954724

c-4947

SUNSPOT—A COMPUTER PROGRAM FOR PRODUCING OPTIMAL SOLAR SAIL PLANETOCENTRIC TRAJECTORIES

by

Lester L. Sackett

September 1977

(NASA-CR-157129) SUNSPOT: A COMPUTER PROGRAM FOR PRODUCING OPTIMAL SOLAR SAIL PLANETOCENTRIC TRAJECTORIES (Draper (Charles Stark) Lab., Inc.) 103 p HC A06/MF A01 CSCL 22A G3/13

N78-24250

Unclas

20707

The Charles Stark Draper Laboratory, Inc.

Cambridge, Massachusetts 02139

TECHNICAL REPORT STANDARD TITLE FAGE

1. Report No.

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle

SUNSPOT—A COMPUTER PROGRAM FOR PRODUCING OPTIMAL SOLAR SAIL PLANETOCENTRIC TRAJECTORIES

5. Report Date

September 1977

6. Performing Organization Code

7. Author(s)

Lester L. Sackett

8. Performing Organization Report No

C-4947

9. Performing Organization Name and Address

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

10. Work Unit No.

11. Contract or Grant No.

NASA 7-100

13. Type of Report and Period Covered

12. Sponsoring Agency Name and Address

NASA
Washington, D.C.

14. Sponsoring Agency Code

15. Supplementary Notes

Technical Monitor, David B. Smith
Jet Propulsion Laboratory, Pasadena, California

16. Abstract

This report contains a description of the input, output and subroutines, including listings, for the SUNSPOT code. SUNSPOT can calculate time optimal planetocentric trajectories including orbit-to-orbit transfer and orbit to a subescape point. Trajectories about the four inner planets can be calculated, and shadowing, oblateness, and solar motion may be included. A penalty function may be included to prevent trajectories which intersect the planet's surface.

17. Key Words Suggested by Author

Solar Sail
Optimal Trajectories

18. Distribution Statement

Unclassified—Unlimited

19. Security Classif. (of this report)	20. Security Classif (of this page)	21. No. of Pages	22. Price
UNCLASSIFIED	UNCLASSIFIED	103	

c-4947

SUNSPOT—A COMPUTER PROGRAM FOR PRODUCING OPTIMAL SOLAR SAIL PLANETOCENTRIC TRAJECTORIES

by

Lester L. Sackett

September 1977

Approved: Donald Fraser

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

ACKNOWLEDGMENT

This report was prepared by The Charles Stark Draper Laboratory, Inc. under Subcontract 954724 with the Jet Propulsion Laboratory; NASA Contract NAS 7-100.

Publication of this report does not constitute approval by the Jet Propulsion Laboratory or NASA of the findings or conclusions contained herein, It is published for the exchange and stimulation of ideas.

TABLE OF CONTENTS

I

SECTION 1

INTRODUCTION

This report is meant to supplement Draper Laboratory Report R-1113, Optimal Solar Sail Planetocentric Trajectories.[(1)*] it contains information about the computer code, called SUNSPOT (for SUN-Sail Program for Optimal Trajectories), which was used to generate the results in the report, and should allow one to use this code to produce additional solar sail trajectories. The principal contents are a description of the input and output and brief descriptions of the subroutines and listings of all the subroutines. The code is functionally similar to SECKSPOT, a solar electric orbit transfer program.[(2,3)]

The program calculates the solution to a two-point boundary-value problem which arises from the application of optimal control theory. A general flow chart is shown in Figure 1. Figure 2 is a system diagram which illustrates the connection of various subroutines which make up the deck, The subroutines above call those below.

Total size of the object deck compiled on an Amdahl 470 in Fortran H is approximately 59,000 bytes. The size of the individual subroutines is shown in Table 1. A Fortran Scientific Subroutine Package (SSP) routine called DRTNI is also needed, but not shown in the table.[(4)]

A plotting capability was developed at Draper Laboratory. Data can be written by the output subroutine and stored for later plotting. Since plotting capabilities vary from facility to facility, that function is not discussed in this report.

---

[*] Superscript numerals refer to similarly numbered items in the List of References.

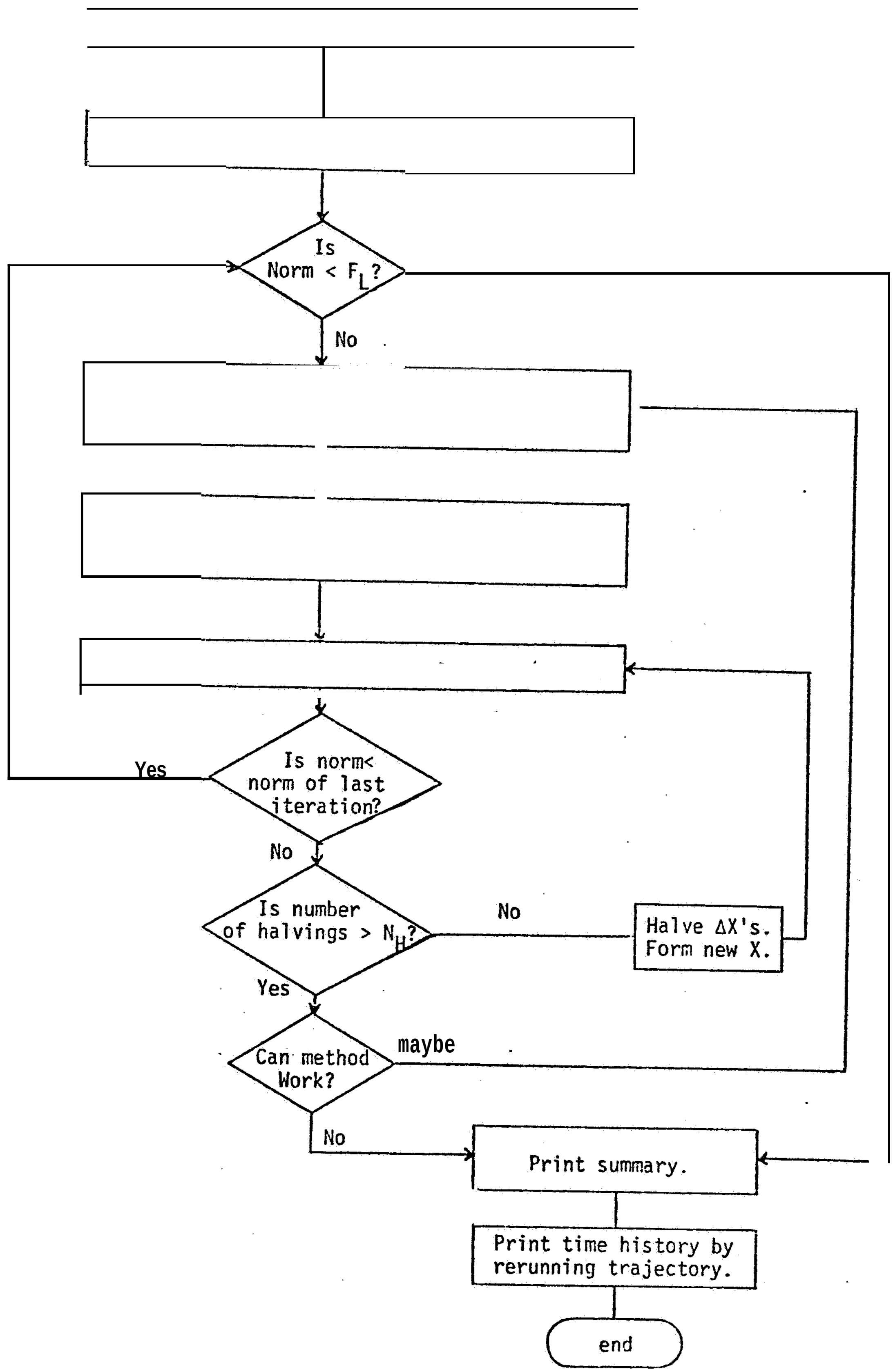

Figure 1. Overall flowchart.

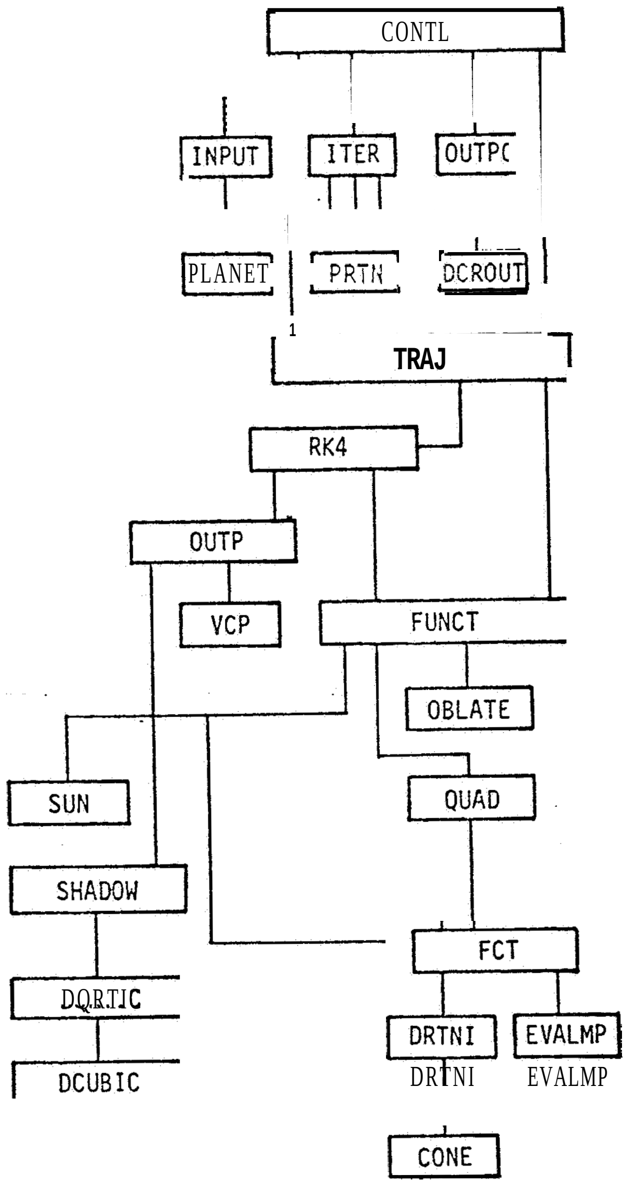

Figure 2. System **diagram.**

ORIGINAL PAGE IS
OF POOR QUALITY

a

. Table i. Program subroutine object deck size (thousands of bytes! .

CONE	0.9	OUTP	4.8
CONTL	0.6	OUTPC	2.2
DCUBIC	1.2	PLANET	1.5
DCROUT	2.5	PRTN	0.6
DQRTIC	1.3	QUAD 4	1.0
EVALMP	5.2	QUAD 8	1.2
FCT	3.4	QUAD 16	1.8
FUNCT	2.5	QUAD 32	2.9
INPUT	9.5	RK4	1.7
ITER (NR)	3.2	SHADOW	3.5
ITER (MODNR)	3.9	SUN	1.4
OBLATE	2.0	TRAJ	2.1

SECTION 2

INPUT

The quantities discussed in this section are all read by the sub-program 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 inputs for the deck are listed in this section along with a brief description and nominal values, if any.

IPNUM planet flag: 1 = Mercury, 2 = Venus, 3 = Earth, 4 = Mars (format, I1)

Initial orbit

W(1) (km) semimajor axis

W(2) eccentricity

W(3) (degrees) inclination

W(4) (degrees) longitude of ascending node

W(5) (degrees) argument of pericenter

Initial guesses

ZL0(1) λ_a, adjoint to semimajor axis

ZL0(2) λ_h, adjoint to orbital element h

ZL0(3) λ_k, adjoint to orbital element k

ZL0(4) λ_p, adjoint to orbital element p

ZL0(5) λ_q, adjoint to orbital element q

The desired final orbit

WF(1) (km) semimajor axis

WF(2) eccentricity, not used if NOP = 3

WF(3) (deg) inclination, not used if NOP = 3

WF(4)	(deg) longitude of ascending node, not used if NOP = 2 or 3
WF(5)	(deg) argument of pericenter, not used if NOP = 2 or 3
TF2	(days) guess for final time
AC	(mm/s^2) characteristic acceleration.
TL	Julian date at initial time

The following input nay 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 (for intermediate trajectories)	0
3	NIMAX	maximum number of iterations (if 0, bypass iteration to print time history)	20
4	TFMAX2	(days) maximum flight time	500.
5	DT2	(days) time step for integrator	10.
6	UEB	upper error bound for integrator (not used with RK4)	1.D10
7	EW	error weights for integrator (5D6.1, 2 lines)(not used with RK4)	1.,1,1,1,1,0 ...
8	UTKM	equatorial radius (km)	Set in PLANET
9	GM	(km3/s2) gravitational constant	Set in PLANET
10	NO?	= 1, five orbital elements specified at TF	1
		= 2, three orbital elements specified at TF	
		= 3, semimajor axis specified at TF	
11	Sets oblateness	AJ2, = 0.D0	Set in PLANET
12	STEP	step size for numerical differentiation in ITER, 6 dim., sixth 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 (6))	

13	ISON	= 0, shadow effect off	0
		= 1, shadow effect on	
14	ISUN	= 0, sun distance effect on thrust off	0
		= 1, effect on	
15	CD(1)	coefficients in	0.5
	CD(2)	acceleration factor	0.5
	CD(3)		0.0
16	IE	= 1, if equatorial frame, solar motion	1
		= 2, if equatorial frame, no solar motion	
		= 3, if ecliptic frame, solar notion	
		= 4, if ecliptic frame, no solar motion	
		= 5, if planetary frame, solar motion	
		= 6, if planetary frame, no solar motion	
17	FLIM	norm limit in iteration routine (format, D6.1)	1.D-6
18	XNU	pericenter penalty function weighting 'factor	0.0
19	NORB	= 0, no orbit print	0
		= 1,..., 999, orbit print on	
		NORB points of an orbit (format, I3)	
20	Sets flag so that plot data is stored		
21	IQ	orbit divided into IQ separate quadrature intervals (1-10)	2

SECTION 3

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 tine 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	semimajor axis
E	eccentricity
I	inclination
LON ASC NODE	longitude of ascending node
ARG PERIC	argument of pericenter
M/S	meters/second
P.R.	planet radii
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/second2

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 (XLO), then Y, the error in the final conditions. The final conditions are the final values of a, h, k, p, q, H, if NOP = 1. If NOP = 2 the final conditions are a, e, tan $\frac{i}{2}$, λ_ω, λ_Ω, 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. If NOP = 3, the final conditions are a, λ_h, λ_k, λ_p, λ_q, H. Then the final time (TF) is printed in internal units, followed by, FO, 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 inputted 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 FO, a new iteration begins; if it is larger than FO, the DELX:S are halved and printed. This continues until F1 < FO 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, and the final time.

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.

The low thrust trajectory time history at each time step is printed. First is printed TIME in various units. The time step number is also printed. Next are printed the equinoctial orbital elements (a, h, k, p, q) followed by the classical orbital elements (a, e, i, Ω, ω). Next is the costate, then the state derivative, then the costate derivative and then the value of the Hamiltonian, the period (hours), pericenter and apocenter (km). The characteristic acceleration divided by the sun distance squared, and the value of C_3. If shadowing is included the time spent in shadow is printed in hours and as a fraction of the period (if the orbit passed through shadow). The entry angle (°) and exit 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 planet-sun distance in units of the planets semimajor axis. 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 planet radii. The following spacecraft parameters are printed with the spacing indicated here:

PRIMER CONE A	THRUST CONE A	PSIV	ACC FACT	T/W RATIO
UF	UG	UW		

All angles are in degrees. The meaning of these terms follows, Equation numbers refer to Reference 1.

PRIMER CONE A	the primer vector cone angle, β, as in Eq, (3.50)
THRUST CONE A	the force vector cone angle, θ, as in Eq. (3.27)
PSIV	the force vector clock angle, ψ, as in Eq. (3.52), except that ψ has the opposite sign, where the reference vector is the velocity vector
ACC FACTOR	the value of the acceleration factor, $c_1 + c_2 \cos 2\theta + c_3 \cos 4\theta$ as in Eq. (3.26)
T/W RATIO	the ratio of the photon force with the local gravitational force
UF, UG, UW	the $\hat{f}$, $\hat{\underline{g}}$, and $\hat{w}$ components, respectively, of the thrust acceleration unit vector, in the equinoctial coordinate system

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 usually arises from small numerical inaccuracies in solving the quartic equation and can usually be ignored unless a number of orbits have intersected the planet's surface.

SECTION 4

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 areas. Equation numbers refer to Reference 1.

Common area: A/A, AMU, PI

A	Characteristic acceleration divided by semimajor axis of planet squared
AMU	Gravitational constant, μ
PI	π

Shared by:

FCT, FUNCT, INPUT, OUTP, TRAJ

Common area: CCOM/CD(4)

CD(1), CD(2), CD(3)	C_1, C_2, C_3 coefficients in acceleration factor, Eq. (3.3)
CD(4)	Cutoff value of θ, Eq. (3.7)

Shared by:

FCT, INPUT, OUTP, OUTPC, CONE

Common area: CON/CB, SB, THETA, CTHETA, STHETA, C2T, C4T, S2T, S4T

CB	Cosine of primer vector cone angle
SB	Sine of primer vector cone angle
THETA	Force vector cone angle
CTHETA	Cosine of force vector cone angle
STHETA	Sine of force vector cone angle

Table 2. Common areas.

SUBROUTINES

COMMAND AREAS	CONE	CONTL	EVALMP	FCT	FUNCT	INPUT	ITER	OUTP	OUTPC	PLANET	PRTN	SHADOW	SUN	TRAJ
A				X	X	X		X						X
CCOM	X			X		X		X	X					
CON	X			X				X						
DY							X							X
ELEM						X			X					X
F						X	X							
INT		X				X	X	X						X
IS					X	X		X						
JD						X				X				
J2					X	X				X				
NU					X	X								X
ORBIT						X		X						
ORBIT1				X				X						
ORBIT2			X	X				X						
PLNTS						X				X				
PLOT						X		X						
Q					X	X								
SHAD					X			X				X		
SOL				X	X			X				X	X	
T						X	X		X		X			X
TC						X			X					X
TERRA						X				X			X	
TRA						X								X
UNITS						X		X	X	X				
WF						X			X					
XMMM						X	X		X		X			X
Z									X					X

C2T	Cosine of two times the force vector cone angle
C4T	Cosine of four times the force vector cone angle
S2T	Sine of two times the force vector cone angle
S4T	Sine of four times the force vector cone angle

Shared by:

FCT, OUTP, CONE

Common area: DY/DYDT(6)

DYDT	The partial of the final conditions with respect to time

Shared by:

ITER, TRAJ

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

ZP0	The initial state
ZPF	The desired final equinoctial orbital elements

Shared by:

INPUT, 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 increment or a fractional increment

Shared by:

INPUT, ITER

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 (10)
IDIM2	Equal to dimension of the state (5)
NIMAX	The maximum number of iterations of the iterator

Shared by:

CONTL, INPUT, ITER, OUTP, TRAJ

Common area: IS/ISUN, ISON

ISUN	Flag indicating if solar distance effect is included
ISON	Flag indicating if shadowing is included

Shared by:

FUNCT, INPUT, OUTP

Common area: JD/TL

TL Julian data at launch

Shared by:

PLANET, INPUT

Common area: J2/AJ2

AJ2 The oblateness coefficient

Shared by:

FUNCT, INPUT, PLANET

Common area: NU/XNU

XNU Weighting constant in penalty function, Eq. (3.73)

Shared by:

FUNCT, INPUT, TRAJ

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), PA

VEC The thrust direction vector

PA Acceleration factor, Eq.. (3.3)

Shared by:

FCT, OUTP

Common area: ORBIT2/X1, Y1, PR(2, 2), X1DOT, Y1DOT

X1 $\hat{f}$ component of spacecraft location in orbit (Eq. 3.16, Reference 1)

Y1 $\hat{g}$ component of spacecraft location in orbit (Eq. 3.17, Reference 1)

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

X1DOT $\hat{f}$ component of velocity, Eq. (3.18)

Y1DOT $\hat{g}$ component of velocity, Eq. (3.19)

Shared by:

EVALMP, FCT, OUTP

Common area: PLNTS/IPNUM

IPNUN Integer indicating planet number

Shared by:

INPUT, PLANET

Common area: PLOT/IPL

IPL Flag indicates if data should be written and stored for later plotting

Shared by:

INPUT, OUTP

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.

Shared by:

FUNCT, INPUT

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

ISHAD A flag indicating whether or not a particular orbit passed through the planet's shadow

Shared by:

FUNCT, OUTP, SHADOW

Common area: SOL/RSUN(3), RS

RSUN The unit vector pointing toward the siin from the planet

RS The distance from planet to sun divided by semimajor-axis

Shared by:

FCT, FUNCT, OUTP, SHADOW, SUN

Common area: T/TF, TO

TF Flight time

TO The initial time (measured from launch time)

Shared by:

INPUT, ITER, OUTPC, PRTN, TRAJ

Common axea: TC/NOP

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

Shared by :

INPUT, OUTPC, TRAJ

Common area: TERRA/AE, EC, W, ENE, AN, ECLMTX, EQUMTX

AE	Planet's semimajor axis (in A.U.)
EC	Eccentricity of planet's orbit
W	Planet's argument of perihelion
ENE	Planet's mean orbital motion
AN	The mean anomaly of the planet at the time of launch
ECLMTX	Matrix €or conversion to ecliptic coordinate frame
EQUMTX	Matrix for conversion to equatorial frame

Shared by:

PLANET, SUN, INPUT

Common area: TRA/TFMAX, DT, UEB, EW(10)

TFMAX	The maximum time of flight allowed
DT	The time step used by the differential equation integrator
UEB	The upper error bound (not used by RK4)
EW	Error weights (not used by RK4)

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/second^2

Shared by

PLANET, INPUT, OUTP, OUTPC

Common area: WF/WF(5)

WF	The desired final classical orbital elements

Shared by:

INPUT, OUTPC

Common area: XMMM/ZL0(5), STEP(6), ZERF(6)

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(10), DZ(10)

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

Shared by:

OUTPC, TRAJ

SECTION 5

SUBROUTINE DESCRIPTION AND LISTINGS

This section ccntains descriptions of all subroutines including input and output, common areas, subroutines which are called or called by. 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 neme. In addition to the subroutines contained in this section, an IBM Scientific Subroutine, DRTNI is needed.[4] Equation numbers refer to the final report, Reference 1.

Subroutine CONE

Description:

Calculates $\frac{\partial H}{\partial \theta}$ and its derivative (see Eq. (3.60) and (3.64)). Used to calculate cone angle by the Newton method.

Argument List:

THETA, FT, DERFT

THETA	Iterative value of thrust cone angle
FT	Iterative value of $\frac{\partial H}{\partial \theta}$
DERFT	Iterative value of $\frac{\partial^2 H}{\partial \theta^2}$

Common Areas:

CCOM/CD(4)

CON/CB, SB, DUB(7)

Called by:

DRTNI

```
C CCHE
C VERSION OF CONE TO BE USED WITH NEWTON ITERATOR
C CALLED BY DRTNI
C CALCULATES DH/DTHETA AND DERIVATIVE
C
      SUBROUTINE CONE(THETA,FT,DERFT)
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON /CCOM/CD(4)
      COMMON /CON/CB,SB,DUB(7)
C
C
C
      CTHETA= DCOS(THETA)
      STHETA= DSIN(THETA)
      C2T= DCOS(2.D0*THETA)
      C4T= DCOS(4.D0*THETA)
      S2T= DSIN(2.D0*THETA)
      S4T= DSIN(4.D0*THETA)
C
      B= CD(1)+CD(2)*C2T+CD(3)*C4T
      BP= -2.D0*CD(2)*S2T-4.D0*CD(3)*S4T
      BPP= -4.D0*CD(2)*C2T-16.D0*CD(3)*C4T
      D= CTHETA*CB+STHETA*SB
      DP= -STHETA*CB+CTHETA*SB
      DPP= -D
C
      FT= BP*D+B*DP
      DERFT= 2.D0*BP*DP+BPP*D+B*DPP
C
      RETURN
      END
```

Subroutine CONTL

Description:

This is the main "driver" program.

Common Areas:

INT/IPR, IDIM, IDIM2, NIMAX

Calls Subroutines:

INPUT, ITER, OUTPC, TRAJ

```
  CCNTL/CONTLSS

C SOLAR SAIL
C THIS IS THE MAIN CONTROLLING PROGRAM FOR FINDING THE
C   OPTIMAL TRAJECTORY FOR PLANETOCENTRIC SOLAR SAIL
C   SPIRAL, ESCAPE OR CAPTURE.
C
C
C
C
      IMPLICIT REAL*8(A-H,O-Z) , INTEGER (I-N)
      COMMON /INT/IPR, IDIM, IDIM2, NIMAX
      EXTERNAL TRAJ, PRTN
C
C
C
C ALL SETTING OF CONSTANTS AND INITIAL READ AND WRITES
C    ARE DONE BY THE SUEEOUTINES INPUT, PLANETS
C
      CBLL INPUT
      IF (NIMAX.EQ.0) GC TO 10
      WRITE: (6,7001)
C
C THE ITERATOR ROUTINE SOLVES THE 2PBVP FOR TEE OPTIMAL TRAJECTORY
C
      CALL ITER(KOUNT, HI, TRAJ, PRTN)
      IF (NI .EQ. 9999) WRITE (6,1002)
      WRITE (6,1003)
C
C OUTPC PRIWTS A SUMMARY OF THE OPTIYAL TRAJECTORY VALUES
C
      CALL OUTPC (KOUNT)
C
C TIME AISTORY OF THE OPTIMAL TRAJECTORY IS CALCULATED BPD PRINTED
C
 10   RRITE (6,1004)
      IPR= 1000000
      CALL TRAJ
C
      IF (NI.EQ.9999) WRITE (6,1002)
      IF (NI.BE.9999)  YRITE (6,1005)
      STOP
 1001 PORKAT (18H1 ITERATION BEGINS)
 1002 FORMAT (29H0 OPTIMIZATION NOT SUCCESSFUL)
 1003 FORMAT (43H1 CONVERGED VALUES FOR OPTIMIZED TRAJECTORY)
 1004 PORKAT (36H1 TIME HISTORY OF OPTIMAL TRAJECTORY)
 1005 FORMAT (30H0 PROGRAM HAS RUN SUCCESSFULLY)
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

Subroutine DCROUT (DCROUT10)

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, B, 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
E?S	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

a

```
DCROUT/DCROUT10   6 DIM.
      SUEHODTINE DCROUT(AA,R,D,EPS,NI,M,IND)
      DOUBLE PRECISION A,R,D,EPS,T,S,P,DT,AA
      DIMENSION A(6,6),R(6,1),AA(6,6)
    5 IND=0
      N=NI
      DO 6 I=1,N
      DO 6 J=1,N
    6 A(I,J)=AA(I,J)
      IF(M)10,25,25
   10 M=N
      DO 20 I=1,N
      DO 15 J=1,N
   15 R(I,J)=0.D0
   20 R(I,I)=1.D0
   25 IC=0
      II=0
      T=DABS(A(1,1)   )
      DO 35 I=2,N
      IF(T-DABS(A(I,1)))30,35,35
   30 II=I
      T=DABS(A(I,1))
   35 CONTINUE
      IF(II)40,65,40
   40 IC=IC+1
      IF(M)45,55,45
   45 DO 50 J=1,M
      S=R(1,J)
      R(1,J)=R(II,J)
   50 R(II,J)=S
   55 DO 60 J=1,N
      S=A(1,J)
      A(1,J)=A(II,J)
   60 A(II,J)=S
   65 P=A(1,1)
      IF(DABS(P)-EPS)70,70,75
   70 IND=1
      D=0.D0
      GO TO 200
   75 DO 80 J=2,N
   80 A(1,J)=A(1,J)/P
      I?(M)85,95,85
   85 DO 90 J=1,M
   90 R(1,J)=R(1,J)/P
   95 DO 170 K=2,N
      KM=K-1
      T=-1.D0
      DO 105 I=K,N
      DO 98 J=1,KM
   98 A(I,K)=A(I,K)-A(I,J)*A(J,K)
      IF(T-DABS(A(I,K)))100,105,105
  100 T=DABS(A(I,K))
      II=I
  105 CONTINUE
      IF(II-K)110,135,110
  110 IC=IC+1
      IF(M)115,125,115
  115 DO 120 J=1,M
      S=R(K,J)
      R(K,J)=R(II,J)
  120 R(II,J)=S
```

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

Subroutine DCUBIC

Description:

Calculates the roots of a cubic equation.

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

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

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

Subroutine EVALMP (EVALMPSS)

Description:

Evaluates M and $\frac{\partial M}{\partial \bar{z}}$ (see Eqs. (3.23) and (3.36)). The form of M which is coded in EVALMP is that shown in Ref. 5, Table 4. It was also from this form that $\frac{\partial M}{\partial \underline{z}}$ 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 $M = \frac{\partial \underline{z}}{\partial \dot{\underline{r}}}$ (Eq. (3.24))
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

```
C EVALMP/EVALMPSS
C
C
C  THIS SUBROUTINE EVALUATES THE 5X3 MATRIX M AND THE
C  5X3X5 PARTIAL OF M WRT X
C
C  IF IMFLAG=1, BOTH M (AM) AND ITS PARTIAL (PAM) ARE EVALUATED
C  IF IMFLAG=2, ONLY M (AM) IS EVALUATED
C  IF IMFLAG=3, ONLY THO, PARTIAL OF M (PAM) IS EVALUATED
C.
C
      SUBROUTINE EVALMP(X, THETA, AMU, AM, PAM, IMFLAG)
      IMPLICIT REAL*8(A-H,O-Z) , INTEGER (I-N)
      DIMENSION X(5), AM(5,3), PAM(5,3,5)
      COMMON /ORBIT2/ X1,Y1,RA,PZ20,PZ26,PZ29,PZ35,X1DOT,Y1DOT
C
C
      EN=DSQRT(AMU/X(1)**3)
      RHO= DSQRT(1.D0- X(2)**2- X(3)**2)
      BETA= 1.D0/(1.D0 +RHO)
      CT= DCOS(THETA)
      ST= DSIN(THETA)
      RA= 1.D0-X(3)*CT -X(2)*ST
      ZETA= X(3)*ST-X(2)*CT
      BETA3= BETA**3/(1.D0 -BETA)
      X1= X(1)*((1.D0 -X(2)**2*BETA)*CT +X(2)*X(3)*BETA*ST -X(3))
      Y1= X(1)*((1.D0 -X(3)**2*BETA)*ST +X(2)*X(3)*BETA*CT -X(2))
      X11=X1
      Y11=Y1
      X1DOT= -((1.D0 -X(2)**2*BETA)*ST -X(2)*X(3)*BETA*CT)*EN*X(1)/RA
      Y1DOT= ((1.D0 -X(3)**2*BETA)*CT -X(2)*X(3)*BETA*ST)*EN*X(1)/RA
      PZ1= X(1)*(ZETA*(BETA+X(2)**2*BETA3) -(X(2)*BETA -ST)*CT/RA)
      PZ2= -X(1)*(-ZETA*X(2)*X(3)*BETA3 +1.D0 +(ST -X(2)*BETA)*ST/RA)
      PZ3= X(1)*(-ZETA*X(2)*X(3)*BETA3-1.D0 +(X(3)*BETA -CT)*CT/RA)
      PZ4= X(1)*(-ZETA*(BETA +X(3)**2*BETA3) +(CT -X(3)*BETA)*ST/RA)
      IF (IMFLAG .EQ. 3) GO TO 10
C  IT DO NOT WANT TO EVALUATE PARTIAL OF M, BRANCH TO 10
      AM(1,1)= 2.D0*X1DOT/(EN**2*X(1))
      AM(1,2)= 2.D0*Y1DOT/(EN**2*X(1))
      AM(1,3)=0.D0
       DUM= RHO/(EN*X(1)**2)
      AM(2,1)= DUM*(PZ2- X(2)*BETA*X1DOT/EN)
      AM(2,2)= DUM*(PZ4 -X(2)*BETA*Y1DOT/EN)
      AM(2,3)= DUM*(X(3)*(X(5)*Y1 -X(4)*X1))/RHO**2
      AM(3,1)= -DUM*(PZ1 +X(3)*BETA*X1DOT/EN)
      AM(3,2)= -DUM*(PZ3 +X(3)*BETA*Y1DOT/EN)
      AM(3,3)= -DUM*(X(2)*(X(5)*Y1 -X(4)*X1)/RHO**2)
      AM(4,1)=0.D0
      AM(4,2)=0.D0
      DUM= (1.D0 +X(4)**2 +X(5)**2)/(2.D0*EN*X(1)**2*RHO)
      AM(4,3)= DUM*Y1
      AM(5,1)=0.D0
      AM(5,2)=0.D0
      AM(5,3)= DUM*X1
      IF (IMFLAG .EQ. 2) RETURN
C  IF WE ONLY WISH TO EVALUATE M THEN PROGRAM RETURNS HERE
 10   CA= DSQRT(AMU/X(1))/RA
      PZ5= X(2)*BETA3
      PZ6= X(3)*BETA3
      PZ9= CA*ST/RA
      PZ10= CA*CT/RA
```

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

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

Subroutine FCT (FCTSS)

Description:

Evaluates what is essentially the preaverzged derivative (see Section 3.3, Reference 1, Eq. (3.33), (3.34)) of the state and costate due to thrusting. Specifically,

$$\left(\frac{\partial H}{\partial \lambda} \frac{dt}{dF}\right) = \underline{\dot{z}} \frac{dt}{dF} \frac{R^2}{a}$$

$$\left[\frac{\partial H}{\partial \underline{\lambda}} \frac{dt}{dF} + H \frac{\partial}{\partial \underline{\lambda}} \left(\frac{dt}{dF}\right)\right] \frac{R}{a}$$

The factor $\frac{a_c}{R^2}$ and the minus sign of Eq. (3.34) and the factor $\frac{1}{2\pi}$ are taken into account in FUNCT. When FCT is called by **OUTP**, the thrust direction is calculated end 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 $\dot{z}$ and $\dot{\lambda}$ corresponding to F1
G	Preaveraged $\underline{\dot{z}}$ and $\underline{\dot{\lambda}}$ at F_2

Common Areas:

A/A, AMU, PI

ORBIT1/VEC(3), PA

ORBIT2/X1, Y1, RA, PR(2,2), X1DOT, Y1DOT

CCOM/CD(4)

SOL/RS(4)

CON/CB, SB, THETA, CTHETA, STHETA, C2T, C4T, S2T, S4T

Subroutine PCT (FCTSS) (Continued)

Called by:

QUAD, FUNCT, OUTP

Calls Subroutines:

EVALMP, DRTNI

```
  FCT/FCTSS

C THIS SUBPROGRAM IS CALLED BY THE QUADRATURE PROGRAH AND
C    EVALUATES THE INTEGRAND
C
C
C
      SUBROUTINE FCT(P1,P2,Z,H,G)
C
      IMPLICIT REAL*8(A-H,O-Z) , INTEGER (I-N)
C
      CCHMON /A/A,AMU,PI
      COMMON /ORBIT1/ YEC(3),PA
      COMMON /ORBIT2/X1,Y1,RA,PR(2,2),X1DOT,Y1DOT
      COMHON /CCON/CD(4)
      COMMON /SOL/RS(4)
      COMMON /CON/CB,SB,THETA,CTHETA,STHETA,C2T,C4T,S2T,S4T
C
      EXTERNAL CONE
      DIMENSION Z(10),G(10),H(10),AM(5,3),PAM(5,3,5),X(5),PRA(5),
     1                    PPA(5),DRSP(3),DRSQ(3),PVEC(3)
C
      N=0
      F=P1
C
C EVALUATE M AND PARTIAL OF M YRT STATE
C
      DO 5 I=1,5
 5     X(I)=Z(I)
 10   CALL EVALMP(X,F,AMU,AM,PAM,1)
-
  EVALUATE THE COMMON SCALOR FACTOR
C
      CT=DCOS(F)
      ST=DSIN[P)
C EVALUATE THE PARTIAL OF BA WRT X
      PRA(1)= 0.D0
      PRA(2)= -ST
      PBA(3)= -CT
      PRA(4)= 0.D0
      PRA(5)= 0.D0
C
C EVALUATE  M TRANSPOSS LAUBDA    (PRIMER VECTOR)
C
      DO 20 I=1,3
      PVEC(I)= 0.D0
      DO 20 J= 1,5
 20   PVEC(I)= PVEC(I)+AM(J,I)*Z(J+5)
C *************
C     WRITE (6,1000) F,Z,CT,ST,RA,     PRA,PYEC
C ***********
C
 19   ABVEC= 0.D0
      DO 22 I=1,3
 22   ABVEC= ABVEC+PVEC(I)*PVEC(I)
      ABVEC= DSQRT(ABVEC)
      DO 23 I=1,3
 23   PVEC(I)= PVEC(I)/ABVEC
C *************
C     WRITE (6,1000)     PYEC,ABVEC
 **************
```

ORIGINAL PAGE OF POOR QUALITY

```
C
C
C PRIMER CONE ANGLE.
      CB=0.D0
      DO 53 I= 1,3
 50   CB= PVEC(I)*RS(I)+CB
      SB= 1.D-10
      IF (DABS(CB).LT.1.D0) SB=DSQRT(1.D0-CB*CB)
C
C PARTIAL OF RS WRT P,Q
      D= 2.D0/(1.D0+Z(4)*Z(4)+Z(5)*Z(5))
      DRSP(1)= (-RS(2)*Z(5)-RS(3))*D
      DRSQ(1)= RS(2)*Z(4)*D
      DRSP(2)= RS(1)*Z(5)*D
      DRSQ(2)= (-RS(1)*Z(4)+RS(3))*D
      DRSP(3)=  RS(1)*D
      DRSQ(3)= -RS(2)*D
C
C THRUST CONE ANGLE
      THG= PI/2.D0
      IF (DABS(SB).GT.1.D-8) THG= DATAN2((3.D0*CB+DSQRT(8.D0*SB*SB
     1    +9.D0*CB*CB)),-4.D0*SB)
      IF (THG.LT.CD(4)) THG= (1.00001D0*CD(4))
      CALL DRTNI(THETA,FT,DFT,CONE,THG,1.E-10,100,IER)
      IF (IER.GT.0) WRITE (6,1001) IER,THG,THETA,FT,DFT
      IF (THETA.GT.PI) WRITE (6,1002) THG,THETA
      IF (THETA.LT.CD(4)) THETA= CD(4)
      STHETA= DSIN(THETA)
      CTHETA= DCOS(THETA)
      C2T= DCOS(2.D0*THETA)
      C4T= DCOS(4.D0*THETA)
C
C
      PCO= STHETA/SB
      RCO= CTHETA-PCO*CB
      DO 70 I=1,3
 70   VEC(I)= RCO*RS(I)+PCO*PVEC(I)
      ABVEC= 0.D0
      DO 72 I=1,3
 72   ABVEC= ABVEC+VEC(I)*VEC(I)
      ABVEC= DSQRT(ABVEC)
      DO 74 I=1,3
 74   VEC(I)= VEC(I)/ABVEC
C
C ACCELERATION FACTOR AND PARTIAL
      PA= CD(1)+CD(2)*C2T+CD(3)*C4T
      DUMMY=  4.D0*CTHETA*(CD(2)+4.D0*CD(3)*C2T)
      DO 80 I= 1,3
 80   PPA(I)= 0.D0
      UPRSP= 0.D0
      UPRSQ= 0.D0
      DO 85 I=1,3
      UPRSP= UPRSP+VEC(I)*DRSP(I)
 85   UPRSQ= UPRSQ+VEC(I)*DRSQ(I)
      PPA(4)= DUMMY*UPRSP
      PPA(5)= DUMMY*UPRSQ
C
C *************
C     WRITE (6,1000) X1,Y1,RS,CB,SB,DRSP,DRSQ,THETA,CTHETA,
C    1      STHETA,C2T,C4T,S2T,S4T,PA,PPA,THG,FT,DFT
C ************
```

```
C
C EVALUATE FUNCTION
 90   IF (F2.EQ.0.D0) RETURN
      DO 100 I=1,5
      G(I)= 0.D0
      DO 100 J=1,3
 100  G(I)= G(I)+AM(I,J)*VEC(J)
      HZ = 0.D0
      DO 120 I=1,5
 120  HZ = HZ +Z(I+5)*G(I)
      DUM1= HZ *RA
      DO 130 I= 1,5
      G(I+5)= 0.D0
      DO 130 J=1,3
      DO 130 L=1,5
 130**G(I+5)= G(I+5)-Z(L+5)*PAM(L,J,I)*VEC(J)
C **  **********
C     BRITE (6,1000) G,        DUM1,PRA, PPA
C **************
      DO 140 I=1,5
 140  G(I+5)= PA*(G(I+5)*RA-HZ*PRA(I))-PPA(I)*DUM1
C **************
C     WRITE (6,1000) G,HZ
C ***********
C
      DUMMY= RA*PA
      DO 150 I= 1,5
 150  G(I)= G(I)*DUMMY
C
      I? (M.EQ.1) RETURN
      DO 200 I= 1,10
 200  H(I)= G(I)
      F=F2
      M=1
      GO TO 10
C1000 FORMAT (1H0,1P10D12.4)
 1001 FORMAT (1H0, 'ERROR IN NERTON ITERATOB FOR CONE ANGLE'/' IER = ',
     1  I2, 'THG= ',1PD20.12,'THETA= ',1PD20.12,'FT= ',1PD20.12,
     2  'DFT= ',1PD20.12)
 1002 FORMAT (1H0,'ERROR--THETA GREATER THAN 180 DEG'/' THG= ',
     1   1PD20.12,'THETA= ',1PD20.12)
      END
```

Subroutine FUNCT (FUNCTSS)

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, and oblateness,

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)

IS/ISUN, ISON

SOL/RSUN(3), RS

NU/XNU

Q/IQ

Called by:

RK4, TRAJ

Calls Subroutines:

SUN, SHADOW, QUAD, FCT, OBLATE

```
C FUNCT/FUNCTSS
C
C SOLAR SAIL
C THIS SUBROUTINE IS AN INTERFACE BETWEEN THE INTEGRATOR ROUTINE
C    AND THE QUADRATURE ROUTINE.
C INCLUDES SHADOW EFFECT.
C THIS ROUTINE ADDS THE EFFECT OF OBLATENESS (J2) TO THE DERIV.
C OBLATE CALCULATES THE EFFECT OF J2. RETURNED AS DZJ2.
C Z IS A VECTOR OF THE AVERAGED STATE AND COSTATE
C DERZ IS THE AVERAGED DERIVATIVE OF Z
C INCLUDES PENALTY FUNCTION.
C
C
C
      SUBROUTINE FUNCT(X,Z,DERZ)
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /A/A,AMU,PI
      COMMON /J2/AJ2
      COMMON /SHAD/ PEN,FEX,DFEN(5),DFEX(5),ISHAD
      COMMON/BCOM/B(9)
      COMMON/IS/ISUN,ISON
      COMMON /SOL/RSUN(3),RS
      COMMON /Q/IQ
      COMMON /NU/XNU
C
C
      DIMENSION Z(10), DERZ(10), G(10), H(10), DZJ2(10),GEX(10),GEN(10)
      DIMENSION DZ10(10)
      EXTERNAL PCT
C
C SET UP COEFFS OF COSF AND SINF IN X1 AND Y1 AND PARTIALS
C
C
      BETA= 1.D0/(1.D0+DSQRT(1.D0-Z(2)**2-Z(3)**2))
      B(1)= 1.D0-Z(2)**2*BETA
      B(2)= Z(2)*Z(3)*BETA
      B(3)= 1.D0-Z(3)**2*BETA
      BETA3= BETA**3/(1.D0-BETA)
      A1= Z(2)**2*BETA3
      A2= Z(3)**2*BETA3
      A3= BETA+A1
      A4= BETA+A2
      B(4)= -Z(2)*(BETA+A3)
      B(5)= Z(3)*A3
      B(6)= -Z(2)*A2
      B(7)= -Z(3)*A1
      B(8)= Z(2)*A4
      B(9)= -Z(3)*(BETA+A4)
C
C
C
      QEX= 0.D0
      QEN= PI+PI
C
      CALL SUN(X,Z)
C
```

```
 23    FAC=   A/(PI+PI)
       IF (ISUN.EQ.1) FAC=FAC/RS**2
C
C
       ISHAD= 0
       IF (ISON.EQ.0) GO TO 24
C
C SHADOW INFLUENCE
C
       CALL SHADOW(Z)
       IF (ISHAD.EQ.0) GO TO 24
       QEN= FEN
       QEX= FEX
       IF (QEN.LT.QEX) QEN= QEN+2.D0*PI
C
 24    AIF= (QEN-QEX)/DFLOAT(IQ)
       IFV=0
       F1= QEX
       DO 26 I=1,10
 26    DERZ(I)= 0.D0
 27    F2= F1+AIF
       CALL QUAD(F1,F2,FCT,DZ10,Z,G,H,10)
       IFV= IFV+1
       F1= F2
       DO 37 I=1,10
 37    DERZ(I)= DERZ(I)+DZ10(I)
       IF (IFV.LT.IQ) GO TO 27
C
 31    DO 32 I=1,10
 32    DERZ(I)=   DERZ(I)*FAC
C PENALTY FUNCTION EFFECT
       IF (XNU.EQ.0.D0) GO TO 38
       E= DSQRT(Z(2)*Z(2)+Z(3)*Z(3))
       DUM=1.D0-E
       EL2=XNU/(Z(1)*Z(1)*DUM*DUM)
       DERZ(6)= -EL2/Z(1)+DERZ(6)
       DUM= EL2/(1.D0-E)
       DUM1=1.D0
       IF (E.GT.1.D-10) DUM1=Z(2)/E
       DERZ(7)= DERZ(7)+DUM1*DUM
       IF (E.GT.1.D-10) DUM1=Z(3)/E
       DERZ(8)= DERZ(8)+DUM1*DUM
C
C
C
 38    IF (ISHAD.EQ.0) GO TO 90
C
C SHADOW INFLUENCE
C
       CALL FCT(QEN,QEX,Z,GEN,GEX)
       HWX=0.D0
       HWN= 0.D0
       DO 40 I=1,5
        HWX= HWX+Z(I+5)*GEX(I)
 40     HWN= HWN+Z(I+5)*GEN(I)
       HWX= HWX*FAC
       HWN= HWN*FAC
       DO 50 I=1,5
 50      DERZ(I+5)= DERZ(I+5)-HWN*DFEN(I)+HWX*DFEX(I)
C
C
```

```
 90   IF (AJ2.LE.0.D0) RETURN

C OBLATENESS BPPECT
C
      CALL OBLATE(AJ2,Z,DZJ2,1)
      DO 120 I=1,10
 120  DERZ(I)= DERZ(I)+DZJ2(I)
      RETURN
C
      END
```

Subroutine INPUT (INPUTSS)

Description:

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

Common Areas:

XMMM/ZLO(5), STEP(6), ZERF(6)

ELEM/ZPO(5), ZPF(5)

INT/IPR, IDIM, IDIM2, NIMAX

TRA/IFMAX, DT, UEB, EW(10)

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

T/TE, TO

A/A, AMU, PI

WF/WF(5)

J2/AJ2

TC/NOP

JD/TL

IS/ISUN, ISON

F/FLIM, RSTEP

ORBIT/NORB

CONSTR/ICON

PLOT/IPL

TERRA/AE, EC, UU, ENS, AN, ECLMTX(3,3), EQUMTX(3,3)

CCOM/CD(4)

Q/IQ

Called by:

CONTL

Calls Subroutines:

PLANET

```
C INPUT/INPUTSS

C SOLAR SAIL--ORBIT RAISING AND ESCAPE
C THIS SUBPROGRAM IS CALLED BY CONT1 AND READS AND PRINTS
C   ALL INITIAL DATA AS WELL AS SETS INITIAL CONSTANTS.
C   THE UNITS ARE BASED ON INTERNAL MU=1.0, INTERNAL DISTANCE
C   UNIT=1 PLANET RADII.  A CIRCULAR
C   ORBIT AT 1 PLANET RADII WOULD HAVE A PERIOD OF 2 PI INTERNAL
C   TIME UNITS,
C INPUT
C  IPNUM - PLANET NUMBER 1-MERCURY,2-VENUS,3-EARTH,4-MARS
C                  INITIAL ORBIT
C  A (KM)
C  E
C  I (DEG)
C  LONG. OF NODE (DEG)
C  ARG. OF PERICENTER (DEG)
C                  INITIAL GUESSES
C  LAMBDA A
C  LAMBDA H
C  LAMBDA K
C  LAMBDA P
C  LAMBDA Q
C                  FINAL CRBIT
C  A
C  E (NOT USED IF NOP=3)
C  I (NOT USED If NOP=3)
C  NODE (NOT USED IF NOP=2 OR 3)
C  PERICENTER (NOT USED IF NOP=2 OR 3)
C
C  TF2 (DAYS), GUESS FOR FINAL TIRE
C  AC (MM/S**2) CHARACTORISTIC ACCELERATICN
C  TL JULIAN DATE AT INITIAL TIME
C  IRDFLG                                          NOMINAL
C  1  END OF INPUT
C  2  IPR, PRINT FLAG                               0
C  3  NIMAX, MAX. NO. OF ITERATIONS                 20
C  4  TFMAX2 (DAYS), MAX. TF                        500.
C  5  DT2 (DAYS), TIME STEP FOR D.E.                10.
C  6  UEB, UPPER ERROR BOUND FOR D.E.               1.D10
C  7  EW, ERROR UEIGHTS FOR D.E.                    1.,1,1,1,1,0,...
C  9  GM (KM**3/SEC**2) GRAV.CONST.                 SET IN PLANETS
C 10  NOP = 1, 5 O.E. SPECIFIED AT TF               1
C         = 2, 3 O.E. SPECIFIED AT TF
C         = 3, SEMIMAJOR AXIS SPECIFIED AT TP
C 11  SETS OBLATENESS, AJ2, = 0.D0                  SET IN PLANETS
C 12  STEP(8), STEP SIZE FOR NUMERICAL DIFF.        1.D-6
C     RSTEP = 0, STEP AS FRACTION IN ITER           0
C           = 1, STEP AS CONSTAHT IN ITER
C 13  ISON = 0, SHACOW EFFECT @FF                   0
C          = 1, SHADOW EFFECT ON
C 14  ISUN = 0 SUN DIST EFFECT ON THRUST OFF        0
C          = 1  EFFECT OH
C 15  CD(1) , COEFFICIENT IN ACCELERATION FACTOR    .5
C     CD(2)                                         .5
C     CD(3)                                         .0
C 16  IE, IF = 1, EQUATORIAL FRAME, SOIAR KOTION     1
C            = 2, EQUATORIAL FRAME, NO SOLAR MOTION
C            = 3, ECLIPTIC FRAME, SCLAR ROTION
C            = 4, ECLIPTIC FRAKE, NO SOLAR MOTION
```


a

```
C           = 5, PLANETARY FRAME,SOLAR MOTION
C           = 6, PLANETARY FRAME, NO SOLAR MOTION
C 17  FLIM, NORM LIMIT IN ITERATION ROUTINE          1.D-06
C 18  XNU, PERICENTER PENALTY FUNCTION FACTOR        0.
C 19  NORB = 0, NO ORBIT PRINT                       0
C          = 1,...,999, ORBIT PRINT ON NORB POINTS OF ORBIT
C 20  SETS IPL=1 SO THAT PLOT DATA IS STORED         0
C 21  IQ, NUMBER OF QUADRATURES (1-10)               2
C
C
      SUBROUTINE INPUT
C
      IMPLICIT REAL*8(A-H,O-Z) , INTEGER (I-N)
C
      NAMELIST /UN/UTKM,UTS,UTD,UTKG,UTKW,UTMS2
C
      COMMON /XMMM/ZLO(5), STEP(6) , ZERP(6)
      COMMON /ELEM/ZPO(5), ZPF(5)
      COMMON /INT/IPR,IDIM,IDIM2,NIMAX
      COMMON /TRA/TFMAX,DT,UEB,EW(10)
      COMMON /UNITS/UTS ,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW ,UTMS2
      COMMON /T/TF,TO
      COMMON /A/A,AMU,PI
      COMMON /WF/WF(5)
      COMMON /J2/ AJ2
      COMMON /TC/NOP
      COMMON /JD/TL
      COMMON /IS/ISUN,ISON
      COMMON /F/FLIM,KSTEP
      COMMON /ORBIT/ NORB
      COMMON /CCOM/CD(4)
      COMMON /Q/IQ
      COMMON /PLOT/IPL
      COMMON /TERRA/AE,EC,UU,ENE,AN,ECLMTX(3,3),EQUMTX(3,3)
      COMMON /PLNTS/ IPNUM
      COMMON /NU/XNU
C
      DIMENSION W(5),PCHAR(4)
C
      DATA PCHAR/'MERCURY ','VENUS   ','EARTH   ','MARS    '/
C
C INTEGER CONSTANTS
C
      IDIM=10
      IDIM2=5
      IDIM3=6
      IPR=0
      ITP=3
      NIMAX=20
      NOP= 1
      ISON= 0
      ISUN= 0
      KSTEP= 1
      NORB= 0
      IQ= 2
      IE= 1
      IPL= 0
C
C REAL CONSTANTS
C
      AMU=1.0D0
```

```
      UEB= 1.0D+10
      DO 10 I=1,IDIM2
       EW(I+IDIM2)= 0.D0
 10    EW(I)= 1.0D0
      DO 12 I= 1,IDIM3
 12    STEP(I)= 1.D-2
      STEP(IDIM3) = 1.D-6
      DT2= 10.D0
      DTR= .017453292519943296D0
      PI= 3.141592653589793 2D0
      TFMAX2= 500.0D0
     .TFMIN= 0.0D0
      T02=0.0D0
      TL= 0.D0
      FLIM= 1.0-06
      CD(1)= .5D0
      CD(2)= .5D0
      CD(3)= 0.D0
      XNU= 0.D0
C
C ALL READ STATEMENTS FOLLOW
C
      REAC (5,1006) IPNUM
      URITE (6,2065) PCHAR(IPNUM)
      IF (IPNUM.LT.1.OR.IPNUM.GT.4) GO TO 240
      READ (5,1001) (W(I),I=1,IDIM2)
      WRITE (6,3001) (W(I),I=1,IDIM2)
      BEAD (5,1001) (ZLO(I),I=1,IDIM2)
      WRITE (6,3001) (ZLO(I),I=1,IDIM2)
      READ (5,1001) (WF(I),I= 1,5)
      WRITE (6,3001) (WF(I),I=1,5)
      READ (5,1001) TF2
      WRITE (6,3001) TP2
      READ (5,7001) AC
      WRITE (6,3001) AC
      READ (5,1001) TL
      WRITE (6,3001) TL
C       TL MUST BE BETWEEN AEOUT A.D. 1950 AND 2000
      IF ((TL. LT. 2.433D6) .OR. (TL.GT.2.452D6)) GO TO 230
C  CALL PLANETS TO SET UP DEFAULTS
      CALL PLANET(GM)
C
 20   BEAD (5,1002) IRDFLG
      WRITE (6,3002) IRDFLG
      IP ((IRDFLG.GT.21).OR.(IRDFLG.LT. 1)) GO TO 200
 25   GO TO (150,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,
     1 49,50,51), IRDFLG
C
C THESE VALUES ARE READ ONLY IP INDICBTEL BY IRDFLG
C
 32   READ (5,1002) IPB
      WRITE (6,3002) IPR
      IF (IPR.LT.0) GO TO 210
      GO TO 20
 33   BEAD (5,1002) NIMAX
      WRITE (6,3002) NIMAX
      IF ((NIMAX.LT.0).OB.(NIMAX.GT.50)) GO TO 220
      GO TO 20
 34   READ (5,1001) TFMAX2
      WRITE (6,3001) TFMAX2
      IF ((TFMAX2.LT.0.D0).OB. (TFMAX2.GT.1.D3)) GO TO 220
```

```
      GO TO 20
35    READ (5,1001) DT2
      WRITE (6,3001) DT2
      IF ((DT2.LE.0.D0).OR.(DT2.GT.1-03)) GO TO 220
      GO TO 20
36    READ (5,1001) UEB
      WRITE (6,3001) UEB
      GO TO 20
37    READ (5,1003) EW
      WRITE (6,3003) EW
      GO TO 20
38    READ (5,1001) UTKM
      WRITE (6,3001) UTKM
      GO TO 20
39    READ (5,1001) GM
      URITE (6,3001) GM
      GO T3 20
40    READ (5,1002) NOP
      WRITE (5,3002) NOP
      IF ((NOP.LT.1).OR.(NOP.GT.3)) GO TO 220
      GO TO 20
41    AJ2= 0.D0
      GO TO 20
42    READ (5,1001) (STEP(I),I=1,IDIM3)
      WRITE (6,3001) (STEP(I),I=1,IDIM3)
      READ (5,1002) KSTEP
      WRITE (6,3002) KSTEP
      IF ((KSTEP.LT.0).OR.(KSTEP.GT.1)) GO TO 220
      GO TO 20
43    READ (5,1002) ISON
      WRITE (6,3002) ISON
      IF ((ISON.LT.0).OR.(ISON.GT.1)) GO TO 220
      GO TO 20
44    READ (5,1002) ISUN
      WRITE (6,3002) ISUN
      IF ((ISUN.LT.0).OR.(ISUN.GT. 1)) GO TO 220
      GO TO 20
45    READ (5,1001) (CD(I),I=1,3)
      WRITE (6,3001) (CD(I),I=1,3)
      IF (DABS(CD(1)+CD(2)+CD(3)-1.D0).GT. 1.D-8) GO TO 220
      GO TO 20
46    READ (5,1002) IE
      WRITE (6,3002) IE
      IF ((IE-LT-1).OR.(IE.GT.6)) GO TO 220
      IF (IE.EQ.2 .OR. IE.EQ.4 .OR. IE.EQ.6) EWE=0.D0
      IF (IE.LE.2) GO TO 20
      EQUMTX(1,1)=1.D0
      EQUMTX(2,1)=0.D0
      EQUMTX(3,1)=0.D0
      EQUMTX(1,2)=0.D0
      EQUMTX(2,2)=1.D0
      EQUMTX(3,2)=0.D0
      EQUMTX(I,3)=0.D0
      EQUMTX(2,3)=0.D0
      EQUMTX(3,3)=1.D0
      IF (IE.LE.4) GO TO 20
      UU=0.D0
      ECLMTX(1,1)=1.D0
      ECLMTX(I,2)=0.D0
      ECLMTX(1,3)=0.D0
      ECLMTX(2,1)=C.D0
```

```
      ECLMTX(2,2)=1.D0
      ECLMTX(2,3)=0.D0
      ECLMTX(3,1)=0.D0
      ECLMTX(3,2)=0.D0
      ECLMTX(3,3)=1.D0
      GO TO 20
 47   READ (5,1003) FLIM
      WRITE (6,3003) FLIM
      IF (FLIM.LT.0.D0) GO TO 220
      GO TO 20
 48   READ (5,1001) XNU
      WRITE (6,3001) XNU
      GO TO 20
 49   READ (5,1005) NORB
      WRITE (6,3005) NORB
      IF ((NORB.LT.0).OR.(NORB.GT.999)) GO TO 220
      GO TO 20
 50   IPL=1
      GO TO 20
 51   READ (5,7002) IQ
      WRITE (6,3002) IQ
      IF ((IQ.LT.1).OR.(IQ.GT.10)) GO TO 220
      GO TO 20
C
C
C
C
C TIME VALUES ARE CHANGED FROM CAPS TO OTHER UNITS
C
 150  UTS= DSQRT(UTKM**3/GM)
      UTM=UTS/60.D0
      UTH=UTS/3600.D0
      UTD=UTH/24.D0
C
C NOW MULTIPLY ENE BY UTD, SINCE WE KNOW WHAT IT IS, THIS
C  USED TO BE DONE IN EARTH/PLANET
      ENE=ENE*UTD
C
      TO= TO2/UTD
      TP= TP2/UTD
      TPMAX= TPMAX2/UTD
      DT= DT2/UTD
      TO1= TO*UTS
      TP1= TP*UTS
      TPMAX1= TPMAX*UTS
      DT1= DT*UTS
C
C MORE CONVERSIONS
C
      UTMS2= UTKM*1.D3/(UTS**2
      UTKG= 1.D3
      UTKW= UTKG*UTMS2*UTKM/UTS
C
C CALCULATE LOWER LIMIT ON THETA
      CD(4)=0.D0
      IF ((CD(2)+CD(3)).EQ.0.D0) GO TO 155
      CK3= 8.D0*CD(3)
      CK1= CD(1)-CD(2)+CD(3)
      CK2= 2.D0*CD(2)-CK3
      IF (CD(3).EQ.0.D0) CD(4)= DARCOS(-DSQRT(CK1/CK2))
      IF (CD(3).NE.0.D0) CD(4)= DARCOS(-DSQRT((-1.D0+DSQRT(1.D0-4.D0*
```

```
     1  CK1*CK3/CK2**2))*CK2/(2.D0*CK3)))
 155  CONTINUE
C
C
C THE PRINTING OF ALL INITIAL VALUES FOLLOWS
C
      WRITE: (6,2000)
      WRITE (6,2065) PCHAR(IPNUM)
      IF (ISON.GE.1) RRITE (6,2042)
      WRITE (6,2002)
      WRITE (6,2003)
C
 158  WRIT3 (6,2004) (W(I),I=1,5)
C
C   CHANGE FROM CLASSICAL O.E. TO EQUINOCTIAL O.E.
      DO 160 I=3,5
 160   W(I)= W(I)*DTR
      ZPO(1)= W(1)/UTKM
      ZPO(2)= W(2)*DSIN(W(5)+W(4))
      ZPO(3)= W(2)*DCOS(W(5)+W(4))
      ZPO(4)= DTAN(W(3)/2.0D0)*DSIN(W(4))
      ZPO(5)= DTAN(W(3)/2.0D0)*DCOS(W(4))
C
      A=(AC*1.D-3)/UTMS2
C
C
      WRITE (6,2005)
      WRITE (6,2004) (ZPO(I),I=1,5)
C
      WRITE (6,2038) AC,A  .
      A=A/AE**2
      CD4= (PI-CD(4))/DTR
      WRITE (6,2063) (CD(I),I=1,3) , CD4
      IF (ISUN.NE.0) WRITE (6,2044)
C
C WRITE FINAL CONDITIONS, CHANGE TO EQUINOCTIAL FINAL COND.
C
      URITE (6,2006)
      ZPF(1)= WF(1)/UTKM
      GO TO (165,170,180), NOP
C
 165  WRITE (6,2003)
      WRITE (6,2004) (WF(I),I= 1,5)
      DO 165 I=3,5
 166   WF(I)= WF(I)*DTR
      ZPF(2)= WF(2)*DSIN(WF(5)+WF(4))
      ZPF(3)= WF(2)*DCOS(WF(5)+HF(4))
      ZPP(4)= DTAN(WF(3)/2.0D0)*DSIN(WF(4))
      ZPP(5)= DTAN(WF(3)/2.0D0)*DCOS(WF(4))
      DC 167 I=3,5
 167   UP(I)=WF(I)/DTR
      URITE (6,2005)
      WRITE (6,2004) (ZPF(I),I=1,5)
      GO T3 190
C
 17C  ZPF(2)= WF(2)
      ZPF(3)= DABS(DTAN(WF(3)*DTR/2.D0))
      ZPF(4)= 0.D0
      ZPF(5)=0.D0
      WRITE (6,2031)
      URITE (6,2004) (WF(I),I=1,3)
```


ORIGINAL PAGE IS
OF POOR QUALITY

```
      WRITE  (6,2032)
      WRITE  (6,2004)  (ZPF(I),I=1,3)
      GO TO 190
 180  C3= -GM/WF(1)
      WRITE  (6,2064)  WF(1),ZPF(1),C3
C
C
 190  WRITE  (6,2007)
 196  WGITZ  (6,2011)  ZL0
      WRITE  (6,2008)
      FRITE  (6,2009)  TF2,TF1,TF
      WRITE  (6,2037)  TL
      WRITE  (6,2036)  BJ2
      WRITE  (6,2013)
      FRITE  (6,2009)  T02,T01,T0
      WRITE  (6,2015)
      WRITE  (6,2009)  TFMAX2,TFMAX1,TFMAX
      WRITE  (6,2010)  KSTEP
      WRITE  (6,2011)  STEP
      PRTTE  (6,201 E)
      WRITE  (6,2009)  DT2,DT1,DT
      WRITE  (6,2017)  UEB
      WRITE  (6,2018)
      WRITE  (6,2019)  EW
      WRITE  (6,2058)  IQ
      WRITE  (6,2020)  IDIM
      WRITE  (6,2022)  NIMAX
      RRITE  (6,2048)  PLIK
      WRITE  (6,2026)  UTKM
      WRITE  (6,2027)  GM
      WRITE  (6,2069)  XNU
      IF (IE.EQ.1) WRITE  (6,2059)
      IF (IE.EQ.2) RRITE  (6,2060)
      IF (IE.EQ.3) WRITE  (6,2061)
      IF (IE.EQ.4) WRITE  (6,2062)
      IF (IE.EQ.5) FRITE  (6,2067)
      IF (IE.EQ.6) WRITE  (6,2068)
      WRITE (6,UN)
      RETURN
 200  WRITE  (6,2023)  IRDFLG
      STOP
 210  WRITE  (6,2024)  IPR
      STOP
 220  URITE  (6,2025)  IRDPLG
      STOP
 230  #RITE  (6,2037)  TL
      STOP
 240  WRITE  (6,2066)  IPNUM
      STOP
C
 1001 FORMAT (F25.15)
 1002 FORHAT (I2)
 1003 FORMAT (506.1)
 1004 FORMAT (P18.11)
 1005 FORHAT (13)
 1006 FORMAT (11)
 2000 FORMAT (1H1,19X,'  MINIMUM TIME OPTIMAL TRAJECTORY PROGRAM FOR PLA
     RNETOCENTRIC SOLAR SAIL SPIRAL' /   1H0,50X, ' ESCAPE OR CAPTURE')
 2001 FORKAT (1H0,53X,13H MINIMUM TIME)
 2002 FORMAT (34H0 THE INITIAL OBBITAL ELEHENTS ARE)
 2003 FORMAT (1H0,10X,6HA (KM),19X,1HE,20X,7HI (DEG),10X,18HLON ASC NODE
```

```
     1 (DEG),6X,15HARG PERIG (DEG))
2004 FORMAT (1P5D23.14)
2005 FORMAT (1H0,5X,14HA (PLANET RAD),16X,1HH,22X,1HK,22X,1HP,22X,1HQ)
2006 FORMAT (40H0 THE DESIRED FINAL ORBITAL ELEMENTS ARE)
2007 FORMAT (32H0 INITIAL GUESSED PARAJETZBS ARE)
2008 FORMAT (21H0 FINAL TIME ESTIMATE)
2009 FORMAT (1H ,10X,1PD22.15,7H DAYS =,1PD22.15,10H SECONDS =,1PD22.15
     1,6H UNITS)
2010 FORMAT (50H0 STEP SIZE FOR NUMERICAL DIPFEREHTIATION, KSTEP =,I2)
2011 FORMAT (1P5D23.14)
2013 FORMAT (17H0 INITIAL TIHE IS)
2015 FORMAT (10H0 TFMAX IS)
2016 FORMAT (27H0 TIME STEP ?OR INTEGRATION)
2017 FOBHAT (36H0 UPPER ERROR BOUND ON INTEGRATION =, 1PD20.10)
2018 FORMAT (35H0 ERROR' UEIGHTS FOR INTEGBATION ARE)
2019 FORMAT (1P10D8.1)
2020 FORMAT (13H0 DIHEHSION =,I5)
2022 FORMAT (32H0 MAXIMUM NUMBER OP ITERATIONS =,I5)
2023 FORHAT (44HO XRDFLG SHOULD BE BETWZEN 1 AND 21, IT IS =,I5)
2024 FORMAT (28H0 IPR SHOULD BE < 0, IT IS =,I5)
2025 FORHAT (27H0 BAD INPUT DATA, IRDFLG = ,I3)
2026 FORHAT (18H0 1 PLANET RADIUS=,F14.5,3H KM)
2027 FORMAT (11H0 MU (GM) =,F16.5,13H KM**3/SEC**2)
2031 FORMAT (1H0,10X,6HA (KM),19X,1HE,20X,7HI (DEG))
2032 FORMAT (1H0,5X,14Rh (PLANET RAD),9X,15HSQRT(H**2+K**2),8X,
     1 15HSQRT(P**2+Q**2))
2037 FORHAT (32H0 JULIAN DATE AT INITIAL TIME IS, PZ0.8)
2036 FORMAT (6H0 J2 =,1PD15.7)
2038 FORMAT ('0 CHARACTERISTIC ACCELERATION = ',F18.11,'MM/S**2 ='
     1 ,1PD15.7, ' INTERNAL UNITS')
2042 FORMAT (24H0 SHADOW EFFECT IHCLUDED)
2044 FORHAT ( '0 SUN DISTANCE EPFECT INCLUDED')
2046 FORMAT (28H0 INITIAL A (PLANET RADII) =, 1PD25.14)
2048 FORMAT (23H0 NORM LIMIT IN ITER = ,1PD12.5)
2058 FORMAT ('0 IQ =',I3)
2059 FORMAT ('0 EQUATORIAL FRAME, SOLAR MOTION')
2060 FORYAT ('0 EQUATORIAL FRAME, NO SOLAR HOTION')
2061 FORHAT ('0 ECLIPTIC FRAME, SOLAR HOTION')
2062 FORMAT ('0 ECLIPTIC FRAME, NC SOLAR MOTION')
2063 FORMAT ('0 ACCELERATION FACTOR COEFFICIENTS = ',1P3D20.12/
     1 '0 MAXIMUM THETA (DEG) = ',1PD20.12)
2064 FORMAT ('0 SEHIHAJOR AXIS= ',1PD23.14,' KR = ',1PD23.14,
     1 ' P.R.; C3 = ',1PD23.14,' KM**2/SEC**2')
2065 FORHAT (' SPACE VEHICLE IS IN ORBIT ABOUT ',A8)
2066 FORHAT (' IPNUH SHOULD BE BETWEEN 1 AND 4, IT IS ',I2)
2067 FORMAT ('0 PLANETARY ORBITAL FRAME, SOLAR MOTION')
2068 FORMAT ('0 PLANETARY ORBITAL FRAME, NO SOLAR XOTION')
2069 FORHAT ('0 PEEXCENTER PEHALTY FUNCTION PXCTOR =',1PD10.2)
3001 FORMAT (1H ,F25.15)
3002 FORMAT (1H ,I2)
3003 FORMAT (1H ,5D6.1)
3004 FORXXT (1H ,F18.11)
3005 FORMAT (1H ,I3)
     END
```

Subroutine ITER (NRSS, MODNRSS)

Description:

A Newton-Raphscn iterator. It calculates a nominal trajectory, receiving the error in the final condition 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 pre-multiplying $\underline{y}$, 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 inputted 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 terms. 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(5), XS(6), Y(6)

INT/IPR, IDIM, IDIM2, MAXNOI

T/TE, TO

DY/DYDT(6)

F/FLIM, KSTEP

Called by:

CONIL

Calls Subroutines:

DCROUT, PRTN, TRAJ

```
C ITER
C NEWTON RAPHSON   NRSS
C
C SPECIAL VERSION FOR 6X6 PARTIAL DER. MATRIX, 6 DIM. Y.
C
      SUBROUTINE ITER(KCUNT,NI,FUNCT,PRTN)
C
      IMPLICIT REAL*8(A-H,O-Z) , INTEGER (I-N)
C
C
C X    VALUES OF THE INDEPENDENT VARIABLES(INITIAL,CURRENT,FINAL)
C XS   STEP SIZE TO PERTURB XS  TO COMPUTE PARTIAL DERIVATIVES
C Y    VALUES OF THE DEPENGENT VARIABLES(CURRENT, FINAL)
      COMMON/XMMM/X(5),XS(6),Y(6)
      COMMON /INT/IPR,IDIM,IDIM2,MAXNCI
      COMMON /T/TF,TO
      COMMON /DY/DYDT(6)
      COMMON /F/FLIM,KSTEP
C
C
C     ARRAYS USED INTERNALLY BY THE ITERATOR
      DIMENSION YNOM(6),XN(5),P(6,6),COEF(6)
      EQUIVALENCE(YNOM,COEF)
C
      N= 5
      M= 6
      NI=1
      KOUST=0
      CALL FUNCT
      KOUNT=KOUNT+1
      FO=0.DO
      DO 15 I=1,M
 15    FO=FO+Y(I)**2
 10   DO 16 I=1,N
      XN(I)=X(I)
 16   YNOM(I)=Y(I)
      YNOM(M)= Y(M)
      TFN=TF
      CALL PRTN(KOUNT,NI)
      WRITE(6,1011)FO
      IF(FO.LE.FLIM) GO TO 90
      IF (NI.GT.MAXNOI) GO TO 80
C COMPUTE NUMERICAL PARTIAL DERIVATIVES
      WRITE (5,1013)
      DO 17 I=1,M
 17   P(I,M)= DYDT(I)
      DO 25 J=1,N
      TEMP=X(J)
      STEP=XS(J)*DABS(X(J))
      IF ((DABS(X(J)).LT.1.D-10).OR.(KSTEP.EQ.1))STEP=XS(J)
      IF (DABS(X(J)).LT. 1.D-10) WRITE (6,1014)
      X(J)=X(J)+STEP
      CALL FUNCT
      WRITE(6,1000)X(J)
      WRITE(6,1001) (Y(I),I=1,M)
      DO 20 I=1,M
   20 P(I,J)=(Y(I)-YNOM(I))/STEP
   25 X(J)=TEMP
      KOUNT=KOUNT+N
      WRITE(5,1002)
      DO 30 I=1,M
```

```
      WRITE(6,1001) (P(I,J) ,J=1,M)
   30 CONTINUE
      DO 35 I=1,M
   35 COEF(I)=-YNOM(I)
      CALL DCROUT(P,COEF,DET,0.D0,M,1,IND)
      IF(IND.NE.0) GO TO 85
      WRITE (6,1015) DET
      DO 40 I=i,n
40    IF (DABS(COEF(I)).LT.1.D-12) COEF(I)= 0.D0
      WRITE (6,1003) (COEF (I),I=1,N)
      SN= COEF(M)
      WRITE (6,1012) SN
      DO 50 J=1,N
   SO X(J)=XN(J)+COEF(J)
      TF=TFN + SN
      IHALV=0
51    IF (DABS(SN).GT.(.25D0*TFN)) GO TO 60
      CALL FUNCT
      KOUNT=KOUNT+1
      F1=0.D0
      DO 52 I=1,M
   52 F1=F1+Y(I)+*2
      WRITE (6,1010) F1
      IF(F1.LT.FO) GO TO 55
60    YRITE: (6,1006)
      IF (IHALV.EQ.10) GO TO 95
      IHALV=IHALV+1
      DO 53 J=1,N
      COEP(J)=COEF(J)/2.D0
      WRITE(6,1000) COEP(J)
   53 X(J)=XN(J)+COEF(3)
      SN= SN/2.DO
      WRITE (6,1012) SN
      TF= TFE + SN
      GO TO 51
   55 IF(NI-MAXNOI) 70,70,80
   70 NI=NI+1
      F0=F1
      GO TO 10
   80 NI=9999
      WRITE (6,1006)
      RETURN
   85 NI=9999
      URITE(6,1007)
      RETURN
   90 WRITE (6,1005) PO
      RETURN
   95 NI=9999
      RRITE (6,1009)
      EETURN
 1000 FORMAT(/1X,1PD23.15)
 1001 FORMAT(1X,1P5D23.15)
 1002 PORKAT(21H0PARTIAL DERIV MATRIX)
 1003 FORMAT(11H0DELX:S ARE/(1X,1PD23.15))
 1005 FORMAT(4H0F0=,1PD22.15,23HCASE CONVERGED...FERTIG)
 1006 FORMAT(38H0EXCEEDED MAXIMUM NUMBER OF ITERATIONS)
 1007 FORMAT(16H0MATRIX SINGULAR)
 1008 FORMAT(11H0DELX:S ARE)
 1009 FORMAT(19H0METHOD CANNOT WORK)
 1010 FORMAT(4H0P1=,1PD23.15)
 1011 FORMAT(4H0F0=,1PD23.15)
```

```
1012 FORMAT (10H0 DEL TF =,1PD23.15)
1013 FORMAT (40H0 X(I)+DX(I) FOLLOWED BY CORRESPONDING Y)
1014 FORMAT (24H0 X(I)=0, SO DX(I)=XS(I))
1015 FORMAT (15H0 DETERMINENT =,1PD23.15)
     END
```

```
C ITER
C MCDNR/MCDNRSS     MODIFIED N-R ITERATOR
C
C
C 6X6 VERSION
C
      SUBROUTINE ITER(KCUNT,NI,FUNCT,PRTN)
C
      IMPLICIT REAL*8(A-H,O-Z)
C
C
C
C X   VALUES OF THE INDEPENDENT VARIABLES(INITIAL,CURRENT,FINAL)
C XS  STEP SIZE TO PERTURB X:S TO COMPUTE PARTIAL DERIVATIVES
C Y   VALUES OF THE DEPENDENT VARIABLES(CURRENT,FINAL)
      COMMON/XMMM/X(5),XS(6),Y(6)
      COMMON /INT/IPR,IDIM,IDIM2,MAXNOI
      COMMON /T/TF,TO
      COMMON /DY/ DYDT(6)
      COMMON /F/FLIM,KSTEP
C
      DIMENSION YNOM(6),XN(5),P(6,6),COEF(6),DYDTN(6)
      N=5
      M=6
      IS=1
      ICONS=1
      ISW=0
      NI=1
      KOUNT=0
      CALL FUNCT
      KOUNT=KOUNT+1
      FO=0.DO
      DO 15 I=1,M
 15   FO=FO+Y(I)**2
 9    DO 16 I=1,N
      DYDTN(I)= DYDT(I)
      XN(I)=X(I)
 16   YNOM(I)=Y(I)
      YNOM(M)= Y(M)
      TFN=TF
      DYDTN(M)= DYDT(M)
 10   CALL PRTN(KOUNT,NI)
      WRITE(6,1011)FO
      IF(FO.LE.FLIM)GO TO 90
      IF (NI.GT.MAXNOI) GO TO 80
      IF(ISW.NE.0)GO TO 27
C COMPUTE NUMERICAL PARTIAL DERIVATIVES
      DO 17 I=1,M
 17    P(I,M)= DYDT(I)
      WRITE (6,1013)
      DO 25 J=1,N
      TEMP=X(J)
      STEP=XS(J)*DABS(X(J))
      IF ((DABS(X(J)).LT.1.D-10).OR.(KSTEP.EQ.1))STEP=XS(J)
C     IF (DABS(X(J)).LT.1.D-10) WRITE (6,1014)
      X(J)=X(J)+STEP
      CALL FUNCT
      WRITE(6,1000)X(J)
      WRITE (6,1001) (Y(I),I=1,M)
      DO 20 I=1,M
 20   P(I,J)=(Y(I)-YNOM(I))/STEP
```

```
   25 X(J)=TEMP
      KCUNT=KOUNT+N
   27 WRITE (6,1002)
      DO 30 I=1,M
      WRITE(6,1001) (P(I,J),J=1,M)
   30 CONTINUE
      DO 35 I=1,M
   35 COST (I)=-YNOM(I)
      CALL DCROUT(P,COEP, DET,0.D0,M,1,IND)
      IF(IND.NE.0) GO TO 85
      WRITE (6,1015) DET
      DO 40 I= 1,M
 40   If (DABS(COEP(I)).LT.1.D-10) COEF(I)= 0.D0
C
      IF (DABS(XN(1)).GT.1.D2) GO TO 47
      RATS= 1.D0
      DO 45 I= 1,5
      RAT= CABS (COEF(I))/(.8D0*DABS(XN (I))+.1D0)
      IF (RAT.GT.RATS) RATS= RAT
 45   CONTINUI
      DO 46 I= 1,8
 46   COE?(I) = COEP (I)/RATS
C     WRITE (5,1016) RATS
C
 47   WRITE: (6,1003)(COEF(I),I=1,N)
      SH= COEF(M)
      WRITE (6,1012) SI
      DO 50 J=1,N
   50 X(J)=XN(J)+COEF(J)
      TF=TFN+SN
      IHALV=0
 51   CALL FUHCT
      KOUNT=KOUNT+1
      F1=0.D0
      DO 52 I=1,M
   52 F1=F1+Y (I)**2
      WRITS(6,1010) Pl
      I?(F1.LT.F0) GO TO 55
      WRITE (6,1008)
      IF(IHALV.EQ.6) GO TO 95
      IHALV=IHALV+1
      DO 53 J=1,N
      COEP (J)=COEF(J)/2.D0
      URITE (6,1000) COEP (J)
   53 X(3)=XH(J)+COEF(J)
      SN=SN/2.0D0
      PRITE (6,1012) SN
      TF=TFN+SN
      GO TO 51
   55 IF(NI-MAXNOI) 70,70,80
   70 NI=NI+1
      ICONS=NI
      F0=P1
      SUMDX=0.D0
      DO 76 J=1,M
   76 SUMDX=COEF(J)**2+SUMDX
      DO 77 I=1,M
      DO 77 J=1,M
      P(I,J)=P(I,J)+(Y(I)*COEF(J))/SUMDX
   77 CONTINUE
      ISW=1
```

```
      GO TO 9
   80 NI=9999
      WRITE(6,1006)
      RETUEN
   85 NI=9999
      WRITE (6,1007)
      RETURN
   90 WRITS(6,1005)F0
      RETURN
   95 IF(NI.EQ.1.OR.I8.EQ.10.OR.ICONS.NE.NI)GO TO 100
      ICONS=ICONSt1
      I8=I8+1
      DO 96 J= 1,N
      DYDT(J)= DYDTN(J)
      X(J)= XN(J)
96    Y(J)= YNOM(J)
      Y(M)= YNOK(E)
      DYDT(M)=DYDTN(M)
      TF= TFN
      ISW=0
      URITE (6,1004)
      GO TO 10
  100 NI=9999
      BRITE(6,1009)
      BETURN
1000 FORMAT(/1X,1PD23. 15)
1001 FORMAT(1X,1P5D23.15)
1002 FORMAT(21HOPARTIAL DEBIY KATRIX)
1003 FORMAT(11HODELX:S ARE/(1X,1PD23.15))
1004 FORMAT(35HOFORM NEW PARTIAL DERIVATIVE MATRIX)
1005 FORMAT(4HOF0=,1PD22.15,23HCASE CONVERGED...FERTIG)
1006 FORMAT(38HOEXCEEDED MAXIMUM NUMBER OF ITERATIONS)
1007 FORMAT(16HOMATRIX SINGULAR)
1008 FORMAT(11HODELX:S ARE)
1009 FORHAT(19HOMETHOD CANNOT YORK)
1010 FORMAT(4HOF1=,1PD23.15)
1011 FORMAT(4HOF0=,1PD23.15)
1012 FORMAT (10H0 DEL TF =,1PD23.15)
1013 FORHAT (40H0 X(I)+DX(I) FOLLOWED BY CORRESPONDING Y)
1014 FORMAT (24H0 X(I)=0, SO DX(I)=XS(I))
1015 FORKAT (15H0 DETERMINENT =,1PD23.15)
1016 FORWBT (8H0 RAPS =,1PD23.15)
      END
```

Subroutine **OBLATE**

Description:

Calculates the oblateness effect on $\dot{\underline{z}}$ and $\underline{\bar{\lambda}}$ (see Section 3.5, Reference 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

```
C OBLATE
C
C THIS SUBPROGRAM EVALUATES THE AVERAGED DERIVATIVE OF THE
C STATE AND COSTATE DUE TO THE OBLATENESS, J2 TERM
C ASSUMES EARTH EQUATORIAL RADIUS=1, MU=1. IF NOT C1 MUST BE
C MODIFIED.
C IF IJ=2, EVALUATE DZ(I),I=1,5 ONLY.
C
C
      SUBROUTINE OBLATE(AJ2,Z,DZJ2,IJ)
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Z(10),DZJ2(10),PJ(4,5)
C
C
C
      C1= 1.5D0*AJ2/Z(1)**3.5
      B1=Z(4)**2+Z(5)**2
      B7= 1.D0-6.D0*B1 + 3.D0*B1**2
      D2= 1.D0 - Z(2)**2 -Z(3)**2
      B2= 1.D0/D2**2
      B6= 1.D0/(1.D0+B1)
      C2= B2*B6
      B4= 1.D0 - B1
      C3= C1*C2
      D16= B6*B7
      D3= C3*D16
C
      DZJ2(1)= 0.D0
      DZJ2(2)= Z(3)*D3
      DZJ2(3)= -Z(2)*D3
C
      D4=B4*C3
C
      DZJ2(4)= -Z(5)*D4
      DZJ2(5)= Z(4)*D4
C
      IF (IJ.EQ.2) RETURN
C
      D5= -3.5D0*C1/Z(1)
      B8= C2*D16*D5
C
      PJ(1,1)=Z(3)*B8
      PJ(2,1)= -Z(2)*B8
C
      B12= C2*B4*D5
C
      PJ(3,1)= -Z(5)*B12
      PJ(4,1)= Z(4)*B12
C
      D2= .25D0*D2**3
      D7= B6*Z(2)/D2
      B9= C1*D16
C
      PJ(1,2)= Z(3)*B9*D7
      PJ(2,2)= -B9*(Z(2)*D7 + C2)
C
      B13= C1*D7*B4
C
      PJ(3,2)= -Z(5)*B13
```

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

Subroutine **OUTP** (OUTPSS)

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 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	Dimensions of state plus costate
PRMT	Parameter vector containing initial and final time, time step, the upper error bound, and a fifth elemsnt not used

Common Areas:

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

INT/IPR, ID, IDIMZ, NIMAX

A/A, AMU, PI

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

IS/ISUN, ISON

SOL/RS(3), RSUN

ORBIT/NORB

ORBIT1/U0(3), PA

ORBIT2/X1, Y1, RA, PR(2,2), XIDOT, YIDOT

CCOM/CD(4)

CON/CB, SB, THETA, DUB(6)

PLOT/IPL

Subroutine OUTP **(OUTPSS)** (Continued)

Called by:

RK4

Calls Subroutines:

SUN, **SHADOW, FCT,** VCP

```
C OUTP/OUTPSS
C
C SCLAB SAIL
C THIS IS THE OUTP PROGRAM FOR THE  R-K
C   INTEGRATOR
C   EQUINOCTIAL O.E.  AND COSTATE ARE USED.
C INCLUDES SHADOW TIME.
C
C
      SUBROUTINE OUTP(T,Z,DERZ,IHLF,IDIM,PRMT)
C
      IMPLICIT REAL*8(A-H,O-Z), INTEGER (I-N)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
      COMMON /INT/IPR,ID,IDIM2,NIMAX
      COMMON /A/A,AMU,PI
      COMMON /SHAD/ FEN,FEX,DFEN(5),DFEX(5),ISHAD
      COMMON /IS/ISUN,ISON
      COMMON /SOL/RS(3),RSUN
      COMMON /ORBIT/NORB
      COMMON /ORBIT1/U0(3),PA
      COMMON /ORBIT2/X1,Y1,RA,PR(2,2),X1DOT,Y1DOT
      COMMON /CCOM /CD(4)
      COMMON /CON/CE,SE,THETA,DUB(6)
      COMMON /PLOT/IPL
C
      DIMENSION PRMT(5), Z(10), DERZ(10), W(5)
      DIMENSION    E1(3),E2(3),       H1(10),G1(10),VEL(3)
C
      INTEGER FRSTSW
      DATA PRSTSW /0/
C
      IF (IPR.EQ.0) RETURN
      IF (T.EQ.PRMT(1)) N=0
      IF (T.EQ.PRMT(1)) M=0
      N=N+1
      IF ((T.LT.(.9999999999D0*DFLOAT(M)*(PRMT(2)-PRMT(1))/DFLOAT(IPR)))
     1.AND.(IHLF.LT.11).AND.(T.LT.(.999999999D0*PRMT(2)))) RETURN
      M=M+1
      CALL SUN(T,Z)
      ACR= (A*UTMS2)*1.D3
      IF (ISON.EQ.0) GO TO 5
      CALL SHADOW(Z)
    5 IF (ISUN.EQ.1) ACR= ACR/RSUN**2
C
      TS=UTS*T
      TM=UTM*T
      TH=UTH*T
      TD=UTD*T
      H= 0.D0
      DO 7 I=1,IDIM2
    7 H= H + Z(I+IDIM2)*DERZ(I)
      W(1)= Z(1)*UTKM
      W(2)=0.D0
      DUMMY=Z(2)**2+Z(3)**2
      IF (DUMMY.GT.1.D-40) W(2)=DSQRT(DUMMY)
      W(3)=0.D0
      DUMMY=Z(4)**2 + Z(5)**2
      IF (DUMMY.GT.1.D-40) W(3)= 2.D0*DATAN(DSQRT(DUMMY))/DTR
      W(4)=0.D0
      IF ((Z(4).NE.0.D0).OR.(Z(5).NE.0.D0)) W(4)=DATAN2(Z(4),Z(5))/DTR
      W(5)= 0.D0
```

```
      IF ((Z(2).NE.0.D0).OR.(Z(3).NE.0.D0)) W(5)=DATAN2(Z(2),Z(3))/DTR
      W(5)=W(5)-W(4)
      IDIM3=IDIM2+1
C
C
C
C
      WRITE (6,1001)
      YRITE (6,1002)
      WRITE (6,1003) T, TS, TH, TD, N
      WRITE (6,1004)
      URITE (6,1005) (Z(I),I=1,IDIM2)
      YRITE (6,1014)
      URITE (6,1005) W
      WRITE (6,1006)
      WRITE (6,1005) (Z(I),I=IDIM3,IDIM)
      URITE (6,1007)
      WRITE (6,1005) (DERZ(I),I=1,IDIM2)
      URITE (6,1008)
      WRITE (6,1005) (DERZ(I),I=IDIM3,IDIM)
      WRITE (6,1009)
      PER=2.D0*PI*DSQRT(Z(1)**3/AMU)*UTH
      bP= W(1)*(1.D0+W(2))
      PE= W(1)*(1.D0-W(2))
      C3= (-AMU/Z(1))*(UTKM/UTS)**2
      WRITE (6,1010) H,PER,PE,AP,ACR,C3
C  IF PLOT DATA DESIRED, PUT IT OUT
      IF (IPL.NE.0 .AND. FRSTSW.EQ.0) WRITE (8) HOBB
      FRSTSW=1
      I? (IPL.NE.0) YRITE (8) TD,Z,W,H,PER,PE,AP,ACR,C3
C
      IP (ISHAD.EQ.0) GO TO 30
      IF (PEX.LT.PEN) PEX=PEX+2.D0*PI
      TSHAD= (PEX-PEN+Z(2)*(DCOS(PEX)-DCOS(PEN))-Z(3)*(DSIN(PEX)
     1        -DSIN(FEN)))*PER/(2.D0*PI)
      FPER= TSHAD/PER
      TSHAD= TSHAD*60.D0
      WRITE (6,1013) TSHAD, FPER
      PEXD= PEX/DTR
      PEND= PEN/DTR
      PRITE (6,1015) PEND,PEXD
C
 30   IP (NORB.EQ.0) GO TO 900
C
C ***********************************
C
      WRIT2 (6,1018) RS,RSUN
 50   WRITE (6,1016)
      F1=0.D0
      CP= 2*PI/DFLOAT(NORB)
      DO 200 I=1,NORB
C F1 IS ECC. LONG.:  H1,G1 ARE DUMMIES
      CALL ?CT (F1,0.D0, Z,H1,G1)
C PRIBER CONE AHGLE
      BETA= (PI-DARCOS(CB))/DTR
C THRUST CONE AHGLE
      TRETAD=(PI-THETA)/DTR
C THRUST CLOCK ANGLE
      VEL(1)= X1DOT
      VEL(2)= Y1DOT
      VEL(3)= 0.D0
```

```
C E1 AND E2 DO NOT HAVE TO BE UNIT VECTORS
      CALL VCP(RS,VEL,E2)
      CALL VCP(E2,RS,E1)
      SCAV=0.D0
      CCAV=0.D0
      DO 100 J=1,3
      SCAV= SCAV+U0(J)*E2(J)
 100  CCAV= CCAV+U0(J)*E1(J)
      PSIV= DATAN2(SCAV,CCAV)/DTR
      FD= F1/DTR
C
C THRUST TO WEIGHT RATIO
      TWR= A*PA*(X1*X1+Y1*Y1)
C
      WRITE (6,1017) FD,X1,Y1,BETA,THETAD,PSIV,PA,TWR,U0
      F1= F1+DF
C  IF PLOT DATA DESIRED, PUT IT OUT
      IF(IPL.NE.0) WRITE (8) FD,X1,Y1,BETA,THETAD,PSIV,PA,TWR,U0
 200  CONTINUE
C
 900  RETURN
C
C
 1001 FORMAT ('0 ******************************************************
     1*************************************************************')
 1002 FORMAT (5H TIME,10X,10HTIME UNITS,15X,7HSECONDS,9X
     1 ,5HHOURS,11X,4HDAYS,12X,1HN)
 1003 FORMAT (1P2D25.7,1P2D15.7,I9/)
 1004 FORMAT (  ' THE EQUINOCTIAL ORBITAL ELEMENTS  ARE')
 1005 FORMAT (1P5D22.12/)
 1006 FORMAT (16H0 THE COSTATE IS)
 1007 FORMAT (32H0 THE DERIVATIVE OF THE STATE IS)
 1008 FORMAT (34H0 THE DERIVATIVE OF THE COSTATE IS)
 1009 FORMAT (1H0,7X,11HHAMILTONIAN,8X,12HPERIOD (HRS),5X,
     1   15HPERICENTER (KM),7X,14HAPOCENTER (KM),5X,14HCHAR ACC/RS**2
     2   ,4X,17HC3 (KM**2/SEC**2))
 1010 FORMAT (2F20.12,1P4D20.10   /)
 1013 FORMAT (24H0 TIME SPENT IN SHADOW =, F12.6,27H MIN,  FRACTION OF P
     1ERIOD =, F12.8)
 1014 FORMAT (  ' THE CLASSICAL O.E.')
 1015 FORMAT (16H0 ENTRY ANGLE = ,F15.8,14H EXIT ANGLE = ,F15.8)
 1016 FORMAT ('0   ECC. LONG.     X1            Y1            PRIMER
     1CONE A  THRUST CONE A  PSIV         ACC FACT        T/W RATIO'
     2              /50X,'UF',13X,'UG',13X,'UW')
 1017 FORMAT (1H0,1P8D15.5/46X,1P5D15.5/46X,1P5D15.5/46X,1P5D15.5)
 1018 FORMAT ('0 SUN DIRECTION AND DISTANCE: ',1P4D15.5)
      END
```

Subroutine OUTPC (OUTPCSS)

Description:

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

Common Areas:

XMMM/ZL0(5), STEP(6), ZERF(6)

Z/ZF(10), DZ(10)

T/TF, TO

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

ELEM/ZP0(5), ZPF(5)

WF/WF(5)

TC/NOP

Called by:

CCNTL

ORIGINAL PAGE IS OF POOR QUALITY

```
C OUTPC/OUTPCSS

C TBIS SUBPROGRAM WRITES THE VALUES FOR THE FINAL CONVERGED
C    TRAJECTORY.  IT IS CALLED BY TRE MAIN PROGRAH CONTL-
C SOLAR SAIL--ORBIT RAISING AND ESCAPE
C
C
C
      SUBROUTINE OUTPC
C
      IMPLICIT REAL*8(A-H,O-Z) , INTEGER (I-N)
C
      COMMON /XMMM/ZLO(5),STEP(6),ZERF(6)
      COMMON /Z/ZF(10),DZ(10)
      COMMON /T/TF,TO
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
      CCMMON /ELEM/ZPO(5),ZPP(5)
      COMHON /WF/WF(5)
      COMMON /CCOM/CD(4)
      COIIMON /TC/NOP
C
      DIMENSION DELZF(5),DELWF(5),WFC(5)
C
      WFC(1)= ZF(1)*UTKM
      WFC(2)= 0.0D0
      DUMMY= ZF(2)**2 + ZF(3)**2
      IF (DUMMY.GT. 1.OD-40) WFC(2)=DSQRT(DUMMY)
      WFC(3)= 0.0D0
      DUMMY= ZF(4)**2 + ZF(5)**2
      IF (DUMMY.GT.1.0D-40) WFC(3)= 2.0D0*DATAN(DSQRT(DUMMY))/DTR
      WFC(4)=0.D0
      IF ((DABS(ZF(4)).GT.1.D-8).AND.(DABS(ZF(5)).GT.1.D-6))
     1 WPC(4)= DATAN2(ZF(4),ZF(5))/DTR
      WFC(5)= 0.D0
      IF ((DABS(ZF(2)).GT.1.D-8).AND.(DABS(ZF(3)).GT.1.D-8))
     1  RPC(5)=DATAN2(ZF(2),ZF(3))/DTR
      PFC(5)=WFC(5)-WFC(4)
      DO 10 I=1,5
       DELWF(I)= WFC(I) - WF(I)
 10    DELZF(I)= ZF(I) - ZPF(I)
      TF2= TF*UTD
      TF1= TF*UTS
C
      RRITE (6,3000)
      RRITB (6,3001)
      URITE (6,3002) WFC
      WRITE (6,3003)
      WRITE (6,3002) (ZF(I),I=1,5)
      RRITE (6,3004)
      WRITE (6,3001)
      IF (NOP.EQ.1) WRITE (6,3002) DELUF
      IF (NOP.EQ.2) WRITE (6,3002) (DELWF(I),I=1,3)
      IF (NOP.EQ.3) WRITE (6,3002) DELWF(1)
      GO TO (20,30,40),NOP
C
 20   YRITE (6,3003)
      WRITE (6,3002) DELZF
      GO TO 100
C
 30   DELZF(2)= DSQRT(ZF(2)**2+ZF(3)**2) -ZPF(2)
      DELZF(3)= DSQRT(ZP(4)**2+ZF(5)**2) -ZPF(3)
```

```
      WRITE (6,3011)
      WRITE (6,3002) (DELZF(I),I=1,3)
      GO TO 100
C
 40   WRITE (6,3003)
      WRITE (6,3002) DELZF(1)
C
 100  RRITE (6,3006)
      YRITE (6,3002) ZLO
      WRITE (6,3008)
      WRITE (6,3009) TF2,TF1,TF
      RETURN
 3000 FORMAT (35H0 ACTUAL PINBL ORBITAL ELEMENTS ARE)
 3001 FORHAT (1H0,10X,6RA (KM),18X,1HE,20X,7HI (DEG),10X,18HLON ASC NODE
     1 (DZG),6X,15HARG PERIC (DEG))
 3002 FORHAT (1P5D23.15)
 3003 FORMAT (1H0,5X,14HA (PLBNET RAD),15X,1HH,22X,1HK,22X,1HP,22X,1HQ)
 3004 FORMAT (51H0 THE ERROR IN THE FINAL O.E. IS (ACTUAL - DESIRED))
 3006 FORMAT (46H0 THE CONVERGED INITIAL GUESSED PARAMETERS ARE)
 3008 FORMAT (29H0 THE MINIMIZED FIHAL TIHE IS)
 3009 FORMAT (1H ,10X,1PD22.15,7H DAYS =,D22.15,10H SECONDS =,1PD22.15,6
     1H UNITS)
 3011 FORMAT (1H0,5X,14HA (PLANET RAD),8X,15HSQRT(H**2+K**2),8X,
     1  15HSQRT(P**2+Q**2))
      END
```

Subroutine PLANET

Description:

Sets various values associated with the planet including orbital elements, gravitational constant, radius, oblateness coefficient, rotation matrices to transform to ecliptic or equatorial coordinate systems.

Argument List:

GM Gravitational constant

Common Areas:

JD/TL

TERRA/AE, EC, W, ENE, AN, ECLMTX(3,3), EQUMTX(3,3)

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

PLNTS/IPNUM

J2/AJ2

```
C PLANET
C
C
C THIS SUBPROGRAM SETS THE VALUES FOR A PLANET'S ORBITAL ELEMENTS
C AND CALCULATES THE MEAN ANOMALY AT THE INITIAL TIME
C O.E. TAKEN FROM BATTIN, 1964. EPOCH 1960 JAN. 1.5, JD=2436935.
C CALCULATES COORDINATE TRANSFORMATION MATRICES AND SETS GRAVITY
C CONSTANT AND PLANET RADIUS.
C INPUT
C      IPNUM--PLANET NUMBER 1-MERCURY,2-VENUS,3-EARTH,4-MARS
C      TL--INITIAL TIME, BEGINNING OF LOW THRUST TRAJ
C OUTPUT
C      AE--EARTH'S SEMIMAJOR AXIS
C      EX--EARTH'S ECCENTRICITY
C      W--ARGUMENT OF PERIH.
C      ENE--MEAN ORBITAL MOTION
C      AN--MEAN ANOMALY AT TL
C      ECLMTX--MATRIX FOR CONVERSION TO EARTH'S ECLIPTIC COORD FRAME
C      EQUMTX--MATRIX FOR CONVERSION TO EQUATORIAL FRAME
C        IMPLEMENTED FOR EARTH ONLY
C      GM--PLANET GRAVITATIONAL CONSTANT
C      UTKM--RADIUS
C      AJ2--OBLATENESS
C
C
C
      SUBROUTINE PLANET(GM)
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON /JD/ TL
      COMMON /TERRA/ AE,EC,W,ENE,AN,ECLMTX(3,3),EQUMTX(3,3)
      COMMON /UNITS/UTS,UTM,UTH,UTD,UTKM,DTR,UTKG,UTKW,UTMS2
      COMMON /PLNTS/ IPNUM
      COMMON /J2/ AJ2
C
      REAL*8 SEMIAX(4),ECCEN(4),MLP(4),MORBM(4),GRAVC(4),RAD(4)
      REAL*8 J2(4),MLE(4),ANGOBL(4),LNODE(4),INCL(4)
C
C  DATA STATEMENTS
      DATA SEMIAX/0.387099D0,0.723322D0,1.D0,1.523691D0/
      DATA ECCEN/0.205627D0,6.793D-3,1.6726D-2,9.3368D-2/
      DATA MLP/76.83309D0,131.00831D0,102.25253D0,335.32269D0/
      DATA MORBM/4.092339D0,1.602131D0,0.985609D0,0.524033D0/
      DATA GRAVC/22181.6D0,324860.1D0,398601.2D0,42828.4D0/
      DATA RAD/2435.D0,6052.D0,6378.16D0,3393.4D0/
      DATA J2/0.D0,0.D0,1.0827D-3,0.D0/
      DATA MLE/222.62165D0,174.29431D0,100.15815D0,258.76729D0/
      DATA ANGOBL/0.D0,0.D0,23.45D0,0.D0/
      DATA LNODE/47.85714D0,76.31972D0,0.D0,49.24903D0/
      DATA INCL/7.00399D0,3.39423D0,0.D0,1.84991D0/
C
C
C  SEMIMAJOR AXIS
      AE=SEMIAX(IPNUM)
C  ECCENTRICITY
      EC=ECCEN(IPNUM)
C ARGUMENT OF PERIH.
      W=MLP(IPNUM)-LNODE(IPNUM)
C  MEAN ORBITAL MOTION
      ENE=MORBM(IPNUM)
```

```
   GM(KM**3S**2)
      GM=GRAVC(IPNUM)
C  RADIUS (KM)
      UTKM=RAD(IPNUM)
C  J2
      AJ2=J2(IPNUM)
C
C MEAN ANOMALY AT EPOCH
      AN= MLE(IPNUM)-MLP(IPNUM)
C MEAN ANOMALY AT TIME TL
      AN= AN+ENE*(TL-2436935.D0)
      AN=AN/360.D0
      AN=AN-IDINT(AN)
C
      AN= AN*360.D0*DTR
      W = W*DTR
C  ENE WILL BE MULTIPLIED BY UTD IN INPUTSS,SINCE UTD DEPENDS
C   ON UTKM,WHICH CAN BE MODIFIED BY THE INPUT STREAM
      ENE= ENE*DTR
C
C  SET UP ARRAY FOR USE IN CONVERTING TO EARTH'S ECLIPTIC FRAME
C
      DCL=DCOS(LNODE(IPNUM)*DTR)
      DSL=DSIN(LNODE(IPNUM)*DTR)
      DCI=DCOS(INCL(IPNUM)*DTR)
      DSI=DSIN(INCL(IPNUM)*DTR)
C
      ECLMTX(1,1)=DCL
      ECLMTX(2,1)=-DCI*DSL
      ECLMTX(3,1)=DSI*DSL
      ECLMTX(1,2)=DSL
      ECLMTX(2,2)=DCI*DCL
      ECLMTX(3,2)=-DSI*DCL
      ECLMTX(1,3)=0.D0
      ECLMTX(2,3)=DSI
      ECLMTX(3,3)=DCI
C
C
C  SET UP EQUMTX FOR USE IN CONVERSION TO EQUATORIAL FRAME
C
C  CURRENTLY IMPLEMENTED ONLY FOR EARTH
      EQUMTX(1,1)=1.D0
      EQUMTX(2,1)=0.D0
      EQUMTX(3,1)=0.D0
      EQUMTX(1,2)=0.D0
      EQUMTX(1,3)=0.D0
      IF (IPNUM.NE.3) GO TO 5
      DUM=ANGOBL(IPNUM)*DTR
      DCOSD=DCOS(DUM)
      DSIND=DSIN(DUM)
      EQUMTX(2,2)=DCOSD
      EQUMTX(3,2)=-DSIND
      EQUMTX(2,3)=DSIND
      EQUMTX(3,3)=DCOSD
      GO TO 10
C
 5    EQUMTX(2,2)=1.D0
      EQUMTX(3,2)=0.D0
      EQUMTX(2,3)=0.D0
      EQUMTX(3,3)=1.D0
 10   CONTINUE
C
      RETURN
      END
```

Subroutine PRTN (PRTNSS)

Description:

Prints information about the Newton iteration.

Argument List:

KOUNT, NOI

KOUNT The number of trajectory calls

NOI The number of iterations

Common Areas:

XMMM/X(5), XS(6), Y(6)

T/TF, TO

Called by:

ITER

```
  PRTN/PRTNSS
C
C THIS PROGRAM IS CALLED BY THE ITERATOR AND PRINTS
C
C
      SUBROUTINE PRTN(KOUNT,NOI)
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /XMMM/X(5),XS(6),Y(6)
      COMMON /T/TF,TO
C
      N=5
      M=6
      URITE (6,1000)
      WRITE (6,1001) NOI,KOUNT
      YRITE (6,1002)
      WRITE (6,1003) (X(J),J=1,N)
      FRITE (6,1004)
      FRITE (6,1003) (Y(J),J=1,M)
      BRITE (6,1005) TF
 1000 FORMAT (29H0 ITER NO.     'IRBJECTOBY CALLS)
 1001 FORHAT (1H0,I6,5X,I6)
 1002 FORHAT (2H0X)
 1003 FORMAT (1X,1P5D23.15)
 1004 FORMAT (2H0Y)
 1005 FORMAT (5H0 TF=,1PD26.16)
      RETURN
      END
```

Subroutine **QUAD** (QUAD4, QUAD8, QUAD16; QUAD32)

Description:

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

Argument List:

<u>XL</u>, <u>XU</u>, <u>FCT</u>, <u>Y</u>, <u>Z</u>, <u>G</u>, <u>H</u>, <u>N</u>

XL	Lower bound
XU	Upper bound
FCT	Dummy program name - called to evaluate integrand
Y	The resultant integrated function
Z	Orbital elements and adjoints
G	Dummy variable
H	Dummy variable
N	Dimension

Called by:

FUNCT

Calls Subroutines:

FCT

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

```
C QUAD/QUAD8
C
C THIS IS A MODIFIED QUADRATURE PROGRAM FOR VECTOR VALUED FUNCTIONS.
C COMPUTES THE INTEGRAL OF THE FUNCTION G (OR H) OVER X FROM XL TO XU.
C THE RESULT IS Y.  ROUTINE USES A 8 POINT GAUSS QUADRATURE.
C
C
C
      SUBROUTINE QUAD(XL,XU,FCT,Y,Z,G,H,N)
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Y(1),H(1),G(1),Z(1)
C
      A=.5D0*(XU+XL)
      B=XU-XL
      C=.4801449282487681 2D0*B
      K=1
      GO TO 500
 10   DO 20 I=1,N
 20    Y(I)=.50614268145188130D-1*G(I)
      C=.39833323870681337D0*B
      K=2
      GO TO 500
 30   DO 40 I=1,N
 40    Y(I)=Y(I)+.11119051722668724D0*G(I)
      C=.26276620495816449D0*B
      K=3
      GO TO 500
 50   DO 60 I=1,N
 60    Y(I)=Y(I)+.15685332293894364D0*G(I)
      C=.91717321247824 90D-1*B
      K=4
      GO TO 500
 70   DO 80 I=1,N
 80   Y(I)=B*(Y(I)+.18134189168918099D0*G(I))
      RETURN
 500  CALL FCT(A-C,A+C,Z,H,G)
      DO 510 I=1,N
 510   G(I)=G(I) + H(I)
      GO TO (10,30,50,70),K
      END
```

QUAD16

```
C THIS IS A MODIFIED QUADRATURE INTEGRATION PROGRAM FOR
C VECTOR VALUED FUNCTIONS FO ONE VARIABLE.  IT INTEGRATES
C G (OR H) OVER X FROM XL TO XU.  THE RESULT IS Y.
C EVALUATION IS BY A 16 POINT GAUSS QUADRATURE FORMULA.
C
      SUBROUTINE QUAD(XL,XU,FCT,Y,Z,G,H,N)
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Y(1),Z(1),G(1),H(1)
C
      A= .5D0*(XU+XL)
      B= XU-XL
      C= .4947004674958250D0*B
      K=1
      GO TO 500
 10   DO 20 I=1,N
 20    Y(I)= .1357622970587705D-1*G(I)
      C= .4722875115366163D0*B
      K=2
      GO TO 500
 30   DO 40 I=1,N
 40    Y(I)= Y(I) + .3112676196932395D-1*G(I)
      C= .4328156011939159D0*B
      K=3
      GO TO 500
 50   DO 60 I=1,N
 60    Y(I)= Y(I)+ .4757925584124639D-1*G(I)
      C= .3777022041775015D0*B
      K=4
      GO TO 500
 70   DO 80 I=1,N
 80    Y(I)= Y(I)+ .6231448562776694D-1*G(I)
      C= .3089381222013219D0*B
      K=5
      GO TO 500
 90   DO 100 I=1,N
 100   Y(I)=Y(I)+.7479799440828837D-1*G(I)
      C= .2290083888286137D0*B
      K=6
      GO TO 500
 110  DO 120 I=1,N
 120   Y(I)= Y(I)+.8457825969750127D-1*G(I)
      C= .1408017753896295D0*B
      K=7
      GO TO 500
 130  DO 140 I=1,N
 140   Y(I)= Y(I)+.9130170752246179D-1*G(I)
      C= .4750625491881872D-1*B
      K=8
      GO TO 500
 150  DO 160 I=1,N
 160   Y(I)= B*(Y(I) +.9472530522753425D-1*G(I))
      RETURN
 500  CALL FCT(A-C,A+C,Z,H,G)
      DO 510 I=1,N
 510   G(I)= G(I) + H(I)
      GO TO (10,30,50,70,90,110,130,150),K
      STOP
      END
```


ORIGINAL PAGE IS
OF POOR QUALITY

```
C QUAD32
C
C THIS IS A MODIFIED QUADRATURE INTEGRATION PROGRAH POR
C VECTOR VALUED FULCTIONS OF ONE VARIABLE. IT INTEGRATES
C G (OR H) OVER X FROM XL TO XU.  ?HE RESULT IS Y.
C EVALUATION IS BY A 32 POINT GAUSS QUADRATURE FORMULA.
C
      SUBROUTINE QUAD(XL,XU,FCT,Y,Z,G,H,N)
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Y(1),Z(1),G(1),H(1)
C
      A= .5D0*(XU-XL)
      E= XU-XL
      C= .49863193092474078D0*B
      K=1
      GO TO 500
 10   DO 20 I=1,N
 20    Y(I)= .35093050047350483D-2*G(I)
      C= .49280575577263417D0*B
      K=2
      GO TO 500
 30   DO 40 I=1,N
 40    Y(I)= Y(I) + .81371973654528350D-2*G(I)
      C= .48238112779375322D0*B
      K=3
      GO TO 500
 50   DO 60 I=1,N
 60    Y(I)= Y(I) + .12696032654631030D-1*G(I)
      C= .46745303796886984D0*B
      K=4
      GO TO 500
 70   DO 80 I=1,N
 80    Y(I)= Y(I) + .17136931456510717D-1*G(I)
      C= .44816057788302606D0*B
      K=5
      GO TO 500
 90   DO 100 I=1,N
 100  Y(I)= Y(I) + .21417949011113340D-1*G(I)
      C= .42468380686628499D0*B
      K=6
      GO TO 500
 110  DO 120 I=1,N
 120   Y(I)= Y(I) + .25499029631188088D-1*G(I)
      C= .39724189798397120D0*B
      K=7
      GO TO 500
 130  DO 140 I=1,N
 140   Y(I)= Y(I) + .29342046739267774D-1*G(I)
      C= .36609105937014484D0*B
      K=8
      GO TO 500
 150  DO 160 I=1,N
 160   Y(I)= Y(I) + .32911111388180923D-1*G(I)
      C= .33152213346510760D0*B
      K=3
      GO TO 590
 170  DO 180 I=1,N
 180   Y(I)= Y(I) + .36172897054424253D-1*G(I)
      C= .29385787862038116D0*B
      K=10
```

```
      GC TO 500
150   DO 200 I=1,N
200    Y(I)= Y(I) + .39096947893535153D-1*G(I)
      C= .25344995446611470D0*B
      K=11
      GO TO 503
21c   DO 220 I=1,N
220   Y(I)= Y(I) + .41655962713473378D-1*G(I)
      C= .21067563806531767D0*B
      K=12
      GO TO 500
23C   DO 240 I=1,N
240    Y(I)= Y(I) + .43826046502201906D-1*G(I)
      C= .16593430114106382D0*B
      K=13
      GG TO 500
250   DO 260 I=1,N
260    Y(I)= Y(I) + .45586939347881942D-1*G(I)
      C= .11964368112606854D0*B
      K=14
      GO TO 500
270   DO 280 I=1,N
280    Y(I)= Y(I) + .46922199540402283D-1*G(I)
      C= .72235980791398250-1*B
      K=15
      GO PO 500
290   DO 300 I=1,N
300    Y(I)= Y(If t .47819360039637430D-1*G(I)
      C= .24153832843869158D-1*B
      K=16
      GO TO 500
310   DO 320 I=1,N
320    Y(I)= B*(Y(I) + .48270044257363900D-1*G(I))
      RETURN
500   CALL FCT(A-C,A+C,Z,H,G)
      DO 510 I=1,N
510    G(I)= G(I) + H(I)
      GO TO (10,30,50,70,90,110,130,150,170,190,210,230,250,270,290,310)
      END
```

Subroutine RK4

Description:

This is 4 point Runge-Kutta integrator which is just the IBM Scientific Subroutine Package version without the accuracy checks. (4)

Argument List:

PRMT, Y, DERV, 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 = TO = 13 if DT(TF - TO) < 0
FCT	Dummy subroutine name - called to evaluate derivative
GUT?	Dummy output subroutine name
AUX	Dummy array used for intermediate calculation

Called by:

TRAJ

```
C   RK4
C
C THIS  IS A RUNGE KUTTA INTEGRATOR  OF 4TR  ORDER-
C
C
      SUBROUTINE RK4(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)
C
      IMPLICIT REAL*8(A-H,O-Z)
      DINENSION Y(1),DERY(1),AUX(1,1),A(4),B(4),C(4),PRMT(1)
C
      X=PRMT(1)
      XEND=PRMT(2)
      H=PRMT(3)
      PRMT(5)=0.D0
      IREC=0
      IHLF=0
      CALL  FCT(X,Y,DERY)
C
C ERROR  TEST
      IF  (H*(XEND-X))38,37,2
C
C PREPABATIOB FOR R-K METHOD
 2    A(1)=.5D0
      A(2)=.29289321881345248D0
      A(3)=1.7071067811865475D0
      A(4)=.16666666666666667D0
      B(1)=2.D0
      B(2)=1.D0
      B(3)=1.D0
      E(4)=2.D0
      C(1)=.5D0
      C(2)=A(2)
      C(3)=A(3)
      C(4)=.5D0
C
C PREPARATIONS  OF  FIRST R-K STEP
      DO 3 I=1,NDIM
 3    AUX(1,I)=0.D0
      IEND=0
C
C
C START OF A R-K STEP
 4    IS=0
      IF  ((X+2.D0*H-XEND)*H)7,6,5
 5    H=(XEND-X)/2.D0
 6    IEND=1
 7    CALL  OUTP(X,Y,DERY,IREC,NDIM,PRMT)
      IF  (PRMT(5))40,8,40
C
C
C START OF R-K LOOP
 8    J=1
 10   AJ=A(J)
      BJ=B(J)
      CJ=C(J)
      DO  11 I=1,NDIM
      R1=H*DERY(I)
      R2=AJ*(R1-BJ*AUX(1,I))
      Y(I)=Y(I)+R2
      R2=R2+R2+R2
 11   AUX(1,I)=AUX(1,I)+R2-CJ*R1
```

Subroutine **SHADOW** (SHADOWSS)

Description:

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

Argument List:

$\underline{Z}$

Z   Orbital elements and their adjoints

Common Areas:

SOL/XSUN, YSUN, ZSUN, RSUN

SHAD/EEN, E, DFEN(5), DFEX(5), ISHAD

Called by:

FUNCT, OUTP

PRECEDING PAGE BLANK NOT FILMED

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

```
      AO= H1**2+C1
C     WRITE (6,EQ2)
      AP(1)=2.D0*(H1*H2+C3)/AO
      AP(2)= (H2**2+2.D0*H3*H1-C1+C2)/AO
      AP(3)= 2.D0*(H3*H2-C3)/AO
      AP(4)= (H3**2-C2)/AO
C
C CALL SUBROUTINE TO SOLVE A QUARTIC EQN.
C
      CALL DQRTIC(AP,RT,NRE)
C
C     WRITE (6,EQ)
C
C NRE= NUMBER OF REAL ROOTS, MUST BE EQUAL TO 0,2,OR 4
C ROOTS ARE RT(I), OR RT(1),RT(2),RT(3)+-RT(4)*I, OR
C RT(1)+-RT(2)*I,RT(3)+-RT(4)*I
      IF ((NRE.EQ.1).OR.(NRE.EQ.3)) GO TO 130
C
      FEN= 100.D0
      FEX= 100.D0
      ISHAD= 0
      I=0
 10   I=I+1
      IF ((ISHAD.EQ.2).OR.(I.EQ.(NRE+1))) GO TO 120
C
      CP= RT(I)
      SP= DSQRT(1.D0-CP**2)
C
C HEMISPHERE CHECK
      II=1
 20   X1= B1*CP+B2*SP-Z(3)
      Y1= B3*SP+B2*CP-Z(2)
C     WRITE (6,PR5)
      IF ((X1*XSUN+Y1*YSUN).LT.0.D0) GO TO 40
 30   IF (II.EQ.2) GO TO 10
      II=2
      SP=-SP
      GO TO 20
C
C IS SHADOW EQUATION ZERO?
 40   EQN= D1*X1**2+D2*Y1**2-D3*X1*Y1-Z(1)**(-2)
C     RRLTE (6,PR6)
      IF (DABS(EQN).GT.1.D-6) GO TO 30
C
C ROOT HAS PASSED TESTS--NOW CHECK TO SEE IF EXIT OR ENTRY ANGLE
      DXDF= -B1*SP+B2*CP
      DYDF= -B2*SP+B3*CP
      DSDP= (2.D0*D1*X1-D3*Y1)*DXDF +(2.D0*D2*Y1-D3*X1)*DYDF
      DUM= DATAN2(SP,CP)
      IF (DSDP) 70,50,60
C
C ORBIT IS TANGENT TO SHADOW
 50   WRITE (6,1010)
      GO TO 30
C
C IS PEX ALREADY FOUND?
 60   IF (PEX.EQ.1.D2) GO TO 80
C YES
      WRITE (6,DUMPEX)
      GO TO 30
C
```

```
C IS FEN ALREADY FOUND?
 70   IF (FEN.EQ.1.D2) GO TO 80
C YES
      WRITE (6,DUMFEN)
      GO TO 30
C
C CALCULATE DSDX
 80   ZETA= Z(3)*SP-Z(2)*CP
      BETA3= BETA**3/(1.D0-BETA)
      PZ5=  Z(2)*BETA3
      PZ6= Z(3)*BETA3
      DXDH= -2.D0*Z(2)*BETA*CP+Z(3)*BETA*SP+PZ5*ZETA*Z(2)
      DXDK= Z(2)*BETA*SP-1.D0+PZ6*Z(2)*ZETA
      DYDH= Z(3)*BETA*CP-1.D0-PZ5*Z(3)*ZETA
      DYDK= -2.D0*Z(3)*BETA*SP+Z(2)*BETA*CP-PZ6*Z(3)*ZETA
      DSDX(1)= 2.D0*Z(1)**(-3)
      DUM1= 2.D0*D1*X1-D3*Y1
      DUM2= 2.D0*D2*Y1-D3*X1
      DSDX(2)= DUM1*DXDH+DUM2*DYDH
      DSDX(3)= DUM1*DXDK+DUM2*DYDK
      D=2.D0/(1.D0+Z(4)**2+Z(5)**2)
      DXSP= (-YSUN*Z(5)-ZSUN)*D
      DXSQ= YSUN*Z(4)*D
      DYSP= XSUN*Z(5)*D
      DYSQ= (-XSUN*Z(4)+ZSUN)*D
      DUM1= -2.D0*X1*(X1*XSUN+Y1*YSUN)
      DUM2= -2.D0*Y1*(Y1*YSUN+X1*XSUN)
      DSDX(4)= DUM1*DXSP+DUM2*DYSP
      DSDX(5)= DUM1*DXSQ+DUM2*DYSQ
C     WRITE (6,PB4)
      ISHAD=ISHAD+1
      IF (DSDF.LT.0.D0) GO TO 100
C
C EXIT ANGLE AND DERIVATIVE
      FEX=DUM
      DO 90 J=1,5
 90    DFEX(J)= -DSDX(J)/DSDF
      GO TO 10
C
C ENTRY ANGLE AND DERIVATIVE
 100  FEN=DUM
      DO 110 J=1,5
 110   DFEN(J)= -DSDX(J)/DSDF
      GO TO 10
C
C
 120  IF ((ISHAD.EQ.0).OR.(ISHAD.EQ.2)) RETURN
      WRITE (6,1020) ISHAD
      ISHAD= 0
      RETURN
C
 130  WRITE (6,1030) NRE
      STOP
C
C
 1010 FORMAT (33H0 DSDF=0, ORBIT TANGENT TO SHADOW)
 1020 FORMAT (15H0 ERROR--ISHAD=,I4)
 1030 FORMAT (49H0 DQRTIC HAS RETURNED WITH NUMBER OF REAL ROOTS =,I4)
      END
```

Subroutine SUN (SUNSS)

Description:

Calculates planet-sun unit vector in the equinoctial frame and the distance in A.U.'s divided by planet's semimajor axis.

Argument List:

T, Z

T	Time
Z	State and costate

Common areas:

SOL/RS(3), R

TERRA/AE, EC, W, ENE, AN, ECLMTX(3,3), EQUMTX(3,3)

Called by:

FUNCT, OUTP

```
C SUN/SUNSS
C
C
C THIS PROGRAM CALCULATES THE PLANET TO SUN DIRECTION AND
C DISTANCE FOR A GIVEN TIME.  OUTPUT IN THE EQUINOCTIAL
C COORDINATE FRAME.  USED WITH PLANETS.
C INPUT
C      Z--10 VECTOR OF EQ. O.E.  AND COSTATE(NOT USED)
C      AE--ORBIT SEMIMAJOR AXIS
C      EC--ORBIT ECCENTRICITY
C      W--ARG. OF PERIH.
C      ENE--MEAN ORBITAL MOTION
C      AN--MEAN ANOMALY AT BEGINNING OF TRAJECTORY (TO)
C      ECLMTX--MATRIX FOR CONVERSION TO ECLIPTIC FRAME
C      EQUMTX--MATRIX FOR CONVERSION TO EQUATORIAL FRAME
C      T--PRESENT TIME
C OUTPUT
C      RS--UNIT VECTOR FROM PLANET TO SUN, EQUINOCTIAL COORD.
C      R--DISTANCE FROM PLANET TO SUN AT TIME T
C
C
C
      SUBROUTINE SUN(T,Z)
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON /SOL/  RS(3), R
      COMMON/TERRA/ AE,EC,W,ENE,AN,ECLMTX(3,3),EQUMTX(3,3)
C
      DIMENSION RS1(3),CM(3,3),Z(10),RS2(3),TMP(3)
C
C
C MEAN ANOMALY AT TIME T
      AA= AN+ENE*T
C
C TRUE ANOMALY--CORRECT THRU ECCENTRICITY CUBED
      P=AA+(2.D0*EC-.25D0*EC**3)*DSIN(AA)+1.25D0*EC**2*DSIN(2.D0*AA)
     1     +1.0833333333333D0*EC**3*DSIN(3.D0*AA)
      B=P+W
C
C DISTANCE BETWEEN PLANET AND SUN IN PLANET RADII
      R=(1.D0-EC**2)/(1.D0+EC*DCOS(P))
C
C  CALCULATE UNIT VECTOR TO SUN, EARTH ECLIPTIC FRAME
      TMP(1)=-DCOS(B)
      TMP(2)=-DSIN(B)
      TMP(3)=0.D0
C
      DO 3 I=1,3
      RS2(I)=0.D0
      DO 3 J=1,3
 3    RS2(I)=RS2(I)+ECLMTX(J,I)*TMP(J)
C
C  CALCULATE UNIT VECTOR TO SUN, EQUATORIAL FRAME
      DO 5 I=1,3
      RS1(I)=0.D0
      DO 5 J=1,3
 5    RS1(I)=RS1(I)+EQUMTX(J,I)*RS2(J)
C
C
C TRANSFORM TO EQUINOCTIAL COORD.
```

```
      AB= 1.D0+Z(4)**2+Z(5)**2
      CM(1,1)= (1.D0-Z(4)**2+Z(5)**2)/AB
      CM(2,1)= 2.D0*Z(4)*Z(5)/AB
      CM(3,1)= -2.D0*Z(4)/AB
      CM(1,2)= CM(2,1)
      CM(2,2)= (1.D0+Z(4)**2-Z(5)**2)/AB
      CM(3,2)= 2.D0*Z(5)/AB
      CM(1,3)= -CM(3,1)
      CM(2,3)= -CM(3,2)
      CM(3,3)= (1.D0-Z(4)**2-Z(5)**2)/AB
      DO 10 I=1,3
       RS(I)= 0.D0
       DO 10 J=1,3
 10     RS(I)= RS(I)+CM(J,I)*RS1(J)
C
      RETURN
      END
```

Subroutine TRAJ (TRAJSS)

Description:

The Runge-Kutta integrator is called to calculate the low thrust trajectory. The errors in the final conditions are calculated to send back to the iterator.

Common Areas:

XMMM/ZLO(5), STEP(6), ZERF(6)

TRA/TFMAX, DT0, UEB, EW(10)

Z/Z(10), DERZ(10)

INT/IPR, IDIM, IDIM2, NIMAX

T/TF, TO

ELEM/ZP0(5), ZPF(5)

DY/DYDT(6)

TC/NOP

A/A, AMU, PI

NU/XNU

Called by:

CONTL, ITER

Calls Subroutines:

RK4, FUNCT

```
C TRAJ/TRAJSS


C SCLAR SAIL
C INCLUDES PENALTY FUNCTION.
C THIS ROUTINE SETS UP THE INPUT TO THE INTEGRATOR WHICH
C    EXTRAPOLATES THE TRAJECTQRY FROM INITIAL TIME TO
C    FINAL TIME.  IT ALSO EVALUATES THE CHANGE IN TF AND
C    THE ERROR IN THE FINAL CONDITIONS.
C THIS PROGRAM IS CALLED BY ITER OR BY COHTROL
C IT CALL THE SUBPROGRAM RK4 (RUNGA-KUTTA)
C MIN J, MAX H.
C 6 DIN. ZERF. T.C. OPTIOHS.
C NOP=1--ALL 5 FINAL O.E.  FIXED, =2--A,E,I ONLY FIXED.
C    =3--SEMIMAJOR AXIS (C3) SPECIFIED, ESCAPE TRAJ
C
      SUBROUTINE TRAJ
C
      IMPLICIT REAL*8(A-H,O-Z) , INTEGER (I-N)
C
      COMMON /XMMM/ZLO(5) , STEP(6), ZERF(6)
      COMMON /TRA/TFMAX, DTO, UEB, EW(10)
      COMMON /Z/Z(10), DERZ(10)
      COMMON /INT/IPR, IDIM, IDIM2, NIMAX
      COMMON /T/TF, TO
      COMMON /ELEM/ZPO(5), ZPF(5)
      COMMON /DY/DYDT(6)
      COMMON /TC/NOP
      COMMON /A/A,AMU,PI
      COMMON /NU/XNU
C
      EXTERNAL FUNCT, OUTP
      DIMENSION PRMT(5), AUX(1,10),DERZ1(10)
C.
C
 9    IF (TF.GT.TFMAX) TF=TFMAX
      PRMT(1)= TO
      PRMT(2)= TF
      PRMT(3)= DTO
      PRMT(4)= UEB
C
C
C Z IS A VECTOR OF STATE AND COSTATE
      DO 10 I=1,IDIM2
       Z(I)=ZPO(I)
 10    Z(I+IDIM2)= ZLO(I)
C
C EW ARE ERROR WEIGHTS--INPUT TO THE INTEGRATOR
C
 15    DO 20 I=1,IDIM
 20    DERZ(I)=EW(I)
C
C CALL THE R-K INTEGRATOR
C
      CALL RK4(PRMT,Z,DERZ,IDIM,IHLF,FUNCT,OUTP,AUX)
      IF (IHLF.GT.10) GC TO 100
C
C Z IS NOW THE FINAL O.E.,
C    ZERF THE ERROR IN THE FINAL CONDITIONS
C
      H=0.DO
```

```
      DO 30 I=1,IDIM2
 30    H= H + Z(I+5)*DERZ(I)
      TF1=TF*(STEP(6)+1.D0)
      CALL FUNCT(TF1,Z,DERZ1)
      H1=0.D0
      DO 35 I=1,IDIM2
 35    H1=H1+Z(I+5)*DERZ1(I)
      DYDT(6)= (H1-H)/(TF1-TF)
C PENALTY FUYCTION ETPECT
      H= H-XNU/(2.D0*Z(1)*Z(1)*(1.DO-DSQRT(Z(2)*Z(2)+Z(3)*Z(3)))**2)
C
      ZERF(6)= H -1.D0
C
C FINAL COYDITION OPTION BRANCH
C
      GO TO (40,50,80),NOP
C
 40   DO 45 I=1,5
       ZERF(I)= Z(I) -ZPF(I)
 45    DYDT(I)= DERZ(I)
      RETURN

 50   ZERF(4)= (Z(3)*Z(7)-Z(2)*Z(8) )*1.D-3
      ZERF(5)= (Z(5)*Z(9) -Z(4)*Z(10))*1.D-3
      DYDT(4)= DERZ(3)*Z(7)+Z(3)*DERZ(7)-DERZ(2)*Z(8) -Z(2)*DERZ(8)
      DYDT(4)= DYDT(4)*1.D-3
      DYDT(5)= DERZ(5)*Z(9) +Z(5)*DERZ(9) -DERZ(4)*Z(10)-Z(4)*DERZ(10)
      DYDT(5)= DYDT(5)*1.D-3
C
 60   ZERF(1)= Z(1) - ZPF(1)
      DUM1= DSQRT(Z(2)**2 + Z(3)**2)
      ZERF(2)= DUM1 - ZPF(2)
      DUM2= DSQRT(Z(4)**2 + Z(5)**2)
      ZERF(3)= DUM2 - ZPF(3)
      DYDT(1)= DERZ(1)
      DYDT(2)= 0.D0
      DYDT(3)= 0.D0
      IF (DUM1.GT.1.D-12) DYDT(2)= (Z(2)*DERZ(2) + Z(3)*DERZ(3))/DUM1
      IF (DUM2.GT.1.D-12) DYDT(3)= (Z(4)*DERZ(4) + Z(5)*DERZ(5))/DUM2
C
C SPECIAL CASE, E=0 AND/OR I=0
C
      IF (ZPF(2).NE.0.D0) GO TO 70
      ZERP(2)= Z(2)
      ZERF(4)= Z(3)
      DYDT(2)= DERZ[2]
      DPDT(4)= DERZ(3)
 70   IF (ZPF(3).NE.0.D0) RETURN
      ZERP(3)= Z(4)
      ZEHF(5)= Z(5)
      DYDT(3)= DERZ(4)
      DYDT(5)= DERZ(5)
      RETURN
C
 80   ZERF(1)= Z(1)-ZPF(1)
      DYDT(1)= DERZ(1)
      DO 85 I=2,5
      ZERF(I)= Z(I+5)*1.D-3
 85   DYDT(I)= DERZ(I+5)*1.D-3
      RETURN
C
```

```
100  IF (IHLF.EQ.11) WRITE (6,1000)
     IF (IHLF.EQ.12) WRITE (6,1001)
     IF (IHLF.EQ.13) WRITE (6,1002)
     STOP
C
1000 FORMAT (68H0 THE NUMBER OF BISECTIONS OF THE ORIGINAL INCREMENT HA
    1S EXCEEDED 10)
1001 FORMAT (27H0 INITIAL INCREMENT IS ZERO)
1002 FORMAT (54H0 INITIAL INCREMENT HAS WRONG SIGN OR BOUNDS ARE WRONG)
     END
```

Subroutine VCP

Description:

Calculates a vector cross product, $\underline{W} = \underline{U} \times \underline{V}$

Argument List:

$\underline{U}$	First 3-vector
$\underline{V}$	Second 3-vector
$\underline{W}$	Cross product result, $\underline{U} \times \underline{V}$

```
C VCP
C
C THIS SUBROUTINE CALCULATES THE VECTOR CROSS PRODUCT U X V
C AND RETURNS PRODUCT AS W
C
      SUBROUTINE VCP(U,V,W)
C
      DOUBLE PRECISION U,V,W
      DIMENSION U(3),V(3),W(3)
C
      W(1)= -U(3)*V(2)+U(2)*V(3)
      W(2)=  U(3)*V(1)-U(1)*V(3)
      W(3)= -U(2)*V(1)+U(1)*V(2)
      RETURN
      END
```

LIST OF REFERENCES

1. Sackett, L. L., Optimal Solar Sail Planetocentric Trajectories, Draper Laboratory Report R-1113, September 1977.

2. Sackett, L. L., H. L. Malchow, and T. N. Edelbaum, Solar Electric Geocentric Transfer with Attitude Constraints: Analysis, NASA CR-13497, Draper Laboratory Report R-901, August 1975.

3. Sackett, L. L., H. L. Halchow, and T. N. Edelbaum, Solar Electric Geocentric Transfer with Attitude Constraints: Program Manual, Draper Laboratory Report R-902, August 1975.

4. System/360 Scientific Subroutine Package Version IV, Programmers Manual, GH20-0205-4, IBM Corp., Fifth edition, 1970.

5. Cefola, P. J., "Equinoctial Orbital Elements - Application to Artificial Satellite Orbits", AIAA Paper 72-937, Palo Alto, California, 1972.