NASA CR–134927

Solar Electric Geocentric Transfer With Attitude Constraints: Program Manual

Lester L. Sackett, Harvey L. Malchow, and Theodore N. Edelbaum
The Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts

August 1975

Acknowledgments

This report was prepared under Project 55–64500 sponsored by the Lewis Research Center, National Aeronautics and Space Administration through Contract NAS3–18886. The Technical Monitor was Mr. James Cake.

CAVEAT EMPTOR!

This copy of the SECKSPOT (Solar Electric Control Knob Setting Program by Optimal Trajectories) documentation may contain errors in sequencing or with overly clipped margins. Every attempt was made to make the best copy possible using state of the art reproduction equipment. Because the original documents have long since passed into the great landfill beyond, this "copy of a copy" is as close to the original as possible and is the best NASA Glenn Research Center can provide.

Use these documents to understand the workings of the program SEPSPOT. Be aware that this documentation describes an input implementation that does not exist in SEPSPOT. Refer to the "SEPSPOT Inputs" document that Glenn Research Center also distributes for the current input mechanism. What the reader will find here are two documents. One (<u>Solar Electric Geocentric Transfer with Attitude Constraints: Analysis</u> NAS3–18886) describes the theoretical underpinnings of SEPSPOT, the other (<u>Solar Electric Geocentric Transfer with Attitude Constraints: Program Manual</u> Draper Laboratory R–902) describes the program structure. Several pages are missing from the program structure manual. They are merely the source code listings that someone removed from our copy. Should you find an error in these documents especially the theoretical formulation manual, please report them to John P. Riehl or Dr. Leslie R. Balkanyi of Glenn Research Center. Current telephone and email addresses for these individuals can be found at http://www.grc.nasa.gov/Doc/GRCfind.htm.

ABSTRACT

This manual is a guide for using the modified version of the SECKSPOT computer program. The program calculates time optimal or nearly time optimal geocentric transfers for a solar electric spacecraft with attitude constraints, which are those of one of the SERT-C designs of NASA/Lewis Research Center, or without attitude constraints. An initial high thrust stage with one or two impulses of fixed total ΔV may be included. The low thrust stage uses a nonsingular set of orbital elements and the method of averaging. The solution to the attitude constrained problem may yield discontinuous changes in the thrust direction. The power degradation due to the Van Allen radiation is modeled analytically. A wide range of solar cell characteristics may be assumed. Oblateness, shadowing, and solar distance effects may be included. The shadow model can include a delay in turning on the thrusters after the spacecraft leaves the shadow. The principal modifications to the SECKSPOT program include the attitude constraint solution, the new radiation and power loss model, the revised shadow model, and extended output.

A costate formulation is used which results in a two point boundary value problem which is solved using a Newton iteration on the initial unknown parameters and the unknown transfer time. The low thrust phase is applicable to general geocentric elliptical orbits. If high thrust is included, the initial orbit is assumed to be circular.

A Runge-Kutta method is used to integrate the state and costate equations. Averaging is done using a Gaussian quadrature. The code is in double precision and Fortran. It has been compiled in Fortran IV and used on an IBM/360 computer, and with minor modifications, in Fortran V on a Univac 1110 computer.

This report contains a general introduction and description of the program and detailed information on input, output, use of the code, and the individual subroutines including listings of the code. The analysis is discussed in a separate report.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

SECTION I

INTRODUCTION

I.1 Problem and Solution Method

This volume is a manual for a computer program which calculates minimum time optimal or nearly optimal trajectories for low thrust solar electric geocentric transfers with an optional initial high thrust stage. Reference 1 is the accompanying volume containing the analysis and model discussion and is referred to frequently in this volume. The code is an extensive expansion and revision of the SECKSPOT code described in References 2 and 3. One of the major extensions is that the new code can calculate trajectories for the attitude constrained case of zero roll and pitch. For this case the thrust direction is constrained to be in a plane perpendicular to the yaw axis which is parallel to the earth-spacecraft vector. This constraint is one of the SERT-C designs of NASA Lewis[4]. Major revisions were made to the radiation and power loss models to make them more accurate and more general. Several solar cell parameters may now be input to the code. The shadow model was also revised and the output was expanded.

The original SECKSPOT code included the following features which are retained in the new code. For the low thrust phase, equinoctial orbital elements[5] are used in order to avoid the singularities at zero eccentricity and inclination when using classical orbital elements. The method of averaging[6] is used to decrease computation time. This allows the averaged equations of motion to be integrated using a time step greater than a single orbital period, rather than small fractions of an orbital period which are necessary for precision orbit calculations. The averaged rate of change of the mean values of the state and costate are found by numerical quadrature. This allows the calculation of many trajectories, in a reasonable length of time, necessary in an optimization program. When the perturbations to the state are small (such as the effect of the low thrust propulsion in a strong gravity field) the error in the resulting trajectory is small compared to the precision trajectory. Thus such trajectories can be used for mission performance studies and also could form the basis of a guidance scheme[7].

For a solar electric spacecraft, the solar cells suffer degradation as a result of passing through the Van Allen radiation. This effect has been modeled analytically to reduce computation time and to facilitate the derivations of the co-state equations. Radiation is modeled as equivalent 1 MEV electrons/cm^2. The new model has a radius and latitude dependence. The model is a fit to actual solar maximum data with the azimuthal variation eliminated by averaging.

The radiation is made axial symmetric about the geographic axis by averaging the variation due to the tilted geomagnetic axis over one day. The new radiation model is dependent upon front and back shield thickness. The new analytic power loss model is a fit to data and is dependent upon cell thickness and base resistivity.

No power is received by the solar cells when the spacecraft is in earth's shadow. The earlier model assumed a cylindrical shadow and assumed that the thrusters were turned off immediately upon entering the shadow and turned on immediately upon exiting the shadow. A new model permits a delay in turning on the thrusters after leaving the shadow. The delay time is the sum of the time for the solar array to achieve operating temperature and the time for the thruster to achieve full thrust after the solar array power is applied to the power processor.

The effect of oblateness (analytically averaged equations are used) and the varying distance of the sun as the earth moves in its orbit may be included. The exhaust velocity and total power efficiency are assumed constant. Initial power and mass are inputs. The low thrust code is applicable to general elliptical orbits.

For the low thrust stage, seven state variables are used: five slowly varying equinoctial orbital elements, spacecraft mass, and equivalent 1 MEV fluence. When the initial high thrust stage is not included, all state variables are specified at the initial time. At the final time five orbital elements may be specified or three: the semimajor axis, eccentricity, and inclination. The remaining state variables are free.

For the attitude constrained case the class of spacecraft modeled is indicated in Figure 1 where the roll axis lies in the orbit plane and is perpendicular to the Earth-spacecraft radius vector; the yaw axis is parallel to the Earth radius vector directed toward the Earth; the pitch axis is perpendicular to the orbit plane and directed south. For the nominal attitude the principle body-centered axes are aligned with this coordinate system. The ion thrusters are mounted on the negative roll face of the spacecraft and directed parallel to the roll axis. The solar arrays are flat panels and are capable of rotation about their longitudinal axis, which is aligned with the spacecraft pitch axis. The required low thrust directions are achieved entirely by the spacecraft attitude rotations.

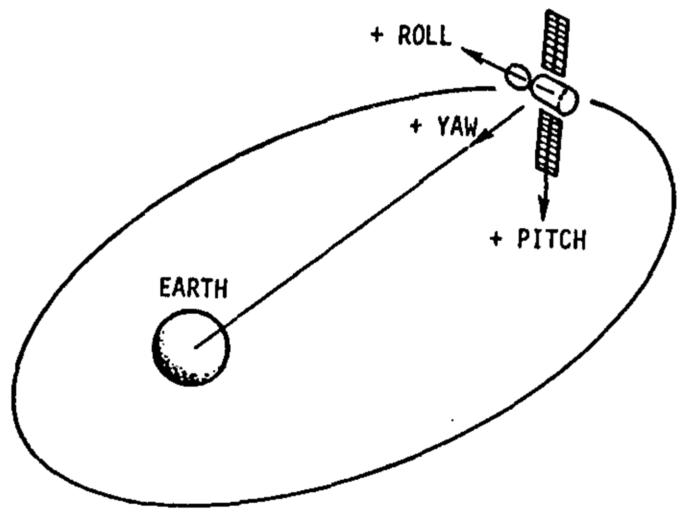


Figure 1 Spacecraft Configuration

The thrust vector must have no component along the vector from Earth to spacecraft. The thrust direction is determined by one control variable, the yaw angle. Since pitch and roll are constrained to be zero, the orientation of the axis of the solar panels is determined by the yaw angle. The panels are allowed to rotate about this axis, resulting in one more control variable. Power is assumed to be proportional to the cosine of the angle between the normal to the panels and the line to the sun. The initial state is given and it is desired to reach a final subset of the state at some, unspecified, final time. Although the original aim was to derive the time optimal control, instead, a particular suboptimal control, which is nearly time optimal, is used. A costate formulation is used along with the method of averaging to set up a two point boundary value problem which can be solved by using a Newton method to iterate on the unknown initial costate and value of the final time in order to meet the desired final conditions on the state and costate.

If initial high thrust is included then it is assumed that there is a specified ΔV of one or two impulses. This transfer is generated using a very efficient code developed by H. Small[8, 9]. The initial orbit is circular with specified semimajor axis and inclination, while the final orbit has specified semimajor axis, eccentricity, and inclination. During the high thrust phase an inverse square gravitational field is assumed. Because this transfer always requires less than a full revolution, its time is negligible compared to the low thrust phase and is not considered.

I.2 Equinoctial Orbital Elements

The low thrust trajectory calculations are done in equinoctial coordinates[5]. 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 north pole. The equinoctial coordinate frame is defined by unit vectors $\underline{\hat{f}}$, $\underline{\hat{g}}$, $\underline{\hat{w}}$ illustrated in Figure 2 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.

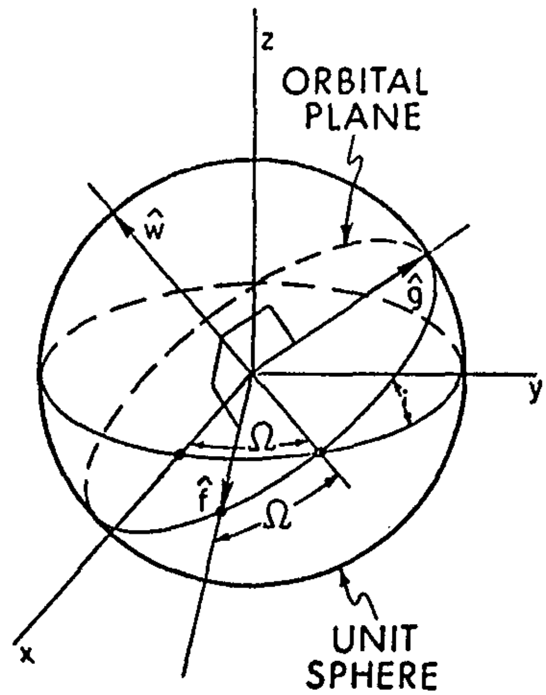


Fig. 2 The equinoctial coordinate frame

For the pure low thrust case five of the initial unknown parameters are the adjoints to the five equinoctial orbital elements. (The other unknown parameters are the adjoints to mass and fluence and the flight time.)

1.3 High Thrust Initial Conditions

Small uses a special set of variables for his high thrust code. This is discussed in Appendix A of Ref. 1. For the high thrust stage the initial argument of perigee, which determines the location in the orbit of the initial impulse, is free. Also, the longitude of the ascending node may be free or specified. As a result of these considerations the initial conditions which are specified are semimajor axis, eccentricity, inclination, the adjoint to the argument of perigee and either the longitude of the ascending node or its adjoint and mass and fluence. The free parameters include Small's variables τ, j, k which are defined in Ref. 1. τ is the out-of-plane thrust direction and j and k determine the in-plane thrust direction, ϕ. The thrust direction is given by

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

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 $\underline{e}_L = \underline{e}_h \times \underline{e}_R$. The initial ϕ is given by

$$\tan \phi = \frac{j \sin T}{1+k \cos T}$$

where we have assumed an initial circular orbit. The variables T, j, k are free, but must satisfy Small's optimality conditions (Eq. A. 4, Ref. 1).

It is convenient to define a transformed set of free initial parameters which have unlimited bounds by using the resulting conditions on T, k and j given in Eqs. A. 15, A. 16, and A. 17 of Ref. 1. If ξ_1, ξ_2, and ξ_3 are the new unbounded parameters, then T, k, and j are given by

$$T = \frac{\pi}{2} \frac{\xi_1}{\sqrt{1+\xi_1^2}}$$

$$k = \cos T \left(.75 + .25 \frac{\xi_2}{\sqrt{1-\xi_2^2}} \right)$$

$$j = (1 + k \cos T) \sqrt{\frac{\cos T - k}{\cos T + k}} \; \frac{\xi_3}{\sqrt{1+\xi_3^2}}$$

The other free initial parameters are the longitude of the ascending node or, if it is fixed, then the argument of perigee, which has the effect of determining the adjoint to the longitude of the ascending node, a scale factor relating the high thrust terminal costate and the initial low thrust costate, and the adjoints to mass and fluence.

SECTION II

DESCRIPTION OF THE DECK AND ITS OPERATION

The deck is made up of several subroutines. The code is in FORTRAN IV in double precision and has been used on an IBM 360 computer, compiled with an H compiler, at Draper Laboratory. It has also been compiled in FORTRAN V on the Lewis UNIVAC 1110 computer. In addition to the subroutines listed in this volume, one IBM Scientific Subroutine, DRTMI, is used, and one other, DRKGS, may be used[10].

The computer program is a version of SECKSPOT[2, 3] which has been extensively modified to generate trajectories for an attitude constrained spacecraft. The earlier version, developed for Goddard Space Flight Center did not include attitude constraints. Other modifications include the calculation of the effect of a delay in thrust turn-on after leaving earth's shadow, the coding of a new radiation and degradation model, and expanded output. Almost all of the features of SECKSPOT are retained.

The computer program solves a two-point-boundary-value-problem (2PBVP) which arises from the application of optimal control theory. The trajectories generated are time optimal, or for the attitude constrained case, nearly time optimal, as discussed in Ref. 1. Certain quantities are known at the initial time: the state consisting of the orbital elements and mass and fluence. When initial high thrust is included, $\omega(t_0)$ is free and at the user's option, $\Omega(t_0)$ may be free. When an initial orbital element is free, its corresponding adjoint must be zero. The final time is to be minimized and is unknown. The Hamiltonian at the final time must be equal to one. Some other final conditions, some of the state or costate or functions thereof, are specified at the final time. The 2PBVP is solved by using a Newton iteration on the unspecified initial conditions (the initial costate, or, if there is an initial high thrust stage, functions of the initial state and costate) and the final time to drive the final conditions to within specified bounds. Optimal (or nearly optimal) high and low thrust trajectories are generated by integrating the state and costate through each stage. A sensitivity matrix, or partial derivative matrix, is generated by varying the unspecified initial conditions and running a set of neighboring trajectories. Partials with respect to the final time can usually be obtained by directly using the state and costate derivatives. Continued calculations of new nominal trajectories and new sensitivity matrices result in many individual trajectories being run.

A single trajectory includes the low thrust solar electric stage and may also include an initial high thrust stage. For the low thrust stage the averaged state and costate are extrapolated using a Runge-Kutta integrator. This integrator receives the averaged derivative at each function call. The averaged derivative is calculated by calling a quadrature routine to calculate the averaged flux rate and the averaged thrusting effect. The averaged effect of oblateness is calculated analytically. Sun location, shadow intersection, and a delay in the thrust turn-on after leaving shadow may be calculated. The initial high thrust effect is calculated using the slightly modified code of Ref. 8.

A flow chart for the overall solution of the 2PBVP is shown in Figure 3.

Figure 4 is a system diagram which illustrates the connection of the various subroutines which make up the deck. In general, those subroutines above call those below to which they are connected.

Typical operation of the program is as follows. A run would begin by a call to INPUT by CONTL in order to set constant values, read and write input values. INPUT calls EARTH which sets earth related constants. Returning to CONTL, ITER is then called (either the Newton-Raphson or modified Newton-Raphson iterator). It calculates a nominal trajectory by calling TRAJ. If initial high thrust is included MAINE is called. MAINE calls START which initializes the S array. The S array contains complete information about the high thrust orbits and impulses. START has a call to SWITCH, then back to MAINE which calls TIME for each of the impulses. TIME calls SWITCH to perform certain calculations. TIME iterates on the value of ΔV in order to satisfy Small's optimality conditions. Then MAINE calls SWITCH to compute the coast angle. SWITCH calls DTDU to update the S array. Total allowed ΔV will be expended using either one or two impulses. The final orbit will be calculated by DTDU by a call from MAINE. Returning then to TRAJ, it calls OUTHI which, using the S array and other information that is in common, calculates the equinoctial state and costate of the last high thrust orbit. OUTHI also contains print statements which can be flaged. TRAJ sets up the input to the differential equation integrator (either RK4 or DRKGS) and calls it in order to calculate the low thrust portion of the trajectory. The integrator calls OUTP which may print information at certain or all time steps on just return. The integrator calls FUNCT which calculates the averaged state and costate derivative. FUNCT calls SUN which calculates the sun's location in equinoctial coordinates. Then SHADOW is called by FUNCT. The intersection (if any) between the orbit and the shadow is calculated as well as certain associated partial derivatives which FUNCT will use. The sun's direction calculated by SUN is used. SHADOW calls DQRTIC in order to solve the shadow quartic equation. DQRTIC calls DCUBIC to solve a cubic equation. If a delay in turn-on of the thrusters after leaving the shadow is included, then SDELAY is called. Next FUNCT calculates the flux effect by calling QUAD with FLUX as an

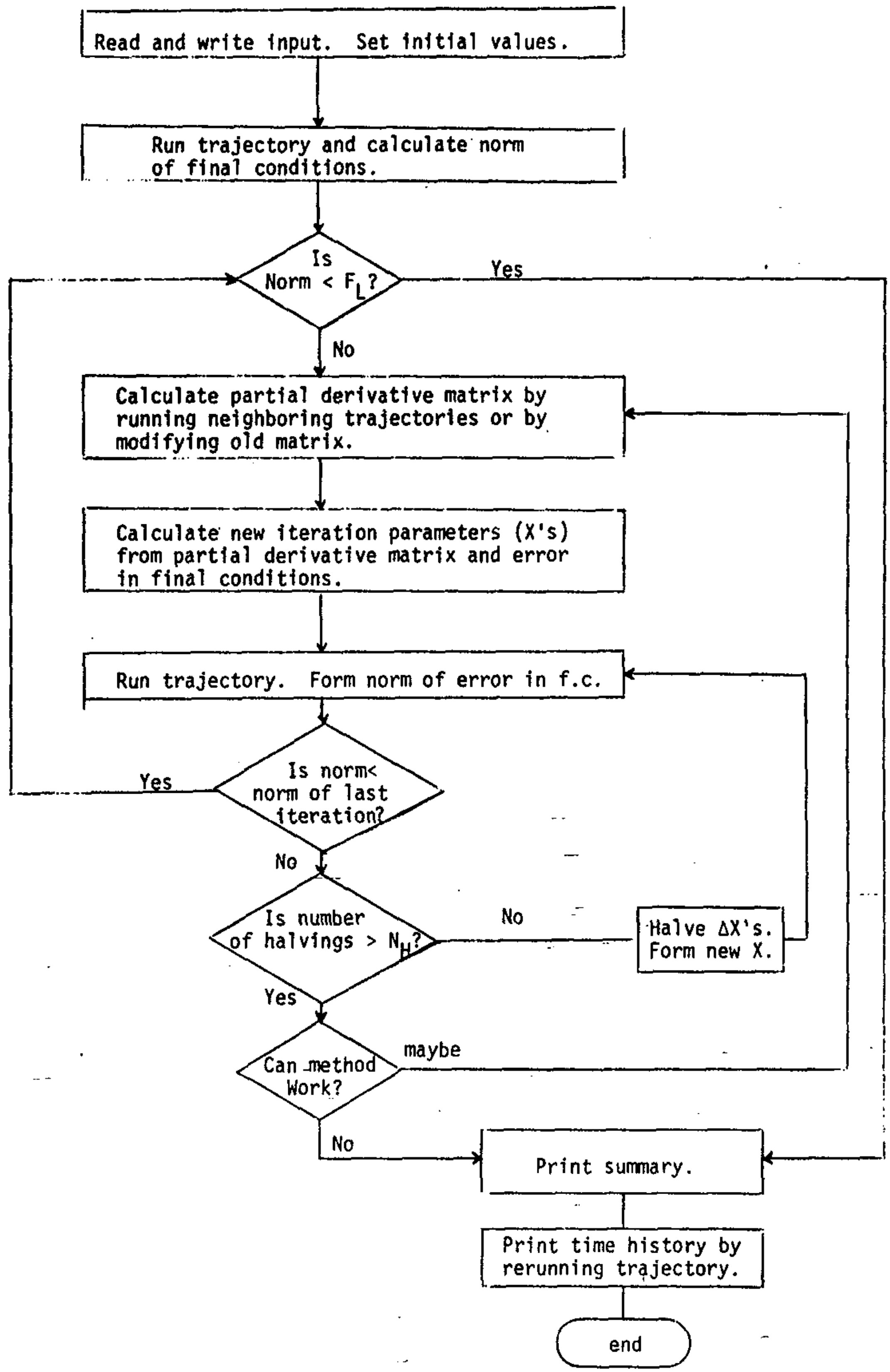


Figure 3. Overall Flowchart

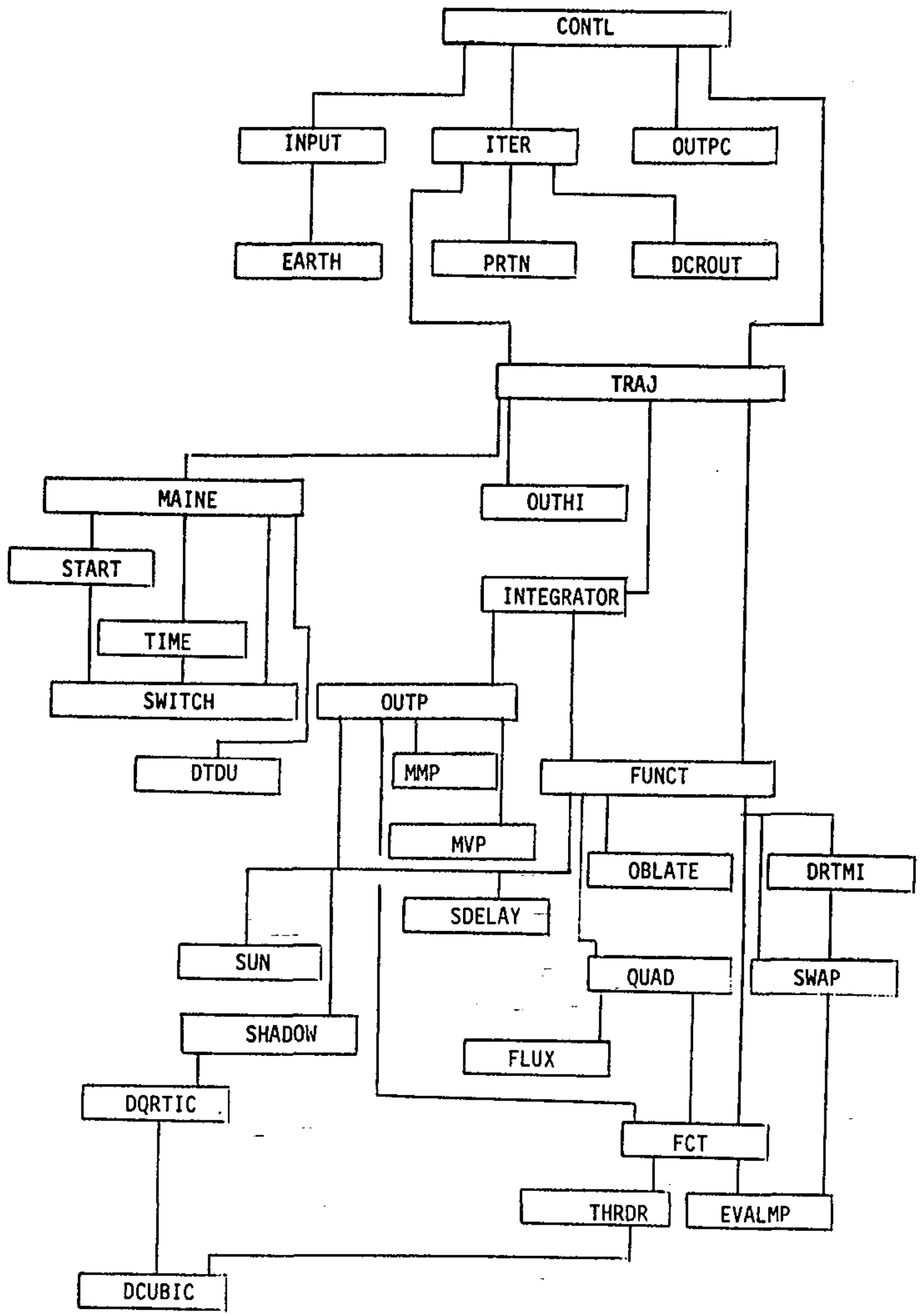


Figure 4. System Diagram

argument. QUAD is the gaussian quadrature routine which averages, in this case, the flux effect on state and costate. The pre-averaged derivatives are calculated by FLUX. For the attitude constrained case, the points on the orbit at which there may be jumps in the control are calculated by calling DRTMI and SWAP. A separate quadrature may be used between each "jump point". Over each such interval FUNCT calls QUAD with FCT as an argument in order to calculate the averaged derivative contribution by thrust. For the attitude constrained case, THRDR is called to calculate the thrust direction and THRDR solves a cubic by a call to DCUBIC. FCT calls EVALMP. FUNCT calls FCT to make certain calculations associated with shadowing. FUNCT then calls OBLATE which determines the J_2 contribution to the averaged derivative. All the contributions are added together appropriately and sent back to the integrator. When the integrator has reached the final time (an input), it returns to TRAJ. In order to calculate $\frac{\partial H}{\partial t_f}(t_f)$ numerically, TRAJ increments t_f by Δt_f and calls FUNCT. Finally, TRAJ calculates the error in the final conditions and the time derivatives of the final conditions and returns to ITER. ITER calculates a partial derivative matrix by varying slightly the iteration parameters. It then calculates changes on the iteration parameters and runs a new trajectory. Continuing with the Newton iteration procedure, the final conditions are either satisfied or else convergence fails. ITER calls PRTN which prints information at every iteration and DCROUT which inverts the partial derivative matrix. CONTL then calls OUTPC which prints a summary of the last trajectory (the optimal if convergence was successful). CONTL then calls TRAJ with the print flag on in order to calculate and print a time history of the last (optimal if converged) trajectory.

Total size of the object deck compiled on the IBM 360/75 in Fortran H is approximately 88000 bytes. The size of the individual subroutines is shown in Table 1. Not listed are the IBM Scientific Subroutines DRTMI and DRKGS.

TABLE 1

PROGRAM SUBROUTINE OBJECT DECK SIZE
(Thousands of bytes)

BLOCK DATA	-	OUTHI	5.8
CONTL	.8	OUTP	7.4
DCUBIC	1.2	OUTPC	2.9
DCROUT	3.3	PRTN	.7
DQRTIC	1.3	QUAD 4	1.0
DTDU	3.3	QUAD 8	1.2
EARTH	.6	QUAD 16	1.8
EVALMP	5.4	QUAD 32	2.9
FCT	4.4	RK4	1.7
FLUX	3.6	SDELAY	1.3
FUNCT	4.9	SHADOW	3.5
INPUT	11.1	START	1.5
ITER(NR)	3.5	SUN	1.5
ITER(MODNR)	3.9	SWAP	1.8
MMP	.9	SWITCH	2.8
MVP	.6	THRDR	1.7
MAINE	1.2	TIME	3.2
OBLATE	2.0	TRAJ(calls RK4)	3.2
		(calls DRKGS)	3.2

SECTION III

CONSTANTS AND CONVERSIONS

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

equatorial earth radius = 6378.16 km (Ref. 11)
earth gravitation coefficient = 398601.2 km^3/sec^2 (Ref. 11)
oblateness J_2 = .0010827 (Ref. 11)

Earth's orbital elements (epoch JD = 2436935.0) (Ref. 12)

a = 1 A.U.
e = .016726
ω = 102°25253

mean orbital motion = .985609°/day (Ref. 12)
obliquity of ecliptic = 23°45 (Ref. 12)
earth rotational frequency = 359°0170416/day (Ref. 12)

Internal units are in equatorial earth radii, 10^3 kg., and internal time units are calculated such that a circular orbit at 1 earth radii would have a period of 2π 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 = .0093381333403884
units to kilowatts = 77458.55283702227

SECTION IV

INPUT

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, l, m, n are read in format, I2. The input for the deck are listed in this section along with a brief description and nominal values, if any.

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) spacecraft mass at beginning of low thrust stage
W(7)	(10^{14} equivalent 1MEV electrons/cm^2) initial influence

Initial guesses

ZL0(1)	λ_a, adjoint to semi major axis or if initial high thrust the transformed T
ZL0(2)	λ_h, adjoint to orbital element h, or if initial high thrust the transformed Small variable k
ZL0(3)	λ_k, adjoint to orbital element k, or if initial high thrust the transformed Small variable j
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 longitude of node (radians) if NODE = 0, or adjoint to longitude of node if NODE = 1 (actually the impulse location)
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.
EF	the total power efficiency (maximum = 1)
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	maximum number of iterations (if 0, bypass iteration to print time history)	20
4	TFMAX2	(days) maximum flight time	400.
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, 2 units)	1., 1, 1, 1,
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 (use only if IHI = 1)	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-2
	KSTEP	= 0, step as fraction in ITER	1
		= 1, step as constant in ITER (except STEP(8))	
13	ISON	= 0, shadow effect off	0
		= 1, shadow effect on	
		= 2, shadow and delay included	

14	ISUN	= 0, sun distance effect on power off	0
		= 1, effect on	
15	IHI	= 1, low thrust only	1
		= 2, high/low	
	DVI1	(m/s) total initial high thrust ΔV	0.
	NODE	= 0, initial line of nodes free, λ_Ω fixed	
		= 1, initial line of nodes fixed, λ_Ω free	1
16	IPOW	= 0, constant power	1 .
		= 1, degradation effect	
	SH(1)	(mils) front shield thickness	6.
	SH(2)	(mils) back shield thickness	∞
	IB	= 0, if infinite back shielding	0
		= 1, if backshielding = front shielding	
		= 2, if unequal front and back shielding	
	IBR	= 0, base resistivity of 1 - 3 ohm-cm	
		= 1, base resistivity of 7 - 13 ohm-cm	1
	CTH	(mils) cell thickness	6.
17	FLIM	norm limit in iteration routine (D6. 1)	1.D-6
18	SGN	= -1, if initial λ_i is negative	SIGN (WF(3)-W(3))
		= +1, if initial λ_i is positive	
		(applicable to high thrust)	
19	NORB	= 0, no orbit print	0
		= 1, ..., 999, orbit print on	
		NORB points of an orbit (format, I3)	
20	ICON	= 1, no attitude constraints	1
		= 2, pitch = 0	
		= 3, pitch and roll zero with jump calculations	
		= 4, pitch and roll zero and no jump calculations	
21	IQ	orbit divided into IQ separate quadrature intervals if ICON = 1 (1 - 10)	2

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 (XL0), then Y, the error in the final conditions. The final conditions are the final values of a, h, k, p, q, λ_m, λ_N, H, if NOP = 1. If NOP = 2 the final conditions are a, e, $\tan\frac{i}{2}$, λ_ω, λ_Ω, λ_m, λ_N, H unless the final eccentricity or inclination is zero. If eccentricity is zero the second condition is h and the fourth k; if inclination is zero, the third is p and the fifth is q. Then the final time (TF) is printed in internal units, followed by, F_0, 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 values of "X" are changed slightly by inputed amounts; these perturbed values of X(X(I) + DX(I)) are next printed followed by the corresponding Y. Then 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 F_0, 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}, the final mass, the ratio of final to initial mass, the final power, the degradation factor 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 $\times$ 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, T and ΔV where the thrust direction is given by

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

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 $\underline{e}_L = \underline{e}_h \times \underline{e}_R$ (see App. A, Ref. 1).

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 (10^3kg) and fluence (10^{14} particles). 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). The delay time (min), the entry angle (°), exit angle (°) and turn-on angle (°) are printed. If NORB was set to a nonzero value then there is additional output.

The optionally printed output for an orbit at each time step includes the sun direction in the equinoctial coordinate frame and the earth-sun distance in A.U.'s. If ICON = 3, the eccentric longitude at which the projection of the primer vector is perpendicular to the sun vector is printed. Jumps in the thrust direction may occur at these points. Several spacecraft parameters are printed at a number of points on the orbit (the number of points = NORB). The points are at equal units of eccentric longitude. At each point the spacecraft location is printed including the eccentric longitude and the $\hat{\underline{f}}$ and $\hat{\underline{g}}$ components X_1 and Y_1 in units of earth radii. The following spacecraft parameters are printed with the spacing indicated here:

IN-PLANE	OUT-OF-PLANE	YAW	PITCH	ROLL
PAN ORIENT.	SUN INC. PAN	SUN INCID. X	SUN INCID. Y	SUN INCID. Z
UF	UG	UW	SUN IN X-Y	PSI
PRIMER ANGLE				

All angles are in degrees. The meanings of these terms follows (most are also discussed in Section 3.13 of Ref. 1).

IN-PLANE, the in-plane thrust acceleration direction angle, θ_i where

$$\hat{\underline{u}} = \cos\theta_o \cos\theta_i\, \hat{\underline{e}}_x + \sin\theta_o\, \hat{\underline{e}}_h + \cos\theta_o \sin\theta_i\, \hat{\underline{e}}_r$$

where $\hat{\underline{e}}_r$ is along the radius vector, $\hat{\underline{e}}_h$ along the angular momentum vector, and

$$\hat{\underline{e}}_x = \hat{\underline{e}}_h \times \hat{\underline{e}}_r$$

OUT-OF-PLANE, the out-of-plane thrust acceleration direction angle, θ_o, defined above.

YAW, θ, and PITCH, ψ_y, are another set of angles defining the thrust acceleration direction where,

$$\underline{\hat{u}} = \cos\theta \cos\psi_y \, \underline{\hat{e}}_x + \cos\theta \sin\psi_1 \, \underline{\hat{e}}_y - \sin\theta \, \underline{\hat{e}}_z$$

Here $\underline{\hat{e}}_z = -\underline{\bar{e}}_r$ is pointed toward the earth from the spacecraft, and $\underline{\hat{e}}_y = -\underline{\hat{e}}_h$ is normal to the orbit plane and pointed "downward".

ROLL defines a spacecraft rotation about the axis which is along the longitudinal axis of the spacecraft. It shifts the orientation of the panel axis, and does not influence thrust direction. The order of rotation is yaw, pitch, and roll. Before yaw and pitch and roll rotations it is assumed that the panel axis is perpendicular to the radius vector with the panels pointed toward the earth.

PAN ORIENT or panel orientation angle defines the rotation about the panel axis. The angle is zero when the normal to the panels is perpendicular to the spacecraft roll axis. Thus before yaw, pitch, and roll rotations the panel angle is zero when the panels are facing the earth. The panel orientation angle is adjusted to maximize the amount of sunlight falling on the panels.

SUN INC PAN or sun incidence angle on the panels is the angle between the normal to the panels and the vector pointing toward the sun. It ranges between 0 and 90°.

SUN INCID X is the sun incidence angle on the forward X-surface of the spacecraft, i.e., the angle between the positive roll axis and the vector pointing toward the sun (after yaw, pitch, roll rotations).

SUN INCID Y is the sun incidence angle on the y surface of the spacecraft, i.e. the angle between the positive pitch axis and the vector pointing toward the sun.

SUN INCID Z is the sun incidence angle on the Z surface of the spacecraft, i.e., the angle between the positive yaw axis and the vector pointing toward the sun.

UF, UG, UW are the $\underline{\hat{f}}$, $\underline{\hat{g}}$, and $\underline{\hat{w}}$ components, respectively, of the thrust acceleration vector, i.e., in the equinoctial coordinate system.

SUN IN X-Y is the angle that the projection of the sun vector onto the $\hat{\underline{e}}_x$, $\hat{\underline{e}}_y$ plane makes with the $\hat{\underline{e}}_x$ vector.

PSI is the control angle, ψ, which is defined in Section 3.6 of Ref. 1. It is equal to the yaw angle minus the above sun angle in X-Y and is meaningful for the constrained case.

PRIMER ANGLE is the angle, α, defined in Section 3.6 of Ref. 1. It yields the orientation of the projection of the primer vector in the plane perpendicular to the radius vector.

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 RUNNING THE PROGRAM

Picking initial conditions is very important for running a program such as this. There are few 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 simplest 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 (4, 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.

For the constrained case which calculates possible jump points the least accurate quadrature is a 4-point between shadow entrance and exit (or thruster turn-on) points and jump points. Otherwise a single 4-point quadrature may be used. Reasonable accuracy requires at least two 4-point or one 8-point quadrature per orbit. A typical moderately accurate time step was one per .2 km/sec of ΔV, e.g., a 10 day time step for a SERT-C type mission.

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.

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

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

ZLO(1) = 3000.
ZLO(2) = 100.
ZLO(3) = 300.
ZLO(4) = - 6000.
ZLO(5) = - 30,000.

ZLO(6) is the adjoint to mass and ZLO(7) the adjoint to fluence. Typical values are

ZLO(6) = - 30000.
ZLO(7) = - 100.

When initial high thrust is included, 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 Υ, k, j (see App. A of Ref. 1).

$$\Upsilon = \frac{\pi}{2} \frac{ZLO(1)}{\sqrt{1 + ZLO(1)^2}}$$

$$k = \cos \Upsilon (.75 + .25 \frac{ZLO(2)}{\sqrt{1 + ZLO(2)^2}})$$

$$j = (1 + k \cos \Upsilon) \sqrt{\frac{\cos \Upsilon - k}{\cos \Upsilon + k}} \cdot \frac{ZLO(3)}{\sqrt{1 + ZLO(3)^2}}$$

The above transformation insures that τ, 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 be around 1 to 10. 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.

If the longitude of the ascending node, Ω, at the initial time is fixed then ZL0(5) is the angle which sets the location of the first impulse. (This has the result of determining λ_{Ω} as shown in Appendix A of Ref. 1.) Alternatively Ω may be free in which case ZL0(5) is Ω. This option should not be used for our model if the sun does not influence the trajectory either through shadowing or for the roll and pitch constraints. If it is used the sensitivity matrix will be singular.

Typical values of ZL0(I), I = 1,...,5 for the high thrust case are

$$
\begin{aligned}
ZL0(1) &= -.2 \\
ZL0(2) &= .4 \\
ZL0(3) &= -.1 \\
ZL0(4) &= 3. \\
ZL0(5) &= 0.
\end{aligned}
$$

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. A smaller number might often be used. 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 400 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 SSP integrator, nominally set to a high value so th it is usually never reached, allowing the user to determine accuracy by picking the time step.

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 arbitrary.

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 398601.2km^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 that 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 fixed step specified with all elements of the array set to 10^{-2} except for the eighth which is set to 10^{-6}. This was found to work pretty well for low thrust only cases. Typical values used for high thrust cases were

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

Changing these values by a couple orders of magnitude had little effect.

STEP (8) remains the fractional variation in the final time used to calculate $\frac{\partial H}{\partial t_f}$.

Nominally, shadow effect is not included.

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

Nominally, low thrust only is assumed. If initial impulses are required, the ΔV must be given. 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.

IPOW is nominally set to 01, so the degradation due to radiation is included. Nominally infinite backshielding is assumed so IB = 0. SH(1) is the front shielding, set to 6 mils, SH(2) is not used if IB = 0. If the front and back shielding are equal IB = 1 is more efficient than setting IB = 2 and setting SH(2) to the same value as SH(1). The nominal base resistivity is 7-13 ohm-cm or IBR = 1. The nominal cell thickness is 6 mils.

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. For "rough" runs a higher value could be 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.

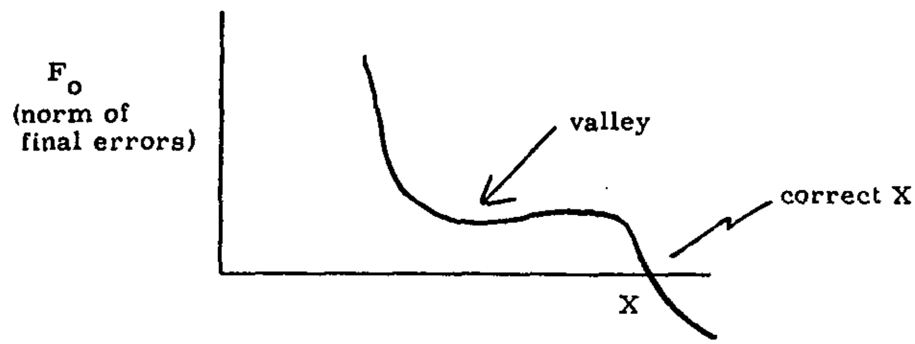
NORB determines the amount of detailed information for an orbit which is printed. Nominally it is set to zero. Setting it 024 will yield printout every 15° of eccentric longitude.

Nominally, no attitude constraints are assumed.

The number of quadratures, IQ, per orbit is applicable only if "jump points" are not calculated, for either the constrained or unconstrained cases, IQ is nominally set to 2.

Because of the functional form of the power versus 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 overrides the internal calculated value. A typical value might be .01 (in units of 10^{14} particles).

Occasionally the program will not converge because the iterator has become hung up in valley, especially when shadowing is included, illustrated in the following figure.



The Newton method keeps sending X to the floor of this valley. These valleys may arise for real physical reasons, but more commonly they are due to numerical causes arising from inaccuracies due to a too large time step, a too small quadrature, or from not calculating the jump points in thrust direction for the constrained case. Sometimes just changing the time step by a small amount, say 20%, will be enough to avoid the valley. A physical cause may arise when including the shadow delay. As the initial "X's" are changed slightly the F_o may be decreasing, but then for a small change in $\underline{X}$ some part of the trajectory may enter the shadow briefly. But even entering the shadow for an instant causes about a 12 minute delay in turn on time, thus introducing a discontinuous change in the trajectory and F_o.

Most of the runs performed have used the special Runge-Kutta integrator without accuracy controls in order to reduce computation time. There has been little experience using the accuracy controls with the SSP integrator.

SECTION VII

COMMON AREAS

There are several common areas which are shared by several of the subroutines. Table 2 lists these common areas and indicates which subroutines share them. Those subroutines which have no common areas are not included. Following is a list of common areas, the variables in these areas, the definition of variables, and a list of subroutines which contain the particular common area.

Common area A/A, AMU, PI

A	Thrust acceleration
AMU	Gravity constant, μ
PI	π

Shared by:

FUNCT, INPUT, OUTP, OUTPC, TRAJ

Common area BCOM / B(9)

B	The coefficients of cosF, sinF in X1, Y1 and partials (Eqs. 3.19, 3.20, Table B-2, Ref. 1)
B(1)	$1 - h^2\beta$
B(2)	$hk\beta$
B(3)	$1 - k^2\beta$
B(4)	$-h(2\beta + h^2\beta_3)$
B(5)	$k(\beta + h^2\beta_3)$
B(6)	$-hk^2\beta_3$
B(7)	$-kh^2\beta_3$
B(8)	$h(\beta + k^2\beta_3)$
B(9)	$-k(2\beta + k^2\beta_3)$ where $\beta_3 = \dfrac{\beta^3}{1-\beta}$

TABLE 2: COMMON AREAS

SUBROUTINES (columns) / COMMON AREAS (rows)

COMMON AREAS	BLOCKDATA	CONTL	DTDU	EARTH	EVALMP	FCT	FLUX	FUNCT	INPUT	ITER	MAINE	OUTHI	OUTP	OUTPC	PRTN	SDELAY	SHADOW	START	SUN	SWAP	SWITCH	TRAJ
A						X		X	X				X			X						X
BCOM							X	X												X		
BLK1	X								X													
BLK2	X								X													
CCOM							X												X			
CONSTR						X		X	X				X									
DCOM								X	X				X	X								
DELAY								X					X			X						
DY										X												X
ELEM									X				X	X								X
F									X	X												
FUOUT								X					X									
HIGH									X													X
INT		X							X	X			X									X
JD				X					X													
J2								X	X													
KCOM							X		X													
ORBIT									X				X									
ORBIT1						X							X									
ORBIT2					X	X							X									
ORBIT3						X							X									
POWER						X		X	X				X	X								
Q								X	X													
SG									X			X										
SHAD								X					X			X	X					
SOL								X					X				X		X	X		
STR			X								X	X						X			X	
T									X	X				X	X							X
TC									X					X								X
TERRA				X															X			
TRA									X													X
UNITS				X					X			X	X	X								
WF									X					X								
XMMM									X	X				X								X
Z														X	X							X
ZSWAP								X												X		

Shared by:

FUNCT, FLUX, SWAP

Common area: BLK1 /EK(15, 5), PK(15, 5)

EK(15, 5)	Electron coefficients, Table 4-4 of Ref. 1
PK(15, 5)	Proton coefficients, Table 4-5 of Ref. 1

Shared by:

BLOCK DATA, INPUT

Common area: BLK2/B(12)

B	The coefficients of the constants, C_1, C_2 of the power loss function given in Eqs. 4.5 and 4.6 of Ref. 1.

Shared by:

BLOCK DATA, INPUT

Common area: CCOM/C(6)

C	Flux factors (see Eq. 3.172-3.176 of Ref. 1)
C(1)	$\hat{f}_3$
C(2)	$\hat{g}_3$
C(3)	$\frac{\partial \hat{f}_3}{\partial p}$
C(4)	$\frac{\partial \hat{f}_3}{\partial q}$
C(5)	$\frac{\partial \hat{g}_3}{\partial p}$
C(6)	$\frac{\partial \hat{g}_3}{\partial q}$

Shared by:

FLUX, SUN

Common area: CONSTR/ICON

ICON	Flag indicating if no attitude constraints, pitch zero, or roll and pitch zero

Shared by:

FCT, FUNCT, INPUT, OUTP

Common area: DCOM/CD1, CD2

CD1	C_1 in the power loss equation (Eq. 4.4 of Ref. 1)
CD2	C_2 in the power loss equation

Shared by:

FUNCT, INPUT, OUTP, OUTPC

Common area: DELAY/TD, FD

TD	The delay time (between leaving shadow and thrust on) (Eq. 3.138 of Ref. 1)
FD	The eccentric longitude at turn on time (see Eq. 3.143, Ref. 1)

Shared by:

FUNCT, OUTP, SDELAY

Common area: DY/DYDT(8)

DYDT	The partial of the final conditions with respect to time

Shared by:

ITER, TRAJ

Common area: ELEM/ZP0(7), ZPF(5)

ZP0	The initial state.
ZPF	The desired final equinoctial orbital elements

Shared by:

INPUT, OUTP, OUTPC, TRAJ

Common area: F/FLIM, KSTEP

FLIM	The requirement on the limit of the norm of the errors in the iterator.
KSTEP	Flag indicates whether STEP refers to a fixed incement or a fractional increment

Shared by:

INPUT, ITER

Common area: FUOUT/FSA(6), MM

FSA	The eccentric longitude at the thrust jump points (maximum of 6), the solutions of Eq. 3.124, Ref. 1
MM	The number of jumps during an orbit

Shared by:

FUNCT, OUTP

Common area: HIGH/DVI1, IHI, NODE

DVI1	The total initial high thrust ΔV
IHI	Flag indicates if initial high thrust is included
NODE	Flag indicates if initial Ω is fixed or free

Shared by:

INPUT, TRAJ

Common area: INT/IPR, IDIM, IDIM2, NIMAX

IPR	Print flag used in OUTP and OUTHI
IDIM	Equal to the dimension of the state plus costate (14)
IDIM2	Equal to dimension of the state (7)
NIMAX	The maximum number of iterations of the iterator

Shared by:

CONTL, INPUT, ITER, OUTP, TRAJ

Common area: JD/TL

TL	Julian date at launch

Shared by:

EARTH, INPUT

Common area: J2/AJ2

AJ2 The oblateness coefficient

Shared by:

FUNCT, INPUT

Common area: KCOM/AK(5, 5, 2, 2)

AK(I, J, K, L) These are the coefficients K in Eq. 4.3, Ref. 1, where I varies as i, J varies with powers of latitude, K is 1 for electrons, 2 for protons, L is 1 for front and 2 for the back shield

Shared by:

FLUX, INPUT

Common area: ORBIT/NORB

NORB Indicates if there should be orbit print and the number of points printed

Shared by:

INPUT, OUTP

Common area: ORBIT1/VEC(3)

VEC The thrust direction vector

Shared by:

FCT, OUTP

Common area: ORBIT2/X1, Y1, PR(2, 2)

X1 $\hat{f}$ component of spacecraft location in orbit (Eq. 3.19, Ref. 1)

Y1 $\hat{g}$ component of spacecraft location in orbit (Eq. 3.20, Ref. 1)

PR(2, 2) Partials of X1 and Y1 with respect to h and k (Table B-2, Ref. 1)

Shared by:

EVALMP, FCT, OUTP

Common area: ORBIT3/CALP, SALP

CALP cosine (α), α is the primer angle (see Eq. 3.97, Ref. 1)

SALP sine (α)

Shared by:

FCT, OUTP

Common area: POWER/P0, C, POW, PH, ISUN, ISON, IPOW

P0	The initial maximum power (to the thrusters) at 1 A. U.
C	The exhaust velocity
POW	Power to the thrusters
PH	Not used
ISUN	Flag indicating if solar distance effect is included
ISON	Flag indicating if shadowing is included
IPOW	Flag indicating if constant power, or power degradation

Shared by:

FCT, FUNCT, INPUT, OUTP, OUTPC

Common area: Q/IQ

IQ	The number of segments into which the orbit is divided, so that the quadrature is called for each segment to calculate the thrust effect; used only if jumps in thrust direction are not calculated

Shared by:

FUNCT, INPUT

Common area: SG/SGN

SGN	Sign of Λ_4 in OUTHI (see Eqs. A. 63, A. 64 of Ref. 1)

Shared by:

INPUT, OUTHI

Common area: SHAD/FEN, FEX, DFEN(5), DFEX(5), ISHAD

FEN	The eccentric longitude at shadow entry
FEX	The eccentric longitude at shadow exit
DFEN(5)	The partial of FEN with respect to the equinoctial orbital elements
DFEX(5)	The partial of FEX with respect to the equinoctial orbital elements (or maybe the partial of the angle at "turn on" if shadow delay is included)
ISHAD	A flag indicating whether or not a particular orbit passed through the earth's shadow

Shared by:

FUNCT, OUTP, SDELAY, SHADOW

Common area: SOL/RSUN(3), RS

RSUN	The unit vector pointing toward the sun from earth
RS	The distance from earth to sun (A. U.)

Shared by:

FCT, FUNCT, OUTP, SHADOW, SUN, SWAP

Common area: STR/S(1, 3, 20)

S(I, J, K)	Used in the high thrust code, I is fixed at 1, J varies over the three possible orbits of a two-impulse trajectory; the quantities for each K are listed in Table A-1, Ref. 1

Shared by:

DTDU, MAINE, OUTHI, START, SWITCH

Common area: T/TF, T0

TF	Flight time
T0	The initial time (measured from launch time)

Shared by:

INPUT, ITER, OUTPC, PRTN, TRAJ

Common area: TC/NOP

NOP	Flag indicates whether three final or five final orbital elements are specified

Shared by:

INPUT, OUTPC, TRAJ

Common area: TERRA/AE, EC, W, ENE, AN, COP, SOB

AE	Earth's semimajor axis (in A. U.)
EC	Eccentricity of earth's orbit
W	Earth's argument of perihelion
ENE	Earth's mean orbital motion
AN	The mean anomaly of Earth at the time of launch
COB	The cosine of the angle of obliquity
SOB	The sine of the angle of obliquity

Shared by:

EARTH, SUN

Common area: TRA/TFMAX, DT, UEB, EW(14)

TFMAX	The maximum time of flight allowed
DT	The time step used by the differential equation integrator
UEB	The upper error bound (used by the DRKGS integrator)
EW	Error weights (used by the DRKGS integrator)

Shared by:

INPUT, TRAJ

Common area: UNITS/UTS, UTM, UTH, UTD, UTKM, DTR, UTKG, UTKW, UTMS2

UTS	Internal units to seconds
UTM	Internal units to minutes
UTH	Internal units to hours
UTD	Internal units to days
UTKM	Internal units to kilometers
DTR	Degrees to radians
UTKG	Internal units to kilograms
UTKW	Internal units to kilowatts
UTMS2	Internal units to meters/sec^2

Shared by:

EARTH, FUNCT, INPUT, OUTHI, OUTP, OUTPC

Common area: WF/WF(5)

WF	The desired final classical orbital elements

Shared by:

INPUT, OUTPC

Common area: XMMM/ZL0 (7), STEP(8), ZERF(8)

ZL0	The iteration parameters
STEP	Step size used to numerically evaluate partial derivative (or sensitivity) matrix in iterator
ZERF	The error in the final conditions

Shared by:

INPUT, ITER, OUTPC, PRTN, TRAJ

Common area: Z/ZF(14), DZ(14)

ZF	The state and costate
DZ	The derivative of the state and costate

Shared by:

OUTPC, TRAJ

Common area: ZSWAP/ZZ(14)

ZZ(14)	The state and costate

Shared by:

FUNCT, SWAP

SECTION VIII

SUBROUTINE DESCRIPTION AND LISTINGS

The section contains descriptions of all subroutines including input and output, common areas, subroutines which are called or called by. For some of the more important subroutines the Fortran variables are defined and flowcharts included. A listing follows the description. A variable in the argument list or the common areas is underlined once if it is output, twice if input or three times if both. Aliases are sometimes given in parentheses following the subroutine name. In addition to the subroutines contained in this section, two other IBM Scientific Subroutines, DRTMI and DRKGS, are needed.[10]

BLOCK DATA

Description:

Contains Table 4-4 and coefficients of Eqs. 4.5 and 4.6 of Reference 1. EK for electrons, PK for protons, and B for the power loss coefficients.

Common Areas:

BLK1/<u>EK</u>(15,5), <u>PK</u>(15,5)
BLK2/<u>B</u>(12)

Subroutine CONTL

Description:

This is the main "driver" program.

Common Areas:

INT/IPR, IDIM, IDIM2, NIMAX

Calls Subroutines:

INPUT, ITER, OUTPC, TRAJ

Subroutine DCROUT (DCROUT14)

Description:

This has a square matrix and a vector as input. It inverts the matrix and premultiplies the vector by the inverse.

Argument List:

AA, R, D, EPS, NI, M, IND

AA	The matrix which is to be inverted
R	As input, the vector to be multiplied by the inverted matrix, as output, the resultant vector
D	The determinant of AA
EPS	An input, a small quantity which is used to check for a singular matrix
NI	Dimension of the matrix and vector
M	Flag set to 1 as input
IND	Flag, if $\neq 0$, matrix is singular

Called by:

ITER

Subroutine DCUBIC

Description:

Calculates the roots of a cubic equations.

Argument List:

C, R, NRE

C	The coefficients of the cubic equation $X^3 + C(1)\,X^2 + C(2)\,X + C(3) = 0$
R	Roots, if one real root, it is R(1)
NRE	The number of real roots

Called by:

DQRTIC, THRDR

Subroutine DQRTIC

Description:

Calculates the roots of a quartic equation.

Argument List:

C, R, NRE

C	The coefficients of the quartic equation $X^4 + C(1)\,X^3 + C(2)\,X^2 + C(3)\,X + C(4) = 0$
R	Roots, if 2 real, they are R(1) and R(2)
NRE	The number of real roots

Called by:

SHADOW

Calls Subroutines:

DCUBIC

Subroutine DTDU

Description:

Updates the S array. When called by SWITCH the S array is that which results after the last impulse (calculated by TIME) and the following coast, Δu, calculated by SWITCH.

When called by MAINE, there are no more impulses and so the coast angle is meaningless, and thus MAINE sets it to 0. Also, rather than using the $d\tau$ (normalized ΔV) calculated by TIME, DTDU uses the $d\tau$ set by MAINE.

Argument List:

I, L, DT, M, ISW

I	Always = 1
J	The orbit number (1, 2 or 3)
DT	Normalized ΔV (Sec. A.2, Ref. 1)
M	Set to J + 1 as input
ISW	Flag, always = 1 as input

Common Areas:

STR/S(1, 3, 20)

Called by:

SWITCH, MAINE

Subroutine EARTH

Description:

Sets the values for the earth's semimajor axis, eccentricity, argument of perihelion, mean orbital motion, mean anomaly at launch, and angle of obliquity.

Common Areas:

JD/TL
TERRA/C(7)
UNITS/UTS, UTM, UTH, UTD, UTKM, DTR, UTKG, UTKW, UTMS2

Called by:

INPUT

Subroutine EVALMP (EVALMPC)

Description:

Evaluates M and $\frac{\partial M}{\partial \underline{z}}$ (see Eqs. 3.33, 3.48 of Ref. 1). The form of M which is coded in EVALMP is that shown in Ref. 5, Table 4. It was also from this form th: $\frac{\partial M}{\partial \underline{z}}$ was calculated, while holding F constant when taking partials with respect to a, h, k, p, q.

Argument List:

X, THETA, AMU, AM, PAM, IMFLAG

X	Five equinoctial orbital elements
THETA	Eccentric longitude
AMU	Gravitational constant, μ
AM	The matrix M = $\frac{\partial \dot{\underline{z}}}{\partial \dot{\underline{r}}}$ (Eq. 3.33 of Ref. 1)
PAM	The partial of M with respect to the orbital elements
IMFLAG	Flag, if = 1, AM and PAM are calculated; if = 2, only M is calculated; if = 3, only PAM is calculated

Common Areas:

ORBIT2/X1, Y1, RA, PZ20, PZ26, PZ29, PZ35

Called by:

FCT, SWAP

Fortran Variables:

EN	n	orbital frequency, (Eq. 3.23, Ref. 1)
BETA	β	(Eq. 3.13, Ref. 1)
RA	r/a	(Eq. 3.24, Ref. 1)
X1	X_1	(Eq. 3.19, Ref. 1)
Y1	Y_1	(Eq. 3.20, Ref. 1)
X1DOT	$\dot{X}_1$	(Eq. 3.21, Ref. 1)
Y1DOT	$\dot{Y}_1$	(Eq. 3.22, Ref. 1)
ZETA	$k \sin F - h \cos F = F - \lambda$ (see Eq. 3.15, Ref. 1)	
BETA3	$\beta_3 = \beta^3 / (1 - \beta)$	

Subroutine EVALMP (Cont'd)

Fortran Variables: (Cont'd)

PZ1	$\frac{\partial X_1}{\partial h}$	(from Eq. 39, Ref. 5) F is a function of h, k, λ)
PZ2	$\frac{\partial X_1}{\partial k}$	
PZ3	$\frac{\partial Y_1}{\partial h}$	
PZ4	$\frac{\partial Y_1}{\partial k}$	
AM	M	(from Table 3 of Ref. 5)
PZ5	$\frac{\partial \beta}{\partial h}$	
PZ6	$\frac{\partial \beta}{\partial k}$	
CA	$A = \sqrt{\mu/a} / (r/a)$	
PZ9	$\frac{\partial A}{\partial h}$	
PZ10	$\frac{\partial A}{\partial k}$	
PZ20	$\frac{\partial X_1}{\partial h}\Big\vert_F$	Table B-2, Ref. 1 F held constant
PZ26	$\frac{\partial X_1}{\partial k}\Big\vert_F$	
PZ29	$\frac{\partial Y_1}{\partial h}\Big\vert_F$	
PZ35	$\frac{\partial Y_1}{\partial k}\Big\vert_F$	
PZ11	$\frac{\partial \dot{X}_1}{\partial a}$	
PZ12	$\frac{\partial \dot{Y}_1}{\partial a}$	

Fortran Variables: (Cont'd)

PZ13	$\frac{\partial \dot{X}_1}{\partial h}$	Table B-8 , Ref. 1
PZ14	$\frac{\partial \dot{X}_1}{\partial k}$	
PZ15	$\frac{\partial \dot{Y}_1}{\partial h}$	
PZ16	$\frac{\partial \dot{Y}_1}{\partial k}$	
PZ17	$\frac{\partial \gamma_1}{\partial \beta}$	where $\gamma_1 = \beta + h \frac{\partial \beta}{\partial h}$
PZ18	$\frac{\partial \gamma_1}{\partial h}$	
PZ19	$\frac{\partial \gamma_1}{\partial k}$	
PZ21	$\frac{\partial}{\partial h}\left(\frac{\partial X_1}{\partial h}\right)$	
PZ22	$\frac{\partial}{\partial k}\left(\frac{\partial X_1}{\partial h}\right)$	
PZ23	$\frac{\partial \beta_3}{\partial \beta}$	
PZ24	$\frac{\partial \beta_3}{\partial h}$	
PZ25	$\frac{\partial \beta_3}{\partial k}$	
PZ27	$\frac{\partial}{\partial h}\left(\frac{\partial X_1}{\partial k}\right)$	
PZ28	$\frac{\partial}{\partial k}\left(\frac{\partial X_1}{\partial k}\right)$	
PZ30	$\frac{\partial}{\partial h}\left(\frac{\partial Y_1}{\partial h}\right)$	

Subroutine EVALMP (Cont'd)

Fortran Variables: (Cont'd)

PZ31	$\frac{\partial}{\partial k}\left(\frac{\partial Y_1}{\partial h}\right)$	
PZ32	$\frac{\partial \gamma_3}{\partial \beta}$	where $\gamma_3 = \beta + k\frac{\partial \beta}{\partial k}$
PZ33	$\frac{\partial \gamma_3}{\partial h}$	
PZ34	$\frac{\partial \gamma_3}{\partial k}$	
PZ36	$\frac{\partial}{\partial h}\left(\frac{\partial Y_1}{\partial k}\right)$	
PZ37	$\frac{\partial}{\partial k}\left(\frac{\partial Y_1}{\partial k}\right)$	
PAM	$\frac{\partial M}{\partial \underline{z}}$	

Subroutine FCT (FCTC)

Description:

Evaluates what is essentially the preaveraged derivative (see Section 3.4, Ref. 1, Eqs. 3.40 ,3.41) of the state and costate due to thrusting. Specifically,

$$\left(\frac{\partial H_z}{\partial \underline{\lambda}_z} \quad \frac{dt}{dF}\right) \frac{c}{2} = \underline{\dot{z}} \, \frac{dt}{dF} \quad \frac{c}{2P}$$

$$\frac{\partial H_m}{\partial \lambda_m} \quad \frac{dt}{dF} \quad \frac{c}{2} = \dot{m} \, \frac{dt}{dF} \quad \frac{c}{2P}$$

$$\left[\frac{\partial H}{\partial \lambda_z} \quad \frac{dt}{dF} + (H_z + H_m) \frac{\partial}{\partial \underline{\lambda}_z} \left(\frac{dt}{dF}\right)\right] \frac{c}{2P}$$

The factor $\frac{2}{c}$ and the minus sign of Eq. 3.41 , Ref. 1 and the factor $\frac{1}{2\pi}$ are taken into account in FUNCT. When FCT is called by OUTP, the thrust direction is calculated and then returns prior to the derivative calculation.

Argument List:

F1, F2, Z, H, G

F1	Eccentric longitude
F2	Another eccentric longitude
Z	Five orbital elements and their adjoints
H	Preaveraged derivative for $\underline{\dot{z}}$ and $\underline{\dot{\lambda}}_z$ corresponding to F1
G	Preaveraged $\underline{\dot{z}}$ and $\underline{\dot{\lambda}}_z$ at F_2

Common Areas:

A/A, AMU, PI
ORBIT1/VEC(3)
ORBIT2/X1,Y1, RA, PR(2, 2)
ORBIT3/CALP, SALP
POWER/P0, C, DUB(2), HI(3)
SOL/RS(4)
CONSTR/ICON

Called by:

QUAD, FUNCT, OUTP

Calls Subroutines:

EVALMP, THRDR

Fortran Variables:

C	exhaust velocity (Eq. 3.32, Ref. 1)
F	eccentric longitude
G	the preaveraged derivative at F2
H	the preaveraged derivative at F1
X	orbital elements
Z	state + costate
AM	M (Eq. 3.33, Ref. 1)
CB	$\cos\beta$ (see Eq. 3.33, Ref. 1)
PA	$\cos\phi$ (Eq. 3.89, Ref. 1)
PR	$\frac{\partial(X1, Y1)}{\partial(h, k)}$ (Table B-2, Ref. 1)
RA	$\frac{r}{a}$ (Eq. 3.24, Ref. 1)
RS	earth-sun unit vector
SB	$\sin\beta$ (see Eq. 3.83, Ref. 1)
TH	T, rotation matrix (Eq. 3.57, Ref. 1)
X1	$\hat{f}$ component of spacecraft vector (Eq. 3.19, Ref. 1)
Y1	$\hat{g}$ component of spacecraft vector (Eq. 3.20, Ref. 1)
AMU	μ, gravity constant
CB2	$\cos^2\beta$ (see Eq. 3.89, Ref. 1)
CMU	multiplier, μ, Eq. 3.53, Ref. 1
HZM	$H_z + H_m$, Hamiltonian segment (see Eq. 3.195, Ref. 1)
PAM	$\frac{\partial M}{\partial \underline{z}}$
PCB	$\frac{\partial \cos\beta}{\partial \underline{z}}$
PPA	$\frac{\partial(\cos\phi)}{\partial \underline{z}}$
PRA	$\frac{\partial}{\partial \underline{z}}(r/a)$

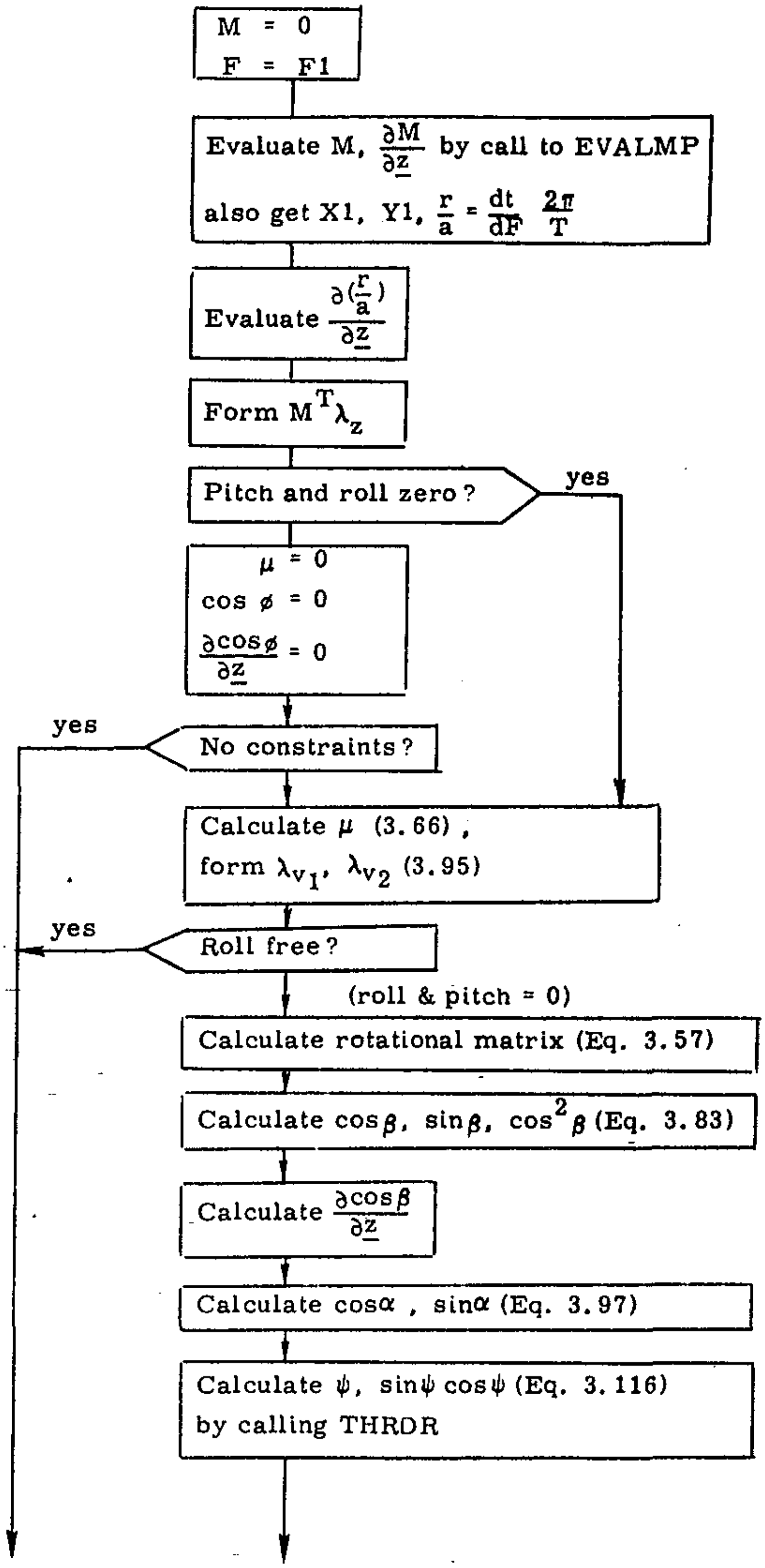
Subroutine FCT (FCTC) Cont'd)

Fortran Variables: (Cont'd)

PSI	ψ , thrust angle (see Eq. 3.84, Ref. 1)
RAA	$\lvert \underline{r} \rvert^2$
VEC	used as primer vector, as primer vector minus component along radius vector, and as thrust direction
CALP	$\cos\alpha$, α is "primer vector angle", (see Eq. 3.97 Ref. 1)
CPSI	$\cos\psi$
DXSP	$\frac{\partial X_s}{\partial p}$
DXSQ	$\frac{\partial X_s}{\partial q}$
DYSP	$\frac{\partial Y_s}{\partial p}$
DYSQ	$\frac{\partial Y_s}{\partial q}$
SALP	$\sin \alpha$
SPSI	$\sin \psi$

(DXSP, DXSQ, DYSP, DYSQ:) X_s, Y_s are components of sun unit vector. (see Eq. D.5, Ref. 1). p and q are equinoctial orbital elements

Flow Diagram for FCT



$\cos\phi = \sqrt{1 - \sin^2\beta \sin^2\psi}$ (Eq. 3.89)

Calculate $\dfrac{\partial \cos\phi}{\partial \underline{z}}$

If F2 = 0 return to OUTP

Evaluate $\dot{\underline{z}} \dfrac{dt}{dF} \dfrac{c}{2} \dfrac{1}{p}$

$\dot{m} \dfrac{dt}{dF} \dfrac{c}{2} \dfrac{1}{p}$

$\left[\dot{\lambda}_z \dfrac{dt}{dF} + \dot{\lambda}_z \dfrac{\partial}{\partial \underline{z}} \left(\dfrac{dt}{dF}\right)\right] \dfrac{c}{2p}$

If M = 1 return, otherwise
M = 1, F = F2, go to top
and repeat for other location

Subroutine FLUX (FLUXC)

Description:

Evaluates the preaveraged values of flux (Eq. 4-1 of Ref. 1) and the partial, $\frac{\partial \dot{N}}{\partial z}$, at the two points on an orbit signified by the eccentric longitudes F1 and F2 and returns the values in H and G respectively.

Argument List:

F1, F2, Z, H, G

F1	Eccentric longitude
F2	Another eccentric longitude
Z	Five orbital elements and adjoints
H	Preaveraged flux ($\dot{N}$) and $\frac{\partial \dot{N}}{\partial z}$ at F1
G	Preaveraged flux ($\dot{N}$) and $\frac{\partial \dot{N}}{\partial z}$ at F2

Common List:

BCOM/B(9)
CCOM/C(6)
KCOM/AK(5, 5, 2, 2), IB

Called by:

QUAD

Fortran Variables:

A(I,J,K)	essentially the A_i of Eq. 4.2 of Ref. 1 where $I \sim i$, J = 1 for electrons, 2 for protons, K = 1 for front shielding, 2 for back shielding
B	coefficients of X1, Y1 etc., set up in FUNCT
C	flux factors evaluated in SUN
E	$\frac{\partial A_i}{\partial w}$ (Eq. 3.168, Ref. 1)
F	eccentric anomaly
G	flux and partials at F1
H	flux and partials at F2
R	spacecraft radius vector magnitude (in earth radii)
S	r/a (Eq. 3.24, Ref. 1)
U	ln(R-1)
W	latitude
Z	state and costate

Subroutine FLUX (FLUXC) (Cont'd)

Fortran Variables: (Cont'd)

AK	values from Table 4-4 and 4-5 and SH contribution of Eq. 4-3 of Ref. 1 for electrons, protons, front and back shielding
CA	cosine of the latitude
CF	cosF
F1	a particular eccentric longitude
F2	a particular eccentric longitude
IB	flag indicating how to include back shielding (∞, = front, $\neq$ front)
PF	essentially $\frac{\partial \dot{N}}{\partial \underline{z}}$ (Eq. 3.166, Ref. 1)
PU	$\frac{\partial(\ln R-1)}{\partial \underline{z}}$
PW	$\frac{\partial}{\partial \underline{z}}$ (latitude)
PX PY	$\left.\begin{array}{l}\frac{\partial X1}{\partial \underline{z}} \\ \\ \frac{\partial Y1}{\partial \underline{z}}\end{array}\right\}$ Table B-2, Ref. 1
SA	sine (latitude)
SF	sin F
X1	$\hat{\underline{f}}$ component of spacecraft radius vector (Eq. 3.19, Ref. 1)
Y1	$\hat{\underline{g}}$ component of spacecraft radius vector (Eq. 3.20, Ref. 1)

Subroutine FUNCT (FUNCTC)

Description:

Called principally by the differential equation integrator to evaluate the averaged derivative for the full state and costate. Can take into account sun location, shadowing, oblateness and calculate jump points for the roll and pitch constrained problem.

Argument List:

X, Z, DERZ

X	Time
Z	State and costate
DERZ	Derivative of state and costate

Common Areas:

A / A,AMU,PI
J2/ AJ2
SHAD / FEN, FEX, DFEN(5), DFEN(5), ISHAD
BCOM / B(9)
POWER/ PO, C, POW,PH,ISON,ISON,IPOW
SOL/ RSUN (3), RS
CONSTR/ICON
ZSWAP/ZZ(14)
FUOUT/FSA(6),MM
DELAY/TD, FD
DCOM/CDI, CD2
Q / IQ

Called by:

RK4 or DRKGS, TRAJ

Calls Subroutines:

SUN, SHADOW, QUAD, FCT, OBLATE, DRTMI, SWAP

Subroutine FUNCT (Cont'd)

Fortran Variables:

B	coefficients of cosF, sinF in X_1, Y_1, $\dot{X}_1$, $\dot{Y}_1$, $\frac{\partial(X_1 Y_1)}{\partial(h,k)}$ (see Eqs. 3.19, 3.20, 3.21, 3.22 and Table B-2 of Ref. 1)
C	exhaust velocity (Eq. 3.32, Ref. 1)
F	eccentric longitude
X	time
Z	state and costate vector
DP	$\frac{\partial P}{\partial N}$, P is power, N is fluence (see Eq. 3.198, Ref. 1)
FD	engine turn-on angle (see Eq. 3.143, Ref. 1)
FS	eccentric longitude at which Eq. 3.124, Ref. 1 holds
HM	$\tilde{H}_m$ (Eq. 3.192, Ref. 1)
HZ	$\tilde{H}_z$ (Eq. 3.191, Ref. 1)
IQ	orbit is divided into IQ-intervals and quadrature called for each one unless ICON = 3, FLUX is calculated over 2*IQ intervals
MM	the number of "jump points"
PI	π
P0	P_o (see Eq. 3.181, Ref. 1)
RS	earth-sun unit vector
TD	the delay time (see Eq. 3.138, Ref. 1)
AJ2	J_2, oblateness coefficient
CD1	C_1 of Eq. 4-4, of Ref. 1
CD2	C_2 of Eq. 4-4 of Ref. 1
DFL	$\left(\dot{\bar{N}}, \frac{\partial \dot{\bar{N}}}{\partial \underline{z}}\right) 2\pi$
FAC	factor $= \frac{P_o D}{\pi c R_s^2}$ (see Section 3.13 of Ref. 1)

Subroutine FUNCT (Cont'd)

Fortran Variables: (Cont'd)

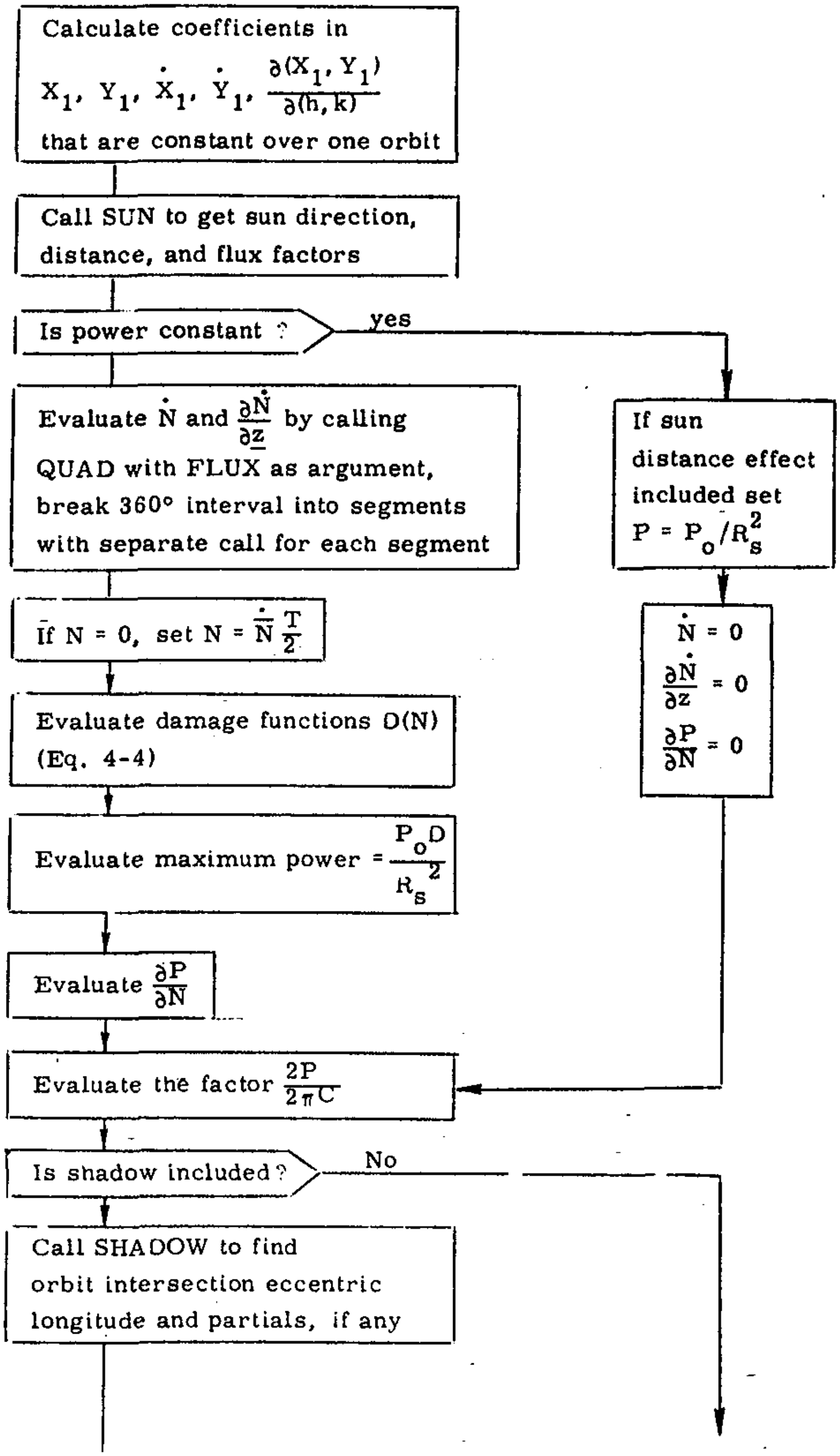
FEN	eccentric longitude at entry to shadow
FEX	eccentric longitude at exit from shadow
FSA	vector containing all the jump points
GEN	proportional to $\dot{\underline{z}}$, $\dot{m}$ at shadow entry
GEX	proportional to $\dot{\underline{z}}$, $\dot{m}$ at shadow exit
HWN	$H_z + H_m$ at entry to shadow
HWX	$H_z + H_m$ at exit from shadow
IER	error flag returned from DRTMI
POW	power $= \frac{P_0 D}{R_s^2}$
QEN	eccentric long. at entrance to shadow
QEX	eccentric long. at entrance to shadow
Z10	orbital elements and their adjoints
BETA	β (Eq. 3.13, Ref. 1)
DERZ	$\dot{\underline{z}}$, $\dot{\underline{\lambda}}_z$, the derivative of the orbital elements and their adjoints
DFEN	$\frac{\partial F}{d\underline{z}}\Big\rvert_{\text{entry}}$ (see Eq. 3.137, Ref. 1)
DFEX	$\frac{dF}{d\underline{z}}\Big\rvert_{\text{exit}}$ (see Eq. 3.137, Ref. 1)
DZJ2	Contribution to $\dot{\underline{z}}$, $\dot{\underline{\lambda}}_z$ due to oblateness
DZ11	essentially $\dot{\underline{z}}$, $\dot{\underline{m}}$, $\dot{\underline{\lambda}}_z$
ICON	Flag indicating attitude constraints (= 1, no constraints; = 2, pitch = 0; = 3, pitch and roll = 0 and calculate jump points; = 4 pitch and roll = 0, don't calculate jump points)
IPOW	Flag = 0 if constant power = 1 if degradation included

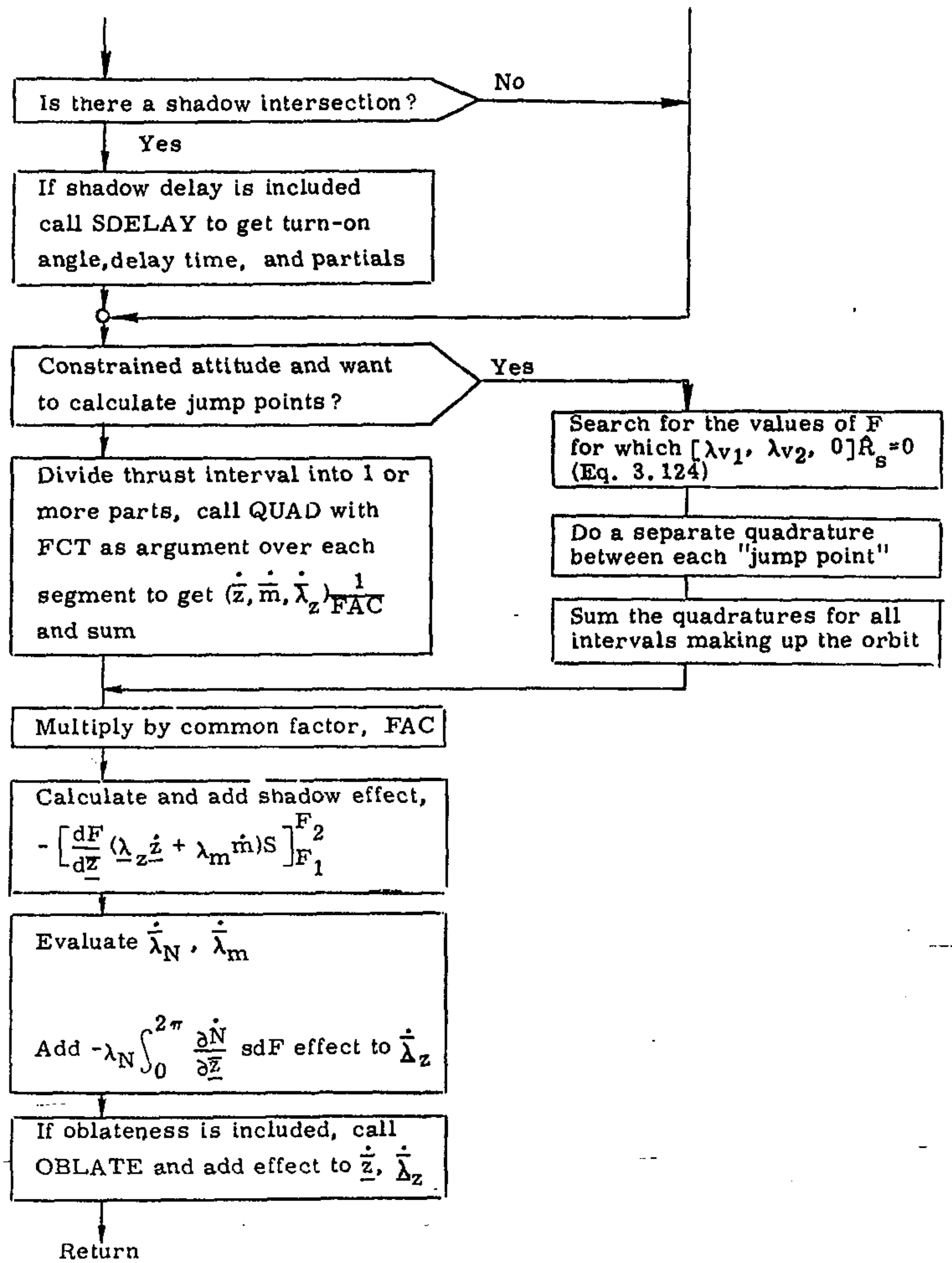
Subroutine FUNCT (Cont'd)

Fortran Variables: (Cont'd)

ISON	flag = 0 if no shadow = 1 if shadow, no delay = 2 if shadow and delay
ISUN	flag = 0 if not include sun distance = 1 if $\frac{1}{R_s^2}$ effect on power
RSUN	Earth-sun distance in A.U.'s
SWAP	function routine which calculates Eq. 3.124, Ref. 1
BETA3	$\beta_3 = \frac{\beta^3}{1-\beta}$, β from Eq. 3.13, Ref. 1
ISHAD	flag = 0 if no shadow intersection = 2 orbit enters and leaves shadow
DERZ11	essentially $\dot{\underline{z}}$, $\dot{m}$, $\dot{\underline{\lambda}}_z$

Flow Chart for FUNCT




Is there a shadow intersection?
No
Yes
If shadow delay is included call SDELAY to get turn-on angle, delay time, and partials
Constrained attitude and want to calculate jump points?
Yes
Search for the values of F for which $[\lambda_{v1}, \lambda_{v2}, 0]\hat{R}_s = 0$ (Eq. 3.124)
Do a separate quadrature between each "jump point"
Sum the quadratures for all intervals making up the orbit
Divide thrust interval into 1 or more parts, call QUAD with FCT as argument over each segment to get $(\dot{\underline{z}}, \dot{\bar{m}}, \dot{\lambda}_z)\frac{1}{FAC}$ and sum
Multiply by common factor, FAC
Calculate and add shadow effect, $-\left[\frac{dF}{d\underline{z}}(\underline{\lambda}_z \dot{\underline{z}} + \lambda_m \dot{m})S\right]_{F_1}^{F_2}$
Evaluate $\dot{\bar{\lambda}}_N$, $\dot{\bar{\lambda}}_m$
Add $-\lambda_N \int_0^{2\pi} \frac{\partial \dot{N}}{\partial \underline{z}}\, s dF$ effect to $\dot{\bar{\lambda}}_z$
If oblateness is included, call OBLATE and add effect to $\dot{\bar{\underline{z}}}$, $\dot{\bar{\lambda}}_z$
Return

Subroutine INPUT (INPUTC)

Description:

Reads initial data, sets initial values, and prints initial information.

Common Areas:

XMMM/ZLO17), STEP16), ZERI16)

ELEM/ZPO(7), ZPF(5)

INT/IPR, IDIM, IDIM2, NIMAX

TRA/IFMAX, DT, UEB, EW(14)

UNITS/UTS, UTM, UTH, STD, UTRM, DTR, UTKG, UTKW, UTMS2

T / TF, TO,

A / A, AMU, PI

WF / WF(5)

J2/AJ2

TC / NOP

JD/ TL

POWER/PO, C, POW, PH, ISUN, ISON, IPOW

HIGH / DVI1, IHI, NODE

F / FLIM, KSTEP

SG / SGN

ORBIT / NORB

CONSTR / ICON

BLK1 / EPK (15, 5, 2)

KCOM / AK (5, 5, 2, 2)

BLK2 / B(12)

DCOM / CD(2)

Q / IQ

Called by:

CONTL

Calls Subroutines:

EARTH

Subroutine ITER (NRC, MODNRC)

Description:

A Newton-Raphson iterator. It calculates a nominal trajectory, receiving the error in the final conditions called $\underline{y}$. It then varies the free initial conditions $\underline{x}$ (including the value of t_f) to get a sensitivity matrix or partial derivative matrix. By inverting this matrix and premultiplying $\underline{y}$ by its new values of $\underline{x}$ are obtained and a new nominal trajectory run. This is continued until the norm of the errors in the final conditions is less than some inputed tolerance or until the convergence fails. There is provision for halving the Δx's several times.

An optional Modified Newton-Raphson iterator does not calculate a new partial derivative matrix by running neighboring trajectories at each iteration but rather updates the partial derivative matrix using the old one and the Δx's. If convergence is poor a new matrix is calculated by running neighboring trajectories, however.

Argument List:

KOUNT, NI, FUNCT, PRTN

KOUNT	Number of trajectory calls
NI	The number of iterations
FUNCT	Dummy subroutine name - that is called to calculate the "y" for the iterator (in our case this is TRAJ)
PRTN	Dummy subroutine name-used to print iteration info (in our case this is called PRTN)

Common Areas:

XMMM/X(7), XS(8), Y(8)
INT/IPR, IDIM, IDIM2, MAXN01
T/TF, TO
DY/DYDT(8)
F/FLIM, KSTEP

Called by:

CONTL

Calls Subroutines:

PCROUT, PRTN, TRAJ

Subroutine MAINE

Description:

The main driving subprogram for the high thrust calculations. It calls START to initialize the S array. Then TIME which finds dτ such that the LHS (Eq. A.4c, Ref. 1) is zero (or actually slightly positive). It then calls SWITCH in order to update the S array. It then checks to see if the total ΔV is greater than the fixed limit. If not, it checks to see if the number of impulses has reached the maximum (set to 2 by TRAJ).

If the total ΔV is not greater than the fixed input value and if the number of impulses is less than the total allowed by the input minus one, then TIME is called again to find the next impulse, otherwise the ΔV of the final impulse is set, Δu is set to zero (it doesn't really mean anything since there will be no more impulses) and DTDU is called to evaluate the S array after this final impulse.

Argument List:

E, F, XK, UP, XJ, HR, IMAX, JM, TTOT

E	Initial eccentricity (set = 0)
F	Initial true anamoly (set = 0)
XK	Initial value for k (Eq. A.8 , Ref. 1)
UP	Initial value for τ (Eq. A.7 , Ref. 1)
XJ	Initial value for j (Eq. A.8 , Ref. 1)
HR	Initial value for angular momentum (set = 1)
IMAX	Not used
JM	Maximum number of impulses + 1 (= 3)
TTOT	The total normalized impulsive ΔV

Common Areas:

STR / S,(1, 3, 20)

Called by:

TRAJ

Calls Subroutines:

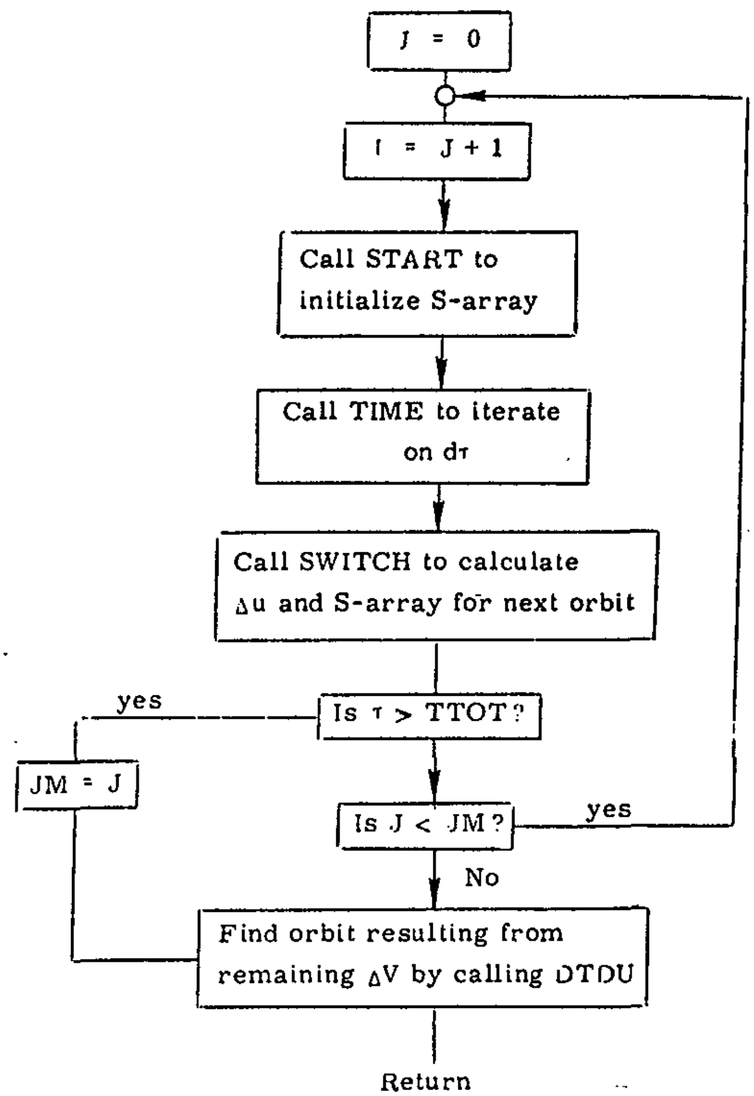
START, TIME, SWITCH, DTDU

Subroutine MAINE (Cont'd)

Fortran Variables:

E	eccentricity
F	true anomaly
S	S array
HR	initial angular momentum (set to 1)
JM	maximum number of impulses (input, set to 2)
NF	flag, if ≤ 0, bad initial data
UP	τ (see Appendix A of Ref. 1)
XJ	j (see Appendix A of Ref. 1)
XK	k (see Appendix A of Ref. 1)
DTF	ΔV for last impulse
DT1	initial guess for $d\bar{\tau}$ (normalized ΔV)
TTOT	total initial normalized ΔV

Flow Chart for MAINE



JM = maximum number of impulses (input set to 2)

TTOT = maximum normalized ΔV

τ = total ΔV

$d\tau$ = ΔV for one impulse

Subroutine MMP

Description:

Calculates a 3×3 matrix-matrix product where if $M = 1$, $C = AB$
$M = 2$, $C = A^T B$
$M = 3$, $C = AB^T$
$M = 4$, $C = A^T B^T$

Argument List:

$\underline{\underline{A}}$, $\underline{\underline{B}}$, $\underline{C}$, $\underline{\underline{M}}$

A	Input matrix
B	Input matrix
C	Output matrix
M	Flag

Called by:

OUTP

Subroutine MVP

Description:

Calculates a 3 dimensional matrix-vector product where if $K = 1 \quad C = AB$

$K = 2 \quad C = A^T B$

Argument List:

$\underline{\underline{A}}$, $\underline{\underline{B}}$, $\underline{C}$, $\underline{\underline{K}}$

A	Input matrix
B	Input vector
C	Output vector
K	Flag

Called by:

OUTP

Subroutine OBLATE

Description:

Calculates the oblateness effect on $\dot{\underline{z}}$ and $\dot{\underline{\lambda}}_z$ (see Section 3.7, Ref. 1).

Argument List:

AJ2, Z, DZJ2, IJ

AJ2	The J_2 coefficient
Z	Orbital elements and adjoints
DZJ2	The oblateness contribution to the state and costate derivative
IJ	Flag, always = 1

Called by:

FUNCT

Subroutine OUTHI

Description:

Forms the interface between the high and the low thrust phases. It takes the contents of the S-array for the last high-thrust orbit, the values of the inclination, longitude of ascending node, and semimajor axis of the initial orbit and calculates the equinoctial orbital elements and may print information for each orbit (see Section A.3, Ref. 1).

Argument List:

JM, PI, A0, AII, OMI, WI, IPR, Z, IDIM2, NODE

JM	The number of impulses + 1
PI	π
A0	Initial semimajor axis
AII	Initial inclination
OMI	Initial longitude of ascending node
WI	Initial argument of perigee ($u = f+\omega$)
IPR	Print flag (= 0 if no print)
Z	State and costate
IDIM2	Dimension of state
NODE	Flag = 0, Ω free = 1, Ω fixed

Common Areas:

STR/S (1, 3, 20)
UNITS/UTS, UTM, UTH, UTD, UTKM, DTR, UTKG, UTRW, UTMS2
SG/SGN

Called by:

TRAJ

Subroutine OUTHI (Cont'd)

Fortran Variables:

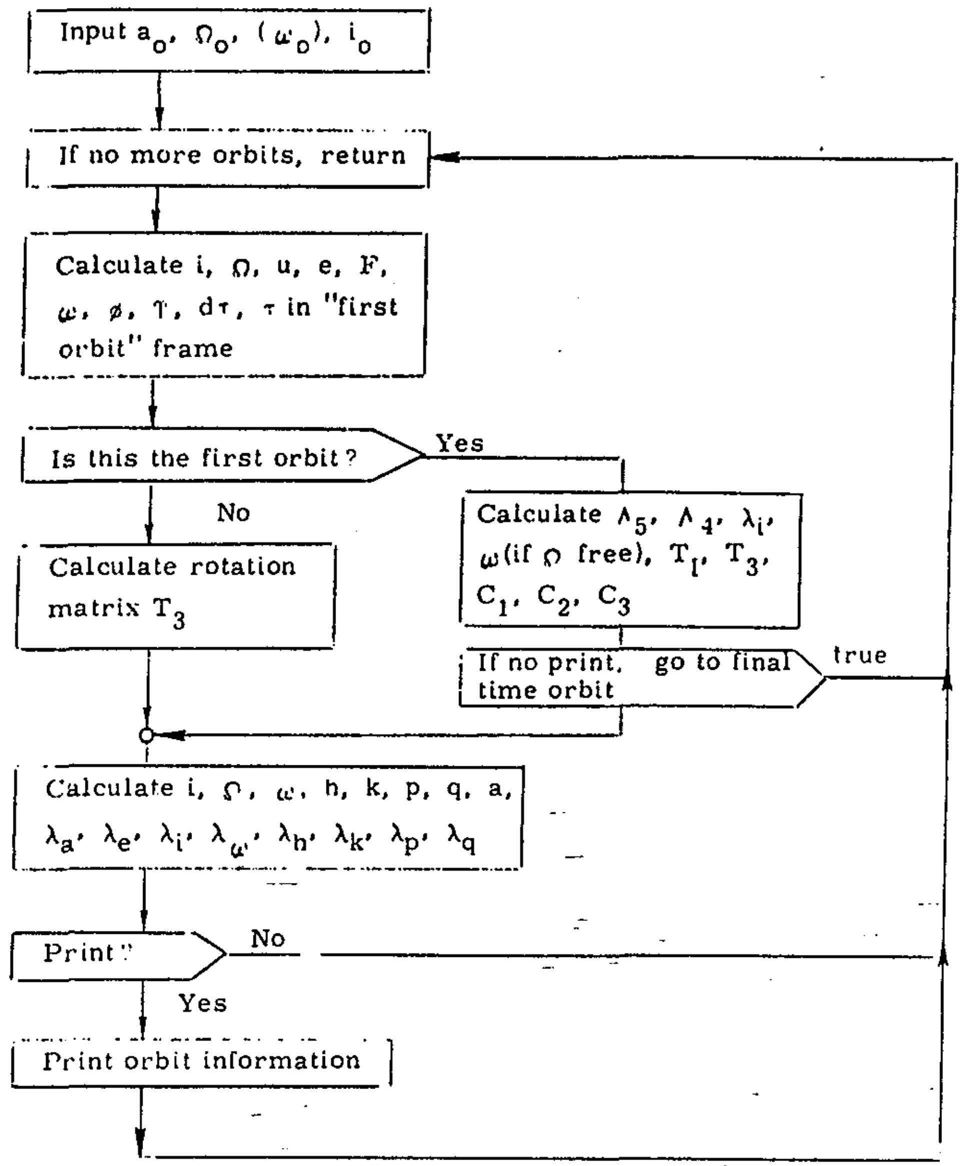
A	semimajor axis
E	eccentricity
F	true anomaly
H	h, angular momentum
S	S array (Table A-1, Ref. 1)
U	u (see Eqs. A.1, A.24, Ref. 1)
ω	ω (first orbit frame, Eq. A.26, Ref. 1)
Z	state and costate
AI	i (first orbit frame, Eqs. A.21, A.22, Ref. 1)
C1	see Eq. A.52, Ref. 1
C2	see Eq. A.52, Ref. 1
C3	see Eq. A.52, Ref. 1
DT	$d\tau$ or normalized ΔV for particular impulse
DV	ΔV
JM	total number of impulses
OM	Ω in first orbit frame (see Eq. A.23, Ref. 1)
PI	π
TR	T, rotation matrix (see Eq. A.29, Ref. 1)
UP	Υ out-of-plane thrust angle (see Eq. A.7, Ref. 1)
WI	initial ω in reference frame (see Eqs. A.48, A.49, Ref. 1)
W3	ω in reference frame (see Eqs. A.33, A.34, Ref. 1)
AII	initial i in reference frame (see Eq. A.29, Ref. 1)
AI3	i in reference frame (from Eq. A.30, Ref. 1)
ALA	λ_a (Eq. A.43, Ref. 1)
ALE	λ_e (Eq. A.44, Ref. 1)
ALI	λ_i (Eq. A.53, Ref. 1)

Subroutine OUTHI (Cont'd)

Fortran Variables: (Cont'd)

ALW	λ_{ω} (Eq. A.55, Ref. 1)
DTR	degrees to radians
IPR	print flag (no print if = 0)
OMI	initial Ω in reference frame (see Eq. A.29, Ref. 1)
OM3	Ω in reference frame (from Eq. A.30, Ref. 1)
PHI	ϕ, in-plane thrust angle (see Eq. A.7, Ref. 1)
TR1	T_I, rotation matrix (Eq. A.29, Ref. 1)
TR3	T_3, rotation matrix (Eq. A.30, Ref. 1)
ALOM	λ_{Ω} (Eq. A.62, Ref. 1)
NODE	flag (= 0 if Ω_o free, = 1 if Ω_o fixed)
ALAM3	Λ_3 (Eq. A.62, Ref. 1)
ALAM4	Λ_4 (see Eqs. A.63, A.64, Ref. 1)
ALAM5	Λ_5 (see Eqs. A.63, A.64, Ref. 1)
IDIM2	equal to dimension of the state (7)

Flow Chart for OUTHI



Subroutine OUTP (OUTPC)

Description:

Used to print information at each time step of the differential equation integration of the trajectory. Optionally it can calculate and print information for several points on an individual orbit such as attitude, thrust direction, sun angles.

Argument List:

T, Z, DERZ, IHLF, IDIM, PRMT

T	Time
Z	State and costate vector
DERZ	State and costate derivative
IHLF	Flag indicating error in integrator
IDIM	Dimension of state plus costate
PRMT	Parameter vector containing initial and final time, time step, the upper error bound, and a fifth element not used

Common Areas:

UNITS/UTS, UTM, UTH, UTD, UTKM, DTR, UTKG, UTCW, UTMS2

INT/IPR, ID, IDIMZ, NIMAX

A / A, AMU, PI

SHAD / FEN, FEX, DFEN(5), DFEX(5), ISHAD

POWER/P0, C, POW, ISUN, ISON, IPOW

ELEM/ Z0(12)

SOL/ RS(3), RSUN

ORBIT/NORB

ORBIT1/U0 (3)

ORBIT2/X1, Y1, RA

CONSTR/ICON

FUOUT/FJUMP(6), MJUMP

ORBIT3/CALP, SALP

DELAY / TT, FDELAY

DCOM / CD1, CD2

Subroutine OUTP (OUTPC) (Cont'd)

Called by:

RK4 or DRKGS

Calls Subroutines:

SUN, SHADOW, SDELAY, FCT, MVP, MMP

Subroutine OUTPC (OUTPCC)

Description:

After the iteration, this subprogram prints a summary of the converged iteration parameters, ΔV, t_f, etc.

Common Areas:

X MMM/ ZL0 (7), STEP(8), ZERF(8)

Z / ZF(14), DZ(14)

T / TF, T0

UNITS/UTS, UTM, UTH, UTD, UTKM, DTR, UTKG, UTKW UTMS2

ELEM/ZP0 (7), ZPF(5)

WF / WF(5)

TC / NOP

POWER / P0, C, POW, PH, ISUN, ISON, IPOW

DCOM / CD1, CD2

Called by:

CONTL

Subroutine PRTN (PRTNC)

Description:

Prints information about the iteration, e.g. x and y (see discussion in Section 2.3, Ref. 1).

Argument List:

KOUNT, NOI

KOUNT	The number of trajectory calls
NOI	The number of iterations

Common Areas:

XMMM / X(7), XS(8), Y(8)

T / TF, T0

Called by:

ITER

Subroutine QUAD (QUAD4, QUAD8, QUAD16, QUAD32)

Description:

These are 4, 8, 16, and 32 point vector Gaussian quadratures used for the averaging.

Argument List:

XL, XU, FCT, Y, Z, G, H, N

XL	Lower bound
XU	Upper bound
FCT	Dummy program name-called to evaluate integrand
Y	The resultant integrated function
Z	Orbital elements and adjoints if calling FLUX, otherwise full state and costate
G	Dummy variable
H	Dummy variable
N	Dimension

Called by:

FUNCT

Calls Subroutines:

FCT or FLUX

Subroutine RK4

Description:

This is 4 point Runge-Kutta integrator which is just the IBM Scientific Subroutine Package version without the accuracy checks. (see Ref. 10) It can be used interchangeably with the SSP integrator.

Argument List:

PRMT, Y, DERY, NDIM, IHLF, FCT, OUTP, AUX

PRMT(5)	Initial time measured from launch time, time of flight, integration step size, final 2 not used.
Y	State and costate (initial as input, final as output)
DERY	Final derivative
NDIM	Dimension of Y
IHLF	Flag = 12 if TF = T0 = 13 if DT(TF-T0)< 0
FCT	Dummy subroutine name - called to evaluate derivative
OUTP	Dummy output subroutine name
AUX	Dummy array used for intermediate calculation

Called by:

TRAJ

Subroutine SDELAY

Description:

Given a shadow entrance angle F_{en}, shadow exit angle, F_{ex} and $\frac{dF_{en}}{d\underline{\underline{z}}}$, $\frac{dF_{ex}}{d\underline{\underline{z}}}$. This subprogram calculates the delay time, the turn on angle and its partial (see Section 3.9, Ref. 1).

Argument List:

$\underline{\underline{Z}}$

Z	State and costate vector

Common areas:

A / AA, AMU, $\underline{\underline{PI}}$

SHAD / $\underline{\underline{FEN}}$, $\underline{\underline{FEX}}$, $\underline{\underline{DFEN}}$(5), $\underline{\underline{\underline{DFEX}}}$(5), IS

DELAY / $\underline{TT}$, $\underline{F1}$

Called by:

FUNCT, OUTP

Subroutine SHADOW (SHADOWC)

Description:

Sets up the shadow quartic equation (App. G, Ref. 1), solves it, checks the roots to find the entrance and exit of the orbit from shadow if there is intersection and calculates $\frac{dF}{d\underline{\bar{z}}}$ (Eq. G.5, Ref. 1) for entry and exit.

Argument List:

$\underline{Z}$

Z Orbital elements and their adjoints

Common Areas:

SOL / XSUN, YSUN, ZSUN, RSUN

SHAD / FEN, FEX, DFEN(5), DFEX(5), ISHAD

Called by:

FUNCT, OUTP

Subroutine START

Description:

Called by MAINE. This subprogram sets the initial values for the S array which contains most of the information about the orbits and the impulses. It calls SWITCH in order to make sure that the input satisfied the inequalities of Eq. A.4, Ref. 1.

Argument List:

E, F, XK, UP, XJ, HR, I, J, NF

E	Initial eccentricity (= 0)
F	Initial time anomaly (= 0)
XK	Initial value of k (Eq. A.8, Ref. 1)
UP	Initial value of T (Eq. A.7, Ref. 1)
XJ	Initial value of j (Eq. A.9, Ref. 1)
HR	Initial value of angular momentum, h(=1)
I	Set to 1
J	Orbit number (here=1)
NF	Flag set in SWITCH, here if ≤ 0 then the input fails to satisfy Eq. A.4 of Ref. 1

Common Areas:

STR / S (1, 3, 20)

Called by:

MAINE

Calls Subroutines:

SWITCH

Subroutine SUN (SUNC)

Description:

Calculates earth-sun unit vector in the equinoctial frame and the distance in A.U.'s and calculates flux factors – coefficients used in FLUX to find the latitude of the spacecraft and the partials of these coefficients.

Argument List:

T, Z

T	Time
Z	State and costate

Common Areas:

SOL / RS(3), R

TERRA/AE, EC, W, ENE, AN, COB, SOB

CCOM/ C(6)

Called by:

FUNCT, OUTP

Subroutine SWAP

Description:

This function routine solves for the cosine of the angle between the earth-sun vector and that portion of the primer vector which is perpendicular to the earth-spacecraft radius vector (Eq. 3.124, Ref. 1). It is a function of F, the eccentric longitude of the spacecraft in its orbit. It is used by FUNCT and DRTMI to find the point on the orbit at which this cosine is zero (and thus an angle of ± 90°) which indicates a possible jump point of the thrust direction.

Argument List:

F

F Eccentric longitude

Common Areas:

ZSWAP / Z(14)

SOL / RS(4)

BCOM / B(9)

Called by:

FUNCT, DRTMI

Calls Subroutines:

EVALMP

Fortran Variables:

AMU	μ
X(5)	orbital elements
Z(14)	state and costate
F	eccentric longitude
X1	$\hat{f}$ component of spacecraft radius vector (Eq. 3.19, Ref. 1)
Y1	$\hat{g}$ component of spacecraft radius vector (Eq. 3.20, Ref. 1)
AM	M (Eq. 3.33, Ref. 1)
VEC	unit of primer vector components perpendicular to radius vector
RS(3)	Earth-sun unit vector
SWAP	unit of VEC · RS

Subroutine SWITCH

Description:

When called by START or TIME this subprogram calculates the LHS (Eq. A.4c, Ref. 1) (called Q1). It also checks that all of (A.4) is not violated before (A.4c).

When called by MAINE it calculates the coast angle Δu (see Eq. A.6, Ref. 1), then calls DTDU to update the S array after the impulse just previously calculated and after the coast through Δu.

Argument List:

I, J, T, Q1, NF, KDU

I	Set to 1
J	Orbit number (= 1, 2, or 3)
T	Normalized ΔV
Q1	Value of LHS (Eq. A.4c, Ref. 1)
NF	Flag if 1 then $Q1 \geq 0$ 0 $Q1 = 0$ -1 $Q1 < 0$
KDU	Flag used in SWITCH to branch to appropriate calculations

Common Areas:

STR/S(1, 3, 20)

Called by:

MAINE, START, TIME

Calls Subroutines:

DTDU

Subroutine THRDR

Description:

Solves for the thrust angle ψ given $\cos^2\beta$, $\sin\alpha$, $\cos\alpha$, by setting up the cubic equation, Eq. 3.116 and then rejecting spurious roots.

Argument List:

CALP, SALP, CB2, PSI, CPSI, SPSI

CALP	$\cos(\alpha)$ α is the primer angle (Eq. 3.97, Ref. 1)
SALP	$\sin(\alpha)$, α is the primer angle
CB2	$\cos^2\beta$, β is the sun angle (Eq. 3.83, Ref. 1)
PSI	ψ, the thruster angle (Eq. 3.84, Ref. 1)
CPSI	$\cos\psi$
SPSI	$\sin\psi$

Called by:

FCT

Calls Subroutines:

DCUBIC

Fortran Variables:

A	coefficients of cubic, Eq. 3.116, Ref. 1
R	roots of the cubic
B1	$\tan^2\beta$, β is the sun angle
B2	$\tan\alpha$, α is primer vector angle
PI	π
RT	correct root of cubic
CB2	$\cos^2\beta$, β is the sun angle
NRE	number of real roots of cubic
ONE	1
PSI	ψ, thrust angle
CALP	$\cos\alpha$, α is the primer vector angle
CPS1	$\cos\psi$, ψ is the thrust angle
SALP	$\sin\alpha$, α is the primer vector angle

Flow Chart for THRDR

Input: $\cos^2\beta$, $\sin\alpha$, $\cos\alpha$

Check for special cases:

If $\cos^2\beta = 1$ or $\sin\alpha = 0$ or

$\cos\alpha = 0$ and $\cos^2\beta \geq .5$, then $\psi = \alpha$

If $\cos^2\beta = 0$, $\psi = \frac{1}{2}\alpha$ if α in quadrant 1 or 4

$= \frac{1}{2}\alpha + \frac{\pi}{2}$ if in quadrant 2

$= \frac{1}{2}\alpha - \frac{\pi}{2}$ if in quadrant 3

If $\cos\alpha = 0$ and $\cos^2\beta < .5$ get ψ from Eq. 3.118

where if $\sin\alpha < 0$, ψ is negative

Calculate $\cos\psi$, $\sin\psi$ and return

otherwise

Calculate coefficients of Eq. 3.116

Call DCUBIC which returns
3 values for $\tan\psi$

Check the real roots for the one
which has the same sign as $\tan\alpha$
(there is only one)

$\psi = \tan^{-1}(\text{root})$, $-\frac{\pi}{2} < \psi < \frac{\pi}{2}$

If $\cos\psi < 0$, $\psi = \psi + \pi$

Evaluate $\cos\psi$, $\sin\psi$ and return

Subroutine TIME

Description:

Called by MAINE, it iterates on τ (or ΔV) to find the $d\tau$ that satisfies A.4c of Ref. 1. It sets $d\tau$ = 1.D-6. Then increments $d\tau$ until A.4c is negative (A.4c is calculated by calling SWITCH). Then it uses a Newton-Raphson procedure to find the exact $d\tau$. If the N-R method has trouble, we have added another segment-halving technique to try to converge to the optimal $d\tau$.

Argument List:

I, J, DT1, SPAN, DTS

I	Set to 1
J	Orbit number (=1, 2, or 3)
DT1	Initial guess for $d\tau$, set to 0.D0 here
SPAN	Not used here
DTS	Resultant $d\tau$ (normalized ΔV)

Called by:

MAINE

Calls Subroutines:

SWITCH

Subroutine TRAJ (TRAJC, TRAJRKC)

Description:

Calculates a trajectory including the high and low thrust phases. If there is high thrust, the input to the high thrust code is set up. MAINE is called, then OUTHI is called to calculate the equinoctial state and costate. Then the Runge-Kutta integrator is called to calculate the low thrust phase. Finally the errors in the final conditions are calculated to send back to the iterator.

Common Areas:

XMMM / ZLO(7),STEP(8),ZERF(8)

TRA / TFMAX, DT0, UEB, EW(14)

Z / Z(14),DERZ(14)

INT / IPR,IDIM,IDIM2,NIMAX

T / TF,T0

ELEM / ZP0(7), ZPF(J)

DY / DYDT(8)

TC / NOP

HIGH / DVI1, IH1,NODE

A / A, AMU, PI

Called by:

CONTL, ITER

Calls Subroutines:

MAINE, OUTHI, RK4 or DRKGS, FUNCT

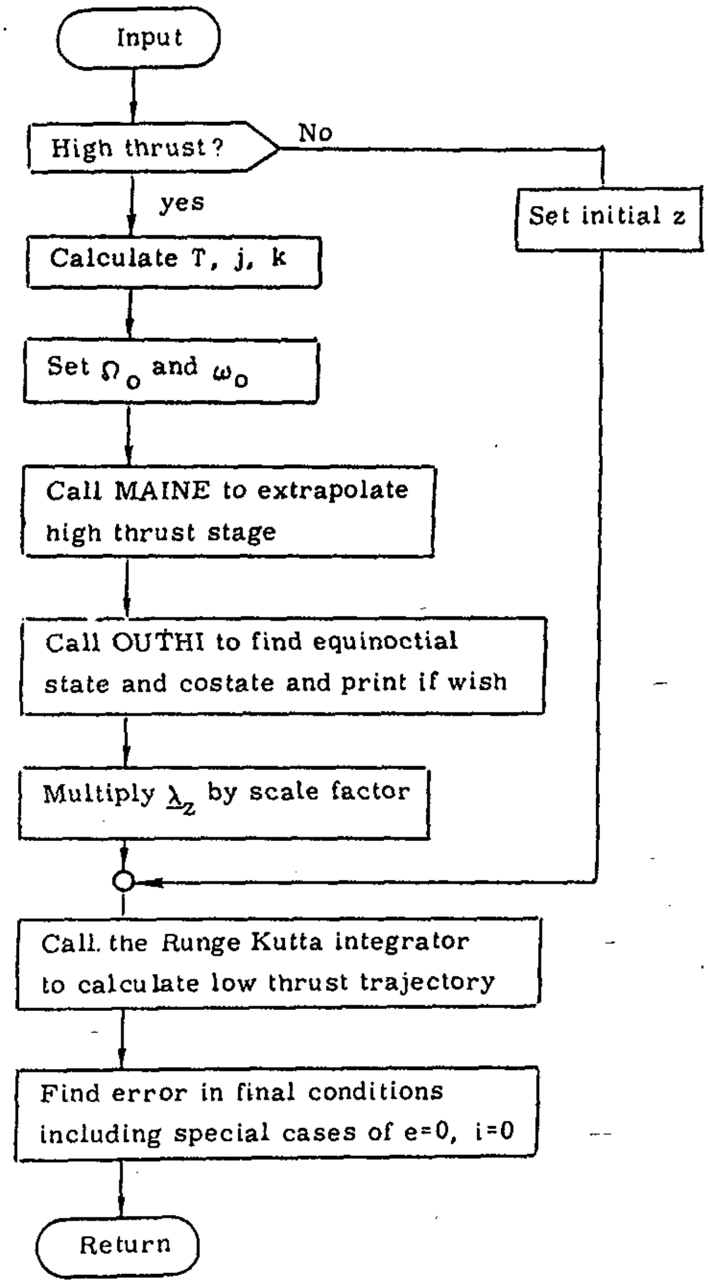
Fortran Variables:

H	Hamiltonian
Z	state and costate
EW	error weights (for PRKGS integrator)
H1	perturbed Hamiltonian
JM	total number of initial impulses
PI	π

Fortran Variables: (Cont'd)

TF	time of flight
T0	initial time (from launch)
UP	τ out-of-plane thrust direction (see Eq. A.7, Ref. 1)
WI	initial ω for high thrust phase
XJ	j, high thrust variable (Eq. A.9, Ref. 1)
XK	k, high thrust variable (Eq. A.8, Ref. 1)
DT0	time step for integrator
IHI	flag = 1 if no high thrust = 2 initial high thrust + low thrust
IPR	print flag (= 0 if no print)
NOP	flag = 1 if final Ω, ω are specified = 2 if final Ω, ω are free
OMI	initial longitude of ascending node (Ω)
TF1	perturbed final time
UEB	upper error bound (used in SSP integrator)
ZL0	initial iteration parameters
ZPF	desired final conditions
ZP0	initial orbital elements (a, h, k, p, q if low thrust only, if high thrust then a, i, (Ω))
DERZ	$\dot{\underline{z}}$, $\dot{\underline{\lambda}}_z$ at the final time
DVI1	total initial ΔV
DYDT	$\frac{\partial \underline{y}}{\partial t_f}$ where $\underline{y}$ is the final conditions
IDIM	dimension = 14
IHLF	error flag from integrator
NODE	flag = 0 if Ω_o free = 1 if Ω_o fixed
ZERF	the error in the final conditions
DERZ1	perturbed $\dot{\underline{z}}$, $\dot{\underline{\lambda}}_z$
IDIM2	dimension of the state (= 7)

Flow Chart for TRAI



LIST OF REFERENCES

1. Sackett, L. L., H. L. Malchow, T. N. Edelbaum, Solar Electric Geocentric Transfer with Attitude Constraints: Analysis, NASA CR-134927 (Draper Lab R-901), The Charles Stark Draper Laboratory, Inc., August, 1975.

2. Sackett, L. L., H. L. Malchow, T. N. Edelbaum, The Computation of Time Optimal Geocentric Transfers Using Solar or Nuclear Electric and High Thrust Propulsion, Report No. R-880, The Charles Stark Draper Laboratory, Inc., May 1975.

3. Sackett, L. L., T. N. Edelbaum, H. L. Malchow, A Users Manual for a Computer Program which Calculates Time Optimal Geocentric Transfers Using Solar or Nuclear Electric and High Thrust Propulsion, Report No. R-827, The Charles Stark Draper Laboratory, Inc., June 1974.

4. SERT-C Project Study, NASA Technical Memorandum, NASA TMX-71508, January 1974.

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

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

7. Cake, J. E., and J. D. Regetz, Jr., "Development of a Unified Guidance System for Geocentric Transfer", NASA-TMX-7-648, 1975.

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

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

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

11. Melbourne, W. G., et. al., Constants and Related Information for Astrodynamic Calculations, 1968, NASA TR32-1306, JPL, Pasadena, 1968.

12. Bottom, R. H., Astronautical Guidance, McGraw Hill, 1964.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1975	Final Contractor Report

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Solar Electric Geocentric Transfer With Attitude Constraints: Program Manual	NAS3–18886
6. AUTHOR(S)	
Lester L. Sackett, Harvey L. Malchow, and Theodore N. Edelbaum	

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
The Charles Stark Draper Laboratory, Inc. 68 Albany Street Cambridge, Massachusetts 02139	E–12239–1

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
National Aeronautics and Space Administration Cleveland, Ohio 44135–3191	NASA CR–134927

11. SUPPLEMENTARY NOTES

Project Manager, James Cake, Spacecraft Technology Division, NASA Lewis Research Center, Cleveland, Ohio.

12a. DISTRIBUTION/AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Unclassified - Unlimited	

13. ABSTRACT ***(Maximum 200 words)***

This report contains a general introduction and description of the program and detailed information on input, output, use of code, and the individual subroutines including listings of the code. The analysis is discussed in a separate report.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Solar electric; Geocentric transfer; Optimal trajectories; Low thrust; Attitude constraints	179
	16. PRICE CODE
	A09

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	