/S

INITIAL MASS (KG) =  1000.00000000000

INITIAL POWER (KW) =   1.46970000000000 01

ACCELERATION LEVEL =    9.99976174203461 4D-04 M/SEC/SEC -    1.07056781891670D-04 G'S

J2 =  1.08270000-03

INITIAL TIME IS                                                                   UNITS
        0.0                    DAYS = 0.0                  SECONDS = 0.0

TEMIN IS                                                                          UNITS
        0.0                    DAYS = 0.0                  SECONDS = 0.0
TEMAX IS
        1.90000000000000 02 DAYS - 1.64160000000000 07 SECONDS = 2.03466788355045D 04 UNITS

STEP SIZE FOR NUMERICAL DIFFERENTIATION, KSTEP = 1
 1.00000000000000D-05   1.00000000000000D-05   1.00000000000000D-06   1.00000000000000D-04   1.00000000000000D-04

TIME STEP FOR INTEGRATION
        3.00000000000000D 00 DAYS = 2.59200000000000 05 SECONDS = 3.21263350034282D 02 UNITS

UPPER ERROR BOUND ON INTEGRATION =    1.00000000000 10

ERROR WEIGHTS FOR INTEGRATION ARE
1.00000 00  1.00000 00  1.00000 00  1.00000 00  1.00000 00  0.0         0.0         0.0         0.0         0.0

DIMENSION =   10

MAXIMUM NUMBER OF ITERATION =   20

NORM LIMIT IN ITER =  1.00000D-08

J2 =  1.08270000-03

1 EARTH RADIUS=        6378.16000000000 KM

MU (SM) =         398601.200000000 KM**3/SEC**2
EUN
UTKM=  6378.16000000000    ,UTS=  806.814720609557 9    ,UTD= 0.93381333403889400D-02,UTMS2=  9.79823345071663 6    ,UTKM=
  77.4585528170222 9
STOP
```

```
ITERATION BEGINS

ITER NO.    TRAJ. CALLS

   1           1

X
-1.6130000000000000D-01  4.7130000000000000D-01 -2.1840000000000000D-01  1.1830000000000000D 00  1.0000000000000000D 00

Y
-2.8062139443148150D-01  5.2107758922302450D-02  1.2079912417463640D-02  6.5664046889678700D-04  0.0
-6.4162291373162250D-03

TF=    4.6047746334247110D 01

F0=  8.1651109012201900D-02

KEIIOPKEII FOLLOWED BY CORRESPONDING Y

-1.6129900000000000D-01
-2.8059000477996340D-01  5.2106678196491330D-02  1.2080330178273750D-02  6.5547395200675970D-04  0.0
-6.4166599630694130D-03

 4.7131000000000000D-01
-2.8058174466303310D-01  5.2101891449527530D-02  1.2091979750081610D-02  6.5672328232822490D-04  0.0
-6.4131397299974860D-03

-2.1839900000000000D-01
-2.8061593106860670D-01  5.2107977072669120D-02  1.2079959244478860D-02  6.5740248572356890D-04  0.0
-6.4529112459470D-03

 1.1831100001000000D 00
-2.8062139443148150D-01  5.2107759922302450D-02  1.2079912417463640D-02  6.5670613294367660D-04  0.0
-6.3168705602459780D-03

 1.0001000000000000D 00
-2.8062139443148150D-01  5.2107758922302450D-02  1.2075212417463440D-02  6.5664046889678700D-04  9.9999999999999990D-05
-6.4162291373162250D-03

PARTIAL DERIV MATRIX
  3.1799651578054510D 01  4.0147768444452940D 00  5.4633628747691180D 00  0.0                      0.0
  1.4429451119547169D-03
  9.1927446381556199D-01 -5.8674777742706520D-01  2.1815026666851520D-01  0.0                      0.0
 -2.6097540246692570D-04
  4.1771081010690690D-01  1.9673376179681790D-01  4.6827015221136130D-02  0.0                      0.0
 -6.6652572825459820D-05
 -1.1665168990027235D 00 -3.1718656856703560D-02  7.6241682676192950D-01  6.5664046889626290D-04  0.0
 -3.0657786742356820D-05
  0.0                     0.0                     0.0                     0.0                      1.0000000000000000D 00
  0.0
 -6.2962575310736360D-01  6.0931073187374910D-01 -6.3784543244405100D-02  9.9358377066247060D-01  0.0
  3.1216199508763470D-05

DETERMINANT =   3.1766975596475940D-03

DELXIS ARE
 -3.9727994777270650D-04
  3.7170764266177520D-03
 -5.4247892973619390D-04
 -2.0677510122305720D-03
  0.0

DEL TF =  1.6962631969577670D 02

F1=  2.3634223584232250D-05

ITER NO.    TRAJ. CALLS

   2           7

X
-1.6169223497722390D-01  4.7504707692661770D-01 -2.1894742597073620D-01  1.1409322489877690D 00  1.0000000000000000D 00

Y
-4.5147149711102210D-03  1.7547584742989410D-03 -2.7661795980574860D-04 -8.7856724036566410D-05  0.0
  3.4935988476593790D-04

TF=    6.2942030333144870D 03

F0=  2.3634223584232250D-05

PARTIAL DERIV MATRIX
  3.1799451514103450D 01  4.0145768443761420D 00  5.4633628748373590D 00  2.6013589710342080D-10  0.0
  1.4590218825764550D-03
  9.1927466995562185D-01 -5.8674727720520470D-01  2.1815026666198930D-01 -1.0111737280912930D-10  0.0
 -2.5171195929163560D-04
  4.1771081010927470D-01  1.9673326177425900D-01  4.6527015224410650D-02  1.2482527807707050D-11  0.0
 -6.7756084073263550D-05
 -1.1665168990026319D 00 -3.1718656264936500D-02  7.6241682478315720D-01  6.5664047372947930D-04  0.0
 -3.5084617098732630D-06
  0.0                     0.0                     0.0                     0.0                      1.0000000000000000D 00
  0.0
 -6.2962575319118720D-01  6.0901073191012350D-01 -6.3784543249686200D-02  9.9358377064233890D-01  0.0
  3.3060469632901580D-05

DETERMINANT =   3.1506861579884060D-03

DELXIS ARE
 -1.7579215195005290D-04
  2.0248762443725630D-05
 -6.6080215779139120D-05
 -1.7600332544959910D-03
  0.0

DEL TF =  1.5510769249129990D 00

F1=  1.2745998736358630D-05

ITER NO.    TRAJ. CALLS

   3           8

X
-1.6198681212922430D-01  4.7705195307095030D-01 -2.1900850513451530D-01  1.1791722167332730D 00  1.0000000000000000D 00

Y
 1.2590100576021960D-04  2.5589719322154760D-02 -9.6041651918577830D-06  1.3312251649122920D-05  0.0
  2.9415151408814110D-04

TF=    4.7957549807354190D 01

F0=  1.2745998736358630D-05
```

```
PARTIAL DERIV MATRIX
  3.1385651547256930 01  4.0145767328993120 00  5.4633629209290120 00  9.8718799003507980-08  0.0
  1.1947849116265620-03
  9.1327480857120720-01 -5.8674476889373530-01  2.1815036844562420-01 -2.1785941071104250-06  0.0
  1.2133452573119900-03
  4.1771081024742940-01  1.9673325702790200-01  4.6827014779789700-02  4.184 135238901580-09  0.0
 -6.8396730619128990-05
 -1.1665168903968620 00 -3.1718649792021150-02  7.6241682729026960-01  6.5653428810434660-04  0.0
 -1.8656826110653520-06
  0.0                    0.0                    0.0                    0.0                    1.0000000000000000 00
  0.0
 -6.2962576253804430-01  6.0901285926915117D-01 -6.3784538718765700-02  9.9358263341655070-01  0.0
  1.0106677877827820-04

 DETERMINANT = -3.4036794966425740-03

DELXIS ARE
 -1.3952874275946970-05
  7.9755950764789150-05
 -1.8004565584374940-05
 -6.3966560216612650-05
  0.0

 DEL TF =  1.2571212697673390-01

FI=  1.3557281736290930-10

 ITER NO.   TRAJ. CALLS

     7          32

X
 -1.6141575296286960-01  4.7627708477668600-01 -2.1865905728623620-01  1.1814367552983260 00  1.0000000000000000 00

Y
 -7.2630504303369070-07 -1.1592366208531550-05 -4.1359874202352460-09  2.4292325943189540-08  0.0
  8.1259518279683670-07

 TF=    4.7033036863728510 03

FO=  1.3557281736290800-10

FO= 1.3557281736290800-10CASE CONVERGED...FERTIG
```

```
CONVERGED VALUES FOR OPTIMIZED TRAJECTORY

ACTUAL FINAL ORBITAL ELEMENTS ARE

       A (KM)                 E                      I (DEG)                LON ASC NODE (DEG)     ARG PERIG (DEG)
 4.2158155836437114D 04  1.4607637791466440-05  9.0550067685227300-02 -2.8696585300442010 01  1.9983080546962930 02
    A (EARTH RAD)          H                      K                      P                      Q
 6.6057676076362380 00  1.3625619098756600-05 -9.7351347670092190-05 -4.1902462502767630-04  7.6547246835238280-04

THE ERROR IN THE FINAL O.E. IS (ACTUAL - DESIRED)

       A (KM)                 E                      I (DEG)                LON ASC NODE (DEG)     ARG PERIG (DEG)
 -4.6356789553446734D-03 -1.1592366208531[illegible]-05 -9.3731477271653550-07

    A (EARTH RAD)          SQRT(H**2+K**2)        SQRT(P**2+Q**2)
 -7.2630504303368070-07 -1.1592366204531550-05 -8.1359874202352460-09

THE CONVERGED INITIAL COSTATE PARAMETERS ARE
 -1.6141535296246960-01  4.7637708477668680-01 -2.1865905728623620-01  1.1814367552983260 00  1.0000000000000000 00

THE MINIMIZED FINAL TIME IS
          5.4737163630973970 01 DAYS = 4.8652909377161970 06 SECONDS = 4.7905036863728510 03 UNITS

FINAL MASS =   868.5069406073160000 KG, FRACTION OF INITIAL MASS =    0.363504948607217

FINAL POWER =    14.8973000000003990 KW, FRACTION OF INITIAL POWER =    1.0000000000000000

LOW THRUST DELV=    4.1440554207095430 00 KM/SEC
```

```
TIME HISTORY OF OPTIMAL TRAJECTORY

HIGH THRUST

ORBIT = 1

EQUINOCTIAL O.E. AND COSTATE/S.F.=1000.
  1.0240518337075?D 00   0.0                      0.0                     0.0                     2.5396764647494?D-01
  3.3141617821995?D-01  -5.789[illegible]D-02     1.2557840656600?D-01    0.0                    -5.3154178522396?D-01

CLASSICAL O.E.
  6.5634700000000D 03    .0                       2.8500000000000D 01     0.0                    -2.3266919826961?D 02

     TRUE ANOMALY           PHI (DEG)               UPSILON (DEG)           DELTA V (KM/S)
  0.0                    8.4000012615678?D-01    -1.4341750128327?D 01    0.0

ORBIT = 2

EQUINOCTIAL O.E. AND COSTATE/S.F.=1000.
  1.665?670?39?7??D 00  -1.524?554466510?-0?     3.5032520792686?D-01    7.9230156533778?D-03    2.3544449814702?D-01
  1.5140543454437?D-01  -7.473[illegible]D-06     1.7174547157416?D-03   -1.8709000262648?D-02   -5.3104917299158?D-01

CLASSICAL O.E.
  1.0?21?3937470?3?D 04  3.8205440199087?D-01    2.6511036469198?D 01    1.9273507057775?D 00   -2.5443002774177?D 01

     TRUE ANOMALY           PHI (DEG)               UPSILON (DEG)           DELTA V (KM/S)
  4.6?30??17602???D-01   ?.4039012615678?D-01    -1.7183654090486?D 01    1.4058019000000?D 00

LOW THRUST

TIME      TIME UNITS              SECONDS        HOURS         DAYS          DV (K/S)       N
          0.0                     0.0            0.0           0.0           0.0            1

THE ORBITAL ELEMENTS ARE (EQ. FOLLOWED BY CL. IN KM & DEG)
  1.665267033958?D 00   -1.524195446645?-01    3.503252079269D-01     7.923015653378D-03     2.354444981471D-01

  1.0?2[illegible]D 04   ?.8205440195?D-01      2.651163694920D 01     1.927350705729D 00    -2.544300277418D 01

     MASS (KG)              POWER (KW)              THRUST (N)             THRUST ACC (M/S**2)
  1.0030000000003D 03    1.4697000000000D 01     9.999761742035D-01     9.999761742035D-04

THE COSTATE IS
  1.7837594532?4?D 03   -5.8?5213671663? 00      2.029064126728D 0?    -2.111296556661D 02    -6.274010118430D 03

THE DERIVITIVES OF THE ORBITAL ELEMENTS ARE
  3.764[illegible]D-06    ?.531[illegible]D-0?    1.6712?1?7?5?9?D-0?    -?.?11?7?3396514?D-05    -2.?145?30787?70-05

THE DERIVITIVE OF THE COSTATE IS
 -?.[illegible]D-01      -?.?7?5?[illegible]D-0?   -2.097537589?17D-01     2.095742?[illegible] 00   -1.5?0654?14?66?D-01

     HAMILTONIAN            PERIOD (HRS)            PERIGEE (KM)           APOGEE (KM)       DIV. TIME STEP
  3.8682070?331?         ?.0?6?5?[illegible]        6.5636099116D 03        1.4479?6?83?D 04       0

TIME      TIME UNITS              SECONDS        HOURS          DAYS           DV (K/S)       N
          ?.?1?6??[illegible] 0?  2.5?9?000?D 05  7.2?00000D 01  3.0000000?D 00  2.6034???D-01  2

THE ORBITAL ELEMENTS ARE (EQ. FOLLOWED BY CL. IN KM & DEG)
  1.7?26?[illegible]D 00  -1.2?[illegible]D-01   3.5417586151?9?D-01   -1.165174714564D-02     2.244464775406D-01

  1.143393?16?3?D 04     ?.7553071?[illegible]D-01   2.535514?827??D 0?   -2.969100465035D 00    -1.644778048383D 01

     MASS (KG)              POWER (KW)              THRUST (N)             THRUST ACC (M/S**2)
  9.9119729604465D 02    1.4677000000000D 01     9.999761742035D-01     1.008872111832D-03

THE COSTATE IS
  1.5811301535330D 03   -1.8632646690330 ?1     -5.140102061377D 01     3.848327001304D 02    -6.297662238022D 03

THE DERIVITIVES OF THE ORBITAL ELEMENTS ARE
  4.1440?410?65?D-04     7.601376341174D-05     5.9608?1836?0?D-06    -5.358436948120D-05    -3.715853681060D-05

THE DERIVITIVE OF THE COSTATE IS
 -6.2707595764470-01    -2.3642996558?37D-02    -2.389665844576D-?1     1.630562495381D 00    -7.485642990200D-03

     HAMILTONIAN            PERIOD (HRS)            PERIGEE (KM)           APOGEE (KM)       DIV. TIME STEP
  0.875961179191         3.375924378358          7.1394872292D 03        1.5728119110D 04        0

TIME      TIME UNITS              SECONDS        HOURS          DAYS           DV (K/S)       N
          6.4752670D 02           5.1840000D 05  1.4400000D 02  6.0000000D 00  5.2301310D-01  3

THE ORBITAL ELEMENTS ARE (EQ. FOLLOWED BY CL. IN KM & DEG)
  1.9?564?0679?5D 00    -1.0??29445895?D-01     3.5452003164?7D-01    -2.509359646634D-02     2.120025037415D-01

  1.2345179331075D 04    3.4927123?37?5?D-01    2.410149126758D 01    -6.750385557968D 00    -9.499914779604D 00

     MASS (KG)              POWER (KW)              THRUST (N)             THRUST ACC (M/S**2)
  9.8236654?929?D 02     1.4697000000000D 01     9.999761742035D-01     1.017927740359D-03

THE COSTATE IS
  1.3?4?4?6?2?7?D 03    -?.1921367?C?5? 01      -1.7324791021??D 02     8.651959325106D 02    -6.288181137650D 03
```

```
THE DERIVITIVES OF THE ORBITAL ELEMENTS ARE
  1.6050577081290-11    1.3422740095120-05   -1.9756977073140-06    1.98 967032230-05   -6.0046717114321-05

THE DERIVITIVE OF THE COSTATE IS
 -1.3667120564457D-01   2.6072172195050-02   -1.2300154501700-01    3.5610236721340-02   -2.4331851308970-02

     HAMILTONIAN          PERIOD (HRS)        PERIGEE (KM)          APOGEE (KM)      DIV. TIME STEP
   3.9809606818210      14.0317426?4139     3.0765495626D 04     4.1613127764D 04          0

******************************************************************************
TIME           TIME UNITS                 SECONDS        HOURS            DAYS          DV (K/S)         N
             4.4976860D 03                3.6288000D 06  1.00800000 03  4.20000000 01  3.67301970 00         15

THE ORBITAL ELEMENTS ARE (EQ. FOLLOWED BY CL. IN KM & DEG)
   6.1760?0465630D 00   -1.1660271047770-02    7.6739717929410-02   -6.0345372205130-03    1.9392108429810-02

   3.6192150567721D 04   7.7520515556670-02    2.3558130957130 00   -1.9414450362170 01    1.0774724550380 01

     MASS (KG)            POWER (KW)            THRUST (N)          THRUST ACC (M/S**2)
   8.7655211662380 02   1.4697000000000 01    9.9997617420350-01   1.1408062968980-03

THE COSTATE IS
   1.2748416957770 02   2.2147167152610 02   -1.3793033333530 03    2.1592524679220 03   -6.2866772649100 03

THE DERIVITIVES OF THE ORBITAL ELEMENTS ARE
   1.5178997406150-03   3.8707731199400-05   -2.4407895377340-06    ? 1149006638990-05   -6.2056339007520-05

THE DERIVITIVE OF THE COSTATE IS
  -2.1400?1071560D-02  -?.?798?0?9?7??D-0     5.8104244724120-03    2.3476863423680-02   -1.0874960430440-02

     HAMILTONIAN          PERIOD (HRS)        PERIGEE (KM)          APOGEE (KM)      DIV. TIME STEP
   0.9908?2?828050      21.6133752?9204     3.6334511373D 04     5.2449789763D 04          0

******************************************************************************
TIME           TIME UNITS                 SECONDS        HOURS            DAYS          DV (K/S)         N
             4.6442453D 03                3.7470455D 06  1.04084600 03  4.3369582D 01  4.0082254D 00         16

THE ORBITAL ELEMENTS ARE (EQ. FOLLOWED BY CL. IN KM & DEG)
   6.?955??72240630 00  -5.9997013670550-03    ?.9511177670500-02   -3.6836478974220-03    1.0200379129380-02

   4.07??????21442D 04   ?.994766?917610-02    1.2427124350900 02   -1.5856097147910 01    1.1378686735260 01

     MASS (KG)            POWER (KW)            THRUST (N)          THRUST ACC (M/S**2)
   ?.7252955261550 02   1.4657000000000 01    9.9997617420350-01   1.1460657055130-03

THE COSTATE IS
   1.2139135658540 02   2.1823277682400 02   -1.3742646275380 03    ?.1623393955760 03   -6.2877608012880 03

THE DERIVITIVES OF THE ORBITAL ELEMENTS ARE
   1.4707049066780-03   ?.9915032571030-05   -2.6764930557830-04    2.1967820786120-05   -6.3??4898503340-05

THE DERIVITIVE OF THE COSTATE IS
  -1.0587469896?0?-01  -3.2071221466640-02    6.2631032107110-02    1.6701262896190-02   -3.8021710399490-03

     HAMILT[illegible]    PE[illegible]  HRS)   PERIGEE (KM)          APOGEE (KM)      DIV. TIME STEP
   0.99?1?0?[illegible]          ?52658     2.9162564065D 04     4.2421567356D 04          0

******************************************************************************
TIME           TIME UNITS                 SECONDS        HOURS            DAYS          DV (K/S)         N
             4.7908017D 03                3.8652909D 06  1.07369180 03  4.4737?660 01  4.1460558D 00         17

THE ORBITAL ELEMENTS ARE (EQ. FOLLOWED BY CL. IN KM & DEG)
   6.6057675076360 00   1.3675609088360-05   -8.7351347670090-05   -4.1902462502770-04    7.6547246839240-04

   4.2158155316320 04   5.0437633791470-05    9.9999067685730-02   -2.8696584300460 01    1.9983380546960 02

     MASS (KG)            POWER (KW)            THRUST (N)          THRUST ACC (M/S**2)
   8.6850696960720 02   1.4457000000000 01    9.9997617420350-01   1.1513771927680-03

THE COSTATE IS
   1.0639729560420 02   2.1195886333560 02   -1.3608896894720 03    2.1647446552040 03   -6.2877548291420 03

THE DERIVITIVES OF THE ORBITAL ELEMENTS ARE
   1.4459390165200-03   4.0507573012760-05   -2.7719611157270-04    1.?.?24?6008380-05   -6.5605114744320-05

THE DERIVITIVE OF THE COSTATE IS
  -1.9919150556490-02  -5.2?? ?5??7?42??-02    1.1583502916460-0?    1.4?40?54??9?40-02    4.0?05?60?1477D-0?

     HAMILTONIAN          PERIOD (HRS)        PERIGEE (KM)          APOGEE (KM)      DIV. TIME STEP
   1.0?3990812995      23.924376020038      4.2156478262D 04     4.2?61882457D 04          0

PROGRAM HAS RUN SUCCESSFULLY
```

APPENDIX E

SOURCE CODE LISTINGS

The source code listings for all the subprograms are given in this appendix. There are five principle groupings. The first is used only for SEP, the second with NEP, the third is a common group used both with SEP and NEP, the fourth includes optional versions of ITER, TRAJ, and QUAD and finally the fifth group includes the driver program ORBIT and special versions of FCT and EVAMP containing print statements which are used to get thrust direction at points on particular orbits.

The SEP group includes:

INPUT
OUTPC
ITER (modified Newton Raphson)
PRTN
DCROUT
TRAJ (Runga-Kutta version)
OUTP
FUNCT
EARTH (non-incluned magnetic axis)
SUN
SHADOW
DQRTIC
DCUBIC
FLUX

The NEP group includes:

INPUT
OUTPC
ITER (modified Newton-Raphson)

The NEP group includes:

PRTN
DCROUT
TRAJ (Runga-Kutta version)
OUTP
FUNCT

The common group includes:

CONTL
OBLATE
QUAD (16 point)
FCT
EVALMP
MAINE
START
TIME
SWITCH
DTDU
OUTHI
IMPLS
YF

The option group includes:

ITER (Newton-Raphson for SEP)
ITER (Newton-Raphson for NEP)
TRAJ (predictor-corrector for SEP)
TRAJ (predictor-corrector for NEP)
QUAD (4 point)
QUAD (8 point)
QUAD (32 point)
EARTH (inclined magnetic axis)

The thrust direction group includes:

ORBIT
FCT (print version)
EVALMP (print version)

```
C INPUT/INPUTS
C
C SEP AND HIGH THRUST.
C THIS SUBPROGRAM IS CALLED BY CONTL AND READS AND PRINTS
C   ALL INITIAL DATA AS WELL AS SETS INITIAL CONSTANTS.
C   THE UNITS ARE BASED ON INTERNAL MU=1.0, INTERNAL DISTANCE
C   UNIT=1 EARTH RADII. AND EXTERNAL MU= 398601.2 KM*KM*KM/
C   SEC*SEC AND EARTH RADII= 6378.16 KM.  A CIRCULAR
C   ORBIT AT 1 EARTH RADII WOULD HAVE A PERIOD OF 2 PI INTERNAL
C   TIME UNITS.
C 14 DIMENSIONAL WITH MASS AND FLUX
C INPUT
C  LOW/HIGH                        HIGH/LOW/HIGH
C                  INITIAL ORBIT
C  A (KM)
C  E                                 SET TO ZERO
C  I (DEG)
C  LONG. OF NODE (DEG)               NOT USED
C  ARG. OF PERIGEE (DEG)             NOT USED
C  MASS (KG)
C  FLUX
C                  INITIAL GUESSES
C  LAMBDA A                          LIKE UPSILON
C  LAMBDA H                          LIKE SMALL'S K
C  LAMBDA K                          LIKE SMALL'S J
C  LAMBDA P                          SCALE FACTOR
C  LAMBDA Q                          NODAL ANGLE (RAD) OR ITS ADJOINT
C  LAMBDA M
C  LAMBDA N
C                  FINAL ORBIT
C  A
C  E
C  I
C  NODE (NOT USED IF NOP=2)          NOT USED
C  PERIGEE (NOT USED IF NOP=2)       NOT USED
C
C  TF2 (DAYS), GUESS FOR FINAL TIME
C  PKW (KW), POWER
C  SPIM (SEC), SPECIFIC IMPULSE OF SEP
C  TL JULIAN DATE AT INITIAL TIME
C  IRDFLG                                              NOMINAL
C  1  END OF INPUT
C  2  IPR, PRINT FLAG                                  0
C  3  NIMAX, MAX. NO. OF ITERATIONS                    20
C  4  TFMAX2 (DAYS), MAX. TF                           190.
C  5  DT2 (DAYS), TIME STEP FOR D.E.                   1.
C  6  UEB, UPPER ERROR BOUND FOR D.E.                  1.D10
C  7  EW, ERROR WEIGHTS FOR D.E.                       1.,1.,1.,1.,0.,...
C  8  UTKM, EQUATORIAL EARTH RADIUS (KM)               6378.16
C  9  GM (KM**3/SEC**2) EARTH GRAV.CONST.              398601.2
C 10  NOP = 1, 5 O.E. SPECIFIED AT TF                  1
C         = 2, 3 O.E. SPECIFIED AT TF
C 11  SETS OBLATENESS, AJ2, = 1.0827D-3                0.
C 12  STEP(8), STEP SIZE FOR NUMERICAL DIFF.           1.D-6
C     KSTEP = 0, STEP AS FRACTION IN ITER              0
C           = 1, STEP AS CONSTANT IN ITER
C 13  ISON = 0, SHADOW EFFECT OFF                      0
```

```
C             = 1. SHADOW EFFECT ON
C 14    ISUN = 0 SUN DIST EFFECT ON POWER OFF          0
C             = 1  EFFECT ON
C       PH (KW) HOUSEKEEPING POWER                    0.
C 15    IHI = 1, LOW THRUST ONLY                      1
C           = 2, HIGH/LOW
C           = 3, HIGH/LOW/HIGH
C           = 4, LOW/HIGH
C       DVI1 (M/S) TOTAL INITIAL HIGH THRUST DEL V    0.
C       DVI2 (M/S) DEL V FOR FINAL IMPULSE            0.
C       NODE = 0, INITIAL LINE OF NODES FREE
C            = 1, INITIAL LINE OF NODES FIXED
C               NODE MEANINGFUL ONLY IF IHI= 2 OR 3
C 16    IPOW = 0, CONSTANT POWER                      1
C             = 1, DEGRADATION EFFECT
C 17    FLIM, NORM LIMIT IN ITERATION ROUTINE         1.D-6
C 18    SGN = -1., INITIAL LAM 1 IS NEGATIVE
C           = +1., INITIAL LAM 1 IS POSITIVE
C
C
      SUBROUTINE INPUT
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      NAMELIST /UN/UTKM,UTS,UTD,UTKG,UTKW,UTMS2
C
      COMMON /XMMM/ZLO(7), STEP(8), ZERF(8)
      COMMON /ELEM/ZPO(7), ZPF(5)
      COMMON /INT/ITF,IPR,IDIM,IDIM2,NIMAX
      COMMON /TRA/TFMAX,DT,UER,FW(14)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
      COMMON /T/TF,S,TO,TFMIN
      COMMON /A/A,AMU,PI
      COMMON /WF/WF(5)
      COMMON /J2/ AJ2
      COMMON /TC/NOP
      COMMON /JD/TL
      COMMON /POWER/PO,C,POW,PH,ISUN,ISON,IPOW
      COMMON /HIGH/DVI1,DVI2,IHI
      COMMON /ACOM/AF(10)
      COMMON /F/FLIM,KSTEP
      COMMON /NOD/NODE
      COMMON /SG/SGN
C
      DIMENSION W(7)
C
C INTEGER CONSTANTS
C
      IDIM=14
      IDIM2=7
      IDIM3=8
      IPR=0
      ITF=3
      NIMAX=20
      NOP= 1
      ISON= 0
      ISUN= 0
```

```
      IHI= 1
      IPOW= 1
      KSTEP= 0
C
C REAL CONSTANTS
C
      AJ2= 0.D0
      AMU=1.0D0
      UEB= 1.0D+10
      DO 10 I=1,IDIM2
       EW(I+IDIM2)= 0.D0
 10    EW(I)= 1.0D0
      EW(6)= 0.D0
      EW(7)= 0.D0
      DO 12 I= 1,IDIM3
 12    STEP(I)= 1.D-6
      DT2= 1.0D0
      GM=398601.2D0
      UTKM= 6378.16D0
      DTR= .017453292519943296D0
      PI= 3.1415926535897932D0
      TFMAX2= 190.0D0
      TFMIN2= 0.0D0
      T02=0.0D0
      TL= 0.D0
      PH= 0.D0
      FLIM= 1.D-6
      DVI1= 0.D0
      DVI2= 0.D0
      AF(1)= 7.2D2
      AF(2)= -1.4D2
      AF(3)= -1.83D1
      AF(4)= 2.06D1
      AF(5)= 9.05D1
      AF(6)= -9.16D1
      AF(7)= -2.27336D3
      AF(8)= 5.97087D3
      AF(9)= -4.25392D3
      AF(10)= 1.19494D-21
C
C ALL READ STATEMENTS FOLLOW
C
      READ (5,1000) (W(I),I= 1,IDIM2)
      READ (5,1000) (ZLO(I),I=1,IDIM2)
      READ (5,1000) (WF(I),I= 1,5)
      SGN= WF(3)-W(3)
      IF (SGN.NE.0.D0) SGN= SGN/DABS(SGN)
      READ (5,1001) TF2
      READ (5,1001) PKW
      READ (5,1001) SPIM
      READ (5,1001) TL
C       TL MUST BE BETWEEN ABOUT A.D. 1950 AND 2000
      IF ((TL.LT.2.433D6).OR.(TL.GT.2.452D6)) GO TO 230
 20   READ (5,1002) IRDFLG
      IF ((IRDFLG.GT.18).OR.(IRDFLG.LT.1)) GO TO 200
 25   GO TO (150,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48),
     1  IRDFLG
```

```
C
C THESE VALUES ARE READ ONLY IF INDICATED BY IRDFLG
C
 32   READ (5,1002) IPR
      IF (IPR.LT.0) GO TO 210
      GO TO 20
 33   READ (5,1002) NIMAX
      IF ((NIMAX.LT.0).OR.(NIMAX.GT.50)) GO TO 220
      GO TO 20
 34   READ (5,1001) TFMAX2
      IF ((TFMAX2.LT.0.D0).OR.(TFMAX2.GT.1.D3)) GO TO 220
      GO TO 20
 35   READ (5,1001) DT2
      IF ((DT2.LT.0.D0).OR.(DT2.GT.1.D3)) GO TO 220
      GO TO 20
 36   READ (5,1001) UER
      GO TO 20
 37   READ (5,1003) EW
      GO TO 20
 38   READ (5,1001) UTKM
      GO TO 20
 39   READ (5,1001) GM
      GO TO 20
 40   READ (5,1002) NOP
      IF ((NOP.LT.1).OR.(NOP.GT.2)) GO TO 220
      GO TO 20
 41   AJ2= 1.0827D-3
      GO TO 20
 42   READ (5,1001) (STEP(I),I=1,IDIM3)
      READ (5,1002) KSTEP
      IF ((KSTEP.LT.0).OR.(KSTEP.GT.1)) GO TO 220
      GO TO 20
 43   READ (5,1002) ISON
      IF ((ISON.LT.0).OR.(ISON.GT.1)) GO TO 220
      GO TO 20
 44   READ (5,1002) ISUN
      IF ((ISUN.LT.0).OR.(ISUN.GT.1)) GO TO 220
      READ (5,1001) PH
      GO TO 20
 45   READ (5,1002) IHI
      IF ((IHI.LT.1).OR.(IHI.GT.4)) GO TO 220
      READ (5,1001) DVI1,DVI2
      IF (IHI.LT.3) DVI2= 0.D0
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) DVI1= 0.D0
      NOP= 2
      READ (5,1002) NODE
      IF ((NODE.LT.0).OR.(NODE.GT.1)) GO TO 220
      GO TO 20
 46   READ (5,1002) IPOW
      IF ((IPOW.LT.0).OR.(IPOW.GT.1)) GO TO 220
      GO TO 20
 47   READ (5,1003) FLIM
      IF (FLIM.LT.0.D0) GO TO 220
      GO TO 20
 48   READ (5,1001) SGN
      IF (DABS(SGN).NE.1.D0) GO TO 220
      GO TO 20
```

```
C
C
C
C
C
C TIME VALUES ARE CHANGED FROM DAYS TO OTHER UNITS
C
 150  UTS= DSQRT(UTKM**3/GM)
      UTM=UTS/60.D0
      UTH=UTS/3600.D0
      UTD=UTH/24.D0
      TO= TO2/UTD
      TF= TF2/UTD
      TFMIN= TFMIN2/UTD
      TFMAX= TFMAX2/UTD
      DT= DT2/UTD
      TO1= TO*UTS
      TF1= TF*UTS
      TFMIN1= TFMIN*UTS
      TFMAX1= TFMAX*UTS
      DT1= DT*UTS
C
      AF(10)= AF(10)*UTS
C
C MORE CONVERSIONS
C
      UTMS2= UTKM*1.D3/(UTS**2)
      UTKG= 1.D3
      UTKW= UTKG*UTMS2*UTKM/UTS
C
C
C
      CALL EARTH
C
C THE PRINTING OF ALL INITIAL VALUES FOLLOWS
C
      WRITE (6,2000)
      WRITE (6,2001)
      WRITE (6,2030)
      WRITE (6,2033)
      WRITE (6,2034)
      WRITE (6,2035) NOP
      IF (ISON.EQ.1) WRITE (6,2042)
      WRITE (6,2045) DVI1,DVI2
      WRITE (6,2002)
      WRITE (6,2003)
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) GO TO 158
C       W(1) IS SEMIMAJOR AXIS
C       W(2) IS ECC. SET TO 0
C       W(3) IS INCLINATION
C       W(4) IS OMEGA IF NODE= 1
C       ZLO SHOULD BE Y1 (UP),Y2 (XK),YS (XJ),C,(LAM) OMEGA,LAM M, LAM N
C
      W(2)= 0.D0
      III=3+NODE
      WRITE (6,2004) (W(I),I= 1,III)
      ZPO(1)= W(1)/UTKM
```

```
      ZPO(2)= W(3)*DTR
      WRITE (6,2046) ZPO(1)
      ZPO(6)= W(6)/UTKG
      IF (NODE.EQ.1) ZPO(3)= W(4)*DTR
      GO TO 162
C
 158  WRITE (6,2004) (W(I),I=1,5)
C
C   CHANGE FROM CLASSICAL O.E. TO EQUINOCTIAL O.E.
      DO 160 I=3,5
 160   W(I)= W(I)*DTR
      ZPO(1)= W(1)/UTKM
      ZPO(2)= W(2)*DSIN(W(5)+W(4))
      ZPO(3)= W(2)*DCOS(W(5)+W(4))
      ZPO(4)= DTAN(W(3)/2.0D0)*DSIN(W(4))
      ZPO(5)= DTAN(W(3)/2.0D0)*DCOS(W(4))
C
      ZPO(6)= W(6)/UTKG
C
C
      WRITE (6,2005)
      WRITE (6,2004) (ZPO(I),I=1,5)
 162  DVI1= (DVI1/(UTMS2*UTS))*DSQRT(ZPO(1)/AMU)
      DVI2= DVI2/(UTMS2*UTS)
      ZPO(7)= W(7)
C
      WRITE (6,2038) W(6)
      IF (IPOW.EQ.0) WRITE (6,2049)
      WRITE (6,2039) PKW
      WRITE (6,2043) PH
      WRITE (6,2040) W(7)
C
      PO= PKW/UTKW
      PH= PH/UTKW
      IF (ISUN.NE.0) WRITE (6,2044)
C
C WRITE FINAL CONDITIONS, CHANGE TO EQUINOCTIAL FINAL COND.
C
      WRITE (6,2006)
      ZPF(1)= WF(1)/UTKM
      GO TO (165,170), NOP
C
 165  WRITE (6,2003)
      WRITE (6,2004) (WF(I),I= 1,5)
      DO 166 I=3,5
 166   WF(I)= WF(I)*DTR
      ZPF(2)= WF(2)*DSIN(WF(5)+WF(4))
      ZPF(3)= WF(2)*DCOS(WF(5)+WF(4))
      ZPF(4)= DTAN(WF(3)/2.0D0)*DSIN(WF(4))
      ZPF(5)= DTAN(WF(3)/2.0D0)*DCOS(WF(4))
      DO 167 I=3,5
 167   WF(I)=WF(I)/DTR
      WRITE (6,2005)
      WRITE (6,2004) (ZPF(I),I=1,5)
      GO TO 190
C
 170  ZPF(2)= WF(2)
```

```
      ZPF(3)= DABS(DTAN(WF(3)*DTR/2.D0))
      ZPF(4)= 0.D0
      ZPF(5)=0.D0
      WRITE (6,2031)
      WRITE (6,2004) (WF(I),I=1,3)
      WRITE (6,2032)
      WRITE (6,2004) (ZPF(I),I=1,3)
C
C
  190 WRITE (6,2007)
  196 WRITE (6,2011) ZLO
      IF ((IHI.GT.1).AND.(IHI.LT.4)) WRITE (6,2050)SGN
      WRITE (6,2008)
      WRITE (6,2009) TF2,TF1,TF
      C= SPIM/UTS
      CC= C*UTMS2*UTS
      WRITE (6,2012) SPIM,CC,C
      A= UTMS2*2.D0*(P0-PF)/(C*ZPO(6))
      WRITE (6,2041) A
      WRITE (6,2037) TL
      WRITE (6,2036) AJ2
      WRITE (6,2013)
      WRITE (6,2009) T02,T01,T0
      WRITE (6,2014)
      WRITE (6,2009) TFMIN2,TFMIN1,TFMIN
      WRITE (6,2015)
      WRITE (6,2009) TFMAX2,TFMAX1,TFMAX
      WRITE (6,2010) KSTEP
      WRITE (6,2011) STEP
      WRITE (6,2016)
      WRITE (6,2009) DT2,DT1,DT
      WRITE (6,2017) UEB
      WRITE (6,2018)
      WRITE (6,2019) EW
      WRITE (6,2020) IDIM
      WRITE (6,2022) NIMAX
      WRITE (6,2048) FLIM
      WRITE (6,2026) UTKM
      WRITE (6,2027) GM
      WRITE (6,UN)
      RETURN
  200 WRITE (6,2023) IRDFLG
      STOP
  210 WRITE (6,2024) IPR
      STOP
  220 WRITE (6,2025) IRDFLG
      STOP
  230 WRITE (6,2037) TL
      STOP
C
 1000 FORMAT (F25.15)
 1001 FORMAT (F25.15)
 1002 FORMAT (I2)
 1003 FORMAT (7D6.1)
 1004 FORMAT (F18.11)
 2000 FORMAT (1H1,22X,75H OPTIMAL TRAJECTORY PROGRAM FOR SATELLITE RAISI
     1NG USING SEP AND HIGH THRUST)
```

```
2001 FORMAT (1H0,53X,13H MINIMUM TIME)
2002 FORMAT (34H0 THE INITIAL ORBITAL ELEMENTS ARE)
2003 FORMAT (1H0,10X,6HA (KM),19X,1HE,20X,7HI (DEG),10X,18HLON ASC NODE
    1 (DEG),6X,15HARG PERIG (DEG))
2004 FORMAT (1P5D23.14)
2005 FORMAT (1H0,6X,13HA (EARTH RAD),16X,1HH,22X,1HK,22X,1HP,22X,1HQ)
2006 FORMAT (40H0 THE DESIRED FINAL ORBITAL ELEMENTS ARE)
2007 FORMAT (32H0 INITIAL GUESSED PARAMETERS ARE)
2008 FORMAT (21H0 FINAL TIME ESTIMATE)
2009 FORMAT (1H ,10X,1PD22.15,7H DAYS =,1PD22.15,10H SECONDS =,1PD22.15
    1,6H UNITS)
2010 FORMAT (50H0 STEP SIZE FOR NUMERICAL DIFFERENTIATION, KSTEP =,I2)
2011 FORMAT (1P5D23.14)
2012 FORMAT (12H0 SPEC IMP =, 1PD23.14,15H SEC, EXH VEL =,1PD23.14,
    16H M/S =,1PD23.14, 10H F.R./T.U.)
2013 FORMAT (17H0 INITIAL TIME IS)
2014 FORMAT (10H0 TFMIN IS)
2015 FORMAT (10H0 TFMAX IS)
2016 FORMAT (27H0 TIME STEP FOR INTEGRATION)
2017 FORMAT (36H0 UPPER ERROR BOUND ON INTEGRATION =,1PD20.10)
2018 FORMAT (35H0 ERROR WEIGHTS FOR INTEGRATION ARE)
2019 FORMAT (1P14D8.1)
2020 FORMAT (13H0 DIMENSION =,I5)
2022 FORMAT (31H0 MAXIMUM NUMBER OF ITERATION =,I5)
2023 FORMAT (44H0 IRDFLG SHOULD BE BETWEEN 1 AND 16, IT IS =,I5)
2024 FORMAT (28H0 IPR SHOULD BE < 0, IT IS =,I5)
2025 FORMAT (27H0 BAD INPUT DATA, IRDFLG = ,I3)
2026 FORMAT (17H0 1 EARTH RADIUS=,F25.12,3H KM)
2027 FORMAT (11H0 MU (GM) =,F25.10,13H KM**3/SEC**2)
2030 FORMAT (25H0 FINAL CONDITION OPTIONS)
2033 FORMAT (43H  1. ALL 5 FINAL ORBITAL ELEMENTS SPECIFIED)
2034 FORMAT (51H  2. A.E.I SPECIFIED, LON ASC NODE AND ARG PER FREE)
2035 FORMAT (24H  FOR THIS RUN, OPTION =,I4)
2031 FORMAT (1H0,10X,6HA (KM),19X,1HE,20X,7HI (DEG))
2032 FORMAT (1H0,6X,13HA (EARTH RAD),9X,15HSQRT(H**2+K**2),8X,
    1  15HSQRT(P**2+Q**2))
2037 FORMAT (32H0 JULIAN DATE AT INITIAL TIME IS, F20.8)
2036 FORMAT (6H0 J2 =,1PD15.7)
2038 FORMAT (21H0 INITIAL MASS (KG) =, F18.11)
2039 FORMAT (22H0 INITIAL POWER (KW) =, 1PD23.14)
2040 FORMAT (17H0 INITIAL FLUX = ,1PD23.14)
2041 FORMAT (24H0 INITIAL ACC (1 A.U.) =,1PD23.15,7H M/S**2)
2042 FORMAT (24H0 SHADOW EFFECT INCLUDED)
2043 FORMAT (27H0 HOUSEKEEPING POWER (KW) =,1PD23.14)
2044 FORMAT (39H0 SUN DISTANCE EFFECT ON POWER INCLUDED)
2045 FORMAT (41H0 TOTAL DELV FOR INITIAL IMPULSES (M/S) =, F20.12,5X,
    1      21HFINAL IMPULSE (M/S) =,F20.12)
2046 FORMAT (27H0 INITIAL A (EARTH RADII) =, 1PD25.14)
C2047 FORMAT (79H0 INITIAL GUESS FOR IMPULSE PARAMETERS, S.F., LONG. NOD
C    1E, MASS AND FLUX COSTATE)
2048 FORMAT (23H0 NORM LIMIT IN ITER = ,1PD12.5)
2049 FORMAT (22H0 NO POWER DEGRADATION)
2050 FORMAT (30H0 SIGN OF INITIAL LAMBDA I IS ,F6.1)
     END
```

```
C OUTPC/OUTPCS
C
C THIS SUBPROGRAM WRITES THE VALUES FOR THE FINAL CONVERGED
C   TRAJECTORY.  IT IS CALLED BY THE MAIN PROGRAM CONTL.
C   TO BE USED WITH 8 DIM. ZERF.
C 14 DIMENSIONAL WITH MASS AND FLUX
C
C
C
      SUBROUTINE OUTPC
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      COMMON /XMMM/ZLO(7),STEP(8),ZERF(8)
      COMMON /Z/ZF(14),DZ(14)
      COMMON /T/TF,S,TC,TFMIN
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
      COMMON /ELEM/ZPO(7),ZPF(5)
      COMMON /WF/WF(5)
      COMMON /A/A,AMU,PI
      COMMON /TC/NOP
      COMMON /POWER/PO,C,POW,PH,ISUN,ISON,IPOW
C
      DIMENSION DELZF(5),DELWF(5),WFC(5)
C
      WFC(1)= ZF(1)*UTKM
      WFC(2)= 0.0D0
      DUMMY= ZF(2)**2 + ZF(3)**2
      IF (DUMMY.GT.1.0D-40) WFC(2)=DSQRT(DUMMY)
      WFC(3)= 0.0D0
      DUMMY= ZF(4)**2 + ZF(5)**2
      IF (DUMMY.GT.1.0D-40) WFC(3)= 2.0D0*DATAN(DSQRT(DUMMY))/DTR
      WFC(4)=0.D0
      IF ((DABS(ZF(4)).GT.1.D-8).AND.(DABS(ZF(5)).GT.1.D-8))
     1  WFC(4)= DATAN2(ZF(4),ZF(5))/DTR
      WFC(5)= 0.D0
      IF ((DABS(ZF(2)).GT.1.D-8).AND.(DABS(ZF(3)).GT.1.D-8))
     1   WFC(5)=DATAN2(ZF(2),ZF(3))/DTR
      WFC(5)=WFC(5)-WFC(4)
      DO 10 I=1,5
       DELWF(I)= WFC(I) - WF(I)
 10    DELZF(I)= ZF(I) - ZPF(I)
      TF2= TF*UTD
      RMASS= ZF(6)/ZPO(6)
      AMASS= ZF(6)*UTKG
      RPOW= POW/PO
      AKWPOW= POW*UTKW
      TF1= TF*UTS
      DELV= C*DLOG(ZPO(6)/ZF(6))*UTKM/UTS
C
      WRITE (6,3000)
      WRITE (6,3001)
      WRITE (6,3002) WFC
      WRITE (6,3003)
      WRITE (6,3002) (ZF(I),I=1,5)
      WRITE (6,3004)
      WRITE (6,3001)
```

```
      IF (NOP.EQ.1) WRITE (6,3002) DELWF
      IF (NOP.EQ.2) WRITE (6,3002) (DELWF(I),I=1,3)
      GO TO (20, 30), NOP
C
 20   WRITE (6,3003)
      WRITE (6,3002) DELZF
      GO TO 100
C
 30   DELZF(2)= DSQRT(ZF(2)**2+ZF(3)**2) -ZPF(2)
      DELZF(3)= DSQRT(ZF(4)**2+ZF(5)**2) -ZPF(3)
      WRITE (6,3011)
      WRITE (6,3002) (DELZF(I),I=1,3)
C
 100  WRITE (6,3006)
      WRITE (6,3002) ZLO
      WRITE (6,3008)
      WRITE (6,3009) TF2,TF1,TF
      WRITE (6,3012) ZF(7)
      WRITE (6,3013) AMASS,RMASS
      WRITE (6,3014) AKWPOW,RPOW
      WRITE (6,3010) DELV
      RETURN
 3000 FORMAT (35H0 ACTUAL FINAL ORBITAL ELEMENTS ARE)
 3001 FORMAT (1H0,10X,6HA (KM),18X,1HE,20X,7HI (DEG),10X,18HLON ASC NODE
     1 (DEG),6X,15HARG PERIG (DEG))
 3002 FORMAT (1P5D23.15)
 3003 FORMAT (1H0,5X,13HA (EARTH RAD),16X,1HH,22X,1HK,22X,1HP,22X,1HQ)
 3004 FORMAT (51H0 THE ERROR IN THE FINAL O.E. IS (ACTUAL - DESIRED))
C3005 FORMAT (60H0 CLASSICAL O.E. MAY HAVE DISCREPANCY OF MULTIPLES OF 9
C    10 DEG)
 3006 FORMAT (46H0 THE CONVERGED INITIAL GUESSED PARAMETERS ARE)
 3008 FORMAT (29H0 THE MINIMIZED FINAL TIME IS)
 3009 FORMAT (1H ,10X,1PD22.15,7H DAYS =,D22.15,10H SECONDS =,1PD22.15,6
     1H UNITS)
 3010 FORMAT (18H0 LOW THRUST DELV=,1PD25.14,7H KM/SEC)
 3011 FORMAT (1H0,5X,13HA (EARTH RAD),9X,15HSQRT(H**2+K**2),8X,
     1      15HSQRT(P**2+Q**2))
 3012 FORMAT (34H0 EQUIVALENT PARTICLES (*1.D-14) =,1PD22.15)
 3013 FORMAT (14H0 FINAL MASS =,F22.15,31H KG, FRACTION OF INITIAL MASS
     1=,F22.15)
 3014 FORMAT (15H0 FINAL POWER =,F22.15,32H KW, FRACTION OF INITIAL POWE
     1R =,F22.15)
      END
```

```
C ITER  MODNRS
C MODNR/MODNRS      MODIFIED N-R ITERATOR
C
C
C HXR VERSION
C
      SUBROUTINE ITER(KOUNT,NI,FUNCT,PRTN)
C
      IMPLICIT REAL*8(A-H,O-$)
C
C
C
C X   VALUES OF THE INDEPENDENT VARIABLES(INITIAL,CURRENT,FINAL)
C XS  STEP SIZE TO PERTURB X:S TO COMPUTE PARTIAL DERIVATIVES
C Y   VALUES OF THE DEPENDENT VARIABLES(CURRENT,FINAL)
      COMMON/XMMM/X(7),XS(8),Y(8)
      COMMON /INT/ITF,IPR,IDIM,IDIM2,MAXNOI
      COMMON /T/TF,S,TO,TFMIN
      COMMON /DY/ DYDT(8)
      COMMON /F/FLIM,KSTEP
C
      DIMENSION YNOM(8),XN(7),P(8,8),COEF(8),DYDTN(8)
      N=7
      M=8
      INORM=ITF
      IB=1
      ICONS=1
      ISW=0
      NI=1
      KOUNT=0
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      FO=0.D0
      DO 15 I=1,M
 15   FO=FO+Y(I)**2
 9    DO 16 I=1,N
      DYDTN(I)= DYDT(I)
      XN(I)=X(I)
   16 YNOM(I)=Y(I)
      YNOM(M)= Y(M)
      TFN=TF
      DYDTN(M)= DYDT(M)
 10   CALL PRTN(KOUNT,NI)
      WRITE(6,1011)FO
      IF(FO.LE.FLIM)GO TO 90
      IF (NI.GT.MAXNOI) GO TO 80
      IF(ISW.NE.0)GO TO 27
C COMPUTE NUMERICAL PARTIAL DERIVATIVES
      DO 17 I=1,M
 17    P(I,M)= DYDT(I)
      WRITE (6,1013)
      DO 25 J=1,N
      TEMP=X(J)
      STEP=XS(J)*DABS(X(J))
      IF ((DABS(X(J)).LT.1.D-10).OR.(KSTEP.EQ.1))STEP=XS(J)
C     IF (DABS(X(J)).LT.1.D-10) WRITE (6,1014)
```

```
      X(J)=X(J)+STEP
      CALL FUNCT
      WRITE(6,1000)X(J)
      WRITE(6,1001)(Y(I),I=1,M)
      DO 20 I=1,M
   20 P(I,J)=(Y(I)-YNOM(I))/STEP
   25 X(J)=TEMP
      KOUNT=KOUNT+[illegible]
   27 WRITE(6,1002)
      DO 30 I=1,M
      WRITE(6,1001)(P(I,J),J=1,M)
   30 CONTINUE
      DO 35 I=1,M
   35 COEF(I)=-YNOM(I)
      CALL DCROUT(P,COEF,DET,0.D0,M,1,IND)
      IF(IND.NE.0)GO TO 85
      WRITE (6,1015) DET
      DO 40 I= 1,M
 40   IF (DABS(COEF(I)).LT.1.D-10) COEF(I)= 0.DC
C
      IF (DABS(XN(1)).GT.1.D2) GO TO 47
      RATS= 1.D0
      DO 45 I= 1,5
      RAT= DABS(COEF(I))/(.8D0*DABS(XN(I))+.1D0)
      IF (RAT.GT.RATS) RATS= RAT
 45   CONTINUE
      DO 46 I= 1,8
 46   COEF(I)= COEF(I)/RATS
C     WRITE (6,1016) RATS
C
 47   WRITE(6,1003)(COEF(I),I=1,N)
      SN= COEF(M)
      WRITE (6,1012) SN
      DO 50 J=1,N
   50 X(J)=XN(J)+COEF(J)
      TF=TFN+SN
      IHALV=0
 51   IF (INORM.EQ.1) ITF=1
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      F1=0.D0
      DO 52 I=1,M
   52 F1=F1+Y(I)**2
      WRITE(6,1010)F1
      IF(F1.LT.F0)GO TO 55
      WRITE(6,1008)
      IF(IHALV.EQ.6)GO TO 95
      IHALV=IHALV+1
      DO 53 J=1,N
      COEF(J)=COEF(J)/2.D0
      WRITE(6,1000)COEF(J)
   53 X(J)=XN(J)+COEF(J)
      SN=SN/2.0D0
      WRITE (6,1012) SN
      TF=TFN+SN
      GO TO 51
```

```
   55 IF(NI-MAXNOI)70,70,80
   70 NI=NI+1
      ICONS=NI
      FO=F1
      SUMDX=0.D0
      DO 76 J=1,M
   76 SUMDX=COEF(J)**2+SUMDX
      DO 77 I=1,M
      DO 77 J=1,M
      P(I,J)=P(I,J)+(Y(I)*COEF(J))/SUMDX
   77 CONTINUE
      ISW=1
      GO TO 9
   80 NI=9999
      WRITE(6,1006)
      RETURN
   85 NI=9999
      WRITE(6,1007)
      RETURN
   90 WRITE(6,1005)FO
      RETURN
   95 IF(NI.EQ.1.OR.I8.EQ.10.OR.ICONS.NE.NI)GO TO 100
      ICONS=ICONS+1
      I8=I8+1
      DO 96 J= 1,N
      DYDT(J)= DYDTN(J)
      X(J)= XN(J)
   96 Y(J)= YNOM(J)
      Y(M)= YNOM(M)
      DYDT(M)=DYDTN(M)
      TF= TFN
      ISW=0
      WRITE(6,1004)
      GO TO 10
  100 NI=9999
      WRITE(6,1009)
      RETURN
 1000 FORMAT(/1X,1PD23.15)
 1001 FORMAT(1X,1P5D23.15)
 1002 FORMAT(21HOPARTIAL DERIV MATRIX)
 1003 FORMAT(11HODELX:S ARE/(1X,1PD23.15))
 1004 FORMAT(35HOFORM NEW PARTIAL DERIVATIVE MATRIX)
 1005 FORMAT(4HOFO=,1PD22.15,23HCASE CONVERGED...FERTIG)
 1006 FORMAT(38HOEXCEEDED MAXIMUM NUMBER OF ITERATIONS)
 1007 FORMAT(16HOMATRIX SINGULAR)
 1008 FORMAT(11HODELX:S ARE)
 1009 FORMAT(19HOMETHOD CANNOT WORK)
 1010 FORMAT(4HOF1=,1PD23.15)
 1011 FORMAT(4HOFO=,1PD23.15)
 1012 FORMAT (10HO DEL TF =,1PD23.15)
 1013 FORMAT (40HO X(I) DX(I) FOLLOWED BY CORRESPONDING Y)
 1014 FORMAT (24HO Y(     . SO DX(I)=XS(I))
 1015 FORMAT (15HO     INENT =,1PD23.15)
 1016 FORMAT (8HO RATS =,1PD23.15)
      END
```

```
C PRTN/PRTNS
C
C THIS PROGRAM IS CALLED BY THE ITERATOR AND PRINTS
C 14 DIMENSIONAL WITH MASS AND FLUX
C
C
      SUBROUTINE PRTN(KOUNT,NOI)
C
      IMPLICIT REAL*8(A-H,O-$)
      COMMON /XMMM/X(7),XS(8),Y(8)
      COMMON /T/TF,S,TO,TFMIN
C
      N=7
      M=8
      WRITE (6,1000)
      WRITE (6,1001) NOI,KOUNT
      WRITE (6,1002)
      WRITE (6,1003) (X(J),J=1,N)
      WRITE (6,1004)
      WRITE (6,1003) (Y(J),J=1,M)
      WRITE (6,1005) TF
 1000 FORMAT (29H0 ITER NO.    TRAJECTORY CALLS)
 1001 FORMAT (1H0,I6,5X,I6)
 1002 FORMAT (2H0X)
 1003 FORMAT (1X,1P5D23.15)
 1004 FORMAT (2H0Y)
 1005 FORMAT (5H0 TF=,1PD26.16)
      RETURN
      END
```

```
C   DCROUT/DCROUT14   8 DIM.
      SUBROUTINE DCROUT(AA,R,  ,EPS,NI,M,IND)
      DOUBLE PRECISION A,-,D,EPS,T,S,P,DT,AA
      DIMENSION A(8,8),  8,1),AA(8,8)
    5 IND=0
      N=NI
      DO 6 I=1,N
      DO 6 J=1,N
    6 A(I,J)=AA(I,J)
      IF(M)10,25,25
   10 M=N
      DO 20 I=1,N
      DO 15 J=1,N
   15 R(I,J)=0.D0
   20 R(I,I)=1.D0
   25 IC=0
      II=0
      T=DABS(A(1,1)   )
      DO 35 I=2,N
      IF(T-DABS(A(I,1)))30,35,35
   30 II=I
      T=DABS(A(I,1))
   35 CONTINUE
      IF(II)40,65,40
   40 IC=IC+1
      IF(M)45,55,45
   45 DO 50 J=1,M
      S=R(1,J)
      R(1,J)=R(II,J)
   50 R(II,J)=S
   55 DO 60 J=1,N
      S=A(1,J)
      A(1,J)=A(II,J)
   60 A(II,J)=S
   65 P=A(1,1)
      IF(DABS(P)-EPS)70,70,75
   70 IND=1
      D=0.D0
      GO TO 200
   75 DO 80 J=2,N
   80 A(1,J)=A(1,J)/P
      IF(M)85,95,85
   85 DO 90 J=1,M
   90 R(1,J)=R(1,J)/P
   95 DO 170 K=2,N
      KM=K-1
      T=-1.D0
      DO 105 I=K,N
      DO 98 J=1,KM
   98 A(I,K)=A(I,K)-A(I,J)*A(J,K)
      IF(T-DABS(A(I,K)))100,105,105
  100 T=DABS(A(I,K))
      II=I
  105 CONTINUE
      IF(II-K)110,135,110
  110 IC=IC+1
      IF(M)115,125,115
```

```
115 DO 120 J=1,M
    S=R(K,J)
    R(K,J)=R(II,J)
120 R(II,J)=S
125 DO 130 J=1,N
    S=A(K,J)
    A(K,J)=A(II,J)
130 A(II,J)=S
135 DT=A(K,K)
    IF(DABS(DT)-EPS)70,70,140
140 P=P*DT
    IF(K-N)145,155,145
145 KP=K+1
    DO 150 J=KP,N
    DO 148 I=1,KM
148 A(K,J)=A(K,J)-A(K,I)*A(I,J)
150 A(K,J)=A(K,J)/DT
155 IF(M)160,170,160
160 DO 165 J=1,M
    DO 162 I=1,KM
162 R(K,J)=R(K,J)-A(K,I)*R(I,J)
165 R(K,J)=R(K,J)/DT
170 CONTINUE
    IF(MOD(IC,2))175,180,175
175 P=-P
180 D=P
    IF(M)185,200,185
185 II=N
    DO 190 K=2,N
    KP=II
    II=II-1
    DO 190 J=1,M
    DO 190 I=KP,N
190 R(II,J)=R(II,J)-A(II,I)*R(I,J)
200 RETURN
    END
```

```
C TRAJ/TRAJRKS
C
C SEP AND HIGH THRUST
C 7/14 DIM. WITH POWER DEGRADATION.
C THIS ROUTINE SETS UP THE INPUT TO THE INTEGRATOR WHICH
C     EXTRAPOLATES THE TRAJECTORY FROM INITIAL TIME TO
C     FINAL TIME.  IT ALSO EVALUATES THE CHANGE IN TF AND
C     THE ERROR IN THE FINAL CONDITIONS.
C THIS PROGRAM IS CALLED BY ITER OR BY CONTROL
C IT CALL THE SUBPROGRAM DRKGS (RUNGA-KUTTA)
C MIN J. MAX H.
C 8 DIM. ZERF. T.C. OPTIONS.
C NOP=1--ALL 5 FINAL O.E. FIXED. =2--A,E,I ONLY FIXED.
C
C
      SUBROUTINE TRAJ
C
      IMPLICIT REAL*8(A-H,O-S), INTEGER (I-N)
C
      COMMON /XMMM/ZLO(7), STEP(8), ZERF(8)
      COMMON /TRA/TFMAX, DTO, UER, EW(14)
      COMMON /Z/Z(14), DERZ(14)
      COMMON /INT/ITF, IPR, IDIM, IDIM2, NIMAX
      COMMON /T/TF, SD, TO, TFMIN
      COMMON /ELEM/ZPO(7), ZPF(5)
      COMMON /DY/DYDT(8)
      COMMON /TC/NOP
      COMMON /HIGH/DVI1,D 12,IHI
      COMMON /A/A,AMU,PI
      COMMON /NOD/NODE
C
      EXTERNAL FUNCT, OUTP
      DIMENSION PRMT(5 , AUX(8,14),DERZ1(14),DP(5)
C
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) GO TO 9
C
C HIGH THRUST
C
      UP= (PI/2.DO)*ZLO(1)/DSQRT(1.DO+ZLO(1)**2)
      CUP= DCOS(UP)
      XK= CUP*(.75DO+.25DO*ZLO(2)/DSQRT(1.DO+ZLO(2)**2))
      DUM= (1.DO+CUP*XK)*DSQRT((CUP-XK)/(CUP+XK))
      XJ= DUM*ZLO(3)/DSQRT(1.DO+ZLO(3)**2)
      JM= 2
      VAR= ZPO(3)
      IF (NODE.EQ.0) VAR= ZLO(5)
      CALL MAINE(0.DO,0.DO,XK,UP,XJ,1.DO,1,JM,DP,DVI1)
      CALL OUTHI(JM,PI,ZPO(1),ZPO(2),VAR,IPR,Z,IDIM2)
      IF (NODE.EQ.0) GO TO 4
      Z(IDIM2+4)= Z(IDIM2+4)+ZLO(5)*Z(5)
      Z(IDIM2+5)= Z(IDIM2+5)-ZLO(5)*Z(4)
 4    DO 5 I= 1,5
 5    Z(I+IDIM2)= ZLO(4)*Z(I+IDIM2)*1.D4
      DO 6 I= 6,7
      Z(I)= ZPO(I)
 6    Z(I+IDIM2)= ZLO(I)
C
```

```
C LOW THRUST
C
 9      PRMT(1)= TO
        PRMT(2)= TF
        PRMT(3)= DTO
        PRMT(4)= UER
C
        IF ((IHI.EQ.2).OR.(IHI.EQ.3)) GO TO 15
C
C Z IS A VECTOR OF STATE AND COSTATE
        DO 10 I=1,IDIM2
         Z(I)=ZPO(I)
 10      Z(I+IDIM2)= ZLO(I)
C
C EW ARE ERROR WEIGHTS--INPUT TO THE INTEGRATOR
C
 15      DO 20 I=1,IDIM
 20      DERZ(I)=EW(I)
C
C CALL THE R-K INTEGRATOR
C
        CALL DRKGS(PRMT,Z,DERZ,IDIM,IHLF,FUNCT,OUTP,AUX)
        IF (IHLF.GT.10) GO TO 100
C
C Z IS NOW THE FINAL O.E..
C       ZERF THE ERROR IN THE FINAL CONDITIONS
C
        H=0.DO
        DO 30 I=1,IDIM2
 30      H= H + Z(I+7)*DERZ(I)
        ZERF(6)= Z(13)*1.D-3
        ZERF(7)= Z(14)*1.D-3
        DYDT(6)= DERZ(13)*1.D-3
        DYDT(7)= DERZ(14)*1.D-3
        TF1=TF*(STEP(8)+1.DO)
        CALL FUNCT(TF1,Z,DERZ1)
        H1=0.DO
        DO 35 I=1,IDIM2
 35      H1=H1+Z(I+7)*DERZ1(I)
        DYDT(8)= (H1-H)/(TF1-TF)
        ZERF(8)= H -1.DO
C
C FINAL CONDITION OPTION BRANCH
C
        GO TO (40,50), NOP
C
 40     DO 45 I=1,5
         ZERF(I)= Z(I) -ZPF(I)
 45      DYDT(I)= DERZ(I)
        RETURN
C
 50     ZERF(4)= (Z(3)*Z(9)-Z(2)*Z(10))*1.D-3
        ZERF(5)= (Z(5)*Z(11)-Z(4)*Z(12))*1.D-3
        DYDT(4)= DERZ(3)*Z(9)+Z(3)*DERZ(9)-DERZ(2)*Z(10)-Z(2)*DERZ(10)
        DYDT(4)= DYDT(4)*1.D-3
        DYDT(5)= DERZ(5)*Z(11)+Z(5)*DERZ(11)-DERZ(4)*Z(12)-Z(4)*DERZ(12)
        DYDT(5)= DYDT(5)*1.D-3
```

```
      IF (IHI.LT.3) GO TO 60
C
C FINAL HIGH THRUST IMPULSE
C
      CALL IMPLS(DVI2,IPR,0,Z,DERZ,IDIM2)
C
 60   ZERF(1)= Z(1) - ZPF(1)
      DUM1= DSQRT(Z(2)**2 + Z(3)**2)
      ZERF(2)= DUM1 - ZPF(2)
      DUM2= DSQRT(Z(4)**2 + Z(5)**2)
      ZERF(3)= DUM2 - ZPF(3)
      DYDT(1)= DERZ(1)
      DYDT(2)= 0.D0
      DYDT(3)= 0.D0
      IF (DUM1.GT.1.D-12) DYDT(2)= (Z(2)*DERZ(2) + Z(3)*DERZ(3))/DUM1
      IF (DUM2.GT.1.D-12) DYDT(3)= (Z(4)*DERZ(4) + Z(5)*DERZ(5))/DUM2
C
C SPECIAL CASE, E=0 AND/OR I=0
C
      IF (ZPF(2).NE.0.D0) GO TO 70
      ZERF(2)= Z(2)
      ZERF(4)= Z(3)
      DYDT(2)= DERZ(2)
      DYDT(4)= DERZ(3)
 70   IF (ZPF(3).NE.0.D0) RETURN
      ZERF(3)= Z(4)
      ZERF(5)= Z(5)
      DYDT(3)= DERZ(4)
      DYDT(5)= DERZ(5)
      RETURN
C
 100  IF (IHLF.EQ.11) WRITE (6,1000)
      IF (IHLF.EQ.12) WRITE (6,1001)
      IF (IHLF.EQ.13) WRITE (6,1002)
      STOP
C
 1000 FORMAT (68H0 THE NUMBER OF BISECTIONS OF THE ORIGINAL INCREMENT HA
     1S EXCEEDED 10)
 1001 FORMAT (27H0 INITIAL INCREMENT IS ZERO)
 1002 FORMAT (54H0 INITIAL INCREMENT HAS WRONG SIGN OR BOUNDS ARE WRONG)
      END
```

```
C OUTP/OUTPS
C
C SEP
C 14 DIMENSIONAL WITH MASS, FLUX
C THIS IS THE OUTP PROGRAM FOR THE P-C OR R-K
C   INTEGRATOR--FIXED TIME ONLY (ITF=3)
C   EQUINOCTIAL O.E. AND COSTATE ARE USED.
C INCLUDES SHADOW TIME.
C
C
      SUBROUTINE OUTP(T,Z,DERZ,IHLF,IDIM,PRMT)
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
      COMMON /INT/ITF,IPR,ID,IDIM2,NIMAX
      COMMON /A/A,AMU,PI
      COMMON /SHAD/ FEN,FEX,DFEN(5),DFEX(5),ISHAD
      COMMON/POWER/PO,C,POW,PH,ISUN,ISON,IPOW
      COMMON /ELEM/ Z0(12)
      COMMON /SOL/RS(4)
C
      DIMENSION PRMT(5), Z(14), DERZ(14), W(9), Z10(10)
C
      IF (ITF.NE.3) GO TO 10
      IF (IPR.EQ.0) RETURN
      IF (T.EQ.PRMT(1)) N=0
      IF (T.EQ.PRMT(1)) M=0
      N=N+1
      IF ((T.LT.(.9999999999D0*DFLOAT(M)*(PRMT(2)-PRMT(1))/DFLOAT(IPR)))
     1     .AND.(IHLF.LT.11).AND.(T.LT.(.999999999D0*PRMT(2)))) RETURN
      M=M+1
      CALL SUN(T,Z)
      IF (ISON.EQ.0) GO TO 2
      DO 1 I= 1,5
      Z10(I)= Z(I)
    1 Z10(I+5)= Z(I+IDIM2)
      CALL SHADOW(Z10)
    2 IF (IPOW) 4,3,4
    3 POW= PO
      IF (ISUN.EQ.1) POW=POW/RS(4)**2
      GO TO 5
    4 DUM1= DLOG10(Z(7))+14.D0
      POW= PO*DEXP(-.4364D-12*DUM1**10)
      IF (ISUN.EQ.1) POW= POW/RS(4)**2
    5 POW= POW-PH
      A= 2.D0*POW/(C*Z(6))
C
    6 TS=UTS*T
      TM=UTM*T
      TH=UTH*T
      TD=UTD*T
      DV= C*DLOG(Z0(6)/Z(6))*UTKM/UTS
      H= 0.D0
      DO 7 I=1,IDIM2
    7 H= H + Z(I+IDIM2)*DERZ(I) .
      W(1)= Z(1)*UTKM
      W(2)=0.D0
```

```
      DUMMY=Z(2)**2+Z(3)**2
      IF (DUMMY.GT.1.D-40) W(2)=DSQRT(DUMMY)
      W(3)=0.D0
      DUMMY=Z(4)**2 + Z(5)**2
      IF (DUMMY.GT.1.D-40) W(3)= 2.D0*DATAN(DSQRT(DUMMY))/DTR
      W(4)=0.D0
      IF ((Z(4).NE.0.D0).OR.(Z(5).NE.0.D0)) W(4)=DATAN2(Z(4),Z(5))/DTR
      W(5)= 0.D0
      IF ((Z(2).NE.0.D0).OR.(Z(3).NE.0.D0)) W(5)=DATAN2(Z(2),Z(3))/DTR
      W(5)=W(5)-W(4)
      IDIM3=IDIM2+1
C
C
      W(6)= Z(6)*UTKG
      W(7)= POW*UTKW
      W(9)= A*UTMS2
      W(8)= W(9)*W(6)
C
C
      WRITE (6,1001)
      WRITE (6,1002)
      WRITE (6,1003) T, TS, TH, TD, DV, N
      WRITE (6,1004)
      WRITE (6,1005) (Z(I),I=1,IDIM2)
      WRITE (6,1014)
      WRITE (6,1005) W
      WRITE (6,1006)
      WRITE (6,1005) (Z(I),I=IDIM3,IDIM)
      WRITE (6,1007)
      WRITE (6,1005) (DERZ(I),I=1,IDIM2)
      WRITE (6,1008)
      WRITE (6,1005) (DERZ(I),I=IDIM3,IDIM)
      WRITE (6,1009)
      PER=2.D0*PI*DSQRT(Z(1)**3/AMU)*UTH
      AP= W(1)*(1.D0+W(2))
      PE= W(1)*(1.D0-W(2))
      WRITE (6,1010) H,PER,PE,AP,IHLF
C
      IF (ISHAD.EQ.0) GO TO 30
      IF (FEX.LT.FEN) FEX=FEX+2.D0*PI
      TSHAD= (FEX-FEN+Z(2)*(DCOS(FEX)-DCOS(FEN))-Z(3)*(DSIN(FEX)
     1        -DSIN(FEN)))*PER/(2.D0*PI)
      FPER= TSHAD/PER
      WRITE (6,1013) TSHAD, FPER
      FEXD= FEX/DTR
      FEND= FEN/DTR
      WRITE (6,1015) FEND,FEXD
C
 30   RETURN
 10   WRITE (6,1000)
      STOP
C
 1000 FORMAT (56H0 ITF MUST EQUAL 3--I.E. NEED FIXED ESTIMATED FINAL TIM
     1E)
 1001 FORMAT (70H0 *****************************************************
     1*****************)
 1002 FORMAT (5H TIME,10X,10HTIME UNITS,15X,7HSECONDS,9X
```

```
     1   ,5HHOURS,11X,4HDAYS,9X,8HDV (K/S),10X,1HN)
1003 FORMAT (1P2D25.7,1P3D15.7,I9//)
1004 FORMAT (54H0 THE EQUINOCTIAL ORBITAL ELEMENTS, MASS, AND FLUX ARE)
1005 FORMAT (1P5D22.12//)
1006 FORMAT (16H0 THE COSTATE IS)
1007 FORMAT (32H0 THE DERIVITIVE OF THE STATE IS)
1008 FORMAT (34H0 THE DERIVITIVE OF THE COSTATE IS)
1009 FORMAT (1H0,7X,11HHAMILTONIAN,8X,12HPERIOD (HRS),7X,
     1     12HPERIGEE (KM),9X,11HAPOGEE (KM),5X,14HDIV. TIME STEP)
1010 FORMAT (2F20.12,1P2D20.10,I9//)
1012 FORMAT (1P4D22.12)
1013 FORMAT (24H0 TIME SPENT IN SHADOW =, F15.8,29H HOURS.  FRACTION OF
     1 PERIOD =, F15.8//)
1014 FORMAT (78H0 THE CLASSICAL O.E., MASS (KG), POWER (KW), THRUST (N)
     1, THRUST ACC (M/SEC**2))
1015 FORMAT (16H0 ENTRY ANGLE = ,F20.10,14H EXIT ANGLE = ,F20.10)
      END
```

```
C FUNCT/FUNCTS
C
C SEP
C COMPLEX MODEL --FLUX AS Z(7)
C 14 DIM. VERSION WITH FLUX AND MASS INCLUDED.
C THIS SUBROUTINE IS AN INTERFACE BETWEEN THE INTEGRATOR ROUTINE
C     AND THE QUADRATURE ROUTINE.
C INCLUDES SHADOW EFFECT.
C THIS ROUTINE ADD  THE EFFECT OF OBLATENESS (J2) TO THE DERIV,
C OBLATE CALCULATES THE EFFECT OF J2. RETURNED AS DZJ2.
C Z IS A VECTOR OF THE AVERAGED STATE AND COSTATE
C DERZ IS THE AVERAGED DERIVITIVE OF Z
C
C
C
      SUBROUTINE FUNCT(X,Z,DERZ)
C
C
      IMPLICIT REAL*8(A-H,O-$)
      COMMON /A/A,AMU,PI
      COMMON /J2/AJ2
      COMMON/JD/ TL
      COMMON /SHAD/ FEN,FEX,DFEN(5),DFEX(5),ISHAD
      COMMON/BCOM/B(9)
      COMMON/POWER/PO,C,POW,PH,ISUN,ISON,IPOW
      COMMON/ACOM/ AF(10)
      COMMON /SOL/RSUN(3),RS
C
      NAMELIST/PRINT/TT,PO,C,Z10,BETA,DUM1,POW,DP,DERZ10,HZ,HM
C
      DIMENSION Z(14), DERZ(14), G(10), H(10), DZJ2(10),GEX(10),GEN(10)
      DIMENSION Z10(10),DERZ10(10),G6(6),H6(6),DFL1(6),DFL(6)
C
      EXTERNAL FCT,FLUX
C
C SET UP COEFFS OF COSF AND SINF IN X1 AND Y1 AND PARTIALS
C
C
      BETA= 1.D0/(1.D0+DSQRT(1.D0-Z(2)**2-Z(3)**2))
      B(1)= 1.D0-Z(2)**2*BETA
      B(2)= Z(2)*Z(3)*BETA
      B(3)= 1.D0-Z(3)**2*BETA
      BETA3= BETA**3/(1.D0-BETA)
      A1= Z(2)**2*BETA3
      A2= Z(3)**2*BETA3
      A3= BETA+A1
      A4= BETA+A2
      B(4)= -Z(2)*(BETA+A3)
      B(5)= Z(3)*A3
      B(6)= -Z(2)*A2
      B(7)= -Z(3)*A1
      B(8)= Z(2)*A4
      B(9)= -Z(3)*(BETA+A4)
C
C
C
C
```

```
C
C
      DO 10 I=1,5
       Z10(I)=Z(I)
 10    Z10(I+5)= Z(I+7)
C
      DUM= 1.D0
      QEX= -PI
      QEN= PI
C
      CALL SUN(X,Z)
      ISHAD=0
      IF (ISON.EQ.0) GO TO 12
      CALL SHADOW(Z10)
C
C CALCULATE POWER, DPDN AND ACCELERATION
C
 12   IF (IPOW) 20,15,20
C
C CONSTANT POWER
C
 15   DO 16 I= 1,6
 16    DFL(I)= 0.D0
      POW= PO
      IF (ISUN.EQ.1) POW= POW/RS**2
      DP= 0.D0
      DERZ(7)= 0.D0
      GO TO 23
C
 20   IF= 4
      AIF= 2.D0*PI/FLOAT(IF)
      IFV= 0
      F1= -PI
      DO 34 I= 1,6
 34    DFL(I)= 0.D0
 35   F2= F1+AIF
      CALL QUAD(F1,F2,FLUX,DFL1,Z10,G6,H6,6)
      IFV= IFV+1
      F1= F2
      DO 36 I= 1,6
 36    DFL(I)= DFL(I)+DFL1(I)
      IF (IFV.LT.IF) GO TO 35
C
      DERZ(7)= AF(10)*DFL(1)
      IF (Z(7).EQ.0.D0) Z(7)= .5D0*DERZ(7)*Z(1)**1.5D0
      DERZ(7)= DERZ(7)/(2.D0*PI)
C
C
      DUM1= DLOG10(Z(7))+14.D0
      POW= PO*DEXP(-.4364D-12*DUM1**10)
      IF (ISUN.EQ.0) GO TO 22
      POW= POW/RS**2
C
 22   DP= -POW*.4364D-11*DUM1**9/(Z(7)*DLOG(10.D0))
C
 23   POW= POW-PH
      IF (POW.LE.0.D0) GO TO 500
```

```
C
C
      A= 2.D0*POW/(C*Z(6))
C
C
      IF (ISHAD.EQ.0) GO TO 30
C
C SHADOW INFLUENCE
C
      IF (FEN.LE.FEX) FEN= FEN+ 2.D0*PI
      QEX= FEX
      QEN= FEN
      DDC= DCOS(FEN)-DCOS(FEX)
      DDS= DSIN(FEN)-DSIN(FEX)
      DUM= (FEN-FEX+Z(2)*DDC-Z(3)*DDS)/(2.D0*PI)
C
      SN= (1.D0-Z(3)*DCOS(FEN)-Z(2)*DSIN(FEN))/(2.D0*PI)
      SX= (1.D0-Z(3)*DCOS(FEX)-Z(2)*DSIN(FEX))/(2.D0*PI)
C
C
   30 CALL QUAD(QEX,QEN,FCT,DERZ10,Z10,G,H,10)
C
      HM= -2.D0*POW/C**2
      DERZ(6)= HM*DUM
      HM= Z(13)*HM
C
      IF (ISHAD.EQ.0) GO TO 60
C
C SHADOW INFLUENCE
C
      CALL FCT(QEN,QEX,Z10,GEN,GEX)
      HWX=0.D0
      HWN= 0.D0
      DO 40 I=1,5
       HWX= HWX+Z10(I+5)*GEX(I)
   40  HWN= HWN+Z10(I+5)*GEN(I)
      HWX= HWX+HM*SX
      HWN= HWN+HM*SN
C
      DO 50 I=1,5
   50   DERZ10(I+5)= DERZ10(I+5)-HWN*DFEN(I)+HWX*DFEX(I)
      DERZ10(7)= DERZ10(7)-HM*DDC/(2.D0*PI)
      DERZ10(8)= DERZ10(8)+HM*DDS/(2.D0*PI)
      HM= HM*DUM
C
C
   60 IF (AJ2.LE.0.D0) GO TO 90
C
C OBLATENESS EFFECT
C
   70 CALL OBLATE(AJ2,Z10,DZJ2,1)
      DO 80 I=1,10
   80  DERZ10(I)= DERZ10(I)+DZJ2(I)
C
C
   90 HZ= 0.D0
      DO 100 I=1,5
```

```
 100    HZ= HZ+Z10(I+5)*DERZ10(I)
C
       DERZ(13)= HZ/Z(6)
       DERZ(14)= -DP*(HZ+HM)/POW
       DO 110 I= 1,5
        DERZ(I)= DERZ10(I)
 110    DERZ(I+7)= DERZ10(I+5)-AF(10)*Z(14)*DEL(I+1)/(2.D0*PI)
C
C
C      TT=X
C      WRITE (6,PRINT)
C
       RETURN
C
 500   WRITE (6,1000)
 1000  FORMAT( 'EITHER POW OR TL <= 0')
       STOP
       END
```

```
C EARTH3
C
C
C THIS SUBPROGRAM SETS THE VALUES FOR EARTH'S ORBITAL ELEMENTS
C AND CALCULATES THE MEAN ANOMALY AT THE INITIAL TIME
C I.E. TAKEN FROM BATTIN, 1964, EPOCH 1960 JAN. 1.5, JD=2436935.
C INPUT
C      TL--INITIAL TIME, BEGINNING OF LOW THRUST TRAJ
C OUTPUT
C      C(1)--EARTH'S SEMIMAJOR AXIS
C      C(2)--EARTH'S ECCENTRICITY
C      C(3)--ARGUMENT OF PERIH.
C      C(4)--MEAN ORBITAL MOTION
C      C(5)--MEAN ANOMALY AT TL
C      C(6)--COSINE OF ANGLE OF OBLIQUITY
C      C(7)--SINE OF ANGLE OF OBLIQUITY
C      C(17)    NG. OF N MAG POLE AT TIME TL
C      C(18        OF N. MAG. POLE **** SET TO 90 DEGREES
C      C(19)--EARTH ROT. FREQ. **** SET TO ZERO
C      C(20)--2*PI
C
C
C
      SUBROUTINE EARTH
C
      IMPLICIT REAL*8(A-H,O-$)
C
      COMMON /JD/ TL
      COMMON /TERRA/ C(20)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
C
C
      C(1) = 1.D0
      C(2)= .016726D0
      C(3)= 102.25253D0
      C(4)= .985609D0
C
C MEAN ANOMALY AT EPOCH
      AN= 100.15815D0-C(3)
      B0= AN
C MEAN ANOMALY AT TIME TL
      AN= AN+C(4)*(TL-2436935.D0)
      AN=AN/360.D0
      AN=AN-IDINT(AN)
C
      C(5)= AN*360.D0*DTR
      C(3)= C(3)*DTR
      C(4)= C(4)*DTR*UTD
C
      DUM= 23.45*DTR
      C(6)= DCOS(DUM)
      C(7)=DSIN(DUM)
C
C CALCULATE THE ROTATION MATRIX FROM EQUATORIAL TO GMT COORD.
C
      E0= B0+(2.D0*C(2)-.25D0*C(2)**3)*DSIN(B0)+1.25D0*C(2)**2
     1    *DSIN(2.D0*B0)+1.083333333333333D0*C(2)**3*DSIN(3.D0*B0)
```

```
      BO= BO+C(3)
      RX= -DCOS(BO)
      RY= -DSIN(BO)*C(6)
      RHO= DSQRT(RX**2+RY**2)
      RX= RX/RHO
      RY= RY/RHO
      C(8)= RX
      C(9)= -RY
      C(10)= 0.D0
      C(11)= RY
      C(12)= RX
      C(13)= 0.D0
      C(14)= 0.D0
      C(15)= 0.D0
      C(16)= 1.D0
C
      C(17)= 289.9D0*DTR
      C(18)= 90.D0*DTR
      C(19)= 0.D0
      C(20)= 360.D0*DTR
      C(17)= C(17)+C(19)*(TL-2436935.D0)/UTD
C
      RETURN
      END
```

```
C SUN
C
C
C THIS PROGRAM CALCULATES THE EARTH TO SUN DIRECTION AND
C DISTANCE FOR A GIVEN TIME.  OUTPUT IN THE EQUINOCTIAL
C COORDINATE FRAME.
C ALSO CALCULATES FLUX FACTORS.
C INPUT
C      Z--10 VECTOR OF EQ. O.E. AND COSTATE(NOT USED)
C      AE--EARTH ORBIT SEMIMAJOR AXIS
C      EC--EARTH ORBIT ECCENTRICITY
C      W--LONG. OF PERIH.
C      ENE--MEAN ORBITAL MOTION
C      AN--MEAN ANOMALY AT BEGINNING OF TRAJECTORY (TO)
C      COB--COS OF ANGLE OF OBLIQUITY
C      SOB--SIN OF ANGLE OF OBLIQUITY
C      CMA--ROTATION MATRIX FROM EQUAT. TO GMT COORD.
C      A1--LONG. OF N MAG POLE AT TL
C      A2--LAT OF N MAG POLE AT TL
C      WE--EARTH ROTATIONAL FREQ.
C      TUPI--2*PI
C      T--PRESENT TIME
C OUTPUT
C      RS--UNIT VECTOR FROM EARTH TO SUN, EQUINOCTIAL COORD.
C      R--DISTANCE FROM EARTH TO SUN AT TIME T
C      C--FLUX FACTORS
C
C
C
      SUBROUTINE SUN(T,Z)
C
      IMPLICIT REAL*8(A-H,O-$)
C
      COMMON /SOL/ RS(3), R
      COMMON/TERRA/ AE,EC,W,ENE,AN,COB,SOB,CMA(3,3),A1,A2,WE,TUPI
      COMMON/CCOM/C(6)
C
      DIMENSION RS1(3),CM(3,3),Z(14)
      DIMENSION VN(3), V2(3),CMP(3,2),CMO(3,2)
C
C
C MEAN ANOMALY AT TIME T
      AA= AN+ENE*T
C
C TRUE ANOMALY--CORRECT THRU ECCENTRICITY CUBED
      F=AA+(2.D0*EC-.25D0*EC**3)*DSIN(AA)+1.25D0*EC**2*DSIN(2.D0*AA)
     1        +1.0833333333333D0*EC**3*DSIN(3.D0*AA)
      B=F+W
C
C DISTANCE BETWEEN EARTH AND SUN
      R=AE*(1.D0-EC**2)/(1.D0+EC*DCOS(F))
C
C UNIT VECTOR TO SUN, EQUATORIAL COORD.
      RS1(1)=-DCOS(B)
      RS1(2)=-COB*DSIN(B)
      RS1(3)=-SOB*DSIN(B)
C
```

```
C TRANSFORM TO EQUINOCTIAL COORD.
      AB= 1.D0+Z(4)**2+Z(5)**2
      CM(1,1)= (1.D0-Z(4)**2+Z(5)**2)/AB
      CM(2,1)= 2.D0*Z(4)*Z(5)/AB
      CM(3,1)= -2.D0*Z(4)/AB
      CM(1,2)= CM(2,1)
      CM(2,2)= (1.D0+Z(4)**2-Z(5)**2)/AB
      CM(3,2)= 2.D0*Z(5)/AB
      CM(1,3)= -CM(3,1)
      CM(2,3)= -CM(3,2)
      CM(3,3)= (1.D0-Z(4)**2-Z(5)**2)/AB
      DO 10 I=1,3
       RS(I)= 0.D0
       DO 10 J=1,3
 10     RS(I)= RS(I)+CM(J,I)*RS1(J)
C
C
C CALCULATION OF C(6), FLUX FACTORS
C
      THETA= WE*T+A1
      THETA= THETA/TUPI
      THETA= THETA-IDINT(THETA)
      THETA= THETA*TUPI
C
C VN IS VECTOR THRU N MAG POLE AT TIME T IN GMT COORD
      A3= DCOS(A2)
      VN(1)= DCOS(THETA)*A3
      VN(2)= DSIN(THETA)*A3
      VN(3)= DSIN(A2)
C
C V2 IS N IN EQUATORIAL COORDS.
      DO 20 I=1,3
       V2(I,=0.D0
       DO 20 J=1,3
 20     V2(I)= V2(I)+VN(J)*CMA(J,I)
C
C CALC PARTIALS OF F AND G UNIT VECTORS
      AB= 2.D0/AB
      DO 30 I=1,3
       CMP(I,1)= -AB*(Z(5)*CM(I,2)+CM(I,3))
       CMP(I,2)= AB*Z(5)*CM(I,1)
       CMQ(I,1)= AB*Z(4)*CM(I,2)
 30    CMQ(I,2)= AB*(-Z(4)*CM(I,1)+CM(I,3))
C
C C(1),C(2) ARE COEFF OF X1,Y1 IN EQN FOR SIN(ALPHA)
C C(3),C(4) ARE COEFF OF X1,Y1 IN DSIN DP
C C(5),C(6) "         "              DQ
      DO 40 K=1,2
       C(K)= 0.D0
       C(K+2)= 0.D0
       C(K+4)= 0.D0
       DO 40 J=1,3
        C(K)= C(K)+V2(J)*CM(J,K)
        C(K+2)= C(K+2)+V2(J)*CMP(J,K)
 40     C(K+4)= C(K+4)+V2(J)*CMQ(J,K)
C
      RETURN
      END
```

```
C SHADOW/SHADOW1
C
C THIS PROGRAM DETERMINES IF A GIVEN ORBIT PASSES THROUGH
C THE EARTH'S SHADOW AND IF SO WHAT THE ENTRY AND EXIT
C ANGLES ARE. IT ALSO CALCULATES THE PARTIAL DERIVITIVE
C VECTOR OF F WRT THE O.E. AT THE ENTRY AND EXIT POINTS.
C INPUT AND OUTPUT IN EQUINOCTIAL COORD.
C INPUT
C     Z-- 10 VECTOR OF O.E. AND COSTATE (COSTATE NOT USED)
C     XSUN,YSUN,ZSUN--SUN'S DIRECTION IN EQ. COORD. (UNIT VECTOR)
C OUTPUT
C     FEN--ENTRY ANGLE
C     FEX--EXIT ANGLE
C     DFEN--DIRIVITIVE OF F AT ENTRY
C     DFEX--DIRIVITIVE OF F AT EXIT
C     ISHAD--FLAG=0 IF ORBIT NOT INTERSECT SHADOW
C                =2 IF ORBIT ENTER AND EXIT FROM SHADOW
C
C
C
      SUBROUTINE SHADOW(Z)
C
      IMPLICIT REAL*8(A-H,O-$)
C
      COMMON /SOL/ XSUN,YSUN,ZSUN,RSUN
      COMMON /SHAD/ FEN,FEX,DFEN(5),DFEX(5),ISHAD
C
      DIMENSION DSDX(5),AP(4),RT(4),Z(10)
C
      NAMELIST /DUMFEN/ FEN,DUM
      NAMELIST /DUMFEX/ FEX,DUM
C     NAMELIST /EQ/ AP,RT,NRE
C     NAMELIST /EQ2/BETA,B1,B2,B3,D1,D2,D3,H1,H2,H3,G1,G2,A0
C     NAMELIST /PR4/ DUM,DSDF,DSDX
C     NAMELIST /PR5/ I,II,CF,SF,X1,Y1
C     NAMELIST /PR6/ I,II,FON
C
C
C CALCULATE POLYNOMIAL COFF.
C
      BETA= DSQRT(1.D0-Z(2)**2-Z(3)**2)
      BETA= 1.D0/(1.D0+BETA)
      B1= 1.D0-Z(2)**2*BETA
      B2= Z(2)*Z(3)*BETA
      B3= 1.D0-Z(3)**2*BETA
      D1= 1.D0-XSUN**2
      D2= 1.D0-YSUN**2
      D3= 2.D0*XSUN*YSUN
      C1=B2**2
      C2=B3**2
      C3=B2*B3
      C4=B1*B2
      H1=D1*(B1**2-C1)+D2*(C1-C2)-D3*(C4-C3)
      H2=-2.D0*(D1*B1*Z(3)+D2*B2*Z(2))+D3*(B2*Z(3)+B1*Z(2))
      H3=D1*(C1+Z(3)**2)+D2*(C2+Z(2)**2)-D3*(C3+Z(2)*Z(3))
     1   -1.D0/Z(1)**2
      G1= 2.D0*(D1*C4+D2*C3)-D3*(C1+B1*B3)
```

```
      G2= -2.D0*(D1*B2*Z(3)+D2*B3*Z(2))+D3*(B3*Z(3)+B2*Z(2))
      C1= G1**2
      C2= G2**2
      C3= G1*G2
      A0= H1**2+C1
C     WRITE (6,EQ2)
      AP(1)=2.D0*(H1*H2+C3)/A0
      AP(2)= (H2**2+2.D0*H3*H1-C1+C2)/A0
      AP(3)= 2.D0*(H3*H2-C3)/A0
      AP(4)= (H3**2-C2)/A0
C
C CALL SUBROUTINE TO SOLVE A QUARTIC EQN.
C
      CALL DQRTIC(AP,RT,NRE)
C
C     WRITE (6,EQ)
C
C NRE= NUMBER OF REAL ROOTS, MUST BE EQUAL TO 0,2,OR 4
C ROOTS ARE RT(I), OR RT(1),RT(2),RT(3)+-RT(4)*I, OR
C RT(1)+-RT(2)*I,RT(3)+-RT(4)*I
      IF ((NRE.EQ.1).OR.(NRE.EQ.3)) GO TO 130
C
      FEN= 100.D0
      FEX= 100.D0
      ISHAD= 0
      I=0
 10   I=I+1
      IF ((ISHAD.EQ.2).OR.(I.EQ.(NRE+1))) GO TO 120
C
      CF= RT(I)
      SF= DSQRT(1.D0-CF**2)
C
C HEMISPHERE CHECK
      II=1
 20   X1= B1*CF+B2*SF-Z(3)
      Y1= B3*SF+B2*CF-Z(2)
C     WRITE (6,PR5)
      IF ((X1*XSUN+Y1*YSUN).LT.0.D0) GO TO 40
 30   IF (II.EQ.2) GO TO 10
      II=2
      SF=-SF
      GO TO 20
C
C IS SHADOW EQUATION ZERO?
 40   EQN= D1*X1**2+D2*Y1**2-D3*X1*Y1-Z(1)**(-2)
C     WRITE (6,PR6)
      IF (DABS(EQN).GT.1.D-6) GO TO 30
C
C ROOT HAS PASSED TESTS--NOW CHECK TO SEE IF EXIT OR ENTRY ANGLE
      DXDF= -B1*SF+B2*CF
      DYDF= -B2*SF+B3*CF
      DSDF= (2.D0*D1*X1-D3*Y1)*DXDF +(2.D0*D2*Y1-D3*X1)*DYDF
      DUM= DATAN2(SF,CF)
      IF (DSDF) 70,50,60
C
C ORBIT IS TANGENT TO SHADOW
 50   WRITE (6,1010)
```

```
      GO TO 30
C
C IS FEX ALREADY FOUND?
 60   IF (FEX.EQ.1.D2) GO TO 80
C YES
      WRITE (6,DUMFEX)
      GO TO 30
C
C IS FEN ALREADY FOUND?
 70   IF (FEN.EQ.1.D2) GO TO 80
C YES
      WRITE (6,DUMFEN)
      GO TO 30
C
C CALCULATE DSDX
 80   ZETA= Z(3)*SF-Z(2)*CF
      BETA3= BETA**3/(1.D0-BETA)
      PZ5=  Z(2)*BETA3
      PZ6= Z(3)*BETA3
      DXDH= -2.D0*Z(2)*BETA*CF+Z(3)*BETA*SF+PZ5*ZETA*Z(2)
      DXDK= Z(2)*BETA*SF-1.D0+PZ6*Z(2)*ZETA
      DYDH= Z(3)*BETA*CF-1.D0-PZ5*Z(3)*ZETA
      DYDK= -2.D0*Z(3)*BETA*SF+Z(2)*BETA*CF-PZ6*Z(3)*ZETA
      DSDX(1)= 2.D0*Z(1)**(-3)
      DUM1= 2.D0*D1*X1-D3*Y1
      DUM2= 2.D0*D2*Y1-D3*X1
      DSDX(2)= DUM1*DXDH+DUM2*DYDH
      DSDX(3)= DUM1*DXDK+DUM2*DYDK
      D=2.D0/(1.D0+Z(4)**2+Z(5)**2)
      DXSP= (-YSUN*Z(5)-ZSUN)*D
      DXSQ= YSUN*Z(4)*D
      DYSP= XSUN*Z(5)*D
      DYSQ= (-XSUN*Z(4)+ZSUN)*D
      DUM1= -2.D0*X1*(X1*XSUN+Y1*YSUN)
      DUM2= -2.D0*Y1*(Y1*YSUN+X1*XSUN)
      DSDX(4)= DUM1*DXSP+DUM2*DYSP
      DSDX(5)= DUM1*DXSQ+DUM2*DYSQ
C     WRITE (6,PR4)
      ISHAD=ISHAD+1
      IF (DSDF.LT.0.D0) GO TO 100
C
C EXIT ANGLE AND DERIVITIVE
      FEX=DUM
      DO 90 J=1,5
 90    DFEX(J)= DSDX(J)/DSDF
      GO TO 10
C
C ENTRY ANGLE AND DERIVITIVE
 100  FEN=DUM
      DO 110 J=1,5
 110   DFEN(J)= DSDX(J)/DSDF
      GO TO 10
C
C
 120  IF ((ISHAD.EQ.0).OR.(ISHAD.EQ.2)) RETURN
      WRITE (6,1020) ISHAD
      ISHAD= 0
```

```
      RETURN
C
 130  WRITE (6,1030) NRF
      STOP
C
C
 1010 FORMAT (33H0 DSDF=0. ORBIT TANGENT TO SHADOW)
 1020 FORMAT (15H0 ERROR--IDSHAD=,I4)
 1030 FORMAT (49H0 DQRTIC HAS RETURNED WITH NUMBER OF REAL ROOTS =,I4)
      END
```

```
      SUBROUTINE DQRTIC(C,R,NRE)
C
C
C     SOLVES POLYNOMIAL EQUATION OF THE TYPE
C     X**4+C(1)*X**3+C(2)*X**2+C(3)*X+C(4)=0
C
C     THE COEFFICIENT OF X**4 IS ASSUMED TO BE 1
C
C     R CONTAINS THE ROOTS
C
C     NRE CONTAINS THE NUMBER OF REAL ROOTS
C
C     IF THERE ARE TWO REAL ROOTS THEY WILL BE IN
C     R(1) AND R(2), WITH THE COMPLEX ROOTS R(3)+-R(4)*I
C
C     IF THERE ARE NO REAL ROOTS,THE COMPLEX
C     ROOTS ARE R(1)+-R(2)*I AND R(3)+-R(4)*I
C
      DIMENSION C(4),R(4),CP(3),Y(3)
      DOUBLE PRECISION C,R,CP,Y,C1SQ,A,B,D,E,F,REAL,DSCR,RAD
      C1SQ=C(1)**2
      CP(1)=-C(2)
      CP(2)=C(1)*C(3)-4.D0*C(4)
      CP(3)=(4.D0*C(2)-C1SQ)*C(4)-C(3)**2
    5 CALL DCUBIC(CP,Y,DUM)
    8 A=C1SQ/4.D0-C(2)+Y(1)
      B=.5D0*C(1)*Y(1)-C(3)
      D=.25D0*Y(1)*Y(1)-C(4)
      IF(A) 10,10,15
   10 E=0.
      GO TO 20
   15 E=DSQRT(A)
   20 IF(D) 25,25,30
   25 F=0.
      GO TO 50
   30 F=DSIGN(DSQRT(D),B)
   50 NRE=0
      REAL=-.25D0*C(1)+.5D0*E
      DSCR=REAL*REAL-.5D0*Y(1)+F
   53 RAD=DSQRT(DABS(DSCR))
      IF(DSCR)60,55,55
   55 NRE=2
      R(1)=REAL+RAD
      R(2)=REAL-RAD
      GO TO 65
   60 R(3)=REAL
      R(4)=RAD
   65 REAL=REAL-E
      DSCR=REAL*REAL-.5D0*Y(1)-F
   68 RAD=DSQRT(DABS(DSCR))
      IF(DSCR)80,70,70
   70 NRE=NRE+2
      R(NRE)=REAL-RAD
      R(NRE-1)=REAL+RAD
      GO TO 90
   80 R(NRE+1)=REAL
      R(NRE+2)=RAD
   90 RETURN
      END
```

```
      SUBROUTINE DCUBIC(C,R,NRE)
C
C
C     SOLVES POLYNOMIAL EQUATION OF THE TYPE
C     X**3+C(1)*X**2+C(2)*X+C(3)=0
C
C     THE COEFFICIENT OF X**3 IS ASSUMED TO BE 1
C
C     R CONTAINS THE ROOTS
C
C     NRE CONTAINS THE NUMBER OF REAL ROOTS
C
C     IF THERE IS ONE REAL ROOT IT WILL BE
C     IN R(1),WITH THE COMPLEX ROOTS R(2)+-R(3)*I
C
      DIMENSION C(3),R(3)
      DOUBLE PRECISION C,R,C1SQ,P,Q,DEL,T,A,CRTA,CRTB,PHI3,CON,Y
      DOUBLE PRECISION B,SQ,HQ
      C1SQ=C(1)**2
      P=C(2)-C1SQ/3.D0
      Q=C(3)-(C(2)/3.D0-2.D0*C1SQ/27.D0)*C(1)
      DEL=4.D0*P**3+27.D0*Q*Q
      T=C(1)/3.D0
    5 IF(DEL)20,10,10
   10 SQ=DSQRT(DEL/108.D0)
      HQ=.5D0*Q
      A=-HQ+SQ
      B=-HQ-SQ
      CRTA=DSIGN(DABS(A)**.33333333333333333D0,A)
      CRTB=DSIGN(DABS(B)**.33333333333333333D0,B)
   15 Y=CRTA+CRTB
      R(1)=Y-T
      R(2)=-.5D0*Y-T
      R(3)=.86602540378443865D0*(CRTA-CRTB)
      NRE=1
      GO TO 40
   20 PHI3=DATAN2(DSQRT(-DEL/27.D0),-Q)/3.D0
   22 CON=2.D0*DSQRT(-P/3.D0)
   30 R(1)=CON*DCOS(PHI3)-T
      R(2)=-CON*DCOS(1.0471975511965977D0-PHI3)-T
      R(3)=-CON*DCOS(1.0471975511965977D0+PHI3)-T
      NRE=3
   40 RETURN
       END
```

```
C FLUX
C
C
C
C THIS PROGRAM CALCULTES THE FLUX DERIVITIVE AND
C THE INTERAND OF THE FLUX CONTRIBUTION TO THE O.F.
C COSTATE EQUATION
C ONLY G(1), PFPX, PFPY, AND PFPSA ARE DEPENDENT
C ON THE FORM OF N DOT
C
C
      SUBROUTINE FLUX(F1,F2,Z,H,G)
C
      IMPLICIT REAL*8(A-H,O-$)
      COMMON /BCOM/B(9)
      COMMON /CCOM/C(6)
      COMMON /ACOM/AF(10)
      DIMENSION FI(9),FIU(9),FIV(9),FIW(9)
      DIMENSION Z(1),H(1),G(1),PF(5)
C
C
      M=0
      F= F1
C
C CALCULTE GEOMAGNETIC COORDINATES
C
 5    CF= DCOS(F)
      SF= DSIN(F)
      X1= Z(1)*(B(1)*CF+B(2)*SF-Z(3))
      Y1= Z(1)*(B(2)*CF+B(3)*SF-Z(2))
      R= DSQRT(X1**2+Y1**2)
      U= 1.D0/R
      SA= (C(1)*X1+C(2)*Y1)/R
      SGN= 0.D0
      DUM1= DABS(SA)
      IF (DUM1.GT.1.D-10) SGN= SA/DUM1
      SA= DUM1
      CA= 0.D0
      IF (SA.LT.1.D0) CA= DSQRT(1.D0-SA**2)
      IF (CA.LE.1.D-10) CA= 1.D-10
      S= 1.D0-Z(3)*CF-Z(2)*SF
C
C FLUX RATE EQUATION
C
      FI(1)= U*CA
      FI(2)= U**2*CA
      FI(3)= FI(2)*SA
      FI(4)= U*SA
      FI(5)= U**3*SA**2
      FI(6)= FI(5)*SA
      FI(7)= DSQRT(U)*CA
      FI(8)= U**.25D0*CA
      FI(9)= U**.2D0*CA
C
      SUM= 0.D0
      DO 6 I=1,9
 6     SUM= SUM+AF(I)*FI(I)
```

```
C
      G(1)= DEXP(SUM)
C
C
C PARTIAL OF X1 AND Y1 WRT H AND K
C
      PXPH= Z(1)*(B(4)*CF+B(5)*SF)
      PYPH= Z(1)*(B(5)*CF+B(6)*SF-1.D0)
      PXPK= Z(1)*(B(7)*CF+B(8)*SF-1.D0)
      PYPK= Z(1)*(B(8)*CF+B(9)*SF)
C
C
      FIU(1)= CA
      FIU(2)= 2.D0*U*CA
      FIU(3)= FIU(2)*SA
      FIU(4)= SA
      FIU(5)= 3.D0*FI(5)/U
      FIU(6)= FIU(5)*SA
      FIU(7)= .5D0*FI(7)/U
      FIU(8)= .25D0*FI(8)/U
      FIU(9)= .2D0*FI(9)/U
C
      FIV(1)= 0.D0
      FIV(2)= 0.D0
      FIV(3)= U**2*CA
      FIV(4)= U
      FIV(5)= 2.D0*U**3*SA
      FIV(6)= FIV(5)*SA*1.5D0
      FIV(7)= 0.D0
      FIV(8)= 0.D0
      FIV(9)= 0.D0
C
      FIW(1)= U
      FIW(2)= U**2
      FIW(3)= FIW(2)*SA
      FIW(4)= 0.D0
      FIW(5)= 0.D0
      FIW(6)= 0.D0
      FIW(7)= U**.5D0
      FIW(8)= U**.25D0
      FIW(9)= U**.2D0
C
      SUM= 0.D0
      DO 7 I= 1,9
 7     SUM= SUM+AF(I)*FIU(I)
      SUM= -G(1)*SUM/R**3
C
      PFPX= SUM*X1
      PFPY= SUM*Y1
C
      SUM= 0.D0
      DO 8 I= 1,9
 8     SUM= SUM+AF(I)*(FIV(I)-FIW(I)*SA/CA)
C
      PFPSA= G(1)*SUM*SGN
C
C
```

```
C
      PF(1)= (PFPX*X1+PFPY*Y1)/Z(1)
      DUM1= PFPX+PFPSA*(C(1)-Y1*SA/R)/R
      DUM2= PFPY+PFPSA*(C(2)-Y1*SA/R)/R
      PF(2)= DUM1*PXPH+DUM2*PYPH
      PF(3)= DUM1*PXPK+DUM2*PYPK
      PF(4)= PFPSA*(C(3)*X1+C(4)*Y1)/R
      PF(5)= PFPSA*(C(5)*X1+C(6)*Y1)/R
C
      DO 10 I= 1,5
 10    G(I+1)= PF(I)*S
      G(3)= G(3)-G(1)*SF
      G(4)= G(4)-G(1)*CF
      G(1)= G(1)*S
C
      IF (M.EQ.1) RETURN
      F= F2
      M= 1
      DO 20 I= 1,6
 20    H(I)= G(I)
      GO TO 5
      END
```

```
C INPUT/INPUTN
C
C NEP AND HIGH THRUST.
C THIS SUBPROGRAM IS CALLED BY CONTL AND READS AND PRINTS
C   ALL INITIAL DATA AS WELL AS SETS INITIAL CONSTANTS.
C   THE UNITS ARE BASED ON INTERNAL MU=1.0, INTERNAL DISTANCE
C   UNIT=1 EARTH RADII, AND EXTERNAL MU= 398601.2 KM*KM*KM/
C   SEC*SEC AND EARTH RADII= 6378.16 KM.  A CIRCULAR
C   ORBIT AT 1 EARTH RADII WOULD HAVE A PERIOD OF 2 PI INTERNAL
C   TIME UNITS.
C   TO BE USED WITH 6 DIM. ZERF.
C INPUT
C  LOW/HIGH                        HIGH/LOW/HIGH
C               INITIAL ORBIT
C  A (KM)
C  E
C  I (DEG)
C  LONG. ASC. NODE (DEG)            NOT USED
C  ARG. OF PERGEE (DEG)             NOT USED
C               INITIAL GUESSES
C  LAMBDA A                        LIKE UPSILON
C  LAMBDA H                        LIKE SMALL'S K
C  LAMBDA K                        LIKE SMALL'S J
C  LAMBDA P                        SCALE FACTOR
C  LAMBDA G                        DUMMY--NOT USED
C            DESIRED FINAL ORBIT
C  A
C  E
C  I
C  NODE (NOT USED IF NOP=2)         NOT USED
C  PERIGEE (NOT USED IF NOP=2)      NOT USED
C
C  TF2 (DAYS), GUESS FOR FINAL TIME
C  PKW (KW), POWER
C  SPIM (SEC), SPECIFIC IMPULSE OF NEP
C  AMO (KG), INITIAL MASS (NEP)
C  IRDFLG                                          NOMINAL
C  1 END OF INPUT
C  2 IPR, PRINT FLAG                               0
C  3 NIMAX, MAX. NO. OF ITERATIONS                 20
C  4 TFMAX2 (DAYS), MAX. TF                        190.
C  5 DT2 (DAYS), TIME STEP FOR D.E.                1.
C  6 UEB, UPPER ERROR BOUND FOR D.E.               1.D10
C  7 EW, ERROR WEIGHTS                             1.,1.,1.,1.,1.,0.,...
C  8 UTKM, EQUATORIAL EARTH RADIUS                 6378.16
C  9 UM (KM**3/SEC**2), EARTH GRAV. CONST.         398601.2
C 10 NOP = 1, 5 O.E. SPECIFIED AT TF               1
C        = 2, 3 O.E. SPECIFIED AT TF
C 11 SETS OBLATENESS COEFF.AJ2.=1.08270-3          0.
C 12 STEP, STEP SIZE FOR NUM. DIFF. (5 DIM)        1.D-6
C    KSTEP = 0, STEP AS FRACTION IN ITER           0
C          = 1, STEP AS CONSTANT IN ITER
C 13 IPOW = 0, CONSTANT POWER                      0
C         = 1, EXPONENTIAL DEGRADATION
C    BB (SEC), TIME CONSTANT                       1.D7
C    PH (KW), HOUSEKEEPING POWER                   0.
C 14 EMPTY
```

```
C 15 IHI = 1. LOW THRUST ONLY                     1
C        = 2. HIGH/LOW
C        = 3. HIGH/LOW/HIGH
C        = 4. LOW/HIGH
C    DVI1 (M/S) TOTAL INITIAL DEL. V               0.
C    DVI2 (M/S), DEL. V FOR FINAL IMPULSE          0.
C 16 EMPTY
C 17 FLIM NORM LIMIT IN ITER ROUTINE               1.D-06
C 18 SGN = -1., INITIAL LAM 1 NEGATIVE
C        = +1., INITIAL LAM 1 POSITIVE
C
C
C
      SUBROUTINE INPUT
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      NAMELIST /UN/UTKM,UTS,UTD,UTMS2,UTKW
C
C
      COMMON /XMMM/ZLO(5), STEP(5), ZERF(6)
      COMMON /ELEM/ZPO(5), ZPF(5)
      COMMON /INT/ITF,IPR,IDIM,IDIM2,NIMAX
      COMMON /TRA/TFMAX,DT,UEB,EW(10)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,DUMMY,UTKW,UTMS2
      COMMON /T/TF,S,TO,TFMIN
      COMMON /A/A,AMU,PI
      COMMON /WF/WF(5)
      COMMON /J2/ AJ2
      COMMON /TC/NOP
      COMMON /POWER/PKW,CC,AMO,A0,A1,A2,A3,IPOW
      COMMON/HIGH/DVI1,DVI2,IHI,ITR
      COMMON /F/FLIM,KSTEP
      COMMON /SG/SGN
C
      DIMENSION W(5)
C
C INTEGER CONSTANTS
C
      IHI= 1
      IDIM=10
      IDIM2=5
      IDIM3=6
      IPR=0
      ITF=3
      NIMAX=20
      NOP= 1
      IPOW= 0
      KSTEP= 0
C
C REAL CONSTANTS
C
      AJ2= 0.D0
      AMU=1.0D0
      UEB= 1.0D+10
      DO 10 I=1,IDIM2
       EW(I+IDIM2)= 0.D0
```


C-2

```
 10     EW(I)= 1.0D0
        DO 12 I=1,IDIM2
 12     STEP(I)= 1.D-6
        DT2= 1.0D0
        GM=398601.2D0
        UTKM= 6378.16D0
        DTR= .017453292519943296D0
        PI= 3.1415926535897932D0
        TFMAX2= 190.0D0
        TFMIN2= 0.0D0
        T02=0.0D0
        BB= 1.D7
        DVI1= 0.D0
        DVI2= 0.D0
        PH= 0.D0
        FLIM= 1.D-6
C
C ALL READ STATEMENTS FOLLOW
C
        READ (5,1001) W
        READ (5,1001) ZLO
        READ (5,1001) WF
        SGN= WF(3)-W(3)
        IF (SGN.NE.0.D0) SGN= SGN/DABS(SGN)
        READ (5,1001) TF2
        READ (5,1001) PKW
        READ (5,1001) SPIM
        READ (5,1001) AMO
 20     READ (5,1002) IRDFLG
        IF ((IRDFLG.GT.18).OR.(IRDFLG.LT.1)) GO TO 200
 25     GO TO (150,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48),
       1  IRDFLG
C
C THESE VALUES ARE READ ONLY IF INDICATED BY IRDFLG
C
 32     READ (5,1002) IPR
        IF (IPR.LT.0) GO TO 210
        GO TO 20
 33     READ (5,1002) NIMAX
        IF ((NIMAX.LT.0).OR.(NIMAX.GT.50)) GO TO 220
        GO TO 20
 34     READ (5,1001) TFMAX2
        IF ((TFMAX2.LT.0.D0).OR.(TFMAX2.GT.1.    )) GO TO 220
        GO TO 20
 35     READ (5,1001) DT2
        IF ((DT2.LT.0.D0).OR.(DT2.GT.1.D3)) GO TO 220
        GO TO 20
 36     READ (5,1001) UEB
        IF (UEB.LT.0.D0) GO TO 220
        GO TO 20
 37     READ (5,1003) EW
        GO TO 20
 38     READ (5,1001) UTKM
        GO TO 20
 39     READ (5,1001) GM
        GO TO 20
 40     READ (5,1002) NOP
```

```
      IF ((NOP.LT.0).OR.(NOP.GT.1)) GO TO 220
      GO TO 20
 41   AJ2= 1.08270-3
      GO TO 20
 42   READ (5,1001) (STEP(I),I=1,5)
      READ (5,1002) KSTEP
      IF ((KSTEP.LT.0).OR.(KSTEP.GT.1)) GO TO 220
      GO TO 20
 43   READ (5,1002) IPOW
      READ (5,1001) BB,PH
C          BB  SHOULD BE IN SECONDS
      IF ((IPOW.LT.0).OR.(IPOW.GT.1)) GO TO 230
      GO TO 20
 44   GO TO 20
 45   READ (5,1002) IHI
      IF ((IHI.LT.0).OR.(IHI.GT.4)) GO TO 220
      NOP= 2
      READ (5,1001) DVI1,DVI2
      IF (IHI.LT.3) DVI2= 0.D0
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) DVI1= 0.D0
      GO TO 20
 46   GO TO 20
 47   READ (5,1003) FLIM
      IF (FLIM.LT.0.D0) GO TO 220
      GO TO 20
 48   READ (5,1001) SGN
      IF ((SGN.NE.1.D0).AND.(SGN.NE.-1.D0)) GO TO 220
      GO TO 20
C
C
C TIME VALUES ARE CHANGED FROM DAYS TO OTHER UNITS
C
 150  UTS= DSQRT(UTKM**3/GM)
      UTM=UTS/60.D0
      UTH=UTS/3600.D0
      UTD=UTH/24.D0
      TO= T02/UTD
      TF= TF2/UTD
      TFMIN= TFMIN2/UTD
      TFMAX= TFMAX2/UTD
      DT= DT2/UTD
      TO1= TO*UTS
      TF1= TF*UTS
      TFMIN1= TFMIN*UTS
      TFMAX1= TFMAX*UTS
      DT1= DT*UTS
C
C MORE CONVERSIONS
      UTMS2= UTKM*1.D3/(UTS**2)
      UTKW= UTMS2*UTKM/UTS
C
      C= SPIM/UTS
      CC= C*UTKM/UTS
      AA= 2.D0*PKW/(AMO*CC)
      AAA= 2.D0*(PKW-PH)/(AMO*CC)
      BB1= BB/UTS
      AO= AA/UTMS2
```

```
      A1= A0/C
      A0A= AAA/UTMS2
      IF (IPOW.EQ.1) A1= A1*BB1
      A2= 1.D0/BB1
      BB2= BB1*UTD
      A3= PH/PKW
C
C
C THE PRINTING OF ALL INITIAL VALUES FOLLOWS
C
      WRITE (6,2000)
      WRITE (6,2001)
      WRITE (6,2030)
      WRITE (6,2033)
      WRITE (6,2034)
      WRITE (6,2035) NOP
      WRITE (6,2043) DVI1,DVI2
C
      WRITE (6,2002)
      WRITE (6,2003)
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) GO TO 158
C       W(1) IS SEMIMAJOR AXIS
C       W(2) IS ECC. SET TO 0
C       W(3) IS INCLINATION
C       ZLO SHOULD BE Y1 (UP),Y2 (XK),Y3 (XJ),C,1.
C
      W(2)= 0.D0
      W(4)= 0.D0
      WRITE (6,2004) (W(I),I=1,4)
      ZPO(1)= W(1)/UTKM
      ZPO(2)= W(3)*DTR
      WRITE (6,2044) ZPO(1)
      ZLO(5)= 1.D0
      GO TO 162
 158  WRITE (6,2004) W
C
C   CHANGE FROM CLASSICAL O.E. TO EQUINOCTIAL O.E.
 159  DO 160 I=3,5
 160   W(I)= W(I)*DTR
      ZPO(1)= W(1)/UTKM
      ZPO(2)= W(2)*DSIN(W(5)+W(4))
      ZPO(3)= W(2)*DCOS(W(5)+W(4))
      ZPO(4)= DTAN(W(3)/2.0D0)*DSIN(W(4))
      ZPO(5)= DTAN(W(3)/2.0D0)*DCOS(W(4))
C
      WRITE (6,2005)
      WRITE (6,2004) ZPO
C
C WRITE FINAL CONDITIONS, CHANGE TO EQUINOCTIAL FINAL COND.
C
 162  DVI1= (DVI1/(UTMS2*UTS))*DSQRT(ZPO(1)/AMU)
      DVI2= DVI2/(UTMS2*UTS)
C
      WRITE (6,2006)
      ZPF(1)= WF(1)/UTKM
      GO TO (165,170), NOP
C
```

```
 165  WRITE (6,2003)
      WRITE (6,2004) WF
      DO 166 I=3,5
 166   WF(I)= WF(I)*DTR
      ZPF(2)= WF(2)*DSIN(WF(5)+WF(4))
      ZPF(3)= WF(2)*DCOS(WF(5)+WF(4))
      ZPF(4)= DTAN(WF(3)/2.0D0)*DSIN(WF(4))
      ZPF(5)= DTAN(WF(3)/2.0D0)*DCOS(WF(4))
      DO 167 I=3,5
 167   WF(I)=WF(I)/DTR
      WRITE (6,2005)
      WRITE (6,2004) ZPF
      GO TO 190
C
 170  ZPF(2)= WF(2)
      ZPF(3)= DABS(DTAN(WF(3)*DTR/2.D0))
      ZPF(4)= 0.D0
      ZPF(5)=0.D0
      WRITE (6,2031)
      WRITE (6,2004) (WF(I),I=1,3)
      WRITE (6,2032)
      WRITE (6,2004) (ZPF(I),I=1,3)
C
 190  IF (IHI.EQ.1) GO TO 194
      WRITE (6,2045)
      GO TO 196
 194  WRITE (6,2007)
 196  WRITE (6,2011) ZLO
      IF ((IHI.GT.1).AND.(IHI.LT.4)) WRITE (6,2047) SGN
      WRITE (6,2008)
      WRITE (6,2009) TF2,TF1,TF
      WRITE (6,2040) SPIM,CC
      WRITE (6,2038) AMO
      WRITE (6,2039) PKW
      IF (IPOW.EQ.1) WRITE (6,2041) BB,BB2
      WRITE (6,2012) AAA,ADA
      WRITE (6,2036) AJ2
      WRITE (6,2013)
      WRITE (6,2009) TO2,TO1,TO
      WRITE (6,2014)
      WRITE (6,2009) TFMIN2,TFMIN1,TFMIN
      WRITE (6,2015)
      WRITE (6,2009) TFMAX2,TFMAX1,TFMAX
      WRITE (6,2010) KSTEP
      WRITE (6,2011) STEP
      WRITE (6,2016)
      WRITE (6,2009) DT2,DT1,DT
      WRITE (6,2017) UEB
      WRITE (6,2018)
      WRITE (6,2019) EW
      WRITE (6,2020) IDIM
      WRITE (6,2022) NIMAX
      WRITE (6,2046) FLIM
      WRITE (6,2036) AJ2
      WRITE (6,2026) U1KM
      WRITE (6,2027) GM
      WRITE (6,UN)
```

```
      RETURN
  200 WRITE (6,2023) IRDFLG
      STOP
  210 WRITE (6,2024) IPR
      STOP
  220 WRITE (6,2025) IRDFLG
      STOP
  230 WRITE (6,2042) IPOW
      STOP
C
 1001 FORMAT (F25.15)
 1002 FORMAT (I2)
 1003 FORMAT (5D6.1)
 2000 FORMAT (1H1,22X,69H OPTIMAL TRAJECTORY PROGRAM FOR NEP AND HIGH TH
     1RUST SATELLITE RAISING)
 2001 FORMAT (1H0,40X,39H MINIMUM TIME                         NEP)
 2002 FORMAT (34H0 THE INITIAL ORBITAL ELEMENTS ARE)
 2003 FORMAT (1H0,10X,6HA (KM),19X,1HE,20X,7HI (DEG),10X,18HLON ASC NODE
     1 (DEG),6X,15HARG PERIG (DEG))
 2004 FORMAT (1P5D23.14)
 2005 FORMAT (1H0,6X,13HA (EARTH RAD),16X,1HH,22X,1HK,22X,1HP,22X,1HQ)
 2006 FORMAT (40H0 THE DESIRED FINAL ORBITAL ELEMENTS ARE)
 2007 FORMAT (32H0 INITIAL GUESSED PARAMETERS ARE)
 2008 FORMAT (21H0 FINAL TIME ESTIMATE)
 2009 FORMAT (1H ,10X,1PD22.15,7H DAYS =,1PD22.15,10H SECONDS =,1PD22.15
     1,6H UNITS)
 2010 FORMAT (50H0 STEP SIZE FOR NUMERICAL DIFFERENTIATION, KSTEP =,I2)
 2011 FORMAT (1P5D23.14)
 2012 FORMAT (22H0 ACCELERATION LEVEL =,1PD25.15,12H M/SEC/SEC =,1PD25.1
     15,4H G'S)
 2013 FORMAT (17H0 INITIAL TIME IS)
 2014 FORMAT (10H0 TFMIN IS)
 2015 FORMAT (10H0 TFMAX IS)
 2016 FORMAT (27H0 TIME STEP FOR INTEGRATION)
 2017 FORMAT (36H0 UPPER ERROR BOUND ON INTEGRATION =,1PD20.10)
 2018 FORMAT (35H0 ERROR WEIGHTS FOR INTEGRATION ARE)
 2019 FORMAT (1P10D12.4)
 2020 FORMAT (13H0 DIMENSION =,I5)
 2022 FORMAT (31H0 MAXIMUM NUMBER OF ITERATION =,I5)
 2023 FORMAT (44H0 IRDFLG SHOULD BE BETWEEN 1 AND 15, IT IS =,I5)
 2024 FORMAT (28H0 IPR SHOULD BE < 0, IT IS =,I5)
 2025 FORMAT (27H0 BAD INPUT DATA, IRDFLG = ,I3)
 2026 FORMAT (17H0 1 EARTH RADIUS=,F25.12,3H KM)
 2027 FORMAT (11H0 MU (GM) =,F25.10,13H KM**3/SEC**2)
 2030 FORMAT (25H0 FINAL CONDITION OPTIONS)
 2033 FORMAT (43H  1. ALL 5 FINAL ORBITAL ELEMENTS SPECIFIED)
 2034 FORMAT (51H  2. A,E,I SPECIFIED, LON ASC NODE AND ARG PER FREE)
 2035 FORMAT (24H  FOR THIS RUN, OPTION =,I4)
 2031 FORMAT (1H0,10X,6HA (KM),19X,1HE,20X,7HI (DEG))
 2032 FORMAT (1H0,6X,13HA (EARTH RAD),9X,15HSQRT(H**2+K**2),8X,
     1  15HSQRT(P**2+Q**2))
 2036 FORMAT (6H0 J2 =,1PD15.7)
 2038 FORMAT (21H0 INITIAL MASS (KG) =, F18.11)
 2039 FORMAT (22H0 INITIAL POWER (KW) =, 1PD23.14)
 2040 FORMAT (12H0 SPEC IMP =,1PD23.14,15H SEC, EXH VEL =,1PD23.14,
     1     5H KM/S)
 2041 FORMAT (33H0 POWER IMIGRATION TIME CONSTANT =,1PD23.15,6H SEC =,
```

```
    1     F20.10,4HDAYS)
 2042 FORMAT (30HO IPOW SHOULD BE 0 OR 1, IPOW=, I3)
 2043 FORMAT (41HO TOTAL DELV FOR INITIAL IMPULSES (M/S) =, F20.12,5X,
    1      21HFINAL IMPULSE (M/S) =, F20.12)
 2044 FORMAT (27HO INITIAL A (EARTH RADII) =,1PD25.14)
 2045 FORMAT (42HO INITIAL GUESS FOR HIGH THRUST, S.F., ONE)
 2046 FORMAT (23HO NORM LIMIT IN ITER = ,1PD12.5)
 2047 FORMAT (29HO SIGN OF INITIAL LAMBDA 1 IS, F6.1)
      END
```

```
C OUTPC/OUTPCN
C
C NEP AND HIGH THRUST
C THIS SUBPROGRAM WRITES THE VALUES FOR THE FINAL CONVERGED
C    TRAJECTORY.  IT IS CALLED BY THE MAIN PROGRAM CONTL.
C    TO BE USED WITH 6 DIM. ZERF.
C
C
C
      SUBROUTINE OUTPC
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      COMMON /XMMM/ZLO(5),STEP(5),ZERF(6)
      COMMON /Z/ZF(10),DZ(10)
      COMMON /T/TF,S,TO,TFMIN
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,DUMMY,UTKW,UTMS2
      COMMON /ELEM/ZPO(5),ZPF(5)
      COMMON /WF/WF(5)
      COMMON /A/A,AMU,PI
      COMMON /TC/NOP
      COMMON /POWER/PKW,CC,AMO,A0,A1,A2,A3,IPOW
C
      DIMENSION DELZF(5),DELWF(5),WFC(5)
C
      WFC(1)= ZF(1)*UTKM
      WFC(2)= 0.0D0
      DUMMY= ZF(2)**2 + ZF(3)**2
      IF (DUMMY.GT.1.0D-40) WFC(2)=DSQRT(DUMMY)
      WFC(3)= 0.0D0
      DUMMY= ZF(4)**2 + ZF(5)**2
      IF (DUMMY.GT.1.0D-40) WFC(3)= 2.0D0*DATAN(DSQRT(DUMMY))/DTR
      WFC(4)=0.D0
      IF ((DABS(ZF(4)).GT.1.D-8).AND.(DABS(ZF(5)).GT.1.D-8))
     1 WFC(4)= DATAN2(ZF(4),ZF(5))/DTR
      WFC(5)= 0.D0
      IF ((DABS(ZF(2)).GT.1.D-8).AND.(DABS(ZF(3)).GT.1.D-8))
     1   WFC(5)=DATAN2(ZF(2),ZF(3))/DTR
      WFC(5)=WFC(5)-WFC(4)
      DO 10 I=1,5
       DELWF(I)= WFC(I) - WF(I)
 10    DELZF(I)= ZF(I) - ZPF(I)
      TF2= TF*UTD
      TF1= TF*UTS
      IF (IPOW.EQ.1) B1= DEXP(-A2*TF)
      IF (IPOW.EQ.1) RMASS= 1.D0+A1*(B1-1.D0+A3*A2*TF)
      IF (IPOW.EQ.0) RMASS= 1.D0-A1*TF
      AMASS= AMO*RMASS
      DELV= -CC*DLOG(RMASS)
      POW= PKW
      IF (IPOW.EQ.1) POW= POW*(B1-A3)
      RPOW= POW/PKW
C
      WRITE (6,3000)
      WRITE (6,3001)
      WRITE (6,3002) WFC
      WRITE (6,3003)
```

```
      WRITE (6,3002) (ZF(I),I=1,5)
      WRITE (6,3004)
      WRITE (6,3001)
      IF (NOP.EQ.1) WRITE (6,3002) DELWF
      IF (NOP.EQ.2) WRITE (6,3002) (DELWF(I),I=1,3)
      GO TO (20, 30), NOP
C
 20   WRITE (6,3003)
      WRITE (6,3002) DELZF
      GO TO 100
C
 30   DELZF(2)= DSQRT(ZF(2)**2+ZF(3)**2) -ZPF(2)
      DELZF(3)= DSQRT(ZF(4)**2+ZF(5)**2) -ZPF(3)
      WRITE (6,3011)
      WRITE (6,3002) (DELZF(I),I=1,3)
C
 100  WRITE (6,3006)
      WRITE (6,3002) ZLO
      WRITE (6,3008)
      WRITE (6,3009) TF2,TF1,TF
      WRITE (6,3013) AMASS,RMASS
      WRITE (6,3014) POW,RPOW
      WRITE (6,3010) DELV
      RETURN
 3000 FORMAT (35H0 ACTUAL FINAL ORBITAL ELEMENTS ARE)
 3001 FORMAT (1H0,10X,6HA (KM),18X,1HE,20X,7HI (DEG),10X,18HLON ASC NODE
     1 (DEG),6X,15HARG PERIG (DEG))
 3002 FORMAT (1P5D23.15)
 3003 FORMAT (1H0,5X,13HA (EARTH RAD),16X,1HH,22X,1HK,22X,1HP,22X,1HQ)
 3004 FORMAT (51H0 THE ERROR IN THE FINAL O.E. IS (ACTUAL - DESIRED))
C3005 FORMAT (60H0 CLASSICAL O.E. MAY HAVE DISCREPANCY OF MULTIPLES OF 90
C    1O DEG)
 3006 FORMAT (46H0 THE CONVERGED INITIAL GUESSED PARAMETERS ARE)
 3008 FORMAT (29H0 THE MINIMIZED FINAL TIME IS)
 3009 FORMAT (1H ,10X,1PD22.15,7H DAYS =,D22.15,10H SECONDS =,1PD22.15,6
     1H UNITS)
 3010 FORMAT (18H0 LOW THRUST DELV=,1PD25.14,7H KM/SEC)
 3011 FORMAT (1H0,5X,13HA (EARTH RAD),9X,15HSQRT(H**2+K**2),8X,
     1      15HSQRT(P**2+Q**2))
 3013 FORMAT (14H0 FINAL MASS =,F22.15,31H KG, FRACTION OF INITIAL MASS
     1=,F22.15)
 3014 FORMAT (15H0 FINAL POWER =, F22.15,32H KW, FRACTION OF INITIAL POW
     1ER =,F22.15)
      END
```

```
C ITER MODNRN
C
C
C 6X6 VERSION
C
      SUBROUTINE ITER(KOUNT,NI,FUNCT,PRTN)
C
      IMPLICIT REAL*8(A-H,O-$)
C
C
C
C X     VALUES OF THE INDEPENDENT VARIABLES(INITIAL,CURRENT,FINAL)
C XS    STEP SIZE TO PERTURB X:S TO COMPUTE PARTIAL DERIVATIVES
C Y     VALUES OF THE DEPENDENT VARIABLES(CURRENT,FINAL)
      COMMON/XMMM/X(5),XS(5),Y(6)
      COMMON /INT/ITF,IPR,IDIM,IDIM2,MAXNOI
      COMMON /T/TF,S,TC,TFMIN
      COMMON /DY/ DYDT(6)
      COMMON /HIGH/DVI1,DVI2,IHI,ITR
      COMMON /F/FLIM,KSTEP
C
      DIMENSION YNOM(6),XN(5),P(6,6),COEF(6),DYDTN(6)
      N=5
      M=6
      INORM=ITF
      18=1
      ICONS=1
      ISW=0
      N  1
      KOUNT=0
      ITR=0
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      FO=0.D0
      DO 15 I=1,M
 15   FO=FO+Y(I)**2
 9    DO 16 I=1,N
      DYDTN(I)= DYDT(I)
      XN(I)=X(I)
   16 YNOM(I)=Y(I)
      YNOM(M)= Y(M)
      DYDTN(M)= DYDT(M)
      TFN=TF
 10   CALL PRTN(KOUNT,NI)
      WRITE(6,1011)FO
      IF(FO.LE.FLIM)GO TO 90
      IF (NI.GT.MAXNOI) GO TO 80
      IF(ISW.NE.0)GO TO 27
C COMPUTE NUMERICAL PARTIAL DERIVATIVES
      DO 17 I=1,M
 17    P(I,M)= DYDT(I)
      WRITE (6,1013)
      DO 25 J=1,N
      ITR=J
      TEMP=X(J)
      STEP=XS(J)*DABS(X(J))
```

```
      IF ((DABS(X(J)).LT.1.D-10).OR.(KSTEP.EQ.1))STEP=XS(J)
C     IF (DABS(X(J)).LT.1.D-10) WRITE (6,1014)
      X(J)=X(J)+STEP
      CALL FUNCT
      WRITE(6,1000)X(J)
      WRITE(6,1001)(Y(I),I=1,M)
      DO 20 I=1,M
   20 P(I,J)=(Y(I)-YNOM(I))/STEP
   25 X(J)=TEMP
      KOUNT=KOUNT+N
   27 WRITE(6,1002)
      DO 30 I=1,M
      WRITE(6,1001)(P(I,J),J=1,M)
   30 CONTINUE
      DO 35 I=1,M
   35 COEF(I)=-YNOM(I)
      CALL DCROUT(P,COEF,DET,0.D0,M,1,IND)
      IF(IND.NE.0)GO TO 65
      WRITE (6,1015) DET
      DO 40 I= 1,M
 40   IF (DABS(COEF(I)).LT.1.D-10) COEF(I)= 0.D0
      WRITE(6,1003)(COEF(I),I=1,N)
      SN= COEF(M)
      WRITE (6,1012) SN
      DO 50 J=1,N
   50 X(J)=XN(J)+COEF(J)
      TF=TFN+SN
      IHALV=0
 51   IF (INORM.EQ.1) ITF=1
      ITR=0
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      F1=0.D0
      DO 52 I=1,M
   52 F1=F1+Y(I)**2
      WRITE(6,1010)F1
      IF(F1.LT.FO)GO TO 55
      WRITE(6,1008)
      IF(IHALV.EQ.8)GO TO 95
      IHALV=IHALV+1
      DO 53 J=1,N
      COEF(J)=COEF(J)/2.D0
      WRITE(6,1000)COEF(J)
   53 X(J)=XN(J)+COEF(J)
      SN=SN/2.0D0
      WRITE (6,1012) SN
      TF=TFN+SN
      GO TO 51
   55 IF(NI-MAXIT)70,70,90
   70 NI=NI+1
      ICONS=NI
      FO=F1
      SUMDX=0.D0
      DO 76 J=1,M
   76 SUMDX=COEF(J)**2+SUMDX
      DO 77 I=1,M
```

```
      DO 77 J=1,M
      P(I,J)=P(I,J)+(Y(I)*COFF(J))/SUMDX
   77 CONTINUE
      ISW=1
      GO TO 9
   80 NI=9999
      WRITE(6,1006)
C     PRINT 1006
      RETURN
   85 NI=9999
      WRITE(6,1007)
C     PRINT 1007
      RETURN
   90 WRITE(6,1005)FO
C  90 PRINT 1005,FO
      RETURN
   95 IF(NI.EQ.1.OR.IB.EQ.10.OR.ICONS.NE.NI)GO TO 100
      ICONS=ICONS+1
      IB=IB+1
      DO 96 J= 1,N
      DYDT(J)= DYDTN(J)
      X(J)= XN(J)
   96 Y(J)= YNOM(J)
      Y(M)= YNOM(M)
      TF= TFN
      DYDT(M)= DYDTN(M)
      ISW=0
C     PRINT 1004
      WRITE(6,1004)
      GO TO 10
  100 NI=9999
      WRITE(6,1009)
C     PRINT 1009
      RETURN
 1000 FORMAT(/1X,1PD23.15)
 1001 FORMAT(1X,1P5D23.15)
 1002 FORMAT(21HOPARTIAL DERIV MATRIX)
 1003 FORMAT(11HODELX:S ARE/(1X,1PD23.15))
 1004 FORMAT(35HOFORM NEW PARTIAL DERIVATIVE MATRIX)
 1005 FORMAT(4HOFO=,1PD22.15,23HCASE CONVERGED...FERTIG)
 1006 FORMAT(38HOEXCEEDED MAXIMUM NUMBER OF ITERATIONS)
 1007 FORMAT(16HOMATRIX SINGULAR)
 1008 FORMAT(11HODELX:S ARE)
 1009 FORMAT(19HOMETHOD CANNOT WORK)
 1010 FORMAT(4HOF1=,1PD23.15)
 1011 FORMAT(4HOFO=,1PD23.15)
 1012 FORMAT (10HO DEL TF =,1PD23.15)
 1013 FORMAT (40HO X(I)+DX(I) FOLLOWED BY CORRESPONDING Y)
 1014 FORMAT (24HO X(I)=0. SO DX(I)=XS(I))
 1015 FORMAT (15HO DETERMINENT =,1PD23.15)
      END
```

```
C PRTN/PRTNN
C
C THIS PROGRAM IS CALLED BY THE ITERATOR AND PRINTS
C USED WITH 6 DIM. Y.
C
C
      SUBROUTINE PRTN(KOUNT,NOI)
C
      IMPLICIT REAL*8(A-H,O-$)
      COMMON /XMMM/X(5),XS(5),Y(6)
      COMMON /T/TF,S,TO,TFMIN
C
      N=5
      M=6
      WRITE (6,1000)
      WRITE (6,1001) NOI,KOUNT
      WRITE (6,1002)
      WRITE (6,1003) (X(J),J=1,N)
      WRITE (6,1004)
      WRITE (6,1003) (Y(J),J=1,M)
      WRITE (6,1005) TF
 1000 FORMAT (24H0 ITER NO.    TRAJ. CALLS)
 1001 FORMAT (1H0,I6,5X,I6)
 1002 FORMAT (2H0X)
 1003 FORMAT (1X,1P5D23.15)
 1004 FORMAT (2H0Y)
 1005 FORMAT (5H0 TF=,1PD26.16)
      RETURN
      END
```

```
C   DCROUT/DCROUT3  6 DIM.
      SUBROUTINE DCROUT(AA,R,D,EPS,NI,M,IND)                            0000000
      DOUBLE PRECISION A,R,D,EPS,T,S,P,DT,AA                            0000000
      DIMENSION A(6,6),R(6,1),AA(6,6)                                   0000000
    5 IND=0                                                             36110050
      N=NI                                                              36110060
      DO 6 I=1,N                                                        0000000
      DO 6 J=1,N                                                        0000000
    6 A(I,J)=AA(I,J)                                                    0000000
      IF(M)10,25,25                                                     36110080
   10 M=N                                                               36110090
      DO 20 I=1,N                                                       36110100
      DO 15 J=1,N                                                       36110110
   15 R(I,J)=0.D0                                                       0000000
   20 R(I,I)=1.D0                                                       0000000
   25 IC=0                                                              36110140
      II=0                                                              36110150
      T=DABS(A(1,1)   )                                                 36110160
      DO 35 I=2,N                                                       36110170
      IF(T-DABS(A(I,1)))30,35,35                                        36110180
   30 II=I                                                              36110190
      T=DABS(A(I,1))                                                    36110020
   35 CONTINUE                                                          36110210
      IF(II)40,65,40                                                    36110220
   40 IC=IC+1                                                           36110230
      IF(M)45,55,45                                                     36110240
   45 DO 50 J=1,M                                                       36110250
      S=R(1,J)                                                          36110260
      R(1,J)=R(II,J)                                                    36110270
   50 R(II,J)=S                                                         36110280
   55 DO 60 J=1,N                                                       36110290
      S=A(1,J)                                                          36110300
      A(1,J)=A(II,J)                                                    36110310
   60 A(II,J)=S                                                         36110320
   65 P=A(1,1)                                                          36110330
      IF(DABS(P)-EPS)70,70,75                                           36110340
   70 IND=1                                                             36110350
      D=0.D0                                                            0000000
      GO TO 200                                                         36110370
   75 DO 80 J=2,N                                                       36110380
   80 A(1,J)=A(1,J)/P                                                   36110390
      IF(M)85,95,85                                                     36110400
   85 DO 90 J=1,M                                                       36110410
   90 R(1,J)=R(1,J)/P                                                   36110420
   95 DO 170 K=2,N                                                      36110430
      KM=K-1                                                            36110440
      T=-1.D0                                                           0000000
      DO 105 I=K,N                                                      36110460
      DO 98 J=1,KM                                                      36110470
   98 A(I,K)=A(I,K)-A(I,J)*A(J,K)                                       36110480
      IF(T-DABS(A(I,K)))100,105,105                                     36110490
  100 T=DABS(A(I,K))                                                    36110500
      II=I                                                              36110510
  105 CONTINUE                                                          36110520
      IF(II-K)110,135,110                                               36110530
  110 IC=IC+1                                                           36110540
      IF(M)115,125,115                                                  36110550
```

```
115 DO 120 J=1,M
    S=R(K,J)
    R(K,J)=R(II,J)
120 R(II,J)=S
125 DO 130 J=1,N
    S=A(K,J)
    A(K,J)=A(II,J)
130 A(II,J)=S
135 DT=A(K,K)
    IF(DABS(DT)-EPS)70,70,140
140 P=P*DT
    IF(K-N)145,155,145
145 KP=K+1
    DO 150 J=KP,N
    DO 148 I=1,KM
148 A(K,J)=A(K,J)-A(K,I)*A(I,J)
150 A(K,J)=A(K,J)/DT
155 IF(M)160,170,160
160 DO 165 J=1,M
    DO 162 I=1,KM
162 R(K,J)=R(K,J)-A(K,I)*R(I,J)
165 R(K,J)=R(K,J)/DT
170 CONTINUE
    IF(MOD(IC,2))175,180,175
175 P=-P
180 D=P
    IF(M)185,200,185
185 II=N
    DO 190 K=2,N
    KP=II
    II=II-1
    DO 190 J=1,M
    DO 190 I=KP,N
190 R(II,J)=R(II,J)-A(II,I)*R(I,J)
200 RETURN
    END
```

```
C TRAJ/TRAJRKN
C
C NEP AND HIGH THRUST.
C THIS ROUTINE SETS UP THE INPUT TO THE INTEGRATOR WHICH
C    EXTRAPOLATES THE TRAJECTORY FROM INITIAL TIME TO
C    FINAL TIME.  IT ALSO EVALUATES THE CHANGE IN TF AND
C    THE ERROR IN THE FINAL CONDITIONS.
C THIS PROGRAM IS CALLED BY ITER OR BY CONTROL
C MIN J, MAX H.
C 6 DIM. ZERF. T.C. OPTIONS.
C NOP=1--ALL 5 FINAL O.E. FIXED, =2--A,E,I ONLY FIXED.
C DYDT USED ONLY IF ITR= 0
C IHI= 1 NO HIGH THRUST
C    = 2 2 INITIAL IMPULSES
C    = 3 2 INITIAL AND ONE FINAL IMPULSE
C    = 4 FINAL IMPULSE
C WHEN IHI=2 OR 3, ZLO(5) AND ZERF(5) ARE DUMMIES.
C
C
      SUBROUTINE TRAJ
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      COMMON /XMMM/ZLO(5), DUMMY(5), ZERF(6)
      COMMON /TRA/TFMAX, DTO, UEB, EW(10)
      COMMON /Z/Z(10), DERZ(10)
      COMMON /INT/ITF, IPR, IDIM, IDIM2, NIMAX
      COMMON /T/TF, SD, TO, TFMIN
      COMMON /ELEM/ZPO(5), ZPF(5)
      COMMON /DY/DYDT(6)
      COMMON /TC/NOP
      COMMON /POWER/PKW,CC,AMO,A0,A1,A2,A3,IPOW
      COMMON /A/A,AMU,PI
      COMMON /HIGH/DVI1,DVI2,IHI,ITR
      COMMON /F/FLIM,KSTEP
C
      EXTERNAL FUNCT, OUTP
      DIMENSION PRMT(5), AUX(8,10), DP(5), ZS(6)
C
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) GO TO 9
      IF (ITR.NE.1) GO TO 4
      DO 3 I=1,6
 3    ZS(I)= ZERF(I)
 4    IF (ITR.EQ.4) GO TO 70
      IF (ITR.EQ.5) GO TO 80
C
C HIGH THRUST
C
      UP= (PI/2.D0)*ZLO(1)/DSQRT(1.D0+ZLO(1)**2)
      CUP= DCOS(UP)
      XK= CUP*(.75D0+.25D0*ZLO(2)/DSQRT(1.D0+ZLO(2)**2))
      DUM= (1.D0+CUP*XK)*DSQRT((CUP-XK)/(CUP+XK))
      XJ= DUM*ZLO(3)/DSQRT(1.D0+ZLO(3)**2)
      JM= 2
      CALL MAINE(0.D0,0.D0,XK,UP,XJ,1.D0,1,JM,DP,DVI1)
      CALL OUTHI(JM,PI,ZPO(1),ZPO(2),0.D0,IPR,Z,IDIM2)
      DO 5 I= 1,IDIM2
```

```
 5     Z(I+IDIM2)= ZLO(4)*Z(I+IDIM2)*1.D4
C
C NEP
C
 9     PRMT(1)= TO
       PRMT(2)= TF
       PRMT(3)= DTO
       PRMT(4)= UEB
C
C Z IS A VECTOR OF STATE AND COSTATE
C
       IF ((IHI.EQ.2).OR.(IHI.EQ.3)) GO TO 15
       DO 10 I=1,IDIM2
        Z(I)=ZPO(I)
 10     Z(I+IDIM2)= ZLO(I)
C
C EW ARE ERROR WEIGHTS--INPUT TO THE INTEGRATOR
C
 15    DO 20 I=1,IDIM
 20     DERZ(I)=EW(I)
C
C FOR ITF LT 3 USE NORM CUTOFF CONDITION
C
       IF (ITF.LT.3) TF=TFMAX
C
C CALL THE R-K INTEGRATOR
C
       CALL DRKGS(PRMT,Z,DERZ,IDIM,IHLF,FUNCT,OUTP,AUX)
       IF (IHLF.GT.10) GO TO 100
C
C Z IS NOW THE FINAL O.E..
C      ZERF THE ERROR IN THE FINAL CONDITIONS
C
       H=0.D0
       DO 30 I=1,IDIM2
 30     H= H + Z(I+5)*DERZ(I)
       ZERF(6)= H -1.D0
       IF (IHI.LT.3) GO TO 35
C
C FINAL IMPULSE
C
       PSI= (Z(3)*Z(7)-Z(2)*Z(8))*1.D-3
       PSIDOT= DERZ(3)*Z(7)+Z(3)*DERZ(7)-Z(2)*DERZ(8)-DERZ(2)*Z(8)
       PSIDOT= PSIDOT*1.D-3
       CALL IMPLS(DVI2,IPR,ITR,Z,DERZ,IDIM2)
C
C FINAL CONDITION OPTION BRANCH
C
 35    GO TO (40,50), NOP
C
 40    DO 45 I=1,IDIM2
        ZERF(I)= Z(I) -ZPF(I)
 45     DYDT(I)= DERZ(I)
       GO TO 55
C
 50    ZERF(1)= Z(1) - ZPF(1)
       DUM1= DSQRT(Z(2)**2 + Z(3)**2)
```

```
      ZERF(2)= DUM1 - ZPF(2)
      DUM2= DSQRT(Z(4)**2 + Z(5)**2)
      ZERF(3)= DUM2 - ZPF(3)
      ZERF(4)= (Z(3)*Z(7) - Z(2)*Z(8))*1.D-3
      ZERF(5)= (Z(5)*Z(9) - Z(4)*Z(10))*1.D-3
      DYDT(1)= DERZ(1)
      DYDT(2)= 0.D0
      DYDT(3)= 0.D0
      IF (DUM1.GT.1.D-40) DYDT(2)= (Z(2)*DERZ(2) + Z(3)*DERZ(3))/DUM1
      IF (DUM2.GT.1.D-40) DYDT(3)= (Z(4)*DERZ(4) + Z(5)*DERZ(5))/DUM2
      DYDT(4)= DERZ(3)*Z(7)+Z(3)*DERZ(7)-DERZ(2)*Z(8)-Z(2)*DERZ(8)
      DYDT(4)=DYDT(4)*1.D-3
      DYDT(5)= DERZ(5)*Z(9)+Z(5)*DERZ(9)-DERZ(4)*Z(10)-Z(4)*DERZ(10)
      DYDT(5)=DYDT(5)*1.D-3
      IF (IHI.EQ.1) GO TO 55
      IF (IHI.EQ.4) GO TO 52
C
C DUMMY
C
      ZERF(5)= ZLO(5)-1.D0
      DYDT(5)= 0.D0
      IF (IHI.LT.3) GO TO 53
C
 52   ZERF(4)= PSI
      DYDT(4)= PSIDOT
 53   IF (ITR.GT.0)  RETURN
C
      DO 54 I= 1,6
 54    ZS(I)= ZERF(I)
C
 55   IF (IPOW.EQ .) GO TO 60
      A4= 1.D0-  *TF
      DYDT(6)= A1*H/A4
      RETURN
C
 60   B1= DEXP(-A2*TF)
      A4= 1.D0+A1*(B1-1.D0+A2*A3*TF)
      DYDT(6)= A2*H*(-B1/(B1-A3)+A1*(B1-A3)/A4)
      RETURN
C
C
C
 70   DO 75 I= 1,3
 75    ZERF(I)= ZS(I)
      DUM= DUMMY(4)
      IF (K..EP.EQ.0) DUM= DUM*ZLO(4)
      ZERF(4)= (1.D0+DUM)*ZS(4)
      ZERF(5)= ZS(5)
      ZERF(6)= (1.D0+DUM)*(ZS(6)+1.D0)-1.D0
      RETURN
C
      DO 85 I= 1,6
 85    ZERF(I)= ZS(I)
      ZERF(5)= ZS(5)+DUMMY(5)
      RETURN
C
 100  IF (IHLF.EQ.11) WRITE (6,1000)
```

```
      IF (IHLF.EQ.12) WRITE (6,1001)
      IF (IHLF.EQ.13) WRITE (6,1002)
      STOP
C
 1000 FORMAT (68H0 THE NUMBER OF BISECTIONS OF THE ORIGINAL INCREMENT HA
     1S EXCEEDED 10)
 1001 FORMAT (27H0 INITIAL INCREMENT IS ZERO)
 1002 FORMAT (54H0 INITIAL INCREMENT HAS WRONG SIGN OR BOUNDS ARE WRONG)
      END
```

```
C OUTP/OUTPN
C
C NEP
C THIS IS THE OUTP PROGRAM FOR THE
C    INTEGRATOR--FIXED GUESS TIME ONLY (ITF=3)
C
C
C
      SUBROUTINE OUTP(T,Z,DERZ,IHLF,IDIM,PRMT)
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,DUMMY,UTKW,UTMS2
      COMMON /INT/ITF,IPR,ID,IDIM2,NIMAX
      COMMON /POWER/PKW,CC,AMO,A0,A1,A2,A3,IPOW
      COMMON /A/A,AMU,PI
C
      DIMENSION PRMT(5), Z(10), DERZ(10), W(5)
C
      IF (ITF.NE.3) GO TO 10
      IF (IPR.EQ.0) RETURN
      IF (T.EQ.PRMT(1)) N=0
      IF (T.EQ.PRMT(1)) M=0
      N=N+1
      IF ((T.LT.(.9999999999D0*DFLOAT(M)*(PRMT(2)-PRMT(1))/DFLOAT(IPR)))
     1     .AND.(IHLF.LT.11).AND.(T.LT.(.9999999999D0*PRMT(2)))) RETURN
      M=M+1
      TS=UTS*T
      TM=UTM*T
      TH=UTH*T
      TD=UTD*T
      IF (IPOW.EQ.1) B1= DEXP(-A2*T)
      IF (IPOW.EQ.1) RMASS= 1.D0+A1*(B1-1.D0+A3*A2*T)
      IF (IPOW.EQ.0) RMASS= 1.D0-A1*T
      DV= -CC*DLOG(RMASS)
      AMASS= AMO*RMASS
      POW= PKW
      IF (IPOW.EQ.1) POW= (B1-A3)*POW
      THR= 2.D0*POW/CC
      AA= THR/AMASS
      H= 0.D0
      DO 5 I=1,5
 5    H= H + Z(I+IDIM2)*DERZ(I)
      W(1)= Z(1)*UTKM
      W(2)=0.D0
      DUMMY=Z(2)**2+Z(3)**2
      IF (DUMMY.GT.1.D-40) W(2)=DSQRT(DUMMY)
      W(3)=0.D0
      DUMMY=Z(4)**2 + Z(5)**2
      IF (DUMMY.GT.1.D-40) W(3)= 2.D0*DATAN(DSQRT(DUMMY))/DTR
      W(4)=0.D0
      IF ((Z(4).NE.0.D0).OR.(Z(5).NE.0.D0)) W(4)= DATAN2(Z(4),Z(5))/DTR
      W(5)= 0.D0
      IF ((Z(2).NE.0.D0).OR.(Z(3).NE.0.D0)) W(5)= DATAN2(Z(2),Z(3))/DTR
      W(5)=W(5)-W(4)
      IDIM3=IDIM2+1
C
C
```

```
      WRITE (6,1001)
      WRITE (6,1002)
      WRITE (6,1003) T, TS, TH, TD, DV, N
      WRITE (6,1004)
      WRITE (6,1005) (Z(I),I=1,IDIM2)
      WRITE (6,1005) W
      WRITE (6, 1013)
      WRITE (6,1012) AMASS,POW,THR,AA
      WRITE (6,1006)
      WRITE (6,1005) (Z(I),I=IDIM3,IDIM)
      WRITE (6,1007)
      WRITE (6,1005) (DERZ(I),I=1,IDIM2)
      WRITE (6,1008)
      WRITE (6,1005) (DERZ(I),I=IDIM3,IDIM)
      WRITE (6,1009)
      PER=2.D0*PI*DSQRT(Z(1)**3/AMU)*UTH
      AP= W(1)*(1.D0+W(2))
      PE= W(1)*(1.D0-W(2))
      WRITE (6,1010) H, PER,PE,AP,IHLF
C
      RETURN
 10   WRITE (6,1000)
      STOP
C
 1000 FORMAT (56H0 ITF MUST EQUAL 3--I.E. NEED FIXED ESTIMATED FINAL TIM
     1E)
 1001 FORMAT (70H0 *****************************************************
     1****************)
 1002 FORMAT (5H TIME,10X,10HTIME UNITS,15X,7HSECONDS,9X
     1   ,5HHOURS,11X,4HDAYS,9X,8HDV (K/S),10X,1HN)
 1003 FORMAT (1P2D25.7,1P3D15.7,I9//)
 1004 FORMAT (60H0 THE ORBITAL ELEMENTS ARE (EQ. FOLLOWED BY CL. IN KM &
     1 DEG))
 1005 FORMAT (1P5D22.12//)
 1006 FORMAT (16H0 THE COSTATE IS)
 1007 FORMAT (45H0 THE DERIVITIVES OF THE ORBITAL ELEMENTS ARE)
 1008 FORMAT (34H0 THE DERIVITIVE OF THE COSTATE IS)
 1009 FORMAT (1H0,7X,11HHAMILTONIAN,8X,12HPERIOD (HRS),7X,
     1    12HPERIGEE (KM),9X,11HAPOGEE (KM),5X,14HDIV. TIME STEP)
 1010 FORMAT (2F20.12,1P2D20.10,I9//)
 1012 FORMAT (1P4D22.12)
 1013 FORMAT (9X,9HMASS (KG),12X,10HPOWER (KW),12X,10HTHRUST (N),
     1    8X,19HTHRUST ACC (M/S**2))
        END
```

```
C FUNCT/FUNCTN
C
C NEP
C THIS SUBROUTINE IS AN INTERFACE BETWEEN THE INTEGRATOR ROUTINE
C     AND THE QUADRATURE ROUTINE.
C THIS ROUTINE ADDS THE EFFECT OF OBLATENESS (J2) TO THE DERIV.
C OBLATE CALCULATES THE EFFECT OF J2. RETURNED AS DZJ2.
C Z IS A VECTOR OF THE AVERAGED STATE AND COSTATE
C DERZ IS THE AVERAGED DERIVITIVE OF Z
C -PI AND PI AND THE LIMITS OF INTEGRATION IN THE QUADRATURE
C 10 IS THE DIMENSION OF Z
C
C
C
      SUBROUTINE FUNCT(X,Z,DERZ)
C
      IMPLICIT REAL*8(A-H,O-$)
      COMMON /A/A,AMU,PI
      COMMON /J2/AJ2
      COMMON /POWER/PKW,CC,AMO,A0,A1,A2,A3,IPOW
      DIMENSION Z(10), DERZ(10), G(10), H(10), DZJ2(10)
      EXTERNAL FCT
      B1= DEXP(-A2*X)
C
      IF (IPOW.EQ.0) A= A0/(1.D0-A1*X)
      IF ((IPOW.EQ.1).AND.(A3.GE.B1)) GO TO 20
      IF (IPOW.EQ.1) A= A0*(B1-A3)/(1.D0+A1*(B1-1.D0+A3*A2*X))
      DO 2 I=1,10
 2     DERZ(I)= 0.D0
C
      CALL QUAD(-PI,PI,FCT,DERZ,Z,G,H,10)
      IF (AJ2.LE.0.D0) RETURN
 5    CALL OBLATE(AJ2,Z,DZJ2,1)
      DO 10 I=1,10
 10    DERZ(I)= DERZ(I) + DZJ2(I)
      RETURN
 20   WRITE (6,1000)
      STOP
 1000 FORMAT (' POWER IS NEGATIVE')
      END
```

```
C CONTL
C
C SEP OR NEP, HIGH THRUST MAY BE INCLUDED.
C THIS IS THE MAIN CONTROLLING PROGRAM FOR FINDING THE
C   OPTIMAL TRAJECTORY FOR SATELLITE RAISING
C   USING ELECTRIC PROPULSION AND HIGH THRUST.
C
C
C
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
      COMMON /INT/ITF, IPR, IDIM, IDIM2, NIMAX
      EXTERNAL TRAJ, PRTN
C
C
C
C ALL SETTING OF CONSTANTS AND INITIAL READ AND WRITES
C    ARE DONE BY THE SUBROUTINE INPUT
C
      CALL INPUT
      IF (NIMAX.EQ.0) GO TO 10
      WRITE (6,1001)
C
C THE ITERATOR ROUTINE SOLVES THE 2PBVP FOR THE OPTIMAL TRAJECTORY
C
      CALL ITER(KOUNT, NI, TRAJ, PRTN)
      IF (NI .EQ. 9999) WRITE (6,1002)
      WRITE (6,1003)
C
C OUTPC PRINTS A SUMMARY OF THE OPTIMAL TRAJECTORY VALUES
C
      CALL OUTPC(KOUNT)
C
C TIME HISTORY OF THE OPTIMAL TRAJECTORY IS CALCULATED AND PRINTED
C
 10   WRITE (6,1004)
      IPR=1000000
      ITF=3
      CALL TRAJ
C
      IF (NI.EQ.9999) WRITE (6,1002)
      IF (NI.NE.9999) WRITE (6,1005)
      STOP
 1001 FORMAT (18H1 ITERATION BEGINS)
 1002 FORMAT (29H0 OPTIMIZATION NOT SUCCESSFUL)
 1003 FORMAT (43H1 CONVERGED VALUES FOR OPTIMIZED TRAJECTORY)
 1004 FORMAT (36H1 TIME HISTORY OF OPTIMAL TRAJECTORY)
 1005 FORMAT (30H0 PROGRAM HAS RUN SUCCESSFULLY)
      END
```

```
C OBLATE
C
C
C THIS SUBPROGRAM EVALUATES THE AVERAGED DERIVATIVE OF THE
C STATE AND COSTATE DUE TO THE OBLATENESS, J2 TERM
C ASSUMES EARTH EQUATORIAL RADIUS=1, MU=1. IF NOT C1 MUST BE
C MODIFIED.
C IF IJ=2, EVALUATE DZ(I),I=1,5 ONLY.
C
C
      SUBROUTINE OBLATE(AJ2,Z,DZJ2,IJ)
C
      IMPLICIT REAL*8(A-H,O-$)
      DIMENSION Z(10),DZJ2(10),PJ(4,5)
C
C
C
      C1= 1.5D0*AJ2/Z(1)**3.5
      B1=Z(4)**2+Z(5)**2
      B7= 1.D0-6.D0*B1 + 3.D0*B1**2
      D2= 1.D0 - Z(2)**2 -Z(3)**2
      B2= 1.D0/D2**2
      B6= 1.D0/(1.D0+B1)
      C2= B2*B6
      B4= 1.D0 - B1
      C3= C1*C2
      D16= B6*B7
      D3= C3*D16
C
      DZJ2(1)= 0.D0
      DZJ2(2)= Z(3)*D3
      DZJ2(3)= -Z(2)*D3
C
      D4=B4*C3
C
      DZJ2(4)= -Z(5)*D4
      DZJ2(5)= Z(4)*D4
C
      IF (IJ.EQ.2) RETURN
C
      D5= -3.5D0*C1/Z(1)
      B8= C2*D16*D5
C
      PJ(1,1)=Z(3)*B8
      PJ(2,1)= -Z(2)*B8
C
      B12= C2*B4*D5
C
      PJ(3,1)= -Z(5)*B12
      PJ(4,1)= Z(4)*B12
C
      D2= .25D0*D2**3
      D7= B6*Z(2)/D2
      B9= C1*D16
C
      PJ(1,2)= Z(3)*B9*D7
      PJ(2,2)= -B9*(Z(2)*D7 + C2)
```

```
C
      B13= C1*D7*B4
C
      PJ(3,2)= -Z(5)*B13
      PJ(4,2)= Z(4)*B13
C
      D9= B6*Z(3)/D2
C
      PJ(1,3)= B9*(Z(3)*D9 + C2)
      PJ(2,3)= -Z(2)*B9*D9
C
      B14= C1*B4*D9
C
      PJ(3,3)= -Z(5)*B14
      PJ(4,3)= Z(4)*B14
C
      D10= -2.D0*B2*B6**2
      D11=Z(4)*D10
      D12= -12.D0*B4
      D13= C1*B6
      D15=C2*D12
      B10= D13*(2.D0*B7*D11 + Z(4)*D15)
C
      PJ(1,4)= Z(3)*B10
      PJ(2,4)= -Z(2)*B10
C
      B15= B4*D11 - 2.D0*Z(4)*C2
C
      PJ(3,4)= -Z(5)*C1*B15
C
      B16= C2*B4
C
      PJ(4,4)= C1*(B16 + Z 4)*B15)
C
      D14= Z(5)*D10
      D15= C2*D12
      B11= D13*(2.D0*B7*D14 + Z(5)*D15)
C
      PJ(1,5)= Z(3)*B11
      PJ(2,5)= -Z(2)*B11
C
      B17= D14*B4 - 2.D0*Z(5)*C2
C
      PJ(3,5)= -C1*(B16 + Z(5)*B17)
      PJ(4,5)= Z(4)*C1*B17
C
C
      DO 10 J=1,5
      DZJ2(J+5)= 0.D0
      DO 10 I=1,4
 10   DZJ2(J+5)= DZJ2(J+5) -Z(I+6)*PJ(I,J)
      RETURN
      END
```

```
C  QUAD16
C
C
C THIS IS A MODIFIED QUADRATURE INTEGRATION PROGRAM FOR
C VECTOR VALUED FUNCTIONS FO ONE VARIABLE.  IT INTEGRATES
C G (OR H) OVER X FROM XL TO XU.  THE RESULT IS Y.
C EVALUATION IS BY A 16 POINT GAUSS QUADRATURE FORMULA.
C
      SUBROUTINE QUAD(XL,XU,FCT,Y,Z,G,H,N)
C
      IMPLICIT REAL*8(A-H,O-$)
      DIMENSION Y(1),Z(1),G(1),H(1)
C
      A= .5D0*(XU+XL)
      B= XU-XL
      C= .49470046749582497D0*B
      K=1
      GO TO 500
 10   DO 20 I=1,N
 20    Y(I)= .13576229705877047D-1*G(I)
      C= .47228751153661629D0*B
      K=2
      GO TO 500
 30   DO 40 I=1,N
 40    Y(I)= Y(I) + .31126761969323946D-1*G(I)
      C= .43281560119391587D0*B
      K=3
      GO TO 500
 50   DO 60 I=1,N
 60    Y(I)= Y(I)+ .47579255841246392D-1*G(I)
      C= .37770220417750152D0*B
      K=4
      GO TO 500
 70   DO 80 I=1,N
 80    Y(I)= Y(I)+ .62314485627766936D-1*G(I)
      C= .30893812220132187D0*B
      K=5
      GO TO 500
 90   DO 100 I=1,N
 100   Y(I)= Y(I) +.74797994408288370D-1*G(I)
      C= .22900838882861369D0*B
      K=6
      GO TO 500
 110  DO 120 I=1,N
 120   Y(I)= Y(I) +.84578259697501270D-1*G(I)
      C= .14080177538962946D0*B
      K=7
      GO TO 500
 130  DO 140 I=1,N
 140   Y(I)= Y(I) +.91301707522461790D-1*G(I)
      C= .47506254918818720D-1*B
      K=8
      GO TO 500
 150  DO 160 I=1,N
 160   Y(I)= B*(Y(I) +.94725305227534250D-1*G(I))
      RETURN
 500  CALL FCT(A-C,A+C,Z,H,G)
      DO 510 I=1,N
 510   G(I)= G(I) + H(I)
      GO TO (10,30,50,70,90,110,130,150),K
      STOP
      END
```

```
C FCT
C
C THIS SUBPROGRAM IS CALLED BY THE QUADRATURE PROGRAM AND
C   EVALUATES THE INTEGRAND
C
C
C
      SUBROUTINE FCT(F1,F2,Z,H,G)
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      COMMON /A/A,AMU,PI
C
      DIMENSION Z(10),G(10),H(10),AM(5,3),PAM(5,3,5),VEC(3),X(5),PRA(5)
C
      M=0
      F=F1
C
C EVALUATE M AND PARTIAL OF M WRT STATE
C
      DO 5 I=1,5
 5     X(I)=Z(I)
 10   CALL EVALMP(X,F,AMU,AM,PAM,1)
C
C EVALUATE THE COMMON SCALOR FACTOR
C
      CT=DCOS(F)
      ST=DSIN(F)
      RA= 1.0D0-Z(3)*CT -Z(2)*ST
      FAC= A*RA/(2.0D0*PI)
C
C EVALUATE THE UNIT OF M TRANSPOSE LAMBDA
C
      ABVEC= 0.D0
      DO 30 I=1,3
       VEC(I)=0.0D0
       DO 20 J=1,5
 20     VEC(I)= VEC(I) + AM(J,I)*Z(J+5)
 30    ABVEC= ABVEC + VEC(I)**2
      ABVEC= DSQRT(ABVEC)
      DO 40 I=1,3
 40    VEC(I)=VEC(I)/ABVEC
C
C EVALUATE THE PARTIAL OF RA WRT X
C
      PRA(1)=0.D0
      PRA(2)=-ST
      PRA(3)=-CT
      PRA(4)=0.D0
      PRA(5)=0.D0
      ABVEC=ABVEC/RA
C
C EVALUATE THE FUNCTION
C
      DO 60 I=1,5
       G(I)= 0.0D0
       G(I+5)=0.0D0
```

```
      DO 50 J=1,3
       G(I)= G(I) + AM(I,J)*VFC(J)
       DO 50 L=1,5
50       G(I+5)= G(I+5) - Z(L+5)*PAM(L,J,I)*VEC(J)
      G(I)= FAC*G(I)
60    G(I+5)=FAC*(G(I+5)-ABVEC*PRA(I))
      IF (M.EQ.1) RETURN
      DO 70 I=1,10
70     H(I)= G(I)
      F=F2
      M=1
      GO TO 10
      END
```

```
C     SUBROUTINE EVALMP
C  THIS SUBROUTINE EVALUATES THE 5X3 MATRIX M AND THE
C  5X3X5 PARTIAL OF M WRT X
C
C  IF IMFLAG=1, BOTH M (AM) AND ITS PARTIAL (PAM) ARE EVALUATED
C  IF IMFLAG=2, ONLY M (AM) IS EVALUATED
C  IF IMFLAG=3, ONLY THE PARTIAL OF M (PAM) IS EVALUATED
C
C
      SUBROUTINE EVALMP(X, THETA, AMU, AM, PAM, IMFLAG)
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
      DIMENSION X(5), AM(5,3), PAM(5,3,5)
      COMMON /EVMP/ X1,Y1,X1DOT,Y1DOT
C
C
      EN=DSQRT(AMU/X(1)**3)
      RHO= DSQRT(1.D0- X(2)**2- X(3)**2)
      BETA= 1.D0/(1.D0 +RHO)
      CT= DCOS(THETA)
      ST= DSIN(THETA)
      RA= 1.D0-X(3)*CT -X(2)*ST
      ZETA= X(3)*ST-X(2)*CT
      BETA3= BETA**3/(1.D0 -BETA)
      X1= X(1)*((1.D0 -X(2)**2*BETA)*CT +X(2)*X(3)*BETA*ST -X(3))
      Y1= X(1)*((1.D0 -X(3)**2*BETA)*ST +X(2)*X(3)*BETA*CT -X(2))
      X1DOT= -((1.D0 -X(2)**2*BETA)*ST -X(2)*X(3)*BETA*CT)*EN*X(1)/RA
      Y1DOT= ((1.D0 -X(3)**2*BETA)*CT -X(2)*X(3)*BETA*ST)*EN*X(1)/RA
      PZ1= X(1)*(ZETA*(BETA+X(2)**2*BETA3) -(X(2)*BETA -ST)*CT/RA)
      PZ2= -X(1)*(-ZETA*X(2)*X(3)*BETA3 +1.D0 +(ST -X(2)*BETA)*ST/RA)
      PZ3= X(1)*(-ZETA*X(2)*X(3)*BETA3-1.D0 +(X(3)*BETA -CT)*CT/RA)
      PZ4= X(1)*(-ZETA*(BETA +X(3)**2*BETA3) +(CT -X(3)*BETA)*ST/RA)
      IF (IMFLAG .EQ. 3) GO TO 10
C  IF DO NOT WANT TO EVALUATE PARTIAL OF M, BRANCH TO 10
      AM(1,1)= 2.D0*X1DOT/(EN**2*X(1))
      AM(1,2)= 2.D0*Y1DOT/(EN**2*X(1))
      AM(1,3)=0.D0
       DUM= RHO/(EN*X(1)**2)
      AM(2,1)= DUM*(PZ2- X(2)*BETA*X1DOT/EN)
      AM(2,2)= DUM*(PZ4 -X(2)*BETA*Y1DOT/EN)
      AM(2,3)= DUM*(X(3)*(X(5)*Y1 -X(4)*X1))/RHO**2
      AM(3,1)= -DUM*(PZ1 +X(3)*BETA*X1DOT/EN)
      AM(3,2)= -DUM*(PZ3 +X(3)*BETA*Y1DOT/EN)
      AM(3,3)= -DUM*(X(2)*(X(5)*Y1 -X(4)*X1)/RHO**2)
      AM(4,1)=0.D0
      AM(4,2)=0.D0
      DUM= (1.D0 +X(4)**2 +X(5)**2)/(2.D0*EN*X(1)**2*RHO)
      AM(4,3)= DUM*Y1
      AM(5,1)=0.D0
      AM(5,2)=0.D0
      AM(5,3)= DUM*X1
      IF (IMFLAG .EQ. 2) RETURN
C  IF WE ONLY WISH TO EVALUATE M THEN PROGRAM RETURNS HERE
 10   CA= DSQRT(AMU/X(1))/RA
      PZ5= X(2)*BETA3
      PZ6= X(3)*BETA3
      PZ9= CA*ST/RA
      PZ10= CA*CT/RA
```

```
      PZ20= X(1)*(-2.D0*X(2)*BETA*CT +X(3)*BETA*ST +PZ5*ZETA*X(2))
      PZ26= X(1)*(X(2)*BETA*ST -1.D0 +PZ6*X(2)*ZETA)
      PZ29= X(1)*(X(3)*BETA*CT -1.D0 -PZ5*X(3)*ZETA)
      PZ35= X(1)*(-2.D0*X(3)*BETA*ST +X(2)*BETA*CT -PZ6*X(3)*ZETA)
      PZ11= -X1DOT/(2.D0*X(1))
      PZ12= -Y1DOT/(2.D0*X(1))
      DUM1= 1.D0 -RA
      PZ13= -CA*(-2.D0*X(2)*BETA*ST -X(3)*BETA*CT -PZ5*X(2)*DUM1)+PZ9
     1                         *X1DOT/CA
      PZ14= -CA*(-X(2)*BETA*CT -PZ6*X(2)*DUM1) +PZ10*X1DOT/CA
      PZ15= -CA*(X(3)*BETA*ST +PZ5*X(3)*DUM1) +PZ9*Y1DOT/CA
      PZ16= -CA*(2.D0*X(3)*BETA*CT +X(2)*BETA*ST +PZ6*DUM1*X(3))
     1         +PZ10*Y1DOT/CA
      DUM= BETA +X(2) *PZ5
      PZ17= 1.D0+ PZ5*X(2)*(3.D0/BETA +1.D0/(1.D0-BETA))
      PZ18= (2.D0 +PZ17)*PZ5
      PZ19= PZ17*PZ6
      DUM2= X(2)*BETA -ST
      PZ21= -X(1)*(CT*DUM -ZETA*PZ18 +CT*DUM/RA +CT*ST*DUM2/RA**2)
      PZ22= X(1)*(ST*DUM +ZETA*PZ19 -CT*X(2)*PZ6/RA-CT**2*DUM2/RA**2)
      PZ23= BETA3*(3.D0/BETA +1.D0/(1.D0 -BETA))
      PZ24= PZ23*PZ5
      PZ25=PZ23*PZ6
      PZ27= X(1)*(-CT*X(2)*X(3)*BETA3 +ZETA*X(3)*(BETA3 +X(2)*PZ24)
     1      +(ST*(BETA +X(2)*PZ5))/RA +ST**2*DUM2/RA**2)
      PZ28= X(1)*(ST*X(2)*X(3)*BETA3 +ZETA*X(2)*(BETA3 +X(3)*PZ25)
     1      +X(2)*ST*PZ6/RA +ST*CT*DUM2/RA**2)
      DUM2= X(3)*BETA-CT
      PZ30= X(1)*(CT*X(2)*X(3)*BETA3 -ZETA*X(3)*(BETA3 +X(2)*PZ24)
     1     +CT*X(3)*PZ5/RA +CT*ST*DUM2/RA**2)
      PZ31= X(1)*(-ST*X(2)*X(3)*BETA3 -ZETA*X(2)*(BETA3 +X(3)*PZ25)
     1      +CT*(BETA +X(3)*PZ6)/RA +CT**2*DUM2/RA**2)
      DUM= BETA +X(3)*PZ6
      PZ32= 1.D0 +PZ6*X(3)*(3.D0/BETA +1.D0/(1.D0 -BETA))
      PZ33= PZ32*PZ5
      PZ34= PZ32*PZ6 +2.D0*X(3)*BETA3
      PZ36= X(1)*(CT*DUM -ZETA*PZ33 -ST*X(3)*PZ5/RA -ST**2*DUM2/RA**2)
      PZ37= X(1)*(-ST*DUM -ZETA*PZ34 -ST*(BETA +X(3)*PZ6)/RA -ST*CT
     1         *DUM2/RA**2)
      DO 20 J=1,2
20    PAM(1,J,1)= 3.D0*AM(1,J)/(2.D0*X(1))
      DUM =2.D0*X(1)**2/AMU
      PAM(1,1,2)= PZ13*DUM
      PAM(1,1,3)= PZ14*DUM
      PAM(1,2,2)= PZ15*DUM
      PAM(1,2,3)=PZ16*DUM
      DUM= DSQRT(AMU*X(1))
      CB=RHO/DUM
      PZ38= -X(2)*CB/RHO**2
      PZ39= -X(3)*CB/RHO**2
      PAM(2,1,1)= AM(2,1)/(2.D0*X(1))
      PAM(2,1,2)= -CB*BETA*X1DOT/EN +PZ38*AM(2,1)/CB +CB*(PZ27
     1          -X(2)*BETA*PZ13/EN -X(2)*X1DOT*PZ5/EN)
      PAM(2,1,3)= PZ39*AM(2,1)/CB +CB*(PZ28 -PZ6*X(2)*X1DOT/EN
     1          -X(2)*BETA*PZ14/EN)
      PAM(2,2,1)= AM(2,2)/(2.D0*X(1))
      PAM(2,2,2)= PZ38*AM(2,2)/CB +CB*(PZ36 -BETA*Y1DOT/EN -X(2)
```

```
     1        *Y1DOT*PZ5/EN -X(2)*BETA*PZ15/EN)
      PAM(2,2,3)= PZ39*AM(2,2)/CB +CB*(PZ37 -X(2)*Y1DOT*PZ6/EN
     1             -X(2)*BETA*PZ16/EN)
      PAM(2,3,1)= AM(2,3)/(2.D0*X(1))
      DUM1= X(5)*Y1 -X(4)*X1
      PAM(2,3,2)= X(3)*(X(5)*PZ29 -X(4)*PZ20)/(RHO*DUM) +X(2)*X(3)
     1        *DUM1/(RHO**3*DUM)
      PAM(2,3,3)= DUM1/(RHO*DUM) +X(3)*(X(5)*PZ35 -X(4)*PZ26)/(RHO
     1        *DUM) +X(3)**2*DUM1/(RHO**3*DUM)
      PAM(2,3,4)= -X(3)*X1/(RHO*DUM)
      PAM(2,3,5)= X(3)*Y1/(RHO*DUM)
      PAM(3,1,1)= AM(3,1)/(2.D0*X(1))
      PAM(3,1,2)= PZ38*AM(3,1)/CB -CB*(PZ21 +X(3)*X1DOT*PZ5/EN
     1        +X(3)*BETA*PZ13/EN)
      PAM(3,1,3)= PZ39*AM(3,1)/CB -CB*(PZ22 +(BETA*X1DOT +X(3)
     1         *X1DOT*PZ6 +X(3)*BETA*PZ14)/EN)
      PAM(3,2,1)= AM(3,2)/(2.D0*X(1))
      PAM(3,2,2)= PZ38*AM(3,2)/CB -CB*(PZ30 +X(3)*(Y1DOT*PZ5
     1       +BETA*PZ15)/EN)
      PAM(3,2,3)= PZ39*AM(3,2)/CB -CB*(PZ31 +(BETA*Y1DOT +X(3)
     1        *Y1DOT*PZ6 +X(3)*BETA*PZ16)/EN)
      PAM(3,3,1)= AM(3,3)/(2.D0*X(1))
      PAM(3,3,2)= -DUM1/(RHO*DUM) -X(2)*(X(5)*PZ29 -X(4)*PZ20)/
     1        (RHO*DUM) -X(2)**2*DUM1/(RHO**3*DUM)
      PAM(3,3,3)= -X(2)*(X(5)*PZ35 -X(4)*PZ26)/(RHO*DUM) -X(2)*X(3)
     1        *DUM1/(RHO**3*DUM)
      PAM(3,3,4)= X(2)*X1/(RHO*DUM)
      PAM(3,3,5)= -X(2)*Y1/(RHO*DUM)
      Z5= (1.D0 +X(5)**2 +X(4)**2)/(2.D0*DUM*RHO)
      PZ40= -Z5/(2.D0*X(1))
      PZ41= X(2)*Z5/RHO**2
      PZ42= X(3)*Z5/RHO**2
      PZ43= X(4)/(DUM*RHO)
      PZ44= X(5)/(DUM*RHO)
      PAM(4,3,1)= AM(4,3)/(2.D0*X(1))
      PAM(4,3,2)= PZ41*Y1+Z5*PZ29
      PAM(4,3,3)= PZ42*Y1 +Z5*PZ35
      PAM(4,3,4)= PZ43*Y1
      PAM(4,3,5)= PZ44*Y1
      PAM(5,3,1)= AM(5,3)/(2.D0*X(1))
      PAM(5,3,2)= PZ41*X1 +Z5*PZ20
      PAM(5,3,3)= PZ42*X1 +Z5*PZ26
      PAM(5,3,4)= PZ43*X1
      PAM(5,3,5)= PZ44*X1
      DO 30 K=1,5
      PAM(1,3,K)=0.D0
      DO 30 I=4,5
      DO 30 J=1,2
   30 PAM(I,J,K)=0.D0
      DO 40 I=1,3
      DO 40 J=1,2
      DO 40 K=4,5
   40 PAM(I,J,K)=0.D0
      RETURN
      END
```

```
C MAINE
C
C
C
C
      SUBROUTINE MAINE (E,F,XK,UP,XJ,HR,IMAX,[illegible]1,DP,TTOT)
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION S(3,7,20),DP(5),D(5)
C
      COMMON/STR/S
      COMMON/BPC/BPO,BPT,E1,PI,DTF
C
  103 FORMAT(6F15.10,' FAILED GENERAL TESTS')
C
      CALL START (E,F,XK,UP,XJ,HR,1,1,NF)
      IF (NF) 399,399,100
C
  100 J=0
      I= 1
      BPO=(DSQRT(2.D0*(1.D0+E))-1.D0-E)/HR
C
  202 J=J+1
  208 DT1= 0.D0
  210 CALL TIME(I,J,DT1,SPAN,S(I,J+1,2))
  220 CALL SWITCH(I,J,S(I,J+1,2),X,NX,0)
      IF (S(I,J+1,1).GT.TTOT) GO TO 230
      IF (J+1-JM) 202,307,307
  230 JM= J
  307 DTF= TTOT-S(I,JM,1)
      S(I,JM+1,14)= 1.D0
      S(I,JM+1,15)= 0.D0
      S(I,JM+1,3)= 0.D0
      CALL DTDU(I,JM,DTF,JM+1,1)
  308 RETURN
  399 WRITE (6,103) E,F,XK,UP,XJ,HR
      RETURN
      END
```

```
C START
C
C
C
C
      SUBROUTINE START (F,E,XK,UP,XJ,HR,I,J,NF)
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION S(3,7,20)
C
      COMMON/STR/S
C
      S(I,J,12)= DCOS(F)
      S(I,J,13)= DSIN(F)
      S(I,J,10)= 1.D0+E*S(I,J,12)
      RAD= DSQRT(S(I,J,10))
      S(I,J,5)= E*S(I,J,13)/RAD
      S(I,J,9)= RAD*DSIN(UP)
      S(I,J,11)= RAD*DCOS(UP)-XK
      S(I,J,7)= (XJ*S(I,J,9)+XK*S(I,J,5))/(1.D0+XK*(XK+S(I,J,11)))
 5    S(I,J,4)= HR/DSQRT((1.D0+S(I,J,7)**2)*S(I,J,10))
      S(I,J,6)= XK
      S(I,J,8)= XJ*(XK+S(I,J,11))+(XK*S(I,J,7)-S(I,J,5))*S(I,J,9)
      IF (I-1) 31,6,7
 6    CALL SWITCH(I,1,0.D0,DX,NF,1)
      IF (NF) 31,31,7
 7    S(I,J,1)= 0.D0
      S(I,J,2)= 0.D0
      S(I,J,3)= 0.D0
      S(I,J,14)= 1.D0
      S(I,J,15)= 0.D0
      S(I,J,16)= 1.D0
      S(I,J,17)= 0.D0
      S(I,J,18)= 0.D0
      S(I,J,19)= 0.D0
      S(I,J,20)= 0.D0
 31   RETURN
      END
```

```
C TIME
C
C
C
C
      SUBROUTINE TIME (I,J,DT1,SPAN,DTS)
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION DT(50),Q(50)
C
 101  FORMAT(8D16.8)
 102  FORMAT(4I3,6D16.8)
 107  FORMAT(' FAILED KSP')
C
      K= 2
      K75= 0
      IKS= 20
      M= 4
      DTNEG= 5.D-10
      IF (DT1) 305,500,600
 500  DT(1)= 1.D-6
      CALL SWITCH(I,J,DT(1),Q(1),NF,1)
      IF (NF) 306,306,501
 501  DT(2)= 1.D-3
 502  CALL SWITCH(I,J,DT(K),Q(K),NF,2)
      IF (NF) 510,510,503
 503  IF (K-7) 504,306,306
 504  DT(K+1)= DT(K)*5.D0
      K= K+1
      GO TO 502
 510  SPAN= (DT(K)-DT(K-1))/5.D0
      GO TO 79
 600  CALL SWITCH(I,J,DT1,Q(1),NF,1)
      IF (NF) 2,2,1
 1    DT(1)= DT1
      DT(2)= DT(1)+SPAN
      CALL SWITCH(I,J,DT(2),Q(2),NF,2)
      IF (NF) 50,50,305
 2    DT(2)= DT1
      Q(2)= Q(1)
      DT(1)= DMAX1(1.D-6,DT1-SPAN)
      CALL SWITCH(I,J,DT(1),Q(1),NF,2)
      IF (NF) 305,305,50
 50   KS= K-1
C NOW GOOD AT DT(KS),BAD AT DT(KS)+SPAN
      KSP= KS+IKS
      DQ= Q(K)-Q(K-1)
 51   DDT= (DT(K-1)-DT(K))/DQ*Q(K)
      IF (DDT*(DTNEG-DDT)) 54,53,53
 53   IF (NF) 75,75,100
 54   IF (KSP-K) 304,304,55
 55   DT(K+1)= DT(K)+DDT-DTNEG/2.D0
      IF ((DT(K+1)-DT(KS)+DTNEG)*(DT(KS)+SPAN-DT(K+1))) 75,75,56
 56   K= K+1
      CALL SWITCH (I,J,DT(K),Q(K),NF,2)
      DQ= Q(K)-Q(K-1)
```

```
      IF (DQ) 51,100,51
75    IF (K75-M) 76,76,305
76    K75= K75+1
      K= KS
      SPAN= SPAN/5.D0
78    K= K+1
79    DT(K)= DT(K-1)+SPAN
      CALL SWITCH(I,J,DT(K),Q(K),NF,2)
80    IF (NF) 50,50,78
100   DT2= DT(K)+DTNEG
      RETURN
304   WRITE (6,107)
305   WRITE (6,102) I,J,K,K75,DT1,SPAN,DT(1),Q(1),DT(2),Q(2)
      WRITE (6,101) (DT(IK),Q(IK), IK=1,K)
      KK= 51
      Q(KK)= 1.D0
306   RETURN
      END
```

```
C SWITCH
C
C
C
C
      SUBROUTINE SWITCH(I,J,T,O1,NF,KDU)
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION S(3,7,20)
C
      COMMON/STR/S
C
 177  FORMAT(8D16.8)
 178  FORMAT(5D25.15)
C
      IF (KDU-1) 40,20,30
 20   ASK= 1.D0-S(I,J,9)*S(I,J,9)-S(I,J,6)*S(I,J,6)
      TF= 1.D0-2.D0*S(I,J,7)*S(I,J,7)
      OS= S(I,J,5)+(S(I,J,6)-S(I,J,11))*S(I,J,7)
      OS8= 8.D0*S(I,J,7)*OS
      ASKO= ASK*(2.D0*TF-1.D0)
      THO= -3.D0*(2.D0*TF-1.D0)
      D22= -(2.D0+TF)*S(I,J,7)
      D21= -D22*S(I,J,6)-OS*TF
      XE= TF-S(I,J,6)**2-((S(I,J,8)+S(I,J,5)*S(I,J,9))**2+
     1         (S(I,J,5)*S(I,J,6)-S(I,J,7))**2)/S(I,J,10)
 30   F= S(I,J,11)+T*S(I,J,4)
      F2= F*F
      P= (F+S(I,J,6))*(F+S(I,J,6))+S(I,J,9)*S(I,J,9)
      XA= P*(ASK+F2)-4.D0*F2
      XB= S(I,J,7)*(2.D0*P*(F+S(I,J,6))-4.D0*F)+(S(I,J,5)+
     1     S(I,J,7)*T*S(I,J,4))*(2.D0*F2-P)
      XC= ASKO+OS8*F+THO*F2+P*XE
      XD= D21+D22*F
      X4= XA*XE-XB*XD
      Z4= XE*XB*XB+XA*(XD*XD-XE*XC)
      O1= (XC*XC+16.D0*X4)*X4*X4-Z4*(XC*(XC*XC+18.D0*X4)+27.D0*Z4)
      IF (XE) 9,9,777
 777  IF (O1) 9,3,3
 3    PP= 2.D0*XC-3.D0*XD*XD/XE
      IF (PP) 4,10,10
 4    IF (XC*XC+12.D0*X4-PP*PP) 9,9,10
 9    NF= 0
      RETURN
 10   NF= 1
      RETURN
 40   DER= XD*XD/XE/XE
      TR= DER/2.D0*(XC/XE-3.D0/4.D0*DER)+2.D0*X4/XE/XE
      OP= XC/XE-3.D0/2.D0*DER
      DO= (XD*(DER-XC/XE)+2.D0*XB)/XE
      IF (DO) 42,41,42
 41   S(I,J+1,3)= 1.D20
      S(I,J+1,14)= -1.D0
      S(I,J+1,15)= 0.D0
      GO TO 43
 42   PPR= OP+DSQRT(XC*XC+12.D0*X4)/XE
```

```
      YS= -DABS(DO)/DO*DSQRT(TR/PPR)
      S3= YS-XD/2.DO/XE
      S(I,J+1,3)= 1.DO/S3
403   DEN= 1.DO+P*S(I,J+1,3)*S(I,J+1,3)
      S(I,J+1,14)= 2.DO/DEN-1.DO
      S(I,J+1,15)= 2.DO*S(I,J+1,3)*DSQRT(P)/DEN
43    CALL DTDU(I,J,T,J+1,1)
300   RETURN
      END
```

```
C DTDU
C
C
C
C
      SUBROUTINE DTDU(I,L,DT,M,ISW)
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION S(3,7,20)
C
      COMMON/STR/S
C
      CDU= 1.D0
      SDU= 0.D0
      DF= S(I,L,4)*DT
      VL= 1.D0+DF*(S(I,L,6)+S(I,L,11))/S(I,L,10)
      VH= DF*S(I,L,9)/S(I,L,10)
      RAD2= VL*VL+VH*VH
      S(I,M,1)= S(I,L,1)+DT
      S(I,M,2)= DT
      S(I,M,5)= S(I,L,5)+DF*S(I,L,7)
      S(I,M,8)= S(I,L,8)*VL-VH*((S(I,L,10)-1.D0)*S(I,L,7)-
     1   S(I,L,11)*S(I,L,5))
      S(I,M,10)= S(I,L,10)*RAD2
      S(I,M,11)= S(I,L,11)+DF
      RAD= DSQRT(RAD2)
      CX= VL/RAD
      SX= VH/RAD
      S(I,M,16)= CX*S(I,L,16)-SX*S(I,L,17)
      S(I,M,17)= CX*S(I,L,17)+SX*S(I,L,16)
      S(I,M,18)= S(I,L,18)
      TX= (CX+S(I,M,16))/(1.D0+S(I,L,16))
      S(I,M,19)= TX*S(I,L,19)+SX*S(I,L,12)
      S(I,M,20)= TX*S(I,L,20)+SX*S(I,L,13)
      IF (ISW) 12,12,11
 11   CDU= S(I,L+1,14)
      SDU= S(I,L+1,15)
      SOR= SDU/DSQRT(S(I,M,10))
      SRT= (1.D0-CDU)/S(I,M,10)
      TP= CDU+SRT-SOR*S(I,M,5)
      TTO= CDU*S(I,L,7)+SOR*S(I,M,11)
      CSOC= DSQRT(1.D0+S(I,L,7)*S(I,L,7)-TTO*TTO)
      FPR= CSOC/DSQRT(TP)
      S(I,M,4)= FPR*S(I,L,4)
      S(I,M,5)= (CDU*S(I,M,5)+(S(I,M,10)-1.D0)*SOR)*FPR/CSOC
      S(I,M,6)= (S(I,L,6)-SOR*S(I,L,7)-SRT*S(I,M,11))/FPR/TP
      S(I,M,7)= TTO/CSOC
      S(I,M,9)= (CDU*S(I,L,9)+SOR*S(I,M,8))/FPR/TP
      S(I,M,8)= (CDU*S(I,M,8)-SOR*S(I,L,9)*S(I,M,10))/CSOC
      S(I,M,11)= (CDU*S(I,M,11)-SOR*S(I,L,7)*S(I,M,10))/FPR
      S(I,M,10)= TP*S(I,M,10)
      T1= S(I,M,17)
      S(I,M,17)= CDU*S(I,M,17)-SDU*S(I,M,18)
      S(I,M,18)= CDU*S(I,M,18)+SDU*T1
 12   SSI= S(I,M,19)*S(I,M,19)+S(I,M,20)*S(I,M,20)
      IF (SSI) 13,13,14
```

```
13    S(I,M,12)= CDU*S(I,L,12)-SDU*S(I,L,13)
      S(I,M,13)= CDU*S(I,L,13)+SDU*S(I,L,12)
      GO TO 300
14    S(I,M,12)= (S(I,M,17)*S(I,M,19)-S(I,M,18)*S(I,M,20))/SSI
      S(I,M,13)= (S(I,M,18)*S(I,M,19)+S(I,M,17)*S(I,M,20))/SSI
300   RETURN
      END
```

```
C OUTHI
C
C
C
      SUBROUTINE OUTHI(JM,PI,A0,AI1,OMI,IPR,Z,IDIM2)
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION S(3,7,20),TR(3,3),TR1(3,3),TR3(3,3),ZP(10),Z(1)
C
      COMMON/STP/S
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
      COMMON /SG/SGN
C
C     NAMELIST/OUT/J,A,H,E,AI,OM,W,F,U,PHI,UP,DT,TT
C     NAMELIST/OUT2/AI3,OM3,W3,ALAM3,ALAM4,ALAM5,Y2,ALA,ALE,ALI
C     NAMELIST/OUT3/ZP
C
C
      JM1=JM+1
      IF (IPR.EQ.0) GO TO 12
C     DO 10 K= 1,20
C10    WRITE (6,999) K,(S(1,J,K), J=1,JM1)
      WRITE (6,1000)
      WRITE (6,1001)
      WRITE (6,1000)
C
 12   DO 100 J= 1,JM1
      IF ((IPR.EQ.0).AND.(J.GT.1).AND.(J.LT.JM1)) GO TO 100
      AI= DATAN2(DSQRT(S(1,J,19)**2+S(1,J,20)**2),S(1,J,16))
      OM= 0.D0
      IF (S(1,J,16).NE.1.D0) OM= DATAN2(S(1,J,20),S(1,J,19))
      DUM= DATAN2(S(1,J,13),S(1,J,12))
      U= DUM-OM
      DUM1= S(1,J,10)-1.D0
      DUM2= DSQRT(S(1,J,10))*S(1,J,5)
      E= DSQRT(DUM1**2+DUM2**2)
      F=U
      IF (E.NE.0.D0) F= DATAN2(DUM2,DUM1)
      W= U-F
      PHI= DATAN(S(1,J,7))
      H= S(1,J,4)*DSQRT(S(1,J,10))/DCOS(PHI)
      H= DABS(H)
      UP= DATAN2(S(1,J,9),S(1,J,11)+S(1,J,6))
      DT= S(1,J,2)
      TT= S(1,J,1)
      IF (J.GT.1) GO TO 40
C
      CPHI= DCOS(PHI)
      G1= S(1,1,8)*CPHI
      G2= S(1,1,9)*DSQRT(S(1,1,10))*CPHI
C
C SGN= +1.D0 IMPLIES ALAM4 IS POSITIVE
C SGN= -1.D0 IMPLIES ALAM4 IS NEGATIVE
C
      ALAM5= (SGN+1.D0)*PI/2.D0
      ALAM4= SGN*DSQRT(G1**2+G2**2)
```

```
      IF (ALAM4.EQ.0.D0) GO TO 14
      ALAM5= DATAN2(SGN*G2,SGN*G1)
   14 WI= ALAM5-PI/2.D0
      ALI= DSQRT(1.D0/A0)*ALAM4
      WD= WI
      AID= AII
      OMD= OMI
      GO TO 110
   20 DO 30 M=1,3
      DO 30 N= 1,3
      TR3(M,N)= TR(M,N)
   30 TR1(M,N)= TR(M,N)
      IF (IPR) 52,100,52
C
   40 WD= W
      AID= AI
      OMD= OM
      GO TO 110
   50 DO 51 M= 1,3
      DO 51 N= 1,3
      TR3(M,N)= 0.D0
      DO 51 IK=1,3
   51 TR3(M,N)= TR3(M,N)+TR(M,IK)*TR1(IK,N)
   52 SI= DSQRT(TR3(3,1)**2+TR3(3,2)**2)
      AI3= DATAN2(SI,TR3(3,3))
      OM3= 0.D0
      W3= DATAN2(TR3(1,2),TR3(1,1))
      IF (DABS(SI).LT.1.D-10) GO TO 60
      OM3= DATAN2(TR3(3,1),-TR3(3,2))
      W3= DATAN2(TR3(1,3),TR3(2,3))
   60 Z(2)= E*DSIN(W3+OM3)
      Z(3)= E*DCOS(W3+OM3)
      Z(4)= TR3(3,1)/(1.D0+TR3(3,3))
      Z(5)= -TR3(3,2)/(1.D0+TR3(3,3))
      ALAM3= S(1,J,4)*(S(1,J,10)*S(1,J,6)-S(1,J,11))/H
      Y2= DSIN(F)*DSIN(PHI)+DCOS(F)*DCOS(PHI)*S(1,J,11)/DSQRT(
     1    S(1,J,10))
      DUM= 1.D0-E**2
      A= A0*H*H/DUM
      Z(1)= A
      ALA= .5D0*DSQRT(1.D0/(A**3*DUM))*(ALAM3-E*Y2)
      ALE= DSQRT(1.D0/(A*DUM**3))*(Y2-E*ALAM3)
      Z(1+IDIM2)= ALA
      Z(2+IDIM2)= DSIN(W3+OM3)*ALE
      Z(3+IDIM2)= DCOS(W3+OM3)*ALE
      DUM= 2.D0*(DCOS(AI/2.D0))**2*ALI
      Z(4+IDIM2)= DSIN(OM3)*DUM
      Z(5+IDIM2)= DCOS(OM3)*DUM
      IF (IPR.EQ.0) GO TO 100
      W3= W3/DTR
      OM3= OM3/DTR
      AI3= AI3/DTR
      AI= AI/DTR
      OM= OM/DTR
      W= W/DTR
      F= F/DTR
      U= U/DTR
```

```
      PHI= PHI/DTR
      UP= UP/DTR
C
      DO 90 I= 1,5
      ZP(I)= Z(I)
 90   ZP(I+5)= Z(I+IDIM2)
C     WRITE (6,OUT)
C     WRITE (6,OUT2)
C     WRITE (6,OUT3)
      A= A*UTKM
      DV= (DT*UTKM/UTS)*DSQRT(1.D0/A0)
      WRITE (6,1002) J
      WRITE (6,1003)
      WRITE (6,1010) (Z(I), I= 1,5)
      WRITE (6,1010) (Z(I+IDIM2), I= 1,5)
      WRITE (6,1004)
      WRITE (6,1010) A,E,AI3,OM3,W3
      WRITE (6,1005)
      WRITE (6,1010) F,PHI,UP,DV
 100  CONTINUE
C
      IF (IPR.EQ.0) RETURN
      WRITE (6,1000)
      WRITE (6,1000)
      WRITE (6,1006)
C
      RETURN
C
 110  CW= DCOS(WD)
      SW= DSIN(WD)
      CI= DCOS(AID)
      SI= DSIN(AID)
      COM= DCOS(OMD)
      SOM= DSIN(OMD)
      TR(1,3)= SW*SI
      TR(2,3)= CW*SI
      TR(3,3)= CI
      TR(3,1)= SI*SOM
      TR(3,2)= -SI*COM
      TR(1,1)= CW*COM-SW*CI*SOM
      TR(1,2)= CW*SOM+SW*CI*COM
      TR(2,1)= -SW*COM-CW*CI*SOM
      TR(2,2)= -SW*SOM+CW*CI*COM
      IF (J.EQ.1) GO TO 20
      GO TO 50
C
C
C
 999  FORMAT (I4,1P5D20.8)
 1000 FORMAT (1H0,'**************************************************')
 1001 FORMAT (1H0,'HIGH THRUST')
 1002 FORMAT (1H0,'ORBIT =',I2)
 1003 FORMAT (1H0,'EQUINOCTIAL O.E. AND COSTATE/S.F.*1000.')
 1004 FORMAT (1H0,'CLASSICAL O.E.')
 1005 FORMAT (1H0,6X,'TRUE ANOMALY',13X,'PHI (DEG)',12X,'UPSILON (DEG)',
     1  9X,'DELTA V (KM/S)')
 1006 FORMAT (1H0,'LOW THRUST')
 1010 FORMAT (1P5D23.14)
      END
```

```
C IMPLS
C
C
C
C CALCULATES EFFECT OF ONE FINAL IMPULSE AT MAX. OF PRIMER VECTOR
C
C
      SUBROUTINE IMPLS(DV12,IPR,ITR,Z,DERZ,IDIM2)
C
      IMPLICIT REAL*8(A-H,O-$)
C
      DIMENSION Y(22),UY(3),V2(3),R(3),CM(3,3),H(3),E(3),C1(3)
      DIMENSION ZZ(5),ZS(5),R1(3),V1(3),Z(1),DERZ(1),DZ5(5),Z10(10)
C
      COMMON /EVMP/ RT(2),VT(2)
      COMMON /A/ A,AMU,PI
      COMMON/UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
C
C     NAMELIST/FIM1/F,UY,ZZ,R1,V1,R,V2,H,E,Y,YS,FS,IS,YO,CM
C     NAMELIST/FIM2/DZ5
C
C
      DO 4 I= 1,5
      Z10(I)= Z(I)
    4 Z10(I+5)= Z(I+IDIM2)
      KK= 0
      DF= PI/10.D0
    5 F= -DF
      YS= -1.D0
      DO 10 I= 2,21
      F= F+DF
      CALL YF(F,Y(I),UY,Z10)
      IF (Y(I).LE.YS) GO TO 10
      YS= Y(I)
      IS= I
      FS= F
   10 CONTINUE
C
      Y(1)= Y(21)
      Y(22)= Y(2)
      FN= FS-DF
      FP= FS+DF
      C= (Y(IS)*FP**2-Y(IS+1)*FS**2)-(Y(IS-1)*FP**2-Y(IS+1)*FN**2)
     1   +(Y(IS-1)*FS**2-Y(IS)*FN**2)
      B= (FS*Y(IS+1)-FP*Y(IS))-(FN*Y(IS+1)-FP*Y(IS-1))
     1   +(FN*Y(IS)-FS*Y(IS-1))
      F=  C/(2.D0*B)
C
      CALL YF(F,YO,UY,Z10)
      DO 15 I= 1,2
      R1(I)= RT(I)
   15 V1(I)= VT(I)
      R1(3)= 0.D0
      V1(3)= 0.D0
      DO 20 I= 1,3
   20 V1(I)= V1(I)+DV12*UY(I)
      P= Z(4)
```

```
      Q= Z(5)
      IFLAG= 1
      GO TO 200
C
 25   DO 30 I= 1,3
      V2(I)= 0.D0
      R(I)= 0.D0
      DO 30 J=1,3
      V2(I)= V2(I)+CM(I,J)*V1(J)
 30   R(I)= R(I)+CM(I,J)*R1(J)
      DO 40 I= 1,3
      I1= I+1
      I2= I+2
      IF (I1.GT.3) I1= I1-3
      IF (I2.GT.3) I2= I2-3
 40   H(I)= R(I1)*V2(I2)-R(I2)*V2(I1)
      DO 50 I= 1,3
      I1= I+1
      I2= I+2
      IF (I1.GT.3) I1= I1-3
      IF (I2.GT.3) I2= I2-3
 50   E(I)= (V2(I1)*H(I2)-V2(I2)*H(I1))/AMU
      HM= 0.D0
      VM= 0.D0
      RAD= 0.D0
      DO 60 I= 1,3
      VM= VM+V2(I)**2
      HM= HM+H(I)**2
 60   RAD= RAD+R(I)**2
      RAD= DSQRT(RAD)
      HM= DSQRT(HM)
      DO 70 I= 1,3
      E(I)= E(I)-R(I)/RAD
 70   H(I)= H(I)/HM
      ZZ(1)= 1.D0/(2.D0/RAD-VM/AMU)
      DUM= 1.D0+H(3)
      ZZ(4)= H(1)/DUM
      ZZ(5)= -H(2)/DUM
C
      P= ZZ(4)
      Q= ZZ(5)
      IFLAG= 2
      GO TO 200
 75   DO 80 I=1,2
      ZZ(4-I)= 0.D0
      DO 80 J= 1,3
 80   ZZ(4-I)= ZZ(4-I)+E(J)*CM(J,I)
      KK= KK+1
C     IF (IPR.GT.0) WRITE (6,FIM1)
C
      IF (ITR.GT.0) GO TO 140
      IF (KK-1) 120, 100,120
 100  DO 110 I= 1,5
      ZS(I)= ZZ(I)
      Z10(I)= Z10(I)+DERZ(I)*1.D-2
 110  Z10(I+5)= Z(I+IDIM2)+DERZ(I+IDIM2)*1.D-2
      GO TO  5
```

```
C
 120  DO 130 I= 1,5
      DERZ(I)= (ZZ(I)-ZS(I))*1.D2
      DZ5(I)=DERZ(I)
 130  Z(I)= ZS(I)
C     IF (IPR.GT.0) WRITE (6,FIM2)
      GO TO 160
C
 140  DO 150 I= 1,5
      Z(I)= ZZ(I)
 150  CONTINUE
C
 160  IF (IPR.EQ.0) RETURN
      WRITE (6,1000)
      WRITE (6,1001)
      WRITE (6,1000)
      WRITE (6,1002)
      WRITE (6,1010) (Z(I),I= 1,5)
      ZZ(1)= Z(1)*UTKM
      ZZ(2)= 0.D0
      DUMMY= Z(2)**2+Z(3)**2
      IF (DUMMY.GT.1.D-40) ZZ(2)= DSQRT(DUMMY)
      ZZ(3)= 0.D0
      DUMMY= Z(4)**2+Z(5)**2
      IF (DUMMY.GT.1.D-40) ZZ(3)= 2.D0*DATAN(DSQRT(DUMMY))/DTR
      ZZ(4)= 0.D0
      IF ((Z(4).NE.0.D0).OR.(Z(5).NE.0.D0)) ZZ(4)=DATAN2(Z(4),Z(5))/DTR
      ZZ(5)= 0.D0
      IF ((Z(2).NE.0.D0).OR.(Z(3).NE.0.D0)) ZZ(5)=DATAN2(Z(2),Z(3))/DTR
      ZZ(5)= ZZ(5)-ZZ(4)
      WRITE (6,1003)
      WRITE (6,1010) ZZ
      F= F/DTR
      WRITE (6,1004) F
      WRITE (6,1005) R1
      WRITE (6,1006) UY
      WRITE (6,1000)
      RETURN
C
C
C CALCULATES TRANSFORMATION MATRIX TO EQUINOCTIAL COORD FRAME
C
 200  AB= 1.D0+P**2+Q**2
      CM(1,1)= (1.D0-P**2+Q**2)/AB
      CM(2,1)= 2.D0*P*Q/AB
      CM(3,1)= -2.D0*P/AB
      CM(1,2)= CM(2,1)
      CM(2,2)= (1.D0+P**2-Q**2)/AB
      CM(3,2)= 2.D0*Q/AB
      CM(1,3)= -CM(3,1)
      CM(2,3)= -CM(3,2)
      CM(3,3)= (1.D0-P**2-Q**2)/AB
      GO TO (25,75),IFLAG
C
 1000 FORMAT (1H0,'*****************************************************')
 1001 FORMAT (1H0,'HIGH THRUST     FINAL ORBIT')
 1002 FORMAT ('0 EQUINOCTIAL O.E.')
```

```
 1003 FORMAT ('0 CLASSICAL O.E.')
 1004 FORMAT ('0 ECCENTRIC LONG. (DEG) = ',F20.10)
 1005 FORMAT ('0 RADIUS VECTOR = ',1P3D23.14)
 1006 FORMAT ('0 IMPULSE DIRECTION = ',1P3D23.14)
 1010 FORMAT (1P5D23.14)
C
      END
```

```
C YF
C
C
C CALLED BY IMPLS, CALLS EVALMP
C
C
      SUBROUTINE YF(F,Y,U,Z)
C
      IMPLICIT REAL*8(A-H,O-$)
C
      DIMENSION U(3),R(3),V(3),AM(5,3),DUMMY(75), X(5),Z(1'
C
C
C
      DO 5 I= 1,5
 5    X(I)= Z(I)
      CALL EVALMP(X,F,1.D0,AM,DUMMY,2)
      Y= 0.D0
      DO 20 I= 1,3
      U(I)= 0.D0
      DO 10 K= 1,5
 10   U(I)= U(I)+AM(K,I)*Z(5+K)
 20   Y= Y+U(I)**2
      Y= DSQRT(Y)
      DO 30 I= 1,3
 30   U(I)= U(I)/Y
      RETURN
      END
```

```
C ITER NRS
C NEWTON RAPHSON   NRS
C
C SPECIAL VERSION FOR 8X8 PARTIAL DER. MATRIX, 8 DIM. Y.
C
      SUBROUTINE ITER(KOUNT,NI,FUNCT,PRTN)
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
C
C X    VALUES OF THE INDEPENDENT VARIABLES(INITIAL,CURRENT,FINAL)
C XS   STEP SIZE TO PERTURB X:S TO COMPUTE PARTIAL DERIVATIVES
C Y    VALUES OF THE DEPENDENT VARIABLES(CURRENT,FINAL)
      COMMON/XMMM/X(7),XS(8),Y(8)
      COMMON /INT/ITF,IPR,IDIM,IDIM2,MAXNOI
      COMMON /T/TF,S,TO,TFMIN
      COMMON /DY/DYDT(8)
      COMMON /F/FLIM,KSTEP
C
C
C     ARRAYS USED INTERNALLY BY THE ITERATOR
      DIMENSION YNOM(8),XN(7),P(8,8),COEF(8)
      EQUIVALENCE(YNOM,COEF)
C
      N= 7
      M= 8
      INORM= ITF
      NI=1
      KOUNT=0
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      FO=0.DO
      DO 15 I=1,M
 15     FO=FO+Y(I)**2
 10   DO 16 I=1,N
      XN(I)=X(I)
 16   YNOM(I)=Y(I)
      YNOM(M)= Y(M)
      TFN=TF
      CALL PRTN(KOUNT,NI)
      WRITE(6,1011)FO
      IF(FO.LE.FLIM)GO TO 90
      IF (NI.GT.MAXNOI) GO TO 80
C COMPUTE NUMERICAL PARTIAL DERIVATIVES
      WRITE (6,1013)
      DO 17 I=1,M
 17   P(I,M)= DYDT(I)
      DO 25 J=1,N
      TEMP=X(J)
      STEP=XS(J)*DABS(X(J))
      IF ((DABS(X(J)).LT.1.D-10).OR.(KSTEP.EQ.1))STEP=XS(J)
      IF (DABS(X(J)).LT.1.D-10) WRITE (6,1014)
      X(J)=X(J)+STEP
      CALL FUNCT
      WRITE(6,1000)X(J)
      WRITE(6,1001)(Y(I),I=1,M)
```

```
      DO 20 I=1,M
   20 P(I,J)=(Y(I)-YNOM(I))/STEP
   25 X(J)=TEMP
      KOUNT=KOUNT+N
      WRITE(6,1002)
      DO 30 I=1,M
      WRITE(6,1001)(P(I,J),J=1,M)
   30 CONTINUE
      DO 35 I=1,M
   35 COEF(I)=-YNOM(I)
      CALL DCROUT(P,COEF,DET,0.D0,M,1,IND)
      IF(IND.NE.0)GO TO 85
      WRITE (6,1015) DET
      DO 40 I= 1,M
 40   IF (DABS(COEF(I)).LT.1.D-12) COEF(I)= 0.D0
      WRITE(6,1003)(COEF(I),I=1,N)
      SN= COEF(M)
      WRITE (6,1012) SN
      DO 50 J=1,N
   50 X(J)=XN(J)+COEF(J)
      TF=TFN + SN
      IHALV=0
 51   IF (INORM.EQ.1) ITF=1
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      F1=0.D0
      DO 52 I=1,M
   52 F1=F1+Y(I)**2
      WRITE(6,1010)F1
      IF(F1.LT.F0)GO TO 55
      WRITE(6,1008)
      IF (IHALV.EQ.10) GO TO 95
      IHALV=IHALV+1
      DO 53 J=1,N
      COEF(J)=COEF(J)/2.D0
      WRITE(6,1000)COEF(J)
   53 X(J)=XN(J)+COEF(J)
      SN= SN/2.D0
      WRITE (6,1012) SN
      TF= TFN + SN
      GO TO 51
   55 IF(NI-MAXNOI)70,70,80
   70 NI=NI+1
      F0=F1
      GO TO 10
   80 NI=9999
      WRITE(6,1006)
      RETURN
   85 NI=9999
      WRITE(6,1007)
      RETURN
   90 WRITE(6,1005)F0
      RETURN
   95 NI=9999
      WRITE(6,1009)
      RETURN
```

```
1000 FORMAT(/1X,1PD23.15)
1001 FORMAT(1X,1P5D23.15)
1002 FORMAT(21HOPARTIAL DERIV MATRIX)
1003 FORMAT(11HODELX:S ARE/(1X,1PD23.15))
1005 FORMAT(4HOFO=,1PD22.15,23HCASE CONVERGED...FERTIG)
1006 FORMAT(38HOEXCEEDED MAXIMUM NUMBER OF ITERATIONS)
1007 FORMAT(16HOMATRIX SINGULAR)
1008 FORMAT(11HODELX:S ARE)
1009 FORMAT(19HOMETHOD CANNOT WORK)
1010 FORMAT(4HOF1=,1PD23.15)
1011 FORMAT(4HOFO=,1PD23.15)
1012 FORMAT (10HO DEL IF =,1PD23.15)
1013 FORMAT (40HO X(I)+DX(I) FOLLOWED BY CORRESPONDING Y)
1014 FORMAT (24HO X(I)=0. SO DX(I)=XS(I))
1015 FORMAT (15HO DETERMINENT =,1PD23 15)
     END
```

```
C ITER    NRN
C NEWTON RAPHSON   NRN
C
C SPECIAL VERSION FOR 6X6 PARTIAL DER. MATRIX, 6 DIM. Y.
C
      SUBROUTINE ITER(KOUNT,NI,FUNCT,PRTN)
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
C
C X   VALUES OF THE INDEPENDENT VARIABLES(INITIAL,CURRENT,FINAL)
C XS  STEP SIZE TO PERTURB X:S TO COMPUTE PARTIAL DERIVATIVES
C Y   VALUES OF THE DEPENDENT VARIABLES(CURRENT,FINAL)
      COMMON/XMMM/X(5),XS(5),Y(6)
      COMMON /INT/ITF,IPR,IDIM,IDIM2,MAXNOI
      COMMON /T/TF,S,TO,TFMIN
      COMMON /DY/DYDT(6)
      COMMON /HIGH/DVI1,DVI2,IHI,ITR
      COMMON /F/FLIM,KSTEP
C
C
C     ARRAYS USED INTERNALLY BY THE ITERATOR
      DIMENSION YNOM(6),XN(5),P(6,6),COEF(6)
      EQUIVALENCE(YNOM,COEF)
C
      N= 5
      M= 6
      INORM= ITF
      NI=1
      KOUNT=0
      ITR= 0
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      F0=0.D0
      DO 15 I=1,M
 15    F0=F0+Y(I)**2
 10   DO 16 I=1,N
      XN(I)=X(I)
 16   YNOM(I)=Y(I)
      YNOM(M)= Y(M)
      TFN=TF
      CALL PRTN(KOUNT,NI)
      WRITE(6,1011)F0
      IF(F0.LE.FLIM)GO TO 90
      IF (NI.GT.MAXNOI) GO TO 80
C COMPUTE NUMERICAL PARTIAL DERIVATIVES
      WRITE (6,1013)
      DO 17 I=1,M
 17   P(I,M)= DYDT(I)
      DO 25 J=1,N
      ITR= J
      TEMP=X(J)
      STEP=XS(J)*DABS(X(J))
      IF ((DABS(X(J)).LT.1.D-10).OR.(KSTEP.EQ.1)) STEP=XS(J)
C     IF (DABS(X(J)).LT.1.D-10) WRITE (6,1014)
      X(J)=X(J)+STEP
```

```
      CALL FUNCT
      WRITE(6,1000)X(J)
      WRITE(6,1001)(Y(I),I=1,M)
      DO 20 I=1,M
   20 P(I,J)=(Y(I)-YNOM(I))/STEP
   25 X(J)=TEMP
      KOUNT=KOUNT+N
      WRITE(6,1002)
      DO 30 I=1,M
      WRITE(6,1001)(P(I,J),J=1,M)
   30 CONTINUE
      DO 35 I=1,M
   35 COEF(I)=-YNOM(I)
      CALL DCROUT(P,COEF,PFT,0.D0,M,1,IND)
      IF(IND.NE.0)GO TO 85
      DO 40 I= 1,M
40    IF (DABS(COEF(I)).LT.1.D-10) COEF(I)= 0.D0
      WRITE(6,1003)(COEF(I),I=1,N)
      SN= COEF(M)
      WRITE (6,1012) SN
      DO 50 J=1,N
   50 X(J)=XN(J)+COEF(J)
      TF=TFN + SN
      IHALV=0
51    IF (INORM.EQ.1) ITF=1
      ITR= 0
      CALL FUNCT
      ITF=3
      KOUNT=KOUNT+1
      F1=0.D0
      DO 52 I=1,M
   52 F1=F1+Y(I)**2
      WRITE(6,1010)F1
      IF(F1.LT.F0)GO TO 55
      WRITE(6,1008)
      IF (IHALV.EQ.10) GO TO 95
      IHALV=IHALV+1
      DO 53 J=1,N
      COEF(J)=COEF(J)/2.D0
      WRITE(6,1000)COEF(J)
   53 X(J)=XN(J)+COEF(J)
      SN= SN/2.D0
      WRITE (6,1012) SN
      TF= TFN + SN
      GO TO 51
   55 IF(NI-MAXNOI)70,70,80
   70 NI=NI+1
      F0=F1
      GO TO 10
   80 NI=9999
      WRITE(6,1006)
      RETURN
   85 NI=9999
      WRITE(6,1007)
      RETURN
   90 WRITE(6,1005)F0
      RETURN
```

```
   95 NI=9999
      WRITE(6,1009)
      RETURN
 1000 FORMAT(/1X,1PD23.15)
 1001 FORMAT(1X,1P5D23.15)
 1002 FORMAT(21HOPARTIAL DERIV MATRIX)
 1003 FORMAT(11HODELX:S ARE/(1X,1PD23.15))
 1005 FORMAT(4HOFO=,1PD22.15,23HCASE CONVERGED...FERTIG)
 1006 FORMAT(38HOEXCEEDED MAXIMUM NUMBER OF ITERATIONS)
 1007 FORMAT(16HOMATRIX SINGULAR)
 1008 FORMAT(11HODELX:S ARE)
 1009 FORMAT(19HOMETHOD CANNOT WORK)
 1010 FORMAT(4HOF1=,1PD23.15)
 1011 FORMAT(4HOFO=,1PD23.15)
 1012 FORMAT (10HO DEL IF =,1PD23.15)
 1013 FORMAT (40HO X(I)+DX(I) FOLLOWED BY CORRESPONDING Y)
 1014 FORMAT (24HO X(I)=0. SO DX(I)=XS(I))
      END
```

```
C TRAJ/TRAJPCS
C
C SEP AND HIGH THRUST
C 7/14 DIM. WITH POWER DEGRADATION.
C THIS ROUTINE SETS UP THE INPUT TO THE INTEGRATOR WHICH
C     EXTRAPULATES THE TRAJECTORY FROM INITIAL TIME TO
C     FINAL TIME.  IT ALSO EVALUATES THE CHANGE IN TF AND
C     THE ERROR IN THE FINAL CONDITIONS.
C THIS PROGRAM IS CALLED BY ITER OR BY CONTROL
C IT CALL THE SUBPROGRAM DHPCG (PRED-CORR)
C MIN J, MAX H.
C P DIM. ZERF. T.C. OPTIONS.
C NOP=1--ALL 5 FINAL O.E. FIXED, =2--A,E,I ONLY FIXED.
C
C
      SUBROUTINE TRAJ
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      COMMON /XMMM/ZLO(7), STEP(8), ZERF(8)
      COMMON /TRA/TFMAX, DTO, UEB, EW(14)
      COMMON /Z/Z(14), DERZ(14)
      COMMON /INT/ITF, IPR, IDIM, IDIM2, NIMAX
      COMMON /T/TF, SD, TO, TFMIN
      COMMON /ELEM/ZPO(7), ZPF(5)
      COMMON /DY/DYDT(8)
      COMMON /TC/NOP
      COMMON /HIGH/DVI1,DVI2,IHI
      COMMON /A/A,AMU,PI
      COMMON /NOD/NODE
C
      EXTERNAL FUNCT, OUTP
      DIMENSION PRMT(5), AUX(16,14),DERZ1(14),DP(5)
C
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) GO TO 9
C
C HI  THRUST
C
      UP= (PI/2.DO)*ZLO(1)/DSQRT(1.DO+ZLO(1)**2)
      CUP= DCOS(UP)
      XK= CUP*(.75DO+.25DO*ZLO(2)/DSQRT(1.DO+ZLO(2)**2))
      DUM= (1.DO+CUP*XK)*DSQRT((CUP-XK)/(CUP+XK))
      XJ= DUM*ZLO(3)/DSQRT(1.DO+ZLO(3)**2)
      JM= 2
      VAR= ZPO(3)
      IF (NODE.EQ.0) VAR= ZLO(5)
      CALL MAINE(0.DO,0.DO,XK,UP,XJ,1.DO,1,JM,DP,DVI1)
      CALL OUTHI(JM,PI,ZPO(1),ZPO(2),VAR,IPR,Z,IDIM2)
      IF (NODE.EQ.0) GO TO 4
      Z(IDIM2+4)= Z(IDIM2+4)+ZLO(5)*Z(5)
      Z(IDIM2+5)= Z(IDIM2+5)-ZLO(5)*Z(4)
 4    DO 5 I= 1,5
 5    Z(I+IDIM2)= ZLO(4)*Z(I+IDIM2)*1.D4
      DO 6 I= 6,7
      Z(I)= ZPO(I)
 6    Z(I+IDIM2)= ZLO(I)
C
```

```
C LOW THRUST
C
 9    DT= (TF-TO)/(IDINT((TF-TO)/DTO)+1.DO)
      PRMT(1)= TO
      PRMT(2)= TF
      PRMT(3)= DT
      PRMT(4)= UEB
C
      IF ((IHI.EQ.2).OR.(IHI.EQ.3)) GO TO 15
C
C Z IS A VECTOR OF STATE AND COSTATE
      DO 10 I=1,IDIM2
       Z(I)=ZPO(I)
 10    Z(I+IDIM2)= ZLO(I)
C
C EW ARE ERROR WEIGHTS--INPUT TO THE INTEGRATOR
C
 15    DO 20 I=1,IDIM
 20    DERZ(I)=EW(I)
C
C CALL THE P-C INTEGRATOR
C
      CALL DHPCG(PRMT,Z,DERZ,IDIM,IHLF,FUNCT,OUTP,AUX)
      IF (IHLF.GT.10) GO TO 100
C
C Z IS NOW THE FINAL O.E..
C     ZERF THE ERROR IN THE FINAL CONDITIONS
C
      H=0.DO
      DO 30 I=1,IDIM2
 30    H= H + Z(I+7)*DERZ(I)
      ZERF(6)= Z(13)*1.D-3
      ZERF(7)= Z(14)*1.D-3
      DYDT(6)= DERZ(13)*1.D-3
      DYDT(7)= DERZ(14)*1.D-3
      TF1=TF*(STEP(8)+1.DO)
      CALL FUNCT(TF1,Z,DERZ1)
      H1=0.DO
      DO 35 I=1,IDIM2
 35    H1=H1+Z(I+7)*DERZ1(I)
      DYDT(8)= (H1-H)/(TF1-TF)
      ZERF(8)= H -1.DO
C
C FINAL CONDITION OPTION BRANCH
C
      GO TO (40,50), NOP
C
 40   DO 45 I=1,5
       ZERF(I)= Z(I) -ZPF(I)
 45    DYDT(I)= DERZ(I)
      RETURN
C
 50   ZERF(4)= (Z(3)*Z(9)-Z(2)*Z(10))*1.D-3
      ZERF(5)= (Z(5)*Z(11)-Z(4)*Z(12))*1.D-3
      DYDT(4)= DERZ(3)*Z(9)+Z(3)*DERZ(9)-DERZ(2)*Z(10)-Z(2)*DERZ(10)
      DYDT(4)= DYDT(4)*1.D-3
      DYDT(5)= DERZ(5)*Z(11)+Z(5)*DERZ(11)-DERZ(4)*Z(12)-Z(4)*DERZ(12)
```

```
      DYDT(5)= DYDT(5)*1.D-3
      IF (IHI.LT.3) GO TO 60
C
C FINAL HIGH THRUST IMPULSE
C
      CALL IMPLS(DVI2,IPR,0,Z,DERZ,IDIM2)
C
 60   ZERF(1)= Z(1) - ZPF(1)
      DUM1= DSQRT(Z(2)**2 + Z(3)**2)
      ZERF(2)= DUM1 - ZPF(2)
      DUM2= DSQRT(Z(4)**2 + Z(5)**2)
      ZERF(3)= DUM2 - ZPF(3)
      DYDT(1)= DERZ(1)
      DYDT(2)= 0.D0
      DYDT(3)= 0.D0
      IF (DUM1.GT.1.D-12) DYDT(2)= (Z(2)*DERZ(2) + Z(3)*DERZ(3))/DUM1
      IF (DUM2.GT.1.D-12) DYDT(3)= (Z(4)*DERZ(4) + Z(5)*DERZ(5))/DUM2
C
C SPECIAL CASE, E=0 AND/OR I=0
C
      IF (ZPF(2).NE.0.D0) GO TO 70
      ZERF(2)= Z(2)
      ZERF(4)= Z(3)
      DYDT(2)= DERZ(2)
      DYDT(4)= DERZ(3)
 70   IF (ZPF(3).NE.0.D0) RETURN
      ZERF(3)= Z(4)
      ZERF(5)= Z(5)
      DYDT(3)= DERZ(4)
      DYDT(5)= DERZ(5)
      RETURN
C
 100  IF (IHLF.EQ.11) WRITE (6,1000)
      IF (IHLF.EQ.12) WRITE (6,1001)
      IF (IHLF.EQ.13) WRITE (6,1002)
      STOP
C
 1000 FORMAT (68H0 THE NUMBER OF BISECTIONS OF THE ORIGINAL INCREMENT HA
     1S EXCEEDED 10)
 1001 FORMAT (27H0 INITIAL INCREMENT IS ZERO)
 1002 FORMAT (54H0 INITIAL INCREMENT HAS WRONG SIGN OR BOUNDS ARE WRONG)
      END
```

```
C TRAJ/TRAJPCN
C
C NEP AND HIGH THRUST.
C THIS ROUTINE SETS UP THE INPUT TO THE INTEGRATOR WHICH
C    EXTRAPOLATES THE TRAJECTORY FROM INITIAL TIME TO
C    FINAL TIME.  IT ALSO EVALUATES THE CHANGE IN TF AND
C    THE ERROR IN THE FINAL CONDITIONS.
C THIS PROGRAM IS CALLED BY ITER OR BY CONTROL
C IT CALLS THE SUBPROGRAM DHPCG  (PREDICTOR CORRECTOR)
C MIN J, MAX H.
C 6 DIM. ZERF. T.C. OPTIONS.
C NOP=1--ALL 5 FINAL O.E. FIXED, =2--A,E,I ONLY FIXED.
C DYDT USED ONLY IF ITR= 0
C IHI= 1 NO HIGH THRUST
C    = 2 2 INITIAL IMPULSES
C    = 3 2 INITIAL AND ONE FINAL IMPULSE
C    = 4 FINAL IMPULSE
C WHEN IHI=2 OR 3, ZLO(5) AND ZERF(5) ARE DUMMIES.
C
C
      SUBROUTINE TRAJ
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      COMMON /XMMM/ZLO(5), DUMMY(5), ZERF(6)
      COMMON /ERA/TFMAX, DTO, UER, EW(10)
      COMMON /Z/Z(10), DERZ(10)
      COMMON /INT/ITF, IPR, IDIM, IDIM2, NIMAX
      COMMON /T/TF, SD, TO, TFMIN
      COMMON /ELEM/ZPO(5), ZPF(5)
      COMMON /DY/DYDT(6)
      COMMON /TC/NOP
      COMMON /POWER/PKW,CC,AMO,A0,A1,A2,A3,IPOW
      COMMON /A/A,AMU,PI
      COMMON /HIGH/DVI1,DVI2,IHI,ITR
      COMMON /F/FLIM,KSTEP
C
      EXTERNAL FUNCT, OUTP
      DIMENSION PRMT(5), AUX(16,10), DP(5), ZS(6)
C
      IF ((IHI.EQ.1).OR.(IHI.EQ.4)) GO TO 9
      IF (ITR.NE.1) GO TO 4
      DO 3 I=1,6
 3    ZS(I)=ZERF(I)
 4    IF (ITR.EQ.4) GO TO 70
      IF (ITR.EQ.5) GO TO 80
C
C HIGH THRUST
C
      UP= (PI/2.D0)*ZLO(1)/DSQRT(1.D0+ZLO(1)**2)
      CUP= DCOS(UP)
      XK= CUP*(.75D0+.25D0*ZLO(2)/DSQRT(1.D0+ZLO(2)**2))
      DUM= (1.D0+CUP*XK)*DSQRT((CUP-XK)/(CUP+XK))
      XJ= DUM*ZLO(3)/DSQRT(1.D0+ZLO(3)**2)
      JM= 2
      CALL MAINE(0.D0,0.D0,XK,UP,XJ,1.D0,1,JM,DP,DVI1)
      CALL OUTHI(JM,PI,ZPO(1),ZPO(2),0.D0,IPR,Z,IDIM2)
```

```
      DO 5 I= 1,IDIM2
 5     Z(I+IDIM2)= ZLO(4)*Z(I+IDIM2)*1.D4
C
C NEP
C
 9    DT= (TF-TO)/(IDINT((TF-TO)/DTO)+1.DO)
      PRMT(1)= TO
      PRMT(2)= TF
      PRMT(3)= DT
      PRMT(4)= UEB
C
C Z IS A VECTOR OF STATE AND COSTATE
C
      IF ((IHI.EQ.2).OR.(IHI.EQ.3)) GO TO 15
      DO 10 I=1,IDIM2
       Z(I)=ZPO(I)
 10    Z(I+IDIM2)= ZLO(I)
C
C EW ARE ERROR WEIGHTS--INPUT TO THE INTEGRATOR
C
 15   DO 20 I=1,IDIM
 20    DERZ(I)=EW(I)
C
C FOR ITF LT 3 USE NORM CUTOFF CONDITION
C
      IF (ITF.LT.3) TF=TFMAX
C
C CALL THE PREDICTOR-CORRECTOR INTEGRATOR
C
      CALL DHPCG(PRMT,Z,DERZ,IDIM,IHLF,FUNCT,OUTP,AUX)
      IF (IHLF.GT.10) GO TO 100
C
C Z IS NOW THE FINAL O.E.,
C     ZERF THE ERROR IN THE FINAL CONDITIONS
C
      H=0.DO
      DO 30 I=1,IDIM2
 30    H= H + Z(I+5)*DERZ(I)
      ZERF(6)= H -1.DO
      IF (IHI.LT.3) GO TO 35
C
C FINAL IMPULSE
C
      PSI= (Z(3)*Z(7)-Z(2)*Z(8))*1.D-3
      PSIDOT= DERZ(3)*Z(7)+Z(3)*DERZ(7)-Z(2)*DERZ(8)-DERZ(2)*Z(8)
      PSIDOT= PSIDOT*1.D-3
C
      CALL IMPLS(DVI2,IPR,ITR,Z,DERZ,IDIM2)
C
C FINAL CONDITION OPTION BRANCH
C
 35   GO TO (40,50), NOP
C
 40   DO 45 I=1,IDIM2
       ZERF(I)= Z(I) -ZPF(I)
 45    DVDT(I)= DERZ(I)
      GO TO 55
```

```
C
 50   ZERF(1)= Z(1) - ZPF(1)
      DUM1= DSQRT(Z(2)**2 + Z(3)**2)
      ZERF(2)= DUM1 - ZPF(2)
      DUM2= DSQRT(Z(4)**2 + Z(5)**2)
      ZERF(3)= DUM2 - ZPF(3)
      ZERF(4)= (Z(3)*Z(7) - Z(2)*Z(8))*1.D-3
      ZERF(5)= (Z(5)*Z(9) - Z(4)*Z(10))*1.D-3
      DYDT(1)= DERZ(1)
      DYDT(2)= 0.D0
      DYDT(3)= 0.D0
      IF (DUM1.GT.1.D-40) DYDT(2)= (Z(2)*DERZ(2) + Z(3)*DERZ(3))/DUM1
      IF (DUM2.GT.1.D-40) DYDT(3)= (Z(4)*DERZ(4) + Z(5)*DERZ(5))/DUM2
      DYDT(4)= DERZ(3)*Z(7)+Z(3)*DERZ(7)-DERZ(2)*Z(8)-Z(2)*DERZ(8)
      DYDT(4)=DYDT(4)*1.D-3
      DYDT(5)= DERZ(5)*Z(9)+Z(5)*DERZ(9)-DERZ(4)*Z(10)-Z(4)*DERZ(10)
      DYDT(5)=DYDT(5)*1.D-3
      IF (IHI.EQ.1) GO TO 55
      IF (IHI.EQ.4) GO TO 52
C
C DUMMY
C
      ZERF(5)= ZLO(5)-1.D0
      DYDT(5)= 0.D0
      IF (IHI.LT.3) GO TO 53
C
 52   ZERF(4)= PSI
      DYDT(4)= PSIDOT
 53   IF (ITR.GT.0)  RETURN
C
      DO 54 I= 1,6
 54    ZS(I)= ZERF(I)
C
 55   IF (IPOW.EQ.1) GO TO 60
      A4= 1.D0-A1*TF
      DYDT(6)= A1*H/A4
      RETURN
C
 60   B1= DEXP(-A2*TF)
      A4= 1.D0+A1*(B1-1.D0+A2*A3*TF)
      DYDT(6)= A2*H*(-B1/(B1-A3)+A1*(B1-A3)/A4)
      RETURN
C
C
C
 70   DO 75 I= 1,3
 75    ZERF(I)= ZS(I)
      DUM= DUMMY(4)
      IF (KSTEP.EQ.0) DUM= DUM*ZLO(4)
      ZERF(4)= (1.D0+DUM)*ZS(4)
      ZERF(5)= ZS(5)
      ZERF(6)= (1.D0+DUM)*(ZS(6)+1.D0)-1.D0
      RETURN
C
 80   DO 85 I= 1,6
 85    ZERF(I)= ZS(I)
      ZERF(5)= ZS(5)+DUMMY(5)
```

```
      RETURN
C
 100  IF (IHLF.EQ.11) WRITE (6,1000)
      IF (IHLF.EQ.12) WRITE (6,1001)
      IF (IHLF.EQ.13) WRITE (6,1002)
      STOP
C
 1000 FORMAT (68HO THE NUMBER OF BISECTIONS OF THE ORIGINAL INCREMENT HA
     1S EXCEEDED 10)
 1001 FORMAT (27HO INITIAL INCREMENT IS ZERO)
 1002 FORMAT (54HO INITIAL INCREMENT HAS WRONG SIGN OR BOUNDS ARE WRONG)
      END
```

```
C QUAD
C
C THIS IS A MODIFIED QUADRATURE PROGRAM FOR VECTOR VALUED FUNCTIONS.
C  COMPUTES INTEGRAL OF THE FUNCTION G (OR H) OVER X FROM XL TO XU.  THE
C RESULT IS Y.   ROUTINE USES A 4 POINT GAUSS QUADRATURE.
C
C
C
      SUBROUTINE QUAD(XL,XU,FCT,Y,Z,G,H,N)
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
      DIMENSION Y(1),H(1),G(1),Z(1)
C
      A= .5D0*(XU+XL)
      B= XU-XL
      C= .43056815579702629D0*B
      K=1
      GO TO 50
 10   DO 20 I=1,N
 20    Y(I)= .17392742256872693D0*G(I)
      C= .16999052179242813D0*B
      K=2
      GO TO 50
 30   DO 40 I=1,N
 40    Y(I)= B*(Y(I) + .32607257743127307D0*G(I))
      RETURN
 50   CALL FCT(A-C,A+C,Z,H,G)
      DO 60 I=1,N
 60    G(I)=G(I) + H(I)
      GO TO (10,30), K
      END
```

```
C QUAD/QUAD8
C
C THIS IS A MODIFIED QUADRATURE PROGRAM FOR VECTOR VALUED FUNCTIONS.
C COMPUTES THE INTEGRAL OF THE FUNCTION G (+ H) OVER X FROM XL TO XU.
C THE RESULT IS Y.  ROUTINE USES A 8 POINT GAUSS QUADRATURE.
C
C
C
      SUBROUTINE QUAD(XL,XU,FCT,Y,Z,G,H,N)
C
      IMPLICIT REAL*8(A-H,O-$)
      DIMENSION Y(1),H(1),G(1),Z(1)
C
      A=.5D0*(XU+XL)
      B=XU-XL
      C=.4801449282487681D0*B
      K=1
      GO TO 500
 10   DO 20 I=1,N
 20    Y(I)=.5061426814518813D-1*G(I)
      C=.3983332387068134D0*B
      K=2
      GO TO 500
 30   DO 40 I=1,N
 40    Y(I)=Y(I)+.1111905172266872D0*G(I)
      C=.2627662049581644D0*B
      K=3
      GO TO 500
 50   DO 60 I=1,N
 60    Y(I)=Y(I)+.1568533229389436D0*G(I)
      C=.9171732124782490D-1*B
      K=4
      GO TO 500
 70   DO 80 I=1,N
 80   Y(I)=B*(Y(I)+.1813418916891809D0*G(I))
      RETURN
 500  CALL FCT(A-C,A+C,Z,H,G)
      DO 510 I=1,N
 510   G(I)=G(I) + H(I)
      GO TO (10,30,50,70),K
      END
```

```
C QUAD32
C
C THIS IS A MODIFIED QUADRATURE INTEGRATION PROGRAM FOR
C VECTOR VALUED FUNCTIONS OF ONE VARIABLE. IT INTEGRATES
C G (OR H) OVER X FROM XL TO XU.  THE RESULT IS Y.
C EVALUATION IS BY A 32 POINT GAUSS QUADRATURE FORMULA.
C
      SUBROUTINE QUAD(XL,XU,FCT,Y,Z,G,H,N)
C
      IMPLICIT REAL*8(A-H,O-$)
      DIMENSION Y(1),Z(1),G(1),H(1)
C
      A= .5D0*(XU-XL)
      B= XU-XL
      C= .49863193092474078D0*B
      K=1
      GO TO 500
 10   DO 20 I=1,N
 20    Y(I)= .35093050047350483D-2*G(I)
      C= .49280575577263417D0*B
      K=2
      GO TO 500
 30   DO 40 I=1,N
 40    Y(I)= Y(I) + .81371973654528350D-2*G(I)
      C= .48238112779375322D0*B
      K=3
      GO TO 500
 50   DO 60 I=1,N
 60    Y(I)= Y(I) + .12696032654631030D-1*G(I)
      C= .46745303796886984D0*B
      K=4
      GO TO 500
 70   DO 80 I=1,N
 80    Y(I)= Y(I) + .17136931456510717D-1*G(I)
      C= .44816057788302606D0*B
      K=5
      GO TO 500
 90   DO 100 I=1,N
 100  Y(I)= Y(I) + .21417949011113340D-1*G(I)
      C= .42468380686628499D0*B
      K=6
      GO TO 500
 110  DO 120 I=1,N
 120   Y(I)= Y(I) + .25499029631188088D-1*G(I)
      C= .39724189798397120D0*B
      K=7
      GO TO 500
 130  DO 140 I=1,N
 140   Y(I)= Y(I) + .29342046739267774D-1*G(I)
      C= .36609105937014484D0*B
      K=8
      GO TO 500
 150  DO 160 I=1,N
 160   Y(I)= Y(I) + .32911113881809230-1*G(I)
      C= .33152213346510760D0*B
      K=9
      GO TO 500
```

```
170  DO 180 I=1,N
180   Y(I)= Y(I) + .36172897054424253D-1*G(I)
     C= .29385787862038116D0*B
     K=10
     GO TO 500
190  DO 200 I=1,N
200   Y(I)= Y(I) + .39096947893535153D-1*G(I)
     C= .25344995446611470D0*B
     K=11
     GO TO 500
210  DO 220 I=1,N
220  Y(I)= Y(I) + .41655962113473378D-1*G(I)
     C= .21067563806531767D0*B
     K=12
     GO TO 500
230  DO 240 I=1,N
240   Y(I)= Y(I) + .43826046502201906D-1*G(I)
     C= .16593430114106382D0*B
     K=13
     GO TO 500
250  DO 260 I=1,N
260   Y(I)= Y(I) + .45586939347881942D-1*G(I)
     C= .11964368112606854D0*B
     K=14
     GO TO 500
270  DO 280 I=1,N
280   Y(I)= Y(I) + .46922199540402283D-1*G(I)
     C= .72235980791398250D-1*B
     K=15
     GO TO 500
290  DO 300 I=1,N
300   Y(I)= Y(I) + .47819360039637430D-1*G(I)
     C= .24153832843869158D-1*B
     K=16
     GO TO 500
310  DO 320 I=1,N
320   Y(I)= B*(Y(I) + .48270044257363900D-1*G(I))
     RETURN
500  CALL FCT(A-C,A+C,Z,H,G)
     DO 510 I=1,N
510   G(I)= G(I) + H(I)
     GO TO (10,30,50,70,90,110,130,150,170,190,210,230,250,270,290,310)
    1    ,K
     END
```

```
C EARTH2
C
C
C THIS SUBPROGRAM SETS THE VALUES FOR EARTH'S ORBITAL ELEMENTS
C AND CALCULATES THE MEAN ANOMALY AT THE INITIAL TIME
C O.E. TAKEN FROM BATTIN, 1964. EPOCH 1960 JAN. 1.5, JD=2436935.
C INPUT
C      TL--INITIAL TIME. BEGINNING OF LOW THRUST TRAJ
C OUTPUT
C      C(1)--EARTH'S SEMIMAJOR AXIS
C      C(2)--EARTH'S ECCENTRICITY
C      C(3)--ARGUMENT OF PERIH.
C      C(4)--MEAN ORBITAL MOTION
C      C(5)--MEAN ANOMALY AT TL
C      C(6)--COSINE OF ANGLE OF OBLIQUITY
C      C(7)--SINE OF ANGLE OF OBLIQUITY
C      C(17)--LONG. OF N MAG POLE AT TIME TL
C      C(18)--LAT. OF N. MAG. POLE
C      C(19)--EARTH ROT. FREQ.
C      C(20)--2*PI
C
C
C
      SUBROUTINE EARTH
C
      IMPLICIT REAL*8(A-H,O-$)
C
      COMMON /JD/ TL
      COMMON /TERRA/ C(20)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
C
C
      C(1) = 1.D0
      C(2)= .016726D0
      C(3)= 102.25253D0
      C(4)= .985609D0
C
C MEAN ANOMALY AT EPOCH
      AN= 100.15815D0-C(3)
      B0= AN
C MEAN ANOMALY AT TIME TL
      AN= AN+C(4)*(TL-2436935.D0)
      AN=AN/360.D0
      AN=AN-IDINT(AN)
C
      C(5)= AN*360.D0*DTR
      C(3)= C(3)*DTR
      C(4)= C(4)*DTR*UTD
C
      DUM= 23.45*DTR
      C(6)= DCOS(DUM)
      C(7)=DSIN(DUM)
C
C CALCULATE THE ROTATION MATRIX FROM EQUATORIAL TO GMT COORD.
C
      B0= B0+(2.D0*C(2)-.25D0*C(2)**3,*DSIN(B0)+1.25D0*C(2)**2
     1   *DSIN(2.D0*B0)+1.0833333333333D0*C(2)**3*DSIN(3.D0*B0)
```

```
      BO= BO+C(3)
      RX= -DCOS(BO)
      RY= -DSIN(BO)*C(6)
      RHO= DSQRT(RX**2+RY**2)
      RX= RX/RHO
      RY= RY/RHO
      C(8)= RX
      C(9)= -RY
      C(10)= 0.D0
      C(11)= RY
      C(12)= RX
      C(13)= 0.D0
      C(14)= 0.D0
      C(15)= 0.D0
      C(16)= 1.D0
C
      C(17)= 289.9D0*DTR
      C(18)= 78.6*DTR
      C(19)= 359.0170416D0*DTR*UTD
      C(20)= 360.D0*DTR
      C(17)= C(17)+C(19)*(TL-2436935.D0)/UTD
C
      RETURN
      END
```

```
C ORBIT
C
C
C THIS PROGRAM IS USED TO PRINT POSITION AND THRUST DIRECTION
C FOR A SINGLE ORBIT.
C INPUT: Z, 10 DIM O.E. AND THEIR ADJOINTS
C         A, ACCELERATION LEVEL, MAY BE SET TO ONE
C         N, ONE HALF NUMBER OF POINTS ON ORBIT PRINTED
C         NCONT, FLAG TO READ DATA FOR ANOTHER ORBIT
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Z(10),H(10),G(10)
      COMMON/A/A,AMU,PI
      NAMELIST/IN/Z,A,N
C
      PI= DARCOS(-1.D0)
      AMU=1.D0
 10   READ (5,1000) Z,A
      READ (5,1001) N
      DF= PI/DFLOAT(N)
      WRITE (6,IN)
      DO 20 I=1,N
      CALL FCT(F1,F2,Z,H,G)
      F1=F1+DF*2.D0
      F2=F2+DF*2.D0
 20   CONTINUE
      READ (5,1001) NCONT
      IF (NCONT.GT.0) GO TO 10
      STOP
 1000 FORMAT (F25.15)
 1001 FORMAT (I2)
      END
```

```
C FCT
C
C  CONTAINS PRINT STATEMENTS
C
C
C
      SUBROUTINE FCT(F1,F2,Z,H,G)
C
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      NAMELIST/PRFCT/VEC
C
      COMMON /A/A,AMU,PI
C
      DIMENSION Z(10),G(10),H(10),AM(5,3),PAM(5,3,5),VEC(3),X(5),PRA(5)
C
      M=0
      F=F1
C
C EVALUATE M AND PARTIAL OF M WRT STATE
C
      DO 5 I=1,5
 5     X(I)=Z(I)
 10   CALL EVALMP(X,F,AMU,AM,PAM,1)
C
C EVALUATE THE COMMON SCALOR FACTOR
C
      CT=DCOS(F)
      ST=DSIN(F)
      RA= 1.0D0-Z(3)*CT -Z(2)*ST
      FAC= A*RA/(2.0D0*PI)
C
C EVALUATE THE UNIT OF M TRANSPOSE LAMBDA
C
      ABVEC= 0.D0
      DO 30 I=1,3
       VEC(I)=0.0D0
       DO 20 J=1,5
 20     VEC(I)= VEC(I) + AM(J,I)*Z(J+5)
 30    ABVEC= ABVEC + VEC(I)**2
      ABVEC= DSQRT(ABVEC)
      DO 40 I=1,3
 40    VEC(I)=VEC(I)/ABVEC
C
      WRITE (6,PRFCT)
C
C
C EVALUATE THE PARTIAL OF RA WRT X
C
      PRA(1)=0.D0
      PRA(2)=-ST
      PRA(3)=-CT
      PRA(4)=0.D0
      PRA(5)=0.D0
      ABVEC=ABVEC/RA
C
C EVALUATE THE FUNCTION
```

```
C
      DO 60 I=1,5
       G(I)= 0.0D0
       G(I+5)=0.0D0
       DO 50 J=1,3
        G(I)= G(I) + AM(I,J)*VEC(J)
        DO 50 L=1,5
 50      G(I+5)= G(I+5) - [illegible]L+5)*PAM(L,J,I)*VEC(J)
       G(I)= FAC*G(I)
 60    G(I+5)=FAC*(G(I+5)-ARVEC*PRA(I))
      IF (M.EQ.1) RETURN
      DO 70 I=1,10
 70    H(I)= G(I)
      F=F2
      M=1
      GO TO 10
      END
```

```
C      SUBROUTINE EVALMP
C   THIS SUBROUTINE EVALUATES THE 5X3 MATRIX M AND THE
C   5X3X5 PARTIAL OF M WRT X
C   CONTAINS PRINT STATEMENTS
C
C   IF IMFLAG=1, BOTH M (AM) AND ITS PARTIAL (PAM) ARE EVALUATED
C   IF IMFLAG=2, ONLY M (AM) IS EVALUATED
C   IF IMFLAG=3, ONLY THE PARTIAL OF M (PAM) IS EVALUATED
C
C
      SUBROUTINE EVALMP(X, THETA, AMU, AM, PAM, IMFLAG)
      IMPLICIT REAL*8(A-H,O-$), INTEGER (I-N)
C
      NAMELIST/PRINT/ F,X1,Y1
C
      DIMENSION X(5), AM(5,3), PAM(5,3,5)
      EN=DSQRT(AMU/X(1)**3)
      RHO= DSQRT(1.D0- X(2)**2- X(3)**2)
      BETA= 1.D0/(1.D0 +RHO)
      CT= DCOS(THETA)
      ST= DSIN(THETA)
      RA= 1.D0-X(3)*CT -X(2)*ST
      ZETA= X(3)*ST-X(2)*CT
      BETA3= BETA**3/(1.D0 -BETA)
      X1= X(1)*((1.D0 -X(2)**2*BETA)*CT +X(2)*X(3)*BETA*ST -X(3))
      Y1= X(1)*((1.D0 -X(3)**2*BETA)*ST +X(2)*X(3)*BETA*CT -X(2))
      X1DOT= -((1.D0 -X(2)**2*BETA)*ST -X(2)*X(3)*BETA*CT)*EN*X(1)/RA
      Y1DOT= ((1.D0 -X(3)**2*BETA)*CT -X(2)*X(3)*BETA*ST)*EN*X(1)/RA
      PZ1= X(1)*(ZETA*(BETA+X(2)**2*BETA3) -(X(2)*BETA -ST)*CT/RA)
      PZ2= -X(1)*(-ZETA*X(2)*X(3)*BETA3 +1.D0 +(ST -X(2)*BETA)*ST/RA)
      PZ3= X(1)*(-ZETA*X(2)*X(3)*BETA3-1.D0 +(X(3)*BETA -CT)*CT/RA)
      PZ4= X(1)*(-ZETA*(BETA +X(3)**2*BETA3) +(CT -X(3)*BETA)*ST/RA)
      IF (IMFLAG .EQ. 3) GO TO 10
C
      F=THETA
      WRITE (6,PRINT)
C
C   IF DO NOT WANT TO EVALUATE PARTIAL OF M, BRANCH TO 10
      AM(1,1)= 2.D0*X1DOT/(EN**2*X(1))
      AM(1,2)= 2.D0*Y1DOT/(EN**2*X(1))
      AM(1,3)=0.D0
       DUM= RHO/(EN*X(1)**2)
      AM(2,1)= DUM*(PZ2- X(2)*BETA*X1DOT/EN)
      AM(2,2)= DUM*(PZ4 -X(2)*BETA*Y1DOT/EN)
      AM(2,3)= DUM*(X(3)*(X(5)*Y1 -X(4)*X1))/RHO**2
      AM(3,1)= -DUM*(PZ1 +X(3)*BETA*X1DOT/EN)
      AM(3,2)= -DUM*(PZ3 +X(3)*BETA*Y1DOT/EN)
      AM(3,3)= -DUM*(X(2)*(X(5)*Y1 -X(4)*X1)/RHO**2)
      AM(4,1)=0.D0
      AM(4,2)=0.D0
      DUM= (1.D0 +X(4)**2 +X(5)**2)/(2.D0*EN*X(1)**2*RHO)
      AM(4,3)= DUM*Y1
      AM(5,1)=0.D0
      AM(5,2)=0.D0
      AM(5,3)= DUM*X1
      IF (IMFLAG .EQ. 2) RETURN
C   IF WE ONLY WISH TO EVALUATE M THEN PROGRAM RETURNS HERE
```

```
10    CA= DSQRT(AMU/X(1))/RA
      PZ5= X(2)*BETA3
      PZ6= X(3)*BETA3
      PZ9= CA*ST/RA
      PZ10= CA*CT/RA
      PZ20= X(1)*(-2.D0*X(2)*BETA*CT +X(3)*BETA*ST +PZ5*ZETA*X(2))
      PZ26= X(1)*(X(2)*BETA*ST -1.D0 +PZ6*X(2)*ZETA)
      PZ29= X(1)*(X(3)*BETA*CT -1.D0 -PZ5*X(3)*ZETA)
      PZ35= X(1)*(-2.D0*X(3)*BETA*ST +X(2)*BETA*CT -PZ6*X(3)*ZETA)
      PZ11= -X1DOT/(2.D0*X(1))
      PZ12= -Y1DOT/(2.D0*X(1))
      DUM1= 1.D0 -RA
      PZ13= -CA*(-2.D0*X(2)*BETA*ST -X(3)*BETA*CT -PZ5*X(2)*DUM1)+PZ9
     1                        *X1DOT/CA
      PZ14= -CA*(-X(2)*BETA*CT -PZ6*X(2)*DUM1) +PZ10*X1DOT/CA
      PZ15= -CA*(X(3)*BETA*ST +PZ5*X(3)*DUM1) +PZ9*Y1DOT/CA
      PZ16= -CA*(2.D0*X(3)*BETA*CT +X(2)*BETA*ST +PZ6*DUM1*X(3))
     1         +PZ10*Y1DOT/CA
      DUM= BETA +X(2) *PZ5
      PZ17= 1.D0+ PZ5*X(2)*(3.D0/BETA +1.D0/(1.D0-BETA))
      PZ18= (2.D0 +PZ17)*PZ5
      PZ19= PZ17*PZ6
      DUM2= X(2)*BETA -ST
      PZ21= -X(1)*(CT*DUM -ZETA*PZ18 +CT*DUM/RA +CT*ST*DUM2/RA**2)
      PZ22= X(1)*(ST*DUM +ZETA*PZ19 -CT*X(2)*PZ6/RA-CT**2*DUM2/RA**2)
      PZ23= BETA3*(3.D0/BETA +1.D0/(1.D0 -BETA))
      PZ24= PZ23*PZ5
      PZ25=PZ23*PZ6
      PZ27= X(1)*(-CT*X(2)*X(3)*BETA3 +ZETA*X(3)*(BETA3 +X(2)*PZ24)
     1       +(ST*(BETA +X(2)*PZ5))/RA +ST**2*DUM2/RA**2)
      PZ28= X(1)*(ST*X(2)*X(3)*BETA3 +ZETA*X(2)*(BETA3 +X(3)*PZ25)
     1       +X(2)*ST*PZ6/RA +ST*CT*DUM2/RA**2)
      DUM2= X(3)*BETA-CT
      PZ30= X(1)*(CT*X(2)*X(3)*BETA3 -ZETA*X(3)*(BETA3 +X(2)*PZ24)
     1     +CT*X(3)*PZ5/RA +CT*ST*DUM2/RA**2)
      PZ31= X(1)*(-ST*X(2)*X(3)*BETA3 -ZETA*X(2)*(BETA3 +X(3)*PZ25)
     1       +CT*(BETA +X(3)*PZ6)/RA +CT**2*DUM2/RA**2)
      DUM= BETA +X(3)*PZ6
      PZ32= 1.D0 +PZ6*X(3)*(3.D0/BETA +1.D0/(1.D0 -BETA))
      PZ33= PZ32*PZ5
      PZ34= PZ32*PZ6 +2.D0*X(3)*BETA3
      PZ36= X(1)*(CT*DUM -ZETA*PZ33 -ST*X(3)*PZ5/RA -ST**2*DUM2/RA**2)
      PZ37= X(1)*(-ST*DUM -ZETA*PZ34 -ST*(BETA +X(3)*PZ6)/RA -ST*CT
     1         *DUM2/RA**2)
      DO 20 J=1,2
20    PAM(1,J,1)= 3.D0*AM(1,J)/(2.D0*X(1))
      DUM =2.D0*X(1)**2/AMU
      PAM(1,1,2)= PZ13*DUM
      PAM(1,1,3)= PZ14*DUM
      PAM(1,2,2)= PZ15*DUM
      PAM(1,2,3)=PZ16*DUM
      DUM= DSQRT(AMU*X(1))
      CB=RHO/DUM
      PZ38= -X(2)*CB/RHO**2
      PZ39= -X(3)*CB/RHO**2
      PAM(2,1,1)= AM(2,1)/(2.D0*X(1))
      PAM(2,1,2)= -CB*BETA*X1DOT/EN +PZ38*AM(2,1)/CB +CB*(PZ27
```

```
     1            -X(2)*BETA*PZ13/EN -X(2)*X1DOT*PZ5/EN)
      PAM(2,1,3)= PZ39*AM(2,1)/CB +CB*(PZ28 -PZ6*X(2)*X1DOT/EN
     1            -X(2)*BETA*PZ14/EN)
      PAM(2,2,1)= AM(2,2)/(2.D0*X(1))
      PAM(2,2,2)= PZ38*AM(2,2)/CB +CB*(PZ36 -BETA*Y1DOT/EN -X(2)
     1          *Y1DOT*PZ5/EN -X(2)*BETA*PZ15/EN)
      PAM(2,2,3)= PZ39*AM(2,2)/CB +CB*(PZ37 -X(2)*Y1DOT*PZ6/EN
     1            -X(2)*BETA*PZ16/EN)
      PAM(2,3,1)= AM(2,3)/(2.D0*X(1))
      DUM1= X(5)*Y1 -X(4)*X1
      PAM(2,3,2)= X(3)*(X(5)*PZ29 -X(4)*PZ20)/(RHO*DUM) +X(2)*X(3)
     1          *DUM1/(RHO**3*DUM)
      PAM(2,3,3)= DUM1/(RHO*DUM) +X(3)*(X(5)*PZ35 -X(4)*PZ26)/(RHO
     1         *DUM) +X(3)**2*DUM1/(RHO**3*DUM)
      PAM(2,3,4)= -X(3)*X1/(RHO*DUM)
      PAM(2,3,5)= X(3)*Y1/(RHO*DUM)
      PAM(3,1,1)= AM(3,1)/(2.D0*X(1))
      PAM(3,1,2)= PZ38*AM(3,1)/CB -CB*(PZ21 +X(3)*X1DOT*PZ5/EN
     1          +X(3)*BETA*PZ13/EN)
      PAM(3,1,3)= PZ39*AM(3,1)/CB -CB*(PZ22 +(BETA*X1DOT +X(3)
     1         *X1DOT*PZ6 +X(3)*BETA*PZ14)/EN)
      PAM(3,2,1)= AM(3,2)/(2.D0*X(1))
      PAM(3,2,2)= PZ38*AM(3,2)/CB -CB*(PZ30 +X(3)*(Y1DOT*PZ5
     1       +BETA*PZ15)/EN)
      PAM(3,2,3)= PZ39*AM(3,2)/CB -CB*(PZ31 +(BETA*Y1DOT +X(3)
     1         *Y1DOT*PZ6 +X(3)*BETA*PZ16)/EN)
      PAM(3,3,1)= AM(3,3)/(2.D0*X(1))
      PAM(3,3,2)= -DUM1/(RHO*DUM) -X(2)*(X(5)*PZ29 -X(4)*PZ20)/
     1         (RHO*DUM) -X(2)**2*DUM1/(RHO**3*DUM)
      PAM(3,3,3)= -X(2)*(X(5)*PZ35 -X(4)*PZ26)/(RHO*DUM) -X(2)*X(3)
     1         *DUM1/(RHO**3*DUM)
      PAM(3,3,4)= X(2)*X1/(RHO*DUM)
      PAM(3,3,5)= -X(2)*Y1/(RHO*DUM)
      Z5= (1.D0 +X(5)**2 +X(4)**2)/(2.D0*DUM*RHO)
      PZ40= -Z5/(2.D0*X(1))
      PZ41= X(2)*Z5/RHO**2
      PZ42= X(3)*Z5/RHO**2
      PZ43= X(4)/(DUM*RHO)
      PZ44= X(5)/(DUM*RHO)
      PAM(4,3,1)= AM(4,3)/(2.D0*X(1))
      PAM(4,3,2)= PZ41*Y1+Z5*PZ29
      PAM(4,3,3)= PZ42*Y1 +Z5*PZ35
      PAM(4,3,4)= PZ43*Y1
      PAM(4,3,5)= PZ44*Y1
      PAM(5,3,1)= AM(5,3)/(2.D0*X(1))
      PAM(5,3,2)= PZ41*X1 +Z5*PZ20
      PAM(5,3,3)= PZ42*X1 +Z5*PZ26
      PAM(5,3,4)= PZ43*X1
      PAM(5,3,5)= PZ44*X1
      DO 30 K=1,5
      PAM(1,3,K)=0.D0
      DO 30 I=4,5
      DO 30 J=1,2
   30 PAM(I,J,K)=0.D0
      DO 40 I=1,3
      DO 40 J=1,2
      DO 40 K=4,5
   40 PAM(I,J,K)=0.D0
      RETURN
      END
```