PROGRAMMER'S MANUAL

for the

Mission Analysis Evaluation and Space

Trajectory Operations Program

MAESTRO

Contract No. NAS 5-11900

March 1973

Report No. 73-10

Prepared by

Analytical Mechanics Associates

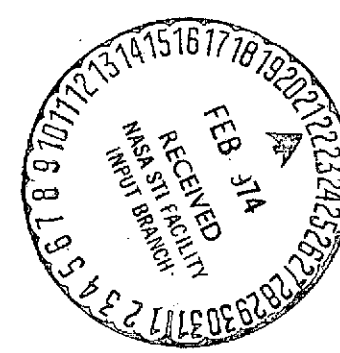
11691 National Boulevard

Los Angeles, California 90064

for

Goddard Space Flight Center

Greenbelt, Maryland



ANALYTICAL MECHANICS ASSOCIATES, INC.
11691 NATIONAL BOULEVARD
LOS ANGELES, CALIFORNIA 90064

PROGRAMMER'S MANUAL

for the

Mission Analysis Evaluation and Space

Trajectory Operations Program

MAESTRO

Contract No. NAS 5-11900

March 1973

Goddard Spaceflight Center
Contracting Officer
Mr. Robert Flick

Technical Officer
Mr. Charles Newman

Prepared by

David Lutzky

William Bjorkman

James Schafer

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

MAESTRO Subroutine Listing

MAIN	Initiates MAESTRO.
ACCEL	Determines spacecraft acceleration.
APROCH	Control subroutine for an approach analysis.
ARMPIT	Determines post injection trim requirements.
ATMO	Determines atmospheric density.
AVEQNS	Determines state derivatives when averaging.
AVERGE	Determines mean solar and lunar accelerations for use in multi-conic.
AVSTRT	Determines initial averaged elements from osculating elements.
BCONIC	Solves Lambert's problem.
BELL	Propagates midcourse execution errors.
BIGMAT	Determines eigenvalues and eigenvectors of a matrix.
BLOCKDATA	Initializes common blocks.
BURND	Computes burn time from impulsive velocity.
BVE	Calculates the impact parameter vector.
CALEND	Converts modified julian date to calendar date.
CLOSE	Determines central planet.
CONTRL	Initialization routine.
COVERT	Transforms a covariance matrix from local tangent plane to inertial.
CRASH	Determines time of closest approach.
CROSS	Vector cross product computation.
DATE	Converts calendar date to modified julian date.
DOPLER	Determines doppler effect.
DOT	Performs vector dot product computation.

DRAG	Calculates acceleration due to atmospheric drag.
DVMAG	Determines magnitude of a vector.
EQNS	Determines derivatives of the state.
FIELD2	Evaluates the disturbing acceleration due to an oblate central planet.
FIND	Retrieves elements from the GTDS 24-hour hold file.
FIXATG	Controls logic for fixed-attitude midcourse scan.
FOWARD	Propagates the state forward in time.
GETTAP	Reads ephemeris tape.
GRAV	Determines disturbing acceleration due to the planets.
HSDTHR	Determines thrust/weight characteristics of midcourse motor.
INPUTF	Data input and editing routine.
INTEG	Sets up arrays for numerical integration.
INTERP	Interpolates for the state as a function of time.
JET	Midcourse pre-targeting determination.
LUNA	Approximate Lunar ephemeris.
MATMPY	Generalized matrix multiplication routine.
MCBURN	Provides the post-midcourse state.
MCSET	Midcourse initialization.
MCVERF	Midcourse verification analysis logic.
MDCORS	Midcourse guidance package.
MINTF	Controls minimum fuel guidance logic.
MINV	Determines minimum post-injection trim velocity for an approach hyperbola.
MONTE	Control subroutine for a Monte-Carlo analysis.
MOTORS	Determines engine thrust/weight characteristics.
MULCON	State propagator using Multi-conic.

MVTRN	Matrix multiplication.
M50EPM	Determines Earth mean of 1950 to Earth prime meridian rotation matrix.
M50JPM	Determines Earth mean of 1950 to prime meridian of planet J rotation matrix.
M50LEQ	Determines Earth mean of 1950 to selenographic rotation matrix.
M50MDT	Determines Earth mean of 1950 to Earth mean of date rotation matrix.
NUTAIT	Determines rotation matrix from Earth mean of date to prime meridian.
NUTATE	Determines rotation matrices from Earth mean of date to Earth prime meridian and selenographic.
OBLATE.	Determines approximate effect due to J_2 when using multiconic.
OBLE	Determines disturbing acceleration due to an oblate Earth.
OBLTY	Determines matrix that rotates through the mean obliquity of the ecliptic.
OBSET	Gravitational field initiation.
ORBIT	Determines cartesian state from orbital elements and vice versa.
ORIENT	Determines rotation so that two vectors have desired dot product.
OUTPUT	Outputs trajectory time history.
OUT1	Determines time to output trajectory.
PLANET	Determines the positions of the planets.
POST	Computes post=targeting information for output.
PRINT	Outputs common blocks
PROTO	Controls midcourse analysis.
PUTELS	Outputs elements into a hold file for later use by the GTDS program.

QUARTC	Solves a quartic.
QUIKIE	Determines the number of days an orbit has shadows.
RANDM1	Determines a random number uniformly distributed between 0 and 1.
READE	Determines the planet's ephemerides from tape or disk ephemeris.
RETDV	Impulsive velocity calculation for retro motor.
RETRO	Determines the retro firing conditions.
RKSEVN	Seventh-order Runge-Kutta numerical integration.
ROTAIT	Rotates two vectors in a plane.
ROTATE	Matrix vector multiplier for 3 x 3 matrix (entry name for MVTRN)
SADOUT	Outputs shadow times.
SENSO	Computes gradient of end constraints w.r.t. midcourse velocity.
SETUP2	Initialization of constants and flags.
SHADOW	Determines times of shadow crossing.
SHORB2	Determines the intersections of an orbit with the shadow cone.
SOL	Approximate Solar ephemeris.
SOLP	Determines the acceleration due to solar pressure.
SPER	Determines spherical coordinates from cartesian coordinates.
SPNM	Calculates Legendre polynomials
SUNMIN	Determines minimum angle between spin axis and Sun as spacecraft is reorientated.
TABINT	Determines thrust and weight from tabular input .
TARGET	Determines midcourse constraint vector at target planet.
TIMEC	Numerical integration control.
TOBODY	Flies along a Keplerian conic for a time interval.
TRIM	Determines post-injection trim velocity.

TRIM2	Determines two-impulse plane change maneuver.
TRMN	Determines mean from true anomaly and vice versa.
TUBE1	Generates graphics data base for use on IBM 2250.
TWELVE	Twelfth-order multi-step numerical integrator.
TWOPIT	Determines optimum position on orbit for 2-impulse plane change maneuver.
UPDATE	Establishes array of back values for interpolation.
VIEW	Determines lighting characteristics of planets.
VISIB	Computes tracking station visibility.
VNORM	Determines unit vector.

Section 1

INTRODUCTION

This manual contains a description of the Mission Analysis Evaluation and Space Trajectory Operations program known as MAESTRO. MAESTRO is an all FORTRAN, block style, computer program designed to perform various mission control tasks. It is intended that this manual serve as a guide to MAESTRO, thereby providing individuals the capability of modifying the program to suit their needs. Of most importance to this task, this manual contains descriptions of each of the subroutines which comprise MAESTRO. These subroutine descriptions consist of input/output description, theory, subroutine description and a flow chart where applicable. The programmer's manual also contains a detailed description of the common blocks, a subroutine cross reference map and a general description of the program structure.

Section 2

MAESTRO STRUCTURE

MAESTRO is an all FORTRAN, block style, computer program designed to perform various mission control tasks. These tasks include

1. Retro motor firing time and attitude determination, APROCH.
2. Midcourse correction determination, PROTO.
3. Verification of midcourse correction, MCVERF.
4. Lunar lifetime prediction, FOWARD.
5. Post-injection trim determination, ARMPIT.

The subroutines responsible for performing the mission control tasks are also shown above. All of these subroutines are essentially self-contained except for PROTO which controls the midcourse determination. The logic involved in this task is more complex than the others, therefore, it will be discussed in detail later in this section.

All of the mission analysis tasks require a means of propagating the state forward in time. Subroutine FOWARD and its related subroutines accomplish this task. The subroutines that make up subroutine FOWARD comprise the major portion of MAESTRO. The second subsection describes the structure of FOWARD.

2.1 Main Control Logic

MAESTRO computation is initialized in the MAIN program. Figure 2.1.1 presents a flow chart of this routine.

The first task of the MAIN program is to clear some common blocks and call subroutine INPUTF. Subroutine INPUTF reads the input data cards and establishes INPUT and FIELDM common blocks. MAIN calls subroutine CONTRL to initialize common blocks. Most of the initialization is actually accomplished in subroutine SETUP2. After the initialization process is complete, MAIN calls one of the mission analysis subroutines to perform the desired analysis, The MODE flag of input common is used to key

the proper subroutine. The logic flow is returned to the input of data to initiate a second case after the analysis is complete.

2.2 Trajectory Propagation, FOWARD

A call to subroutine FOWARD will propagate the state forward in time. The state will be propagated by numerical integration or by the multiconic algorithm. Subroutine MULCON is used when the multiconic algorithm is employed to propagate the state. Subroutines TIMEC, INTEG, EQNS, and ACCEL are employed for numerical integration.

Figure 2.1.1 presents a flow chart of the basic control logic involved in trajectory propagation. The initial state is brought into the subroutines via STATE common. The METH flag in INPUT common is used to determine the trajectory propagation scheme. If the multi-conic technique is desired, subroutine MULCON is called and the state propagated forward in that routine. If one of the numerical intregration schemes is requested, subroutine FOWARD calls subroutine TIMEC to initiate the numerical integration. As seen from Figure 2.1.1, the numerical integration logic consists of looping through subroutines TIMEC, INTEG and RKSEVN or TWELVE. Each loop through the subroutines numerically integrates the state one compute interval. The size of the compute interval is determined in TIMEC. Thus, when flow returns to TIMEC at the end of a loop, the state in STATE common corresponds to the spacecraft's state at the end of the time step.

Subroutine INTEG is used to establish the integration array before integration and determine the position and velocity vectors after the step is complete. An intermediate array is used in the numerical integration subroutines RKSEVN and TWELVE. This array is necessary because a variety of variables can be integrated. The trajectory propagation flag, METH, determines which set of variables are transferred from STATE common to the integration array in INTVAR common. After integration is complete, the variables at the end of the state are transferred back into STATE common. The position and velocity vectors are also determined because they are needed for use in other auxiliary calculations after integration. The integration array separates the numerical integration logic from the trajectory propagation logic. Thus, new numerical integration or trajectory propagation techniques can be easily added

without affecting logic in the other segment. Integration of other quantities can also easily be included. For example, the equations which describe the state transition matrix can be included in the numerical integration without any changes to the numerical integration subroutines.

The actual integration that is carried out is a loop comprised of subroutines EQNS, ACCEL and RKSEVN or TWELVE. Subroutines RKSEVN and TWELVE are single and multi-step numerical integration schemes. The derivatives of the quantities to be integrated are determined in subroutines ACCEL and EQNS. EQNS contains logic to determine the derivatives according to the trajectory propagation scheme employed. The derivatives are transferred to the numerical integration routines via RATES of INTVAR common.

There are many other subroutines besides the ones mentioned which are involved in propagating the state forward in time. These routines are used in the determination of shadow times, closest approach times, trajectory output and interpolation. Most of the calls to perform these auxiliary calculations are made from subroutine TIMEC.

2.3 Midcourse Guidance Structure

The objective of the midcourse guidance package is to determine the velocity correction which nulls a constraint vector at the target planet. The velocity correction is determined through a generalized Newton-Raphson technique. The partial derivatives of the constraints with respect to the velocity correction are determined by finite differences.

The midcourse guidance calculations are initiated in subroutine PROTO, (see Figure 2.3.1). PROTO's functions are to initialize constants, increment execution times, write the midcourse tape and output displays.

The Newton-Raphson logic to determine the correction is contained in a loop inside subroutine MDCORS. This loop uses subroutines SENSO, MCBURN and TARGET to perform many of the midcourse calculations along with subroutine JET to pre-target. Subroutine JET uses a variety of conic techniques to determine an initial value of the correction velocity. This velocity is used as the initial guess in the Newton-Raphson

iteration process. The iteration process consists of the following steps in MDCORS:

1. The constraint vector is determined at the target planet. Subroutines SENSO, MCBURN and TARGET are required. SENSO sets up the logic to propagate the state to the target planet and calls MCBURN to apply the current correction. SENSO next calls FOWARD to obtain the state at the target planet and then TARGET to convert the state to the constraint vector.

2. Tests are made on the constraint vector to determine if each component is within tolerances. If the vector is within the desired tolerances, the solution is converged and flow transferred back to subroutine PROTO.

3. The secant matrix is determined if the solution is outside the bounds. The secant matrix is determined by incrementing each of the control variables independently. The constraint vector is found for each control variable in the same manner as described in step 1.

4. The velocity correction is found by inverting the secant matrix and multiplying by the difference of the current constraint vector from the desired constraint vector.

5. Flow is transferred to step 1.

After the midcourse correction is determined, subroutine PROTO calls POST to obtain auxiliary output quantities. These quantities along with the correction and the pre-midcourse state are written on an output device. If more midcourse execution times are to be analyzed, flow is transferred back to the call to MDCORS to evaluate the next correction. After the last correction is determined, the midcourse tape is read and the appropriate displays output.

There are subroutines involved in the midcourse calculation other than those already discussed. Most of these are used to perform calculations necessary in JET, TARGET, MCBURN and POST. A complete description of all the subroutines is presented in Section 5.

MAIN CONTROL FLOW CHART

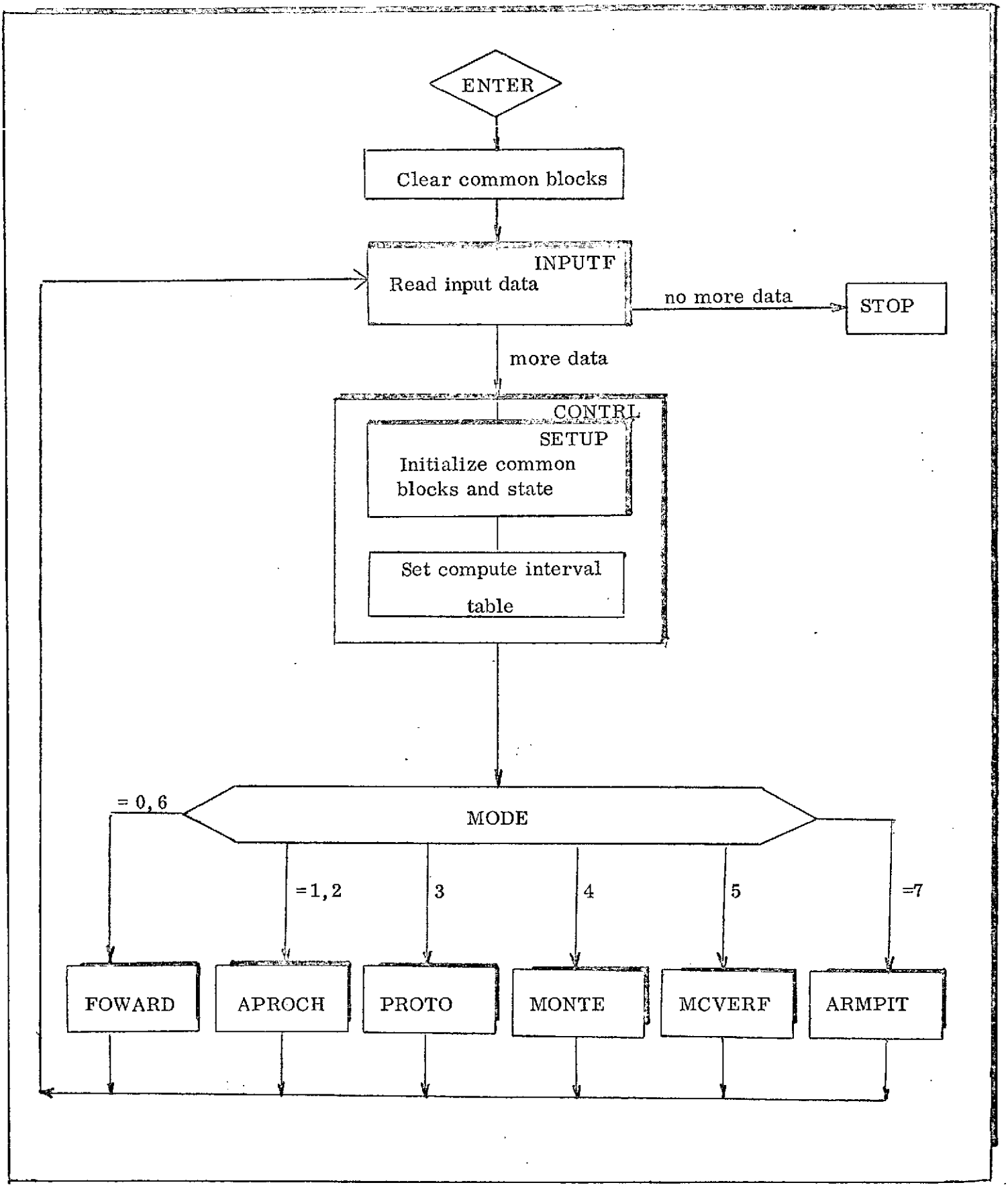


FIGURE 2.2.1

SUBROUTINE FOWARD

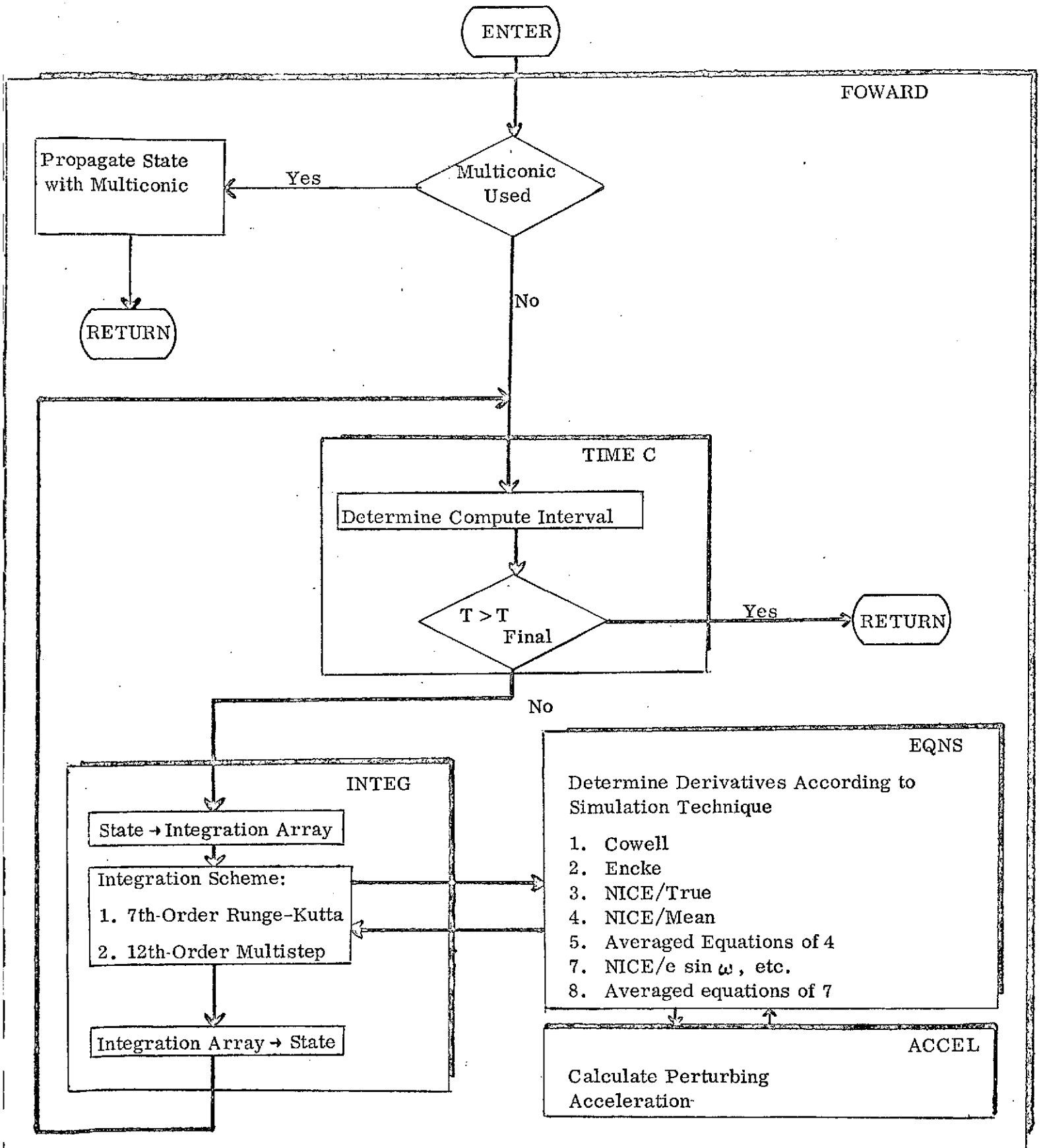
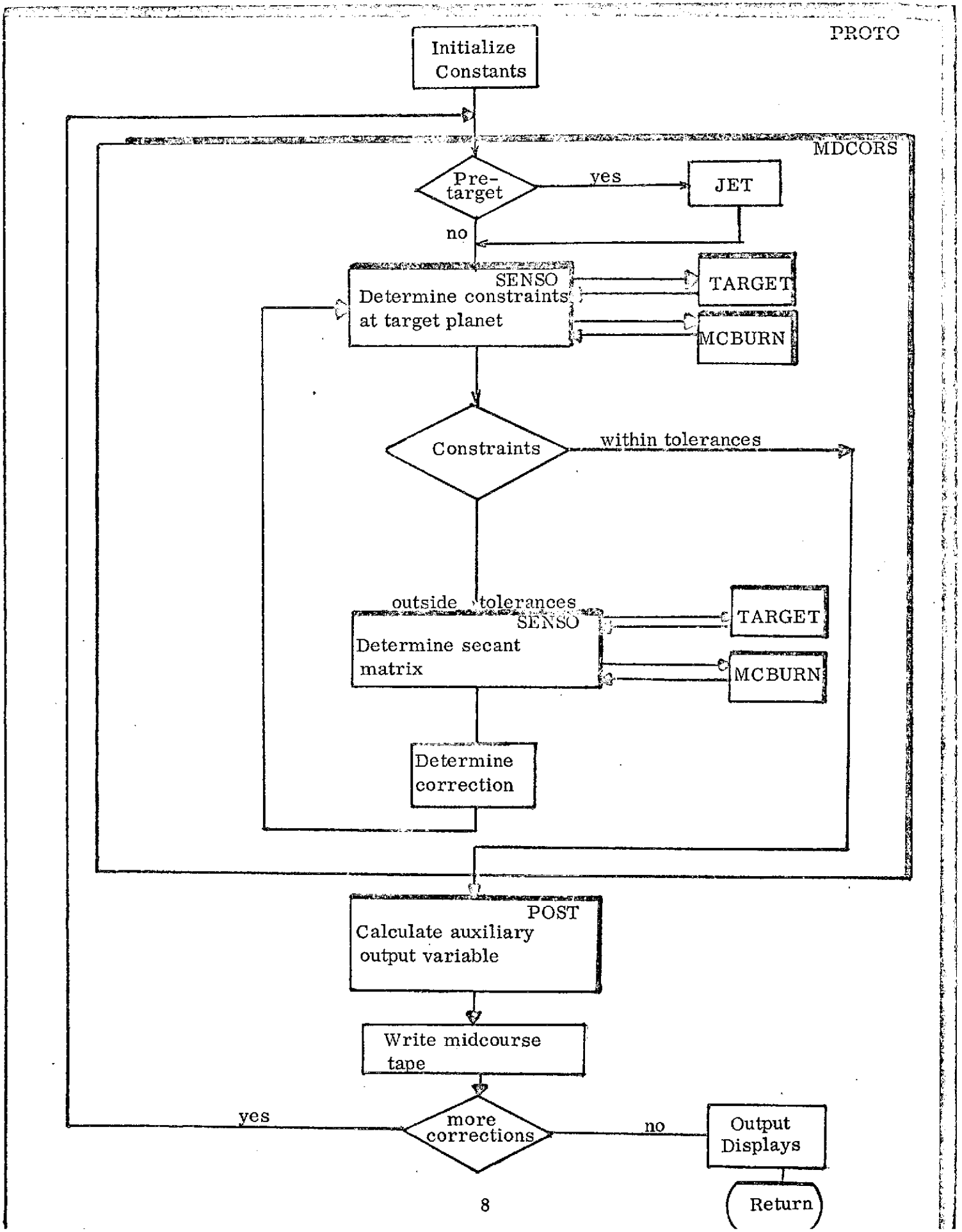


Figure 2.3.1

MIDCOURSE GUIDANCE STRUCTURE



Section 3

CROSS REFERENCE MAP

3.0 Introduction

The MAESTRO system is comprised of more than 90 subroutines. It is difficult to understand the structure of a program of this complexity without some sort of a visual guide. The maps on the following pages are designed to meet this need. These maps present a hierarchy of subroutine calls for each division of the program.

The program is divided into three sections for clarification. The first section consists of the prime control logic. The second division presents the midcourse logic, while the third section consists of the hierarchy of subroutine FOWARD.

3.1 Primary Control Logic

The primary control logic consists of those subroutines required for input, initialization and selecting the analysis modes of MAESTRO. The cross reference map of the primary control logic is shown in Table 3.1.

3.2 Midcourse Logic

The midcourse guidance section comprises a substantial portion of MAESTRO. The midcourse logic can be entered directly from the MAIN program by a call to PROTO or from subroutine MONTE via calls to MDCORS, SENSO and MCBURN. A cross reference map of the subroutines involved in the midcourse logic is presented in Table 3.2.

3.3 Subroutine FOWARD Logic

Subroutine FOWARD's sole aim is to propagate the state forward in time. This subroutine comprises the major portion of MAESTRO and is called from many other subroutines. Table 3.3 presents a cross reference map of the subroutines involved in this portion of the logic.

TABLE 3.1

MAIN CONTROL LOGIC CROSS REFERENCE MAP

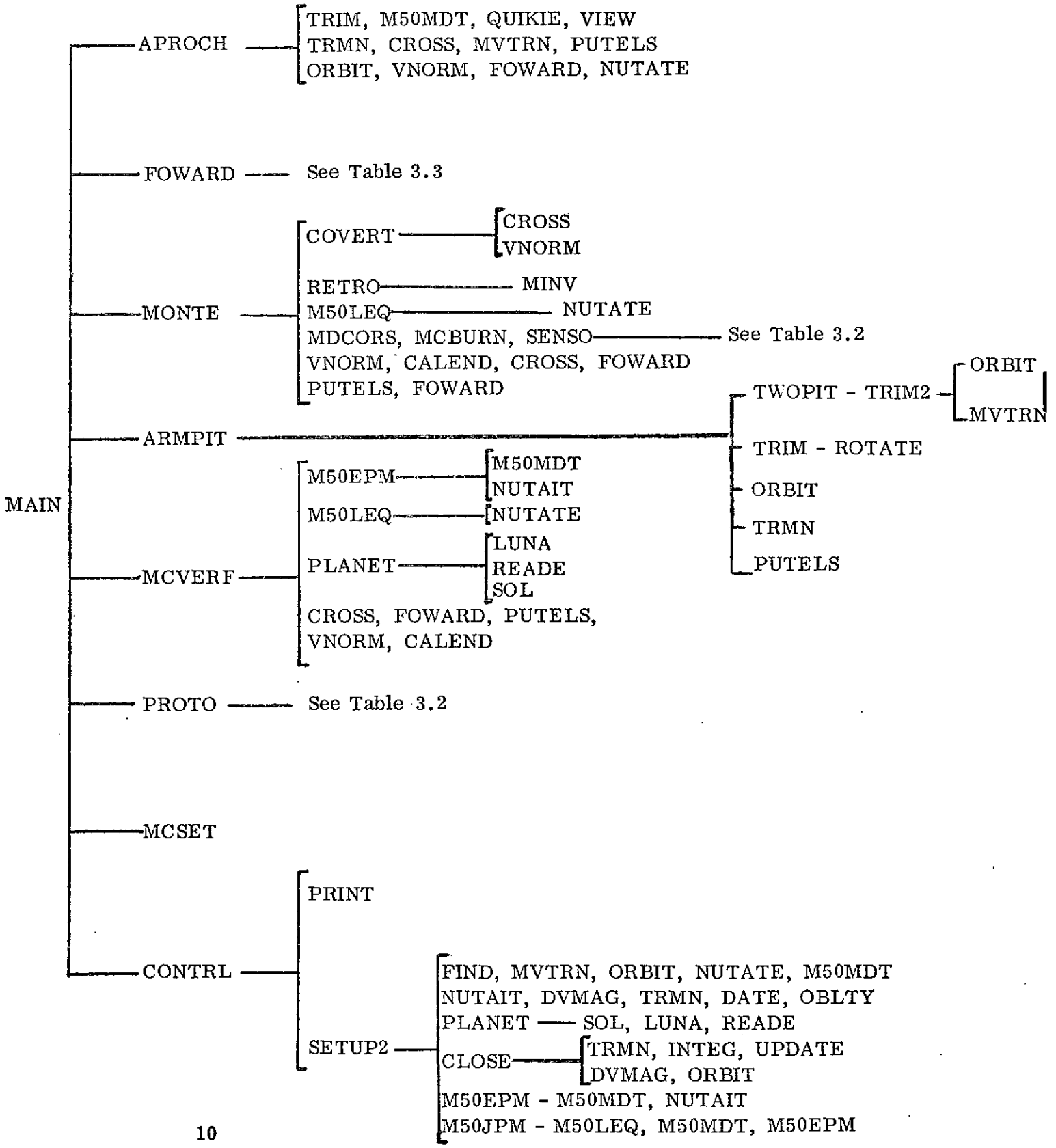


Table 3.2

MIDCOURSE GUIDANCE CROSS REFERENCE MAP

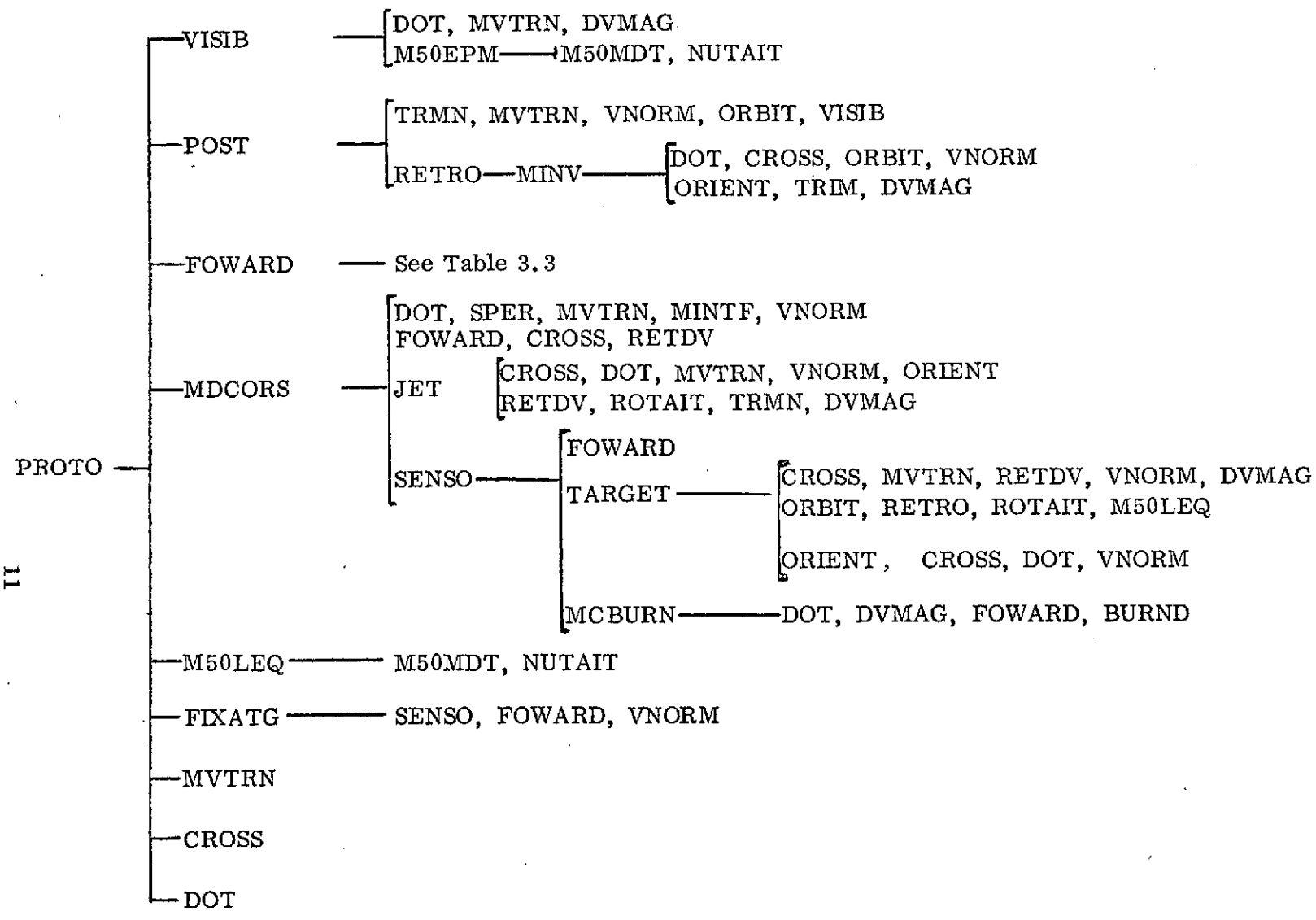
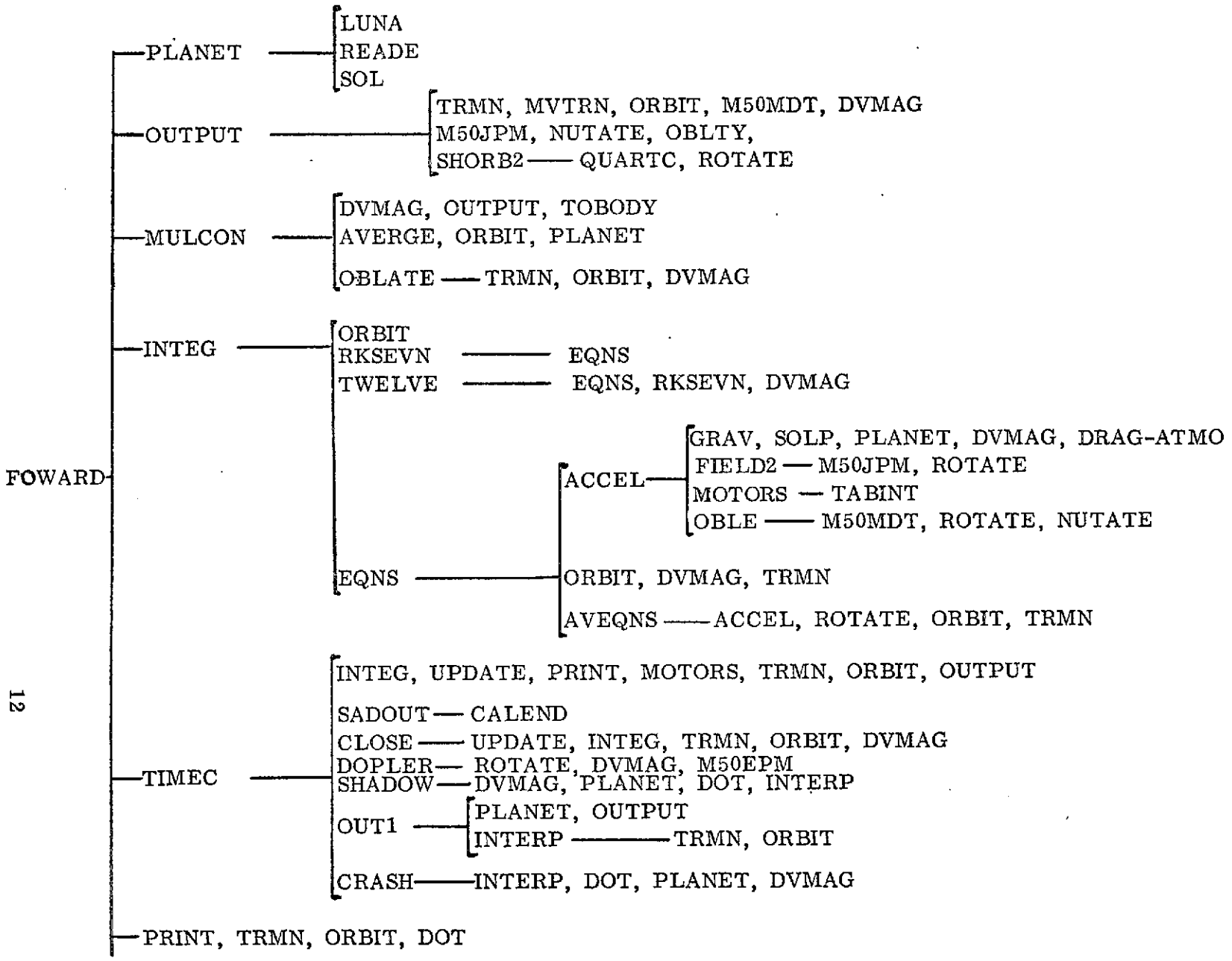


Table 3.3

SUBROUTINE FOWARD CROSS REFERENCE MAP



SECTION 4

COMMON BLOCKS

Currently, there are 25 common blocks established in MAESTRO. Most of these common blocks are required for the numerical integration portion of MAESTRO. The table below presents a list of the current common blocks:

Used in Numerical Integration	Used in Ephemeris Calculation	Used in Auxiliary Calculations
AVG	CETBL2	ANKOR
CNTRL	CETBL3	ELMNT
CONST	CETBL9	INPUTS
DUM	MOON	INTER
FIELDM		MCCOM
GRAVTY		OBSIT
INPUT		PIT
INTVAR		SHAD
INTVRX		
PERT		
PLNET		
SAVE		
STATE		

The following pages in this section present a description of each of the above common blocks. The descriptions include the usage of the common block, its length, the subroutines which required the common block, and a description of each of the elements that comprise the common block.

COMMON ANKOR

Description: This common block contains the current anchor vector

Length: 6 8-byte words

Subroutines Using: CONTRL MONTE PROTO SETUP2

Location	Symbolic Name	Description
1	ANKUE(6)	Cartesian position and velocity vectors of the current anchor vector in Earth mean equator and equinox of 1950, km and km/sec.

COMMON AVG

Description: This common block contains the weights and the corresponding abscissa for the Gaussian quadrature formulae.

Length: 156 8-byte words

Subroutines Using: AVEQNS, BLOCK DATA, MAIN

LOCATION	SYMBOLIC NAME	DESCRIPTION
1	WEIGHT (78)	Weights for Gaussian quadrature formulae*
79	ABSCIS (78)	Abscissa for Gaussian quadrature formulae*

*See Scarborough, James B, Numerical Mathematical Analysis, The Johns Hopkins Press, 1930.

COMMON CETBL2

Description: This common block contains arrays that determine which planets are desired in the disk or tape ephemeris call using subroutine READE

Length: 15 4-byte integer words

Subroutines Using: MAIN GETTAP PLANET READE SETUP2

Location	Symbolic Name	Description
1	ICW	Flag indicating the status of the common block CETBL3
2	ICENTR	Central planet number defined as 1 Mercury, 2 Venus, 3 Earth, 4 Mars, 5 Jupiter, 6 Saturn, 7 Uranus, 8 Neptune, 9 Pluto 10 Sun 11 Moon 12 Oddball
3	IREQ(13)	Array used to determine which planets the ephemeris is desired. The planets correspond to the index of the array as described in ICENTR. The value of IREQ is determined from: IREQ(J) = 0 no ephemeris = 1 position only = 2 position and velocity

COMMON CETBL3

Description: This common block is used to transfer information read from the ephemeris tape to subroutines GETTAP and READE.

Length: 829 8-byte words followed by
205 real 4-byte words

Subroutines Using: MAIN GETTAP PLANET READE SETUP2

Location	Symbolic Name	Description
1	TAB3(829)	Array of raw data obtained from the JPL ephemeris tape. Data contains 8 days of information.
830	NUT(204)	Array of 4-byte real words describing the nutation data obtained from the JPL ephemeris tape.
1034	CKSUM	A 4-byte word used for checksum.

COMMON CETBL9

Description: This common block contains quantities required in the tape or disk ephemeris calculations.

Length: 3 8-byte words followed by
1 4-byte integer word

Subroutines Using: GETTAP READE

Location	Symbolic Name	Description
1	JD1	Reference julian date of ephemeris call.
2	TDAY	Time since reference julian date ephemeris is desired, days.
3	JDIF	Difference in the time of the desired ephemeris minus the time of the ephemeris data read from the tape or disk.
4	IERR1	Error return flag from subroutine GETTAP

COMMON CNTRL

Description: CNTRL common is used to store flags that control state propagation.

Length: 28 4-byte integer words

Subroutines Using:

MAIN	ACCEL	APROCH	AVEQNS
AVSTRT	BELL	CLOSE	CRASH
DOPLER	DRAG	EQNS	FIELD2
FIXATG	FOWARD	GETTAP	GRAV
INTEG	INTERP	LUNA	MCBURN
MCVERF	MDCORS	MONTE	MOTORS
MULCON	OBLE	OUTPUT	PLANET
PRINT	PROTO	RKSEVN	SETUP2
SHADOW	SOL	TARGET	TIMEC
TRIM2	TWELVE		

Location	Symbolic Name	Description
1	-	Not used
2	KTHRST	Thrusting flag. Set to 1 when an engine is thrusting.
3-4	-	Not used
5	KDIS	Discontinuity flag. Flag equals 1 when the current time is a discontinuity time, otherwise set to 0.
6	KHALT	Error return flag. Flag is set to 1 to cause an error return from the numerical integration.
7	JC	Central planet number.
8	KREAD	Flag used to determine if the ephemeris tape or disk is to be read at the current time.
9	KNT	Counter that contains the number of calls to the derivative subroutine, EQNS.
10	KDOPWT	Doppler write flag. Flag equals zero on the first call to DOPLER. On subsequent calls this flag is set to 1.
11	KCA	Counter used in the closest approach iteration between subroutines CRASH and TIMEC. KCA equals the number of iterations.

Location	Symbolic Name	Description
12	KFIRST	First pass flag. Flag equals one on the first step of a numerical integration
13	IEVG	Engine number of engine thrusting
14-28	-	Not used

COMMON CONST

Description: This common block contains physical constants and unit conversion factors

Length: 50 8-byte words

Subroutines Using:

MAIN	APROCH	ARMPIT	AVEQNS
AVERGE	AVSTRT	BLOCK DATA	BURND
CLOSE	CONTRL	DOPLER	DRAG
EQNS	FIELD2	FIXATG	FOWARD
GRAV	INTEG	INTERP	JET
LUNA	MCBURN	MCSET	MCVERF
MDCORS	MINV	MONTE	MULCON
M50EPM	M50JPM	OBLATE	OBLE
OBLTY	ORBIT	OUTPUT	POST
PRINT	PROTO	QUIKIE	RETDV
SETUP2	SHADOW	SOL	SOLP
SPER	SUNMIN	TARGET	TIMEC
TOBODY	TRIM	TRIM2	TRMN
TWELVE	TWOPIT	VISIB	

LOCATION	SYMBOLIC NAME	DESCRIPTION
1	RAD	Radian to degree conversion factor
2	PI	Pi, π
3	PI2	Twice pi
4	Au	Number of kilometers in an astronomical unit
5	GM(12)	Gravitational coefficients of the planets, km^3/sec^2. The planets are ordered in the array as follows,
17	RE(12)	Equatorial radii of the planets, km. The order is the same as the gravitational coefficients.
29	WP(12)	Rotation rates of the planets, rad/sec. The order is the same as the gravitational coefficients.
41	THS	Hour to second conversion factor
42	TSH	Second to hour conversion factor
43	TDS	Day to second conversion factor
44	TSD	Second to day conversion factor
45	G	Gravitation acceleration at the surface of the Earth, km/sec^2

1. Mercury	4. Mars	7. Uranus	10. Sun
2. Venus	5. Jupiter	8. Neptune	11. Moon
3. Earth	6. Saturn	9. Pluto	12. Oddball

COMMON DUM

Description: This common block contains quantities used to propagate the state using the multi-conic scheme.

Length: 12 8-byte words

Subroutines Using: AVERGE MULCON OBLATE TOBODY

Location	Symbolic Name	Description
1	A1S(3)	Acceleration due to the Moon's indirect term, kms/sec^2
4	A2S(3)	Acceleration due to the Sun, km/sec^2
7	PM	Spacecraft's mean motion, rad/sec
8	-	Not used
9	AM	Value of mean anomaly at end of a compute step, rad
10	DT	Current multi-conic compute step, sec.
11	AMO	Value of the mean anomaly at the beginning of the step, rad.
12	-	Not used.

COMMON ELMNT

Description: This common block is used to transfer input data read from the 24-hour hold file, It is also used for writing data for GTDS retrieval.

Length: 40 8-byte words followed by
8 4-byte integer words.

Subroutines Using: APROCH FIND MCVERF PUTELS SETUP2

Location	Symbolic Name	Description
1	DATE	Epoch date in year, month and day written as YYMMDD.
2	TIME	Epoch GMT time in hours, minutes and seconds written as HHMMSS.SSS.
3	X(3)	Position vector, km.
6	DX(3)	Velocity vector, km/sec.
9	ELM(6)	Orbital elements
15	COV(21)	Covariance matrix of the state. Upper triangle presented.
36-40	-	Not used.
41	ID	Satellite identification number.
42	ICOR	Coordinate system of state. 1 = Earth mean equator and equinox of 1950.
43	ICENT	Central body indicator. 1 = Earth 2 = Moon
44	ISET	Element set number of desired data.
45-48	-	Not used

COMMON FIELDM

Description: This common block contains constants that describe the gravitational field of a planet.

Length: 297 8-byte words followed by
2 4-byte integer words.

Subroutines Using: MAIN FIELD2 INPUTF OBSET

Location	Symbolic Name	Description
1	ZONL(16)	Zonal coefficients of the spherical harmonic potential term. ZONL(i) = $- C_{i0}$ $i = 1, 16$
17	TSRL(16, 17)	Tesseral coefficients of the spherical harmonic potential term. TSRL (i, j) = C_{ij} $i = 1, 16$ $0 < j \leq i$ TSRL (j, i+1) = S_{ij} $i = 1, 16$ $0 < j \leq i$
513	SELNEQ(9)	Rotation matrix from the Earth mean equator and equinox of 1950 to true equator and prime meridian of the planet for which the gravitational field is to be evaluated.
522	NMOD	Maximum number of zonals used to define the gravity field.
523	MMOD	Maximum number of tesserals used to define the gravity field.

COMMON GRAVTY

Description: This common block contains vectors used in the determination of the disturbing acceleration when numerically integrating

Length: 6 8-byte words

Subroutines Using:

ACCEL	AVEQNS	DRAG	EQNS
FIELD2	GRAV	OBLE	SOLP

Location	Symbolic Name	Description
1	POS(3)	Position vector with respect to central planet in Earth mean equator and equinox of 1950, km
4	VEL(3)	Velocity vector in same system, km/sec

COMMON INPUT

Description: This common block contains the data input to MAESTRO.

Length: 1000 8-byte real words followed by
100 4-byte integer words

Subroutines Using:

MAIN	ACCEL	APROCH	ARMPIT
AVEQNS	AVSTRT	BELL	BLOCK DATA
BURND	CALEND	CLOSE	CONTRL
CRASH	DOPLER	DRAG	EQNS
FIELD2	FIXATG	FOWARD	GRAV
INPUTF	INTEG	INTERP	JET
LUNA	MCBURN	MCSET	MCVERF
MDCORS	MINV	MONTE	MOTORS
MULCON	OBLATE	OBLTY	OUTPUT
OUT1	PLANET	POST	PRINT
PROTO	RETDV	RETRO	RKSEVN
SADOUT	SENSO	SETUP2	SHADOW
SOL	SOLP	TABINT	TARGET
TIMEC	TRIM	TRIM2	TWELVE
TWOPIT	UPDATE	VISIB	

MAESTRO INPUT ARRAY*

LOCATION	FORTRAN SYMBOL	USE***	PRESET VALUE	DESCRIPTION
1	ERRC	I	-	Error control for automatic integration step size. If = 0 assume in fixed-step mode.
2	DELTO	I	-	Initial computer step when in automatic mode. Note: Need not be input when fixed-step.
3	DELMN	I	-	Minimum compute interval. Note: Not necessary in fixed-step mode.
4	TF	I	**	Run stop-time.
5	PCON	I	1.0	Unit conversion factor for positions. The value is determined such that, when PCON multiplies the input units, they will be converted to KMS. The output units will be scaled by 1/PCON so that the output units will be the same units as input.
6	VCON	I	1.0	Used the same as PCON except that velocities are scaled to KM/SEC.
7	EMPTY			
8	TOL	I	0.0	Corrector convergence tolerance in twelfth-order predictor-corrector integration scheme. Values used depend on the accuracy requirements. They range from 10^{-6} to 10^{-12}.
9	EMPTY			

* unless otherwise stated, the units of the input quantities are: KM, SEC, KG, degrees

** preset value depends on program operating mode

*** I integration; V Midcourse verification
A approach analysis; L Lifetime analysis
M midcourse analysis; P Post-injection trim
C Monte Carlo analysis

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
10-12	TMETH(3)	I	**	Times for method table. See KMETH (location 1036) for description.
13-19	EMPTY			
20 21 22 23 24 25	XMONL DAYL YRL HRL XMINL SECL	I	-	Launch epoch
26-29	EMPTY			
30 31 32 33 34 35	ELM(6)	I	-	Input initial conditions as orbital elements. The order is a, e, f, ω, i, Ω. The initial conditions may be accepted as position and velocity vectors (see locations 40-45). The coordinate system is defined in KINPT (location 1019).
36	EMPTY			
37	DJL			Modified julian launch date. Not a program input.
38	WTO	I	331.40	Initial weight
39	ETC	I	-	Ephemeris time correction. If no input, the ephemeris time correction will be calculated from ETC = 38.66 + .0025921DJ where DJ is the number of days since the julian date of 2440000.0.
40 41 42	X(3)	I	-	Initial conditions as position and velocity vectors or spherical coordinates. When spherical coordinates are used the order is velocity, flight path elevation angle, flight path azimuth angle, radius, geocentric latitude, geocentric longitude. The input coordinate system is defined by KINPT (location 1019)
43 44 45	DX(3)	I	-	
46	DJO			Modified julian launch date. Not a program input.
47	RA	I	-	Initial right ascension and declination in Earth equator and equinox of 1950
48	DEC	I	-	

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
50	XMON0			
51	DAY0			Initial time
52	YR0			
53	HR0	I	-	Input is month, day and year,
54	MIN0			hour, min, second (GMT).
55	SEC0			
56-76	COV	C	-	Upper right-hand triangle of the tracking covariance matrix loaded across the rows. Can be input in mean equator and equinox of 1950 or in a local tangent plane. Location 1085 determines the system. When mean of 1950 is used, the order is $X, Y, Z, \dot{X}, \dot{Y}, \dot{Z}$. When local tangent plane is input, the order is the position components as R, (RxV) x R, R x V and then the velocity components along the same axes.
77-99	EMPTY			
100-111	GM(12)	I	***	Gravitational coefficient. The order of the planets is defined in locations 1001-1012.
112-123	RE(12)	I	***	Planetary radii.
124-135	WP(12)	I	***	Planetary rotation rates
136-169	EMPTY			
170-179	TCOMP(10)	I	**	Compute interval table when in fixed compute interval mode, (loc (1) = 0.). Compute interval = DELT (I) when TCOMP (I-1) < T < TCOMP(I) or Compute interval = DELT (1) when T < TCOMP(1). Complete compute interval table must be loaded when alterations are made to preset values.
180-189	DELT(10)	I	**	
190-192	XMØØN(3)	I	-	Initial position of Moon if osculating elements are used for Moon's position;
193-195	DXMØØN(3)	I	-	Earth radii and Earth radii/mean solar day.
196	TMM	I	-	Moon's epoch in modified julian date.
197	SPRESS	I	$4.7(10)^{-5}$	Solar pressure at 1 AU, dynes/cm^2

*** See Table 3.3 of User's Manual

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
198	REFLK	I	0.2	Solar pressure reflectivity coefficient
199	EMPTY			
200-219	TF1(20)	I, A, M, V	***	Tabular thrust history of Motor 1. TF1(20) are times since ignition and F1(20) are the thrust values (newtons) at the corresponding times. For the RAE-B application, this motor is used as the midcourse motor.
260-279	F1(20)	I, A, M, V	***	
220-239	TF2(20)	I	-	Thrust table for motor 2.
280-299	F2(20)	I	-	
240-259	TF3(20)	I	-	Thrust table for motor 3.
300-319	F3(20)	I	-	
320-329	TWDOT1(10)	I, A, M, V	***	Tabular weight flow rate for motor 1. TWDOT1 is the time past ignition and WDOT is the flow rate at the corresponding time. The flow rate is in KG/sec.
350-359	WDOT1(10)	I, A, M, V	***	
330-339	TWDOT2(10)	I	-	Flow rate table for motor 2.
360-369	WDOT2(10)	I	-	
340-349	TWDOT3(10)	I	-	Flow rate table for motor 3.
370-379	WDOT3(10)	I	-	
380-382	TIG(3)	I, V	-	Ignition times for motors 1, 2 and 3.
383-385	TBØ(3)	I, V	-	Burnout times for motors 1, 2 and 3.
386-399	EMPTY			
400	TFIRE1	A	-	First and last trial retro firing time on an approach analysis. Time reference to liftoff epoch. Used when KAPOPT = 3 (location 1055)
401	TFIRE2	A	-	
402	DTFIRE	A	-	Increment in retro firing time.
403	RA0	A	-	Initial right ascension and declination used in the attitude sweep in the approach analysis. If both zero, velocity vector at closest approach is used.
404	DEC0	A	-	

*** See Table 3.3

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION	
405	DELRA	A	-	Increment in right ascension and declination in attitude sweep. Used when KAPOPT = 2 or 3	
406	DELDEC	A	-		
407	EMPTY				
408	CAR1	V	4.0095D8	Telemetry carrier frequency No. 1	
409	CAR2	V	4.0D8	Telemetry carrier frequency No. 2	
410-419	OBSLON(10)	A,M,V	***	Observation site geocentric longitude for sites 1-10.	
420	PSID(1)	M	2838.	Lunar radius	Midcourse analysis desired end conditions
421	PSID(2)	M	116.5	inclination w.r.t. target planet equator.	
422	PSID(3)	M	.396D6	time of flight w.r.t. liftoff	
423	PSID(4)	M	0.62	Hyperbolic excess speed	
424	PSID(5)	M	0.0	circular excess speed	
425-429	PSID(6-10)	M		Empty	
430-432	DV(3)	I	-	Impulsive velocity of engines 1 to 3. Used when KFMØN (location 1047) is set to 2. Velocities are added at ignition times in location (380-382).	
433	SOLARA	I	13560	Area of the spacecraft used in solar pressure calculation, square centimeters	
434	DELTMC	M	7200.	Increment in execution time	
435	SIGATM	C,M	0.7	One sigma pointing error during midcourse maneuver.	
436	SIGDVM	C,M	0.02	One sigma percentage error of midcourse velocity. Loaded as percent/100.	
437	SIGATR	C	0.7	One sigma pointing error during retro.	

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
438	SIGDVR	C	0.0003	One sigma percentage error of retro velocity. Loaded as percent/100.
439	TMC1	C	36000.	Time of first correction
440	TMC2	C	324000.	Time of second correction
441	ASPMC	A,M,C	226.0	Midcourse motor specific impulse
442	ASPR	A,M,C	282.5	Retro motor specific impulse
443	WRETRO	A,M,C	71.44	Mass of retro fuel
444	RO	A,M,C,P	2838.	Desired lunar orbit radius used in trim maneuver
445	VC	A,M,C,P		Circular velocity at RO. Program calculation, not to be input.
446	CIO	A,M,C,P	116.5	Desired lunar orbit inclination w.r.t. lunar equator used in trim maneuver
447	BVD(1)	M	6000.	Desired B · T Desired B · R Midcourse analysis desired end conditio Used when IBTR flag set to 1. (loc 1062)
448	BVD(2)	M	6000.	
449	TRINC	M,C,P	0.	Tolerance band on inclination correction during post-injection trim. If inclination change is less than TRINC, no inclination adjustment is made.
450-459	TOUT(10)	I	TOUT(1)= 1.D20	Print table
460-469	DTOUT(10)	I	DTOUT(1)= 18000.	Print interval = DTOUT(1) when TOUT(I-1) < T < TOUT(I) or print interval = DTOUT(1) when T < TOUT(1)
470	ATFULA	C	6.0	Available attitude control fuel and constant to determine attitude control fuel, KG/RAD.
471	AFUEL	C	0.1678	Attitude fuel used = AFUEL * attitude angle change.
472	FTØT	C	20.4	Total midcourse fuel available

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
473	WDROP	A,M,C	13.77	Retro drop weight. Weight dropped after retro firing.
474	CONE	A, M	5.0	Cone angle used in approach analysis. The attitude range is this cone angle about the velocity vector or input attitude when KAPOPT=1 (loc 1055). Also used in midcourse fixed attitude scan.
475	TRUE	A,M, C	20	True anomaly range for trial retro firings in approach analysis, deg. Trial retro firings are made from -True to +True true anomaly on the approach hyperbola. Used when KAPOPT =1. Also used as firing range in retro optimization.
476	BURNT	V	-	Midcourse motor burn time. Used when the initial state is not obtained from a midcourse analysis.
477	EMPTY			
478	TMC	M	7200.	Initial execution time
479	DINK	M	0.0003	Secant partial step size in midcourse analysis also velocity increment when using the fixed attitude mode.
480-489	OBSLAT(10)	A,M,V	***	Observation site geocentric latitude for sites 1-10
490	TØL(1)	M	10.	B·T — Midcourse analysis tolerances on desired end conditions
491	TØL(2)	M	10.	B·R — Midcourse analysis tolerances on desired end conditions
492	TØL(3)	M	10.	time of flight — Midcourse analysis tolerances on desired end conditions
493	TØL(4)	M	.0001	hyperbolic excess speed — Midcourse analysis tolerances on desired end conditions
494	TØL(5)	M	.0001	circular excess speed — Midcourse analysis tolerances on desired end conditions
495	TØL(6)	M	.02	Total fuel optimization — Midcourse analysis tolerances on desired end conditions
496	TØL(7)	M	5.	closest approach radius — Midcourse analysis tolerances on desired end conditions
497	TØL(8)	M	.2	inclination — Midcourse analysis tolerances on desired end conditions
498-499	TØL(9,10)			Empty
500-511	RSWTCH(12)	I	***	Spheres of influence of the planets used to determine the central planet. If the distance from the planet is less than the value in RSWTCH, the planet is the central planet. If none of the planets are central, then the Sun is considered central.

*** See Table 3.3

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
512	RBURN	A	20.	Retro burn time
513	TCATST	I	0.	Time to begin closest approach testing logic.
516	SPNRA	I	12.0	Spin rate, RPM.
517	PTI	I	238.0	Initial midcourse motor tank pressure, PSIA.
518	TPI	I	20.	Initial midcourse motor temperature, ^{o}C.
519	FMC	I	20.4	Current midcourse motor fuel, ke.
520	FMULT	I	1.	Thrust multiplier.
521	WMULT	I	1.	Weight multiplier.
522	WDOTT	M	.0183	Midcourse motor weight flow used with Hamilton Standard thrust in midcourse targeting.
523	DTM	M	0.	Compute interval during motor burn in midcourse targeting. Compute interval equals burntime when value is zero.

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1001-1012	KP(12)	I	-	Bodies in system. Set to 1 if only position is necessary, 2 if both position and velocity are desired. The order is as follows:

1001 MERCURY	1005 JUPITER	1009 PLUTO
1002 VENUS	1006 SATURN	1010 SUN
1003 EARTH	1007 URANUS	1011 MOON
1004 MARS	1008 NEPTUNE	1012 ODDBALL

Note: 1003 = 2, 1010 = 1, 1011 = 2 are preset

1013	METH			Current trajectory propagation method. Not a program input.
1014	KINT	I	3	Numerical Integration Scheme 3. 7th-Order Runge-Kutta 5. 12th-Order Multistep, single step size only
1015	JL	I	3	Launch planet number.
1016	ENGID	A	2	Engine number of the retro motor. Used in a MODE=1 analysis.
1017	JMN	I	5	Lunar and solar ephemeris flag 1. Mean elements 2. Mean elements for Sun Mean elements + 1st-order corrections for Moon 3. Ephemeris tape 4. Mean elements for Sun, osculating elements for Moon (loaded in locations 190-196) 5. Ephemeris tape using Goddard's direct read feature
1018	KOBLTE	I	1	Set to 1 for Earth oblateness. Should be set to 0 when 1029 = 3.

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1019	KINPT	I	1	Input coordinate system flag 1. Mean equator and equinox of 1950 2. Mean equator and equinox of date 3. Mean ecliptic and equinox of date 4. True equator and prime meridian of launch planet 5. True equator and equinox of date 6. True lunar equator and node. 7. Earth spherical. See location 40-50 of INPUT common
1021-1028	EMPTY			
1029	KOBL	I	**	Set to the number of the planet in which the gravitational field is to be simulated using FIELD2. See location 1035.
1030	KOUT	I	**	If 1, output according to print table. If zero, output only at beginning and end of run. If -1, no output.
1031	JT	I	11	Target planet number
1032	KCRASH	I	**	Closest approach flag 0 No closest approach test 1 Continue after closest approach 2 Stop on closest approach
1033	EMPTY			
1034	JOCC	I	0	Occultation flag. Set to the planet number where the observer is located.
1035	MODLEM	I, L	1	Lunar gravity model flag 1 for Houston L1 model of Lunar field 2 for Earth J_2, J_3, J_4 3 for JPL 15 by 8 Lunar Field 5 zeroes initial field so new field can be input 10 Used field set in last case
1036-1038	KMETH(3)	I	**	Trajectory propagation method If T < TMETH(1) METH = KMETH(1) If TMETH (I-1) < TC < TMETH(I) METH = KMETH(I) TMETH is in location 10. 1. Cowell 2. Encke 3. NICE/True 4. NICE/Mean 5. Averaged Equations of 4 6. Multiconic, fixed stepsize only 7. Integrals e cos ω, ω+ f 8. Equations 7 are used in averaging

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1039-1040	KOUTPT(2)	I	3,2	Output coordinate systems - Launch and target planets 1. Mean equinox and ecliptic of date 2. True equator and prime meridian 3. Mean Earth equator and equinox of 1950 4. True Earth equator and equinox of date 5. No output
1041	JRA	A, M	10	Number of right ascensions and declination angles used in the attitude sweep of the approach analysis. Must be less than 16. Also used in midcourse fixed attitude analysis.
1042	JDEC	A, M	15	
1043	KFIRE	A, M	20	Number of trial retro firing times on approach analysis less than 20
1044	MODE	ALL		Program mode 0 Fly to TF or closest approach 1, 2 Approach analyses 3 Midcourse analyses 4 MonteCarlo analyses 5 Midcourse verification mode 6 Lunar lifetime mode 7 Post-injection trim
1045	KDØP	V	0	Set to one for doppler analysis
1046	KAPOUT	A	0	Set to one if retro firing time analysis is to be output for each attitude
1047	KFMOD	I	0	Thrusting mode 0 Tabular thrust / weight 2 Impulsive velocity 3 Hamilton Standard subroutine
1048	KAPSAD	A	1	Set to one for lunar orbit shadow calculations during approach analysis.
1049	KSADØW	I	0	Shadow calculation flag during trajectory propagation: 0 No shadows determined. 1 Shadow time determined by interpolation while numerically integrating trajectory. 2 Osculating orbit used to determine times. Note: KSADOW=1 should not be used when averaging.

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1050	MCOUT	M	0	Extra output flag in midcourse analysis. 1. Outputs targeted solution from PROTO 2. Prints ΔV, constraint errors for each iteration. 3. Prints jet iteration information along with 2.
1051	JMC	M	10	Number of midcourse execution times simulated
1052	KMONTE	C	2	Monte Carlo description flag 1. retro only 2. midcourse and retro 3. two midcourses and retro
1053	KMAX	C	50	If negative, the first correction will be calculated from the nominal assuming that the tracking data is good. Sample size of Monte Carlo analysis.
1054	KSTART	C	17	Random number starter.
1055	KAPOPT	A	1	Approach analysis option flag. KAPOPT = 1 Firings made between an input range of true anomaly about perigee on the approach hyperbola. The attitude range is an input cone angle about the velocity vector at closest approach. KAPOPT = 2 Firings made from asymptote to asymptote on the approach hyperbola. The attitude range is input as initial right ascension and declination, increment in right ascension and declination, and number of attitude angles. KAPOPT = 3 Firings made between input times. The attitude range is input as in 2.
1056	NGRAPH			If positive, Element set number, for MAESTRO graphics data base. If negative, replace |NGRAPH| element set.
1057	KREAD	A, V	-	Integer used to obtain initial conditions from a midcourse analysis. KREAD corresponds to the midcourse correction number desired. If KREAD is zero, this option is ignored and the initial state must be input.
1058	KOUT9	I, L	**	Auxiliary peripheral output unit number

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1059	KTERM	I, L	0	Extra output flag. Used when KOUT9 ≠ 0 0. Orbital elements printed 1. Position and velocity vectors printed 2. Both 0 and 1
1060	INPWT	I	1	Input array write flag 0 no write 1 write
1061	MCUNIT	A, M, V	11	Unit number of auxiliary midcourse output unit.
1062	IBTR	M	2	Miss vector option flag in Midcourse Analysis 1. Use B·T and B·R loaded in BVD 2. Use PSID(1) and PSID(2)
1063	KGLAW	M	2	Midcourse guidance law 1. Minimum fuel 2. Fixed time of arrival 3. Fixed target energy 4. Variable target energy 5. Total fuel optimization
1064	NGROPT	M	1	Number of trials for which secant matrix is recomputed in Midcourse Analysis.
1065	NT	M	10	Number of trials allowed in Midcourse convergence.
1066	JET	M	1	If set, preliminary targeting will be done in the Midcourse Analysis.
1067	MCLIM	M	100	Limiting factor in the Midcourse Analysis. The midcourse correction is limited to MCLIM * DINK (KM/SEC) on each iteration.

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1068	NORD	I, L	6	Number of ordinates in averaging – less than 16.
1069	INT	I, L	3	Number of intervals in averaging – less than 16
1070	KPROB	M	95	Output probability for midcourse execution error. Second midcourse execution time and errors are input through locations 440, 435, and 436. Negative or zero skips error propagation.
1071	IBURN	M	6	Trajectory propagation method during finite burn of midcourse motor. Impulsive calculations are used when set to zero.
1072	KROUT	M, C	0	Extra output flag during retro optimization calculations.
1073	KORECT	I	0	If set nonzero, the derivative at the end of an integration step will not be calculated.
1074	KMTOUT	C	0	Flag used to output initial state for each sample in a Monte Carlo analysis.
1075	KMETHP	M	6	Trajectory propagation method when generating partial derivatives.
1076	IFIND	A11	-	Element set number of the anchor vector to be transferred from the differential correction program.
1077	KTF	M	0	Flag used to determine the type of Midcourse analysis. = 0 One-dimensional scan of midcourse execution times > 0 Two-dimensional scan of midcourse execution times and flight times. Flight times are scanned in KTF one-hour steps beginning at the desired flight time in location 422. < 0 Two-dimensional scan of midcourse execution times and midcourse impulsive velocity. The impulsive velocity is centered about value, loaded into 426 and varied in -KTF steps of size DINK (location (479).

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1078	IVTI	M	0	Overburn option key 0. in-plane retro antiparallel at periapsis, ±1. variable inclination procedure approaching above or below desired inclination ±2. variable periapsis procedure circularizing in-plane before or after periapsis.
1079	KHIGH	I	0	Farthest approach flag. If set, furthest approach will be found.
1080	NORMIN	M, C	0	Retro optimization flag 0 in-plane at periapsis maneuver for underburns, according to IVTI for overburns ±1. optimize retro to trim inclination in PROTO 2. Same as 1, but in TARGET also.
1081	NREV	A	0	Number of revolutions the transfer orbit completes before stopping at closest approach in an approach analysis
1082	KAPWT	A	0	Flag used to write information from the approach analysis in an auxiliary unit. It is set to the output unit number when the option is desired.
1083	NAPUNT	P, L	11	Unit numbers where initial conditions are stored from a previous approach analysis on post-injection trim analysis.
1084	KSOLP	I	0	Solar pressure flag. 0. no solar pressure 1. pressure in radial direction only (s/c assumed to be sphere) 2. spacecraft assumed to be a cylinder spinning along attitude vector.
1085	KCOV	C	0	Coordinate system of covariance matrix 0. mean Equator and equinox of 1950. 1. local tangent plane
1086				Used in PEST version
1087	IDATT	all	0	Element set number of attitude file. Used when attitude is to be input via read from ADP.

LOCATION	FORTRAN SYMBOL	USE	PRESET VALUE	DESCRIPTION
1088	NATUNT	all	12	Unit number of attitude data.
1089	IDSAT	all	1234567	Satellite identification number.
1090	INIT	all	0	Initial line number of MAESTRO's space allotted on the director's display. If INIT is zero, no writes are made on the director's display. This option is only used during real-time operations.
1091	KPIT	P	0	Flag used to indicate post-injection trim targeting.
1092	ISET	all	0	Element set number when using subroutine PUTELS to transfer state to GTDS program.
1093	KPLOT	all	0	Set to the plot unit number when plotting.
1094	NMOD	I, L	0	Maximum number of zonals and tesserals used to define the gravity field.
1095	NMOD	I, L	0	
1096				Used in PEST version.
1097	KATMOS	I	0	Atmosphere drag flag. Set to 3 for Earth Drag.
1098	KASTRT	all	0	Flag used to initiate start-up procedure to obtain average elements from osculating elements.
1099	KTIST	I	0	Flag used to determine thrusters used in Hamilton Standard program 0 both thrusters 1 first thruster only 2 second thruster only

COMMON INPUTS

Description:	This common block contains the saved input array. This common block is required to stack cases.
Length:	1000 8-byte real words followed by 100 4-byte integer words
Subroutines Using:	MAIN INPUTF

This common block is essentially the same as INPUT common.

COMMON INTER

Description: This common block contains quantities used in the interpolation logic of state propagation by numerical integration.

Length: 130 8-byte words followed by
1 4-byte integer word

Subroutines Using: CLOSE INTERP TIMEC UPDATE

Location	Symbolic Name	Description
1	X(10)	Table of times corresponding to the saved values
11	POS(6, 10)	Array of saved values of the integration variables at the times corresponding to X.
71	ACL(6, 10)	Array of derivatives of the integration variables at the times corresponding to X.
131	INT	Integer used to indicate the current value of the arrays X, POS and VEL.

COMMON INTVAR

Description: This common block contains the variables used in numerical integration

Length: 14 8-byte words

Subroutines Using:

MAIN	ARMPIT	AVEQNS	AVSTRT
CRASH	EQNS	FIELD2	FOWARD
GRAV	INTEG	JET	LUNA
MOTORS	OUTPUT	OUT1	PLANET
POST	RKSEVN	SETUP2	SHADOW
SOL	TIMEC	TWELVE	UPDATE

Location	Symbolic Name	Description
1	X	Independent variable of the numerical integration, usually time (sec)
2	Y(6)	Dependent variables of the numerical integration. The quantities depend upon the trajectory propagation technique in use.
8	RATES(6)	Derivatives of the dependent variables with respect to the independent variable
14	H	Compute interval is used, same units as X.

COMMON INTVRX

Description: INTVRX is used in averaging osculating elements.

Length: 12 8-byte words plus
1 4-byte integer word

Subroutines Using: AVSTRT EQNS RKSEVN

Location	Symbolic Name	Description
1	Z	Osculating elements, Method 7
7	QATES	Time derivatives of Z
13	NQ	Number of elements to be averaged (6)

COMMON MCCOM

Description: This common block contains variables pertaining to the midcourse correction analysis.

Length: 150 8-byte words followed by
50 4-byte integer words

Subroutines Using:

APROCH	BELL	BURND	FIXATG
JET	MCBURN	MCSET	MCVERF
MDCORS	MINTF	MONTE	POST
PROTO	SENSO	SETUP2	TARGET

Location	Symbolic Name	Description
1	ALIMIT	Maximum allowable change in midcourse velocity on each iteration, km/sec.
2	TR	Epoch of target planets state XS (location 32 in MCCOM common), seconds since state epoch.
3	PRX	Probability level of midcourse execution errors, percent.
4-5	-	Not used.
6	XMC(6)	Midcourse pre-maneuver state. Cartesian position and velocity vectors in Earth mean equinox and equator of 1950, km and km/sec.
12	DV(3)	Midcourse impulsive velocity correction vector in Earth mean equator and equinox of 1950, km/sec
15	DVS(3)	Spherical components of midcourse velocity DV. Magnitude, declination and right ascension, respectively, km/sec and deg.
18	TMCS	Midcourse correction execution time, seconds since state epoch.
19	BVD(2)	Desired impact parameter vector at target planet, B·T and B·R, respectively, kms.
21	XSUN(3)	Vector from the spacecraft to the Sun in Earth mean equator of 1950.

Location	Symbolic Name	Description
24	DVB4	Midcourse impulsive velocity correction on previous correction, km/sec.
25	DVRET	Impulsive velocity of retro motor, km/sec
26	EXFUEL	Expected second midcourse fuel due to first midcourse errors (kg)
27	TV(3)	Unit vector along the spin-axis at retro maneuver time (true equator and prime meridian of target body)
30	DVT	Required post-injection trim velocity, km/sec.
31	FUELT	Fuel required for the post-injection trim maneuver, kg.
32	XS(6)	Cartesian target planet state resulting from current midcourse maneuver in true equator and prime meridian of target planet, km and km/sec.
38	PFAC	Midcourse variable target energy guidance constant. Set to PFAC = ASPR * WRETRO/ASPMC where ASPR is the retro motor's specific impulse, sec. ASPMC is the midcourse motor's specific impulse, sec. WRETRO is the retro motor's fuel, kg.
39	DJDIF	Time from liftoff to state epoch, sec.
40	SIGOUT(1)	Time, sec.
41	SIGOUT(2)	Hyperbolic excess speed, km/sec
42	SIGOUT(3)	Arrival circular excess speed, km/sec
43	SIGOUT(4)	Required post injection trim fuel, kg.
44	SIGOUT(5)	Radius, km
45	SIGOUT(6)	Inclination, deg.

} errors at target planet due to midcourse execution errors.

Location	Symbolic Name	Description	
46	FFIRE	True anomaly of retro-fire on the approach hyperbola (rad)	
47	WTF	Spacecraft weight after midcourse correction, kg.	
48	TMC2	Time of second midcourse, seconds since state epoch.	
49	EXV2	Expected magnitude of second midcourse correction, km/sec	
50	DPT(3,10)	Partial derivative matrix. This matrix represents the partial derivative of the output variables with respect to the midcourse correction velocity. The order of the output quantities is the same as the PSI Array starting in location 100.	
80	PSID(1)	Radius, km.	Desired end conditions at target planet. Array is ten long, only six are currently in use.
81	PSID(2)	Inclination, deg.	
82	PSID(3)	Time of flight, sec.	
83	PSID(4)	Hyperbolic excess speed, km/sec.	
84	PSID(5)	Circular excess speed, km/sec.	
85	PSID(6)	Total fuel, kg.	
86	PSID(7)	Central velocity value for fixed- attitude guidance scan (km/sec)	
90	TOL(1)	B·T, km.	Convergence tolerance on end conditions. Only six locations of a ten long array are currently in use.
91	TOL(2)	B·R, km.	
92	TOL(3)	Time of flight, sec.	
93	TOL(4)	Hyperbolic excess speed km/sec	
94	TOL(5)	Circular excess speed km/sec.	
95	TOL(6)	Minimum gain in total fuel, kg.	

Location	Symbolic Name	Description
100	PSI(1)	B·T, km
101	PSI(2)	B·R, km
102	PSI(3)	Time of flight, sec.
103	PSI(4)	Hyperbolic excess velocity, km/sec.
104	PSI(5)	Circular excess velocity, km/sec
105	PSI(6)	Fuel expended
		Constraint error vector. Difference between the desired parameters and the actual parameters.
110	DW(10)	Expended weight table - See MCSET, MCBURN (kg)
120-129	-	Not used
130	XMCOM(3)	Vector from s/c to Moon in Earth mean equator of 1950
133	XEARTH(3)	Vector from s/c to Earth in Earth mean equator of 1950
136-139	-	Not used
140	EC	Eccentricity of post trim orbit
141	SEMT	Semi-major axis of post trim orbit
142-150	-	Not used
151	NT	Maximum number of iterations allowed
152	KM	Minimum total fuel return key
153	JUMPTF	Minimum total fuel (MINTF) logic indicator
154	KBURN	Midcourse motor burn computation method = 0 impulsive =1-8 corresponds to the trajectory propagation methods. See location 1013 of INPUT common
155	I3	Indicator for third constraint parameter in MDCORS
156	KENTRY	Entry key for MDCORS - skips initialization
157	IT	Current iteration number
158	IR	Return key set by subroutine MDCORS
159	KMC	Midcourse execution counter
160	NP	Number of constraints = 2 when minimum fuel guidance is used = 3 otherwise
161	KT	Flight time scan counter
162	-	Not used

Location	Symbolic Name	Description
163	NGROPT	Number of trials to recompute secant matrix
164	KGLAW	Guidance law = 1 Minimum fuel 2 Fixed time of arrival 3 Fixed target energy 4 Variable target energy 5 Minimum total fuel
165	ICB	Central body number of XMC state in location 6.
166	ISP	Key to determine if gradient has been computed.
167	IBTR	Miss vector option key = 1 B · T and B · R 2 Radius of closest approach and inclination
169	IPD(3)	Constraint indicator array - constraints pointed out by this array are tested against tolerances for convergence.
172	KEL(10)	Array of elevation angles of the spacecraft at the midcourse execution time. The array corresponds to observation sites 1 - 10, (deg / 10)
182	KELR(10)	An array similar to KEL except at the retro firing time.

COMMON MOON

Description: This common block is used to describe the Lunar ephemeris when the Moon is represented by osculating elements.

Length: 15 8-byte words

Subroutines Using: LUNA SETUP2

Location	Symbolic Name	Description
1	ELMMN(12)	Osculating orbital elements of the Moon and their sines and cosines, km and rad
13	TMOON	Epoch of the orbital elements, days since 2400000 julian date
14	PM	Mean motion, rad/sec
15	FOM	Mean anomaly of osculating Moon at TMOON epoch, rad.

COMMON OBSIT

Description: This common block contains quantities which describe the location of the observation sites on the Earth.

Length: 140 8-byte words

Subroutines Using: DOPLER MCVERF SETUP2 VISIB

Location	Symbolic Name	Description
1	DOBS(10,2)	Velocity of the 10 observation sites in the Earth true equator and prime meridian, km/sec.
21	XOBS(10,3)	Position vector of the 10 observation sites with respect to the center of the Earth in Earth true equator and Greenwich, km.
51	OBSROT(9,10)	The nine elements of a rotation matrix for each of the 10 tracking stations. The rotation matrix transforms a vector in the Earth true equator and Greenwich to an observation site local with the X-axis north, Y-axis west and the Z-axis up.

COMMON PERT

Description: This common contains quantities which have to do with the evaluation of the disturbing acceleration and the current Encke reference orbit.

Length: 30 8-byte words

Subroutines Using:

MAIN	ACCEL	AVEQNS	DRAG
EQNS	FIELD2	GRAV	INTEG
INTERP	MOTORS	OBLE	OUT1
PRINT	SETUP2	SOLP	TIMEC
TWELVE			

Location	Symbolic Name	Description
1	RCART(3)	Perturbing acceleration in Earth mean equator and equinox of 1950, km/sec^2
4	RSW(3)	Perturbing acceleration with respect to orbit plane. In radial, circumferential and normal to orbit plane, kms/sec^2
7-9	-	Not used
10	REFORB(12)	Orbital elements and sine and cosine of orbital elements representing Encke's reference orbit, km and rad.
22	EN	Mean motion of Encke's reference orbit, rad/sec
23	TREF	Epoch of Encke's reference orbit, time since state epoch, sec.
24	RECT	Parameter used to determine when to rectify Encke's reference orbit.
25	DOM(6)	Current position and velocity vectors of Encke's reference orbit, kms and kms/sec.

COMMON PIT

Description: PIT is used for communication during orbit trim calculations

Length: 20 8-byte words

Subroutines Using: ARMPIT TRIM2

Location	Symbolic Name	Description
1	CIF	Cosine of final orbital inclination.
2	SIF	Sine of final orbital inclination.
3	DV1	First trim maneuver's velocity impulse.
6	DV2	Second trim maneuver's velocity impulse.
9	DV3	Third trim maneuver's velocity impulse.
12	XS	Initial state vector.
18	DMGG	Storage vector of length 3.

COMMON PLNET

Description: This common block contains quantities that describe the position and velocity of the planets with respect to the central planet.

Length: 100 8-byte words

Subroutines Using:

MAIN	ACCEL	APROCH	ARMPIT
AVERGE	CLOSE	CRASH	DRAG
FIXATG	GRAV	JET	LUNA
MCVERF	MDCORS	MULCON	OBLE
OUTPUT	OUT1	POST	PRINT
READE	SETUP2	SHADOW	SOL
SOLP	TARGET	UPDATE	VISIB

Location	Symbolic Name	Description
1	XP(6, 12)	Position and velocity vectors of the planets with respect to the central planet in Earth mean equator and equinox of 1950, km and km/sec. The first index denotes the vectors while the second index denotes the planet number. The order of the planets is, 1 Mercury, 2 Venus, 3 Earth, 4 Mars, 5 Jupiter, 6 Saturn, 7 Uranus, 8 Neptune, 9 Pluto, 10 Sun, 11 Moon, 12 Oddball
73	DST(12)	Distance from the central planet to the planets, except that DST (JC) is distance to spacecraft where JC is central planet number.
85	NUT(4)	Earth nutation variables. Only available when tape ephemeris is used.
89-100	-	Not used

COMMON SAVE

Description: This common block is principally used to save various quantities for use in restoring the state.

Length: 45 8-byte words

Subroutines Using:

MAIN	CRASH	FOWARD	OUT1
PRINT	SETUP2	TIMEC	

Location	Symbolic Name	Description
1	SAVE(6)	Saved position and velocity vectors in mean equator and equinox of 1950, km and km/sec
7	TSAV	Epoch of saved variables, time since state epoch, sec
8	SAVE1(6)	Saved integration variables at time corresponding to TSAV.
14	SAVE2(14)	Saved Encke's reference orbit, mean motion and epoch of reference orbit.
28	SRATES(6)	Saved derivatives of the integration variables at TSAV.
34-39	-	Not used
40	TOUTL	Last time trajectory output was obtained during numerical integration, sec. since state epoch.
41	RSAV	Last sine of the flight path angle with respect to target planet. Used during closest approach iteration of the numerical integration.

COMMON SHAD

Description: This common block contains quantities used to determine the times of umbral and penumbral crossings.

Length: 28 8-byte words

Subroutines Using: FOWARD SADOUT SHADOW

Location	Symbolic Name	Description
1	DSAD(3, 5)	Saved distances from the shadow cone, km. The first index corresponds to 3 back times. The second index corresponds to the type of shadow as follows: 1. launch planet umbra 2. launch planet penumbra 3. target planet umbra 4. target planet penumbra 5. occultation
16	TSAD(3)	Times of the 3 back values of the DSAD array
19	TSX(10)	Times of crossings, seconds from state epoch. This array is segmentated into groups of 2. The first of a group is the entrance time while the second is the time of exit from the shadow cone. The 5 groups are ordered in the same manner as the DSAD array.

COMMON STATE

Description: STATE common contains information which describes the state of the spacecraft at the end of a numerical integration step.

Length: 40 8-byte words

Subroutines Using:

APROCH	ARMPIT	AVEQNS	AVERGE
AVSTRT	BELL	CLOSE	CONTRL
CRASH	DOPLER	EQNS	FIELD2
FIXATG	FOWARD	INTEG	JET
LUNA	MCBURN	MCSET	MCVERF
MDCORS	MONTE	MOTORS	MULCON
OBLATE	OBLE	OUTPUT	OUT1
PLANET	POST	PRINT	PROTO
SADOUT	SETUP2	SHADOW	SOL
SOLP	TARGET	TIMEC	TOBODY
UPDATE	MAIN		

Location	Symbolic Name	Description
1	X(3)	Spacecraft's position vector in Earth mean equator and equinox of 1950, kms.
4	DX(3)	Spacecraft's velocity vector in Earth mean equator and equinox of 1950, kms/sec.
7	D2X(3)	Spacecraft's acceleration vector in Earth mean equator and equinox of 1950, kms/sec^2.
10	T	Epoch of the above state, second since state epoch. Note: state epoch in location 46 of INPUT common.
11	ATT(3)	Unit vector along spacecraft's attitude in Earth mean equinox and equator of 1950.
14	ELM(6)	An array of quantities that describe the current state. Location 13 of INPUT common. METH, determines the quantities:

METH =
1. not used
2 Encke's variables
3, 6 Orbital elements
4, 5 Orbital elements with mean anomaly, μ
7 Orbital elements with e cos ω, e sin ω and f + ω.
8 Same as 7 except f+M instead of f + ω

Location	Symbolic Name	Description
20-25	-	Not used
26	EJO	Modified ephemeris date of state epoch, days since 2400000 julian date.
27	TRU	Contains true or mean anomaly according to the setting of the METH flag in location 1013 of INPUT common. METH = 4, 5 mean anomaly METH = 8 true anomaly plus argument of perigee
28	EJT	Current ephemeris time, days since 2400000 julian date. Also valid at intermediate times of integration.
29	TCA	Time of closest approach, seconds since state epoch.
30-31	-	Not used
32	UJT	Current modified julian date, days since 2400000. Also valid at intermediate times of integration.
33	THRUST	Engine thrust, newtons.
34	WT	Spacecraft mass at engine ignition, kg.
35	W	Current spacecraft mass, kg.
36	SOL	Solar pressure constant. = solar pressure area x solar pressure at 1 au from the Sun x au^2 x 1 $(10)^{-8}$.
37	ACL	Magnitude of thrust acceleration (km/sec^2)
38-40	-	Not used

SECTION 5

Input/Output Units

This section describes the purpose of the input/output units required for the operation of MAESTRO. A description of these units is shown in Table 5.1. Some of the unit numbers are program inputs. If the unit number is greater than 1000, the number under unit number specifies the location in the input array where the unit number is to be input.

TABLE 5.1

Input/Output Units

Unit Number	Record Length (Bytes)	Subroutine Used	Purpose
1	3512	FIND	Directory of the GTDS 24-hour hold file. Required when input input location 1076 is non-zero.
5	36	INPUTF	Program data inputs. Unit required for all operations.
6	variable	Many routines	Program primary output device. Unit required for all operations.
10	7452	GETTAP	Ephemeris disk or tape. Direct read used with disk. Input location 1017 used to specify whether disk or tape. Unit required when 1017 equals 3 or 5.
12	280	SETUP2	Attitude input unit. Used to pass attitude from Attitude Determination Program to MAESTRO. Required when input location 1087 is non-zero.
13	80	MAIN	Director's display. Used during in-flight operations. Required when input location 1090 is non-zero.
21	64	POST, PROTO	Plot Unit. Used to write tape for post processing. Required when input location 1093 is set to 21.
26	7112	FIND	State input unit. Used to pass state from GTDS to MAESTRO via the 24-hour hold file. Required when input location 1076 is non-zero.
27	7200	PUTELS	State output unit. Used to transfer state to GTDS program. Required when input location 1092 is non-zero.

Unit Number	Record Length (Bytes)	Subroutine Used	Purpose
*1058	variable	OUTPUT PROTO POST	Auxiliary output unit. Required when input location 1058 is non-zero.
*1061	360	POST PROTO	Midcourse output unit. Unit number in input location 1061 must be specified when the midcourse analysis mode is used.

* The unit number is a program input. The number shown is the input location where the unit number is input.

SECTION 6

SUBROUTINE DESCRIPTION

This section presents a description of the subroutines which comprise the MAESTRO program. The descriptions include

1. Calling sequence
2. Purpose
3. Common Blocks Required
4. Subroutines Required
5. Input / Output descriptions
6. Theory
7. Description
8. Flow chart where applicable

MAIN PROGRAM

Purpose: The main program initializes some common blocks and calls the subroutine (s) which perform the desired MAESTRO analysis.

Common Blocks Required: AVG, CETBL2, CETBL3, CNTRL, CONST, FIELDM, INPUT, INPUTS, INTVAR, PERT, PLNET, SAVE, STATE

Subroutines Required: APROCH, ARMPIT, CONTRL, FOWARD, MCSET, MCVERF, MONTE, PROTO, INPUTF

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	ICW	1	CETBL2(1)	Tape ephemeris error flag
I	INIT	1	INPUT(1090)	Flag used to write director's file
I	MODE	1	INPUT(1044)	MAESTRO analysis flag
O	TAB3	829	CETBL3(1)	Raw ephemeris tape data

Description:

The main program initializes some common blocks and calls the appropriate subroutines to perform the desired MAESTRO analysis. No computations are done in this routine. MAIN is also used to define the length of many of the common blocks.

Initially common blocks CNTRL, PLNET and STATE are cleared. Next subroutine INPUTF is called to initialize the input array from program inputs. Writes on unit 13 are made if INIT is not equal to zero. These writes are messages which are displayed on a file known as the director's file and are only used in real-time operations. Next, the saved common is cleared and subroutines CONTRL and MCSET are called to initialize variables for later use. Finally, subroutines APROCH, MONTE, MCVERF, ARMPIT or PROTO are called according to the setting of the MODE flag. These subroutines are used to control the logic for one of the MAESTRO analysis modes. After the analysis is complete, flow returns to the call to INPUTF to initiate the next case.

SUBROUTINE ACCEL

Calling Sequence: CALL ACCEL

Purpose: This subroutine evaluates the acceleration of the spacecraft.

Common Blocks Required: CNTRL, INPUT, GRAVTY, PERT, PLNET

Subroutines Required: DVMAG, FIELD2, GRAV, MOTORS, OBLE, PLANET, SOLP

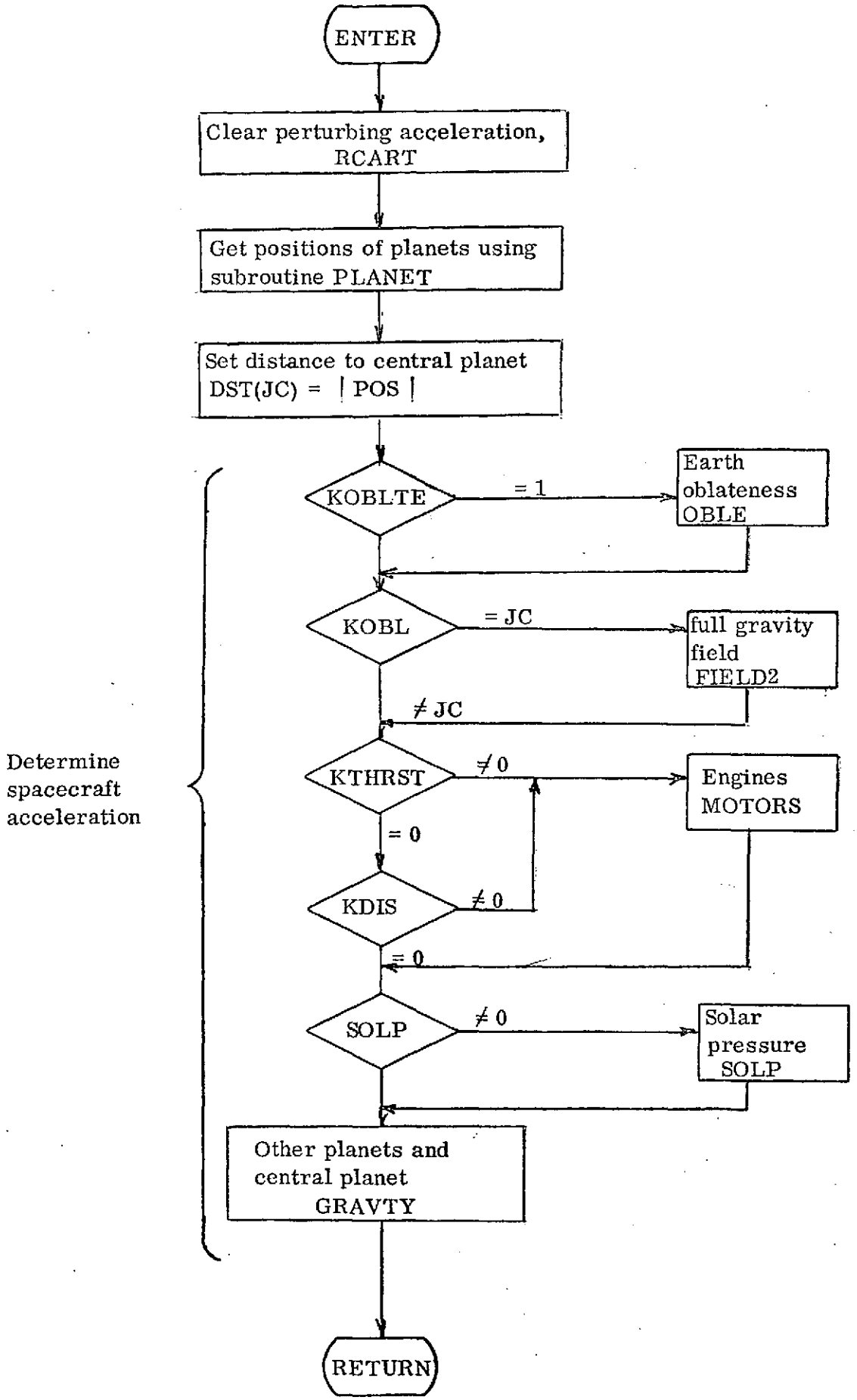
Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	JC	1	CNTRL(7)	Central planet number
I	KOBL	1	INPUT(1029)	Planet number of the planet used in the gravitation field evaluation
I	KOBLTE	1	INPUT(1018)	Earth oblateness flag
I	KSOLP	1	INPUT(1084)	Solar pressure flag
I	KTHRST	1	CNTRL(2)	Thrusting flag
I	K	1	CNTRL(5)	Discontinuity flag
I	POS	3	GRAVTY(1)	Position of the spacecraft w.r.t. central planet
O	RCART	3	PERT(1)	Spacecraft's acceleration

Description:

This subroutine controls the logic to determine the spacecraft's acceleration. Initially, the acceleration in RCART is cleared. Next, subroutine PLANET is called to obtain the current position of the planets. Subroutines OBLE, FIELD2, MOTORS and SOLP are called to evaluate the acceleration due to Earth oblateness, general gravitational field, engine firing and solar pressure. These subroutines are called only if input flags are set. Finally subroutine GRAV is called to determine the acceleration due to the planets in the system and the central planet if the Cowell trajectory propagation scheme is used.

SUBROUTINE ACCEL



SUBROUTINE APROCH

Calling Sequence: CALL APROCH

Purpose: This subroutine performs the analysis to determine the retromotor's firing time and attitude

Common Block Required: CNTRL, CONST, ELMNT, INPUT, MCCOM, PLNET, STATE

Subroutines Required: CROSS, FOWARD, M50MDT, NUTATE, ORBIT, QUIKIE, PUTELS, TRIM, TRMN, VNORM, VIEW, VISIB

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	TF	1	INPUT(4)	Final time
I	DJI	1	INPUT(37)	Launch epoch
I	WO	1	INPUT(38)	Initial S/C mass
I	DJO	1	INPUT(46)	State epoch
O	RTAC	1	INPUT(47)	Right ascension in retro motor altitude
O	DECL	1	INPUT(48)	Declination of retro motor altitude
I	HRO	1	INPUT(53)	Hour of state epoch
I	XMINO	1	INPUT(54)	Minutes of state epoch
I	SECO	1	INPUT(55)	Seconds of state epoch

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	TCOMP	10	INPUT(170)	Switching times of compute interval table
O	DELT	10	INPUT(180)	Compute intervals
O	TIG	1	INPUT(381)	Retro motor ignition time
O	TBO	1	INPUT(384)	Retro motor burnout time
I	TFIRE1	1	INPUT(400)	Initial retromotor firing time
I	TFIRE2	1	INPUT(401)	Final retro motor firing time
I	RAO	1	INPUT(403)	Initial right ascension of retro
I	DECO	1	INPUT(404)	Initial declination of retro
I	DELRA	1	INPUT(405)	Increment in right ascension
I	DELDEC	1	INPUT(406)	Increment in declination
I/O	DELV	1	INPUT(407)	Retro motor's impulsive velocity
I	ASPMC	1	INPUT(441)	Midcourse motor's ISP
I	ASPR	1	INPUT(442)	Retro motor's ISP
I	WRETRO	1	INPUT(443)	Mass of retro fuel
I	WDROP	1	INPUT(473)	Retro drop mass
I	CONE	1	INPUT(474)	Altitude cone angle
I	TRUE	1	INPUT(475)	True anomaly firing range

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	RBURN	1	INPUT(512)	Retro motor burn time
O	TCATST	1	INPUT(513)	Time to begin closest approach testing
I	JRA	1	INPUT(1041)	Number of trial right ascensions
I	JDEC	1	INPUT(1042)	Number of trial declinations
I	KFIRE	1	INPUT(1043)	Number of firing times
I	MODE	1	INPUT(1044)	Program mode flag
I	KAPOUT	1	INPUT(1046)	Firing analysis output flag
I	KAPSAD	1	INPUT(1048)	Lunar orbit shadow flag
I	KAPOPT	1	INPUT(1055)	Retro analysis option flag
I	KREAD	1	INPUT(1057)	Midcourse firing number when initial conditions from midcourse analysis
I	MCVNIT	1	INPUT(1061)	Midcourse unit number
I	NREV	1	INPUT(1081)	Number of transfer trajectory revolutions
I	KAPWT	1	INPUT(1082)	Unit number where approach firings are saved
I/O	X	3	STATE(1)	S/C position vector
I/O	DX	3	STATE(4)	S/C velocity vector

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	T	1	STATE(10)	Time since state epoch
O	ATT	3	STATE(11)	Unit vector along S/C altitude
O	ELMD	6	STATE(14)	S/C orbital elements
I	UJT	1	STATE(32)	Current modified julian date
I	WT	1	STATE(34)	S/C mass prior to motor ignition
I	JC	1	CNTRL(7)	Central planet number

Description:

This subroutine is used to determine the retro firing time and altitude in order to meet specific mission criteria. This version of subroutine APROCH was designed for the RAE-B mission. The most important criterion for this mission is the post injection trim fuel requirement. Other constraints are the spin axis-sun angle at firing, tracking station coverage, shadow times and orbit orientation. The retro conditions are determined in a brute-force manner. Repeated trial retro firings are made at various altitudes and firing times. The mission constraints are determined for each firing and output in displays so that the user can determine the firing time and altitude.

The retro motor can be simulated numerically or with the impulsive velocity approximation. Also, the pre-retro state can be determined numerically or with conic approximations. The MODE flag is used to determine which approximations apply. If the MODE flag is set to one, the entire analysis is performed using numerical techniques. The impulsive velocity and conic approximations are used when the MODE flag equals two.

The first part of the subroutine sets up constants and has logic to pick up the initial conditions from the midcourse tape if desired. The KREAD flag is used for this option. Logic is also included to set the closest approach test start time , TCATST, if more than one orbit of the transfer trajectory is desired. This option was included for Earth orbit missions and does not apply to a RAE-B type of mission. NREV should equal zero. At this point the logic flow divides according to the type of analysis desired - approximate or precise.

Let's discuss the approximate case first. When MODE = 2, the closest approach flag is set to 2 to cause the propagator to stop at closest approach. Next, subroutine FOWARD is used to propagate the state to the closest approach of the target planet. The coordinate rotation is determined from the integration frame (Earth mean of 1950) to the true equator of the target planet. Subroutines M50MDT and NUTATE are used. The state at the closest approach is rotated to the desired system using MVTRN and orbital elements determined using subroutine ORBIT. These elements are the elements of the approach hyperbola.

The firing time on the approach hyperbola and altitude are varied in one of three ways through the KAPOPT flag as follows:

KAPOPT=1

The true anomaly of the firing is determined from

$$f_i = (f_o - \text{TRUE}) + \sum_{i=1}^{\text{KFIRE}} i * (2\ \text{TRUE}/\text{KFIRE})$$

where f_o is the true anomaly at closest approach

TRUE is the input true anomaly range
KFIRE is the input number of firings

The attitude is determined by

$$ra_i = (RAO - CONE) + \sum_{i=1}^{JRA} i * (2CONE/JRA)$$

$$DEC_i = (DECO - CONE) + \sum_{i=1}^{JDEC} i * (2CONE/JDEC)$$

Where

RAO	is the input starting right ascension
DECO	is the input starting declination
JRA	is the input number of right ascensions
JDEC	is the input number of declinations
CONE	is the input altitude cone angle

If the starting right ascension and declination are not input (equal to zero), the right ascension and declination of the velocity vector at closest approach will be used.

KAPOPT=2

The starting increment and number of altitudes must be input. Then the altitude is determined from

$$RA_i = RAO + \sum_{i=1}^{JRA} i * DELRA$$

$$DEC_i = DECO + \sum_{i=1}^{JDEC} i * DELDEC$$

The firing true anomaly is determined in a manner similar to the KAPOPT=1 option except that the range in true anomaly is from asymptote to asymptote instead of CONE.

KAPOPT=3

The first and last firing point on the approach hyperbola is input via the firing time since liftoff. The firings are made at KFIRE number of constant true anomaly steps between the bounds specified by the times. The altitude is varied as described for KAPOPT=2.

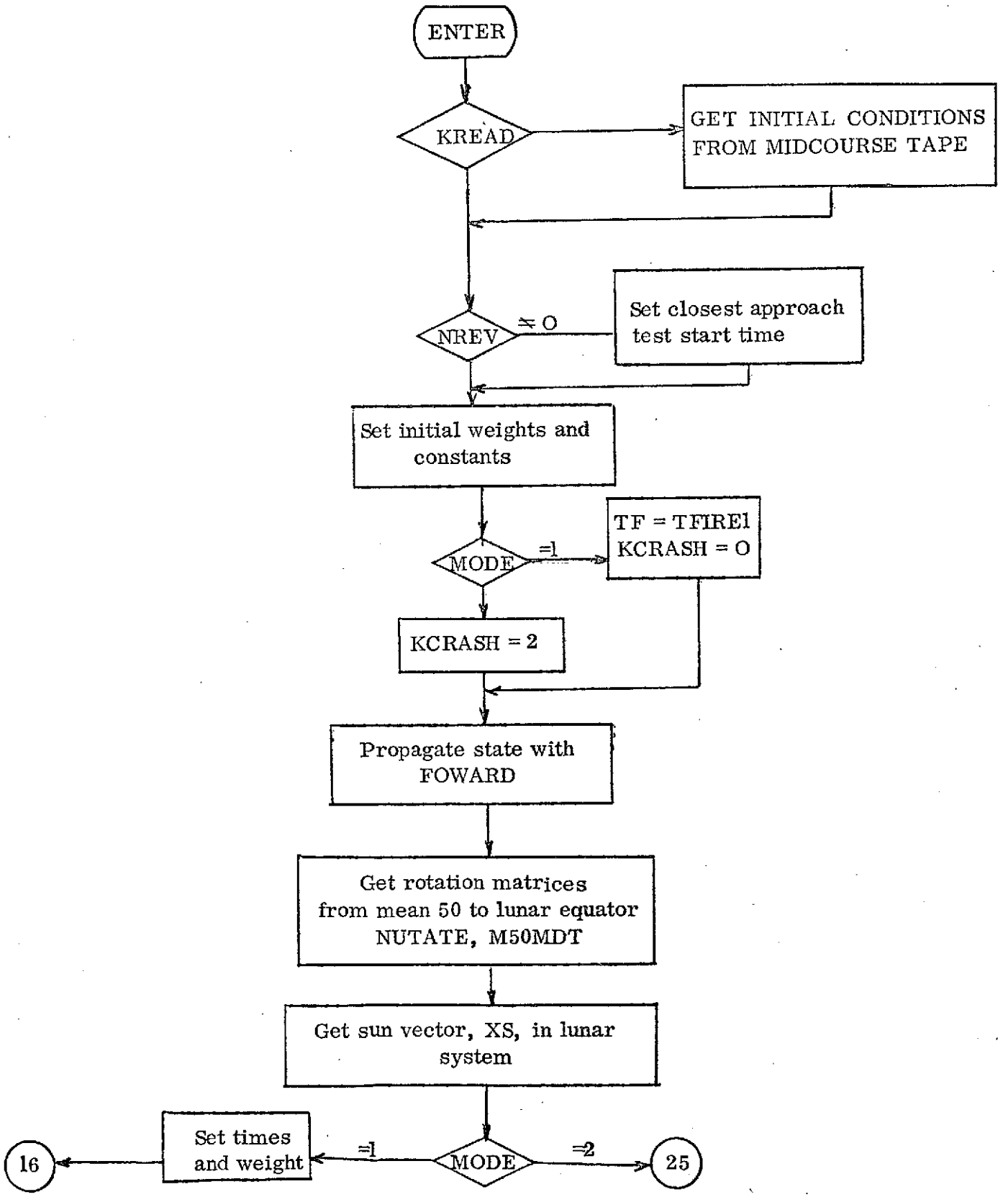
The IF statement just before statement 8 is used to transfer to the logic which sets up the necessary constants to increment the true anomaly and altitude.

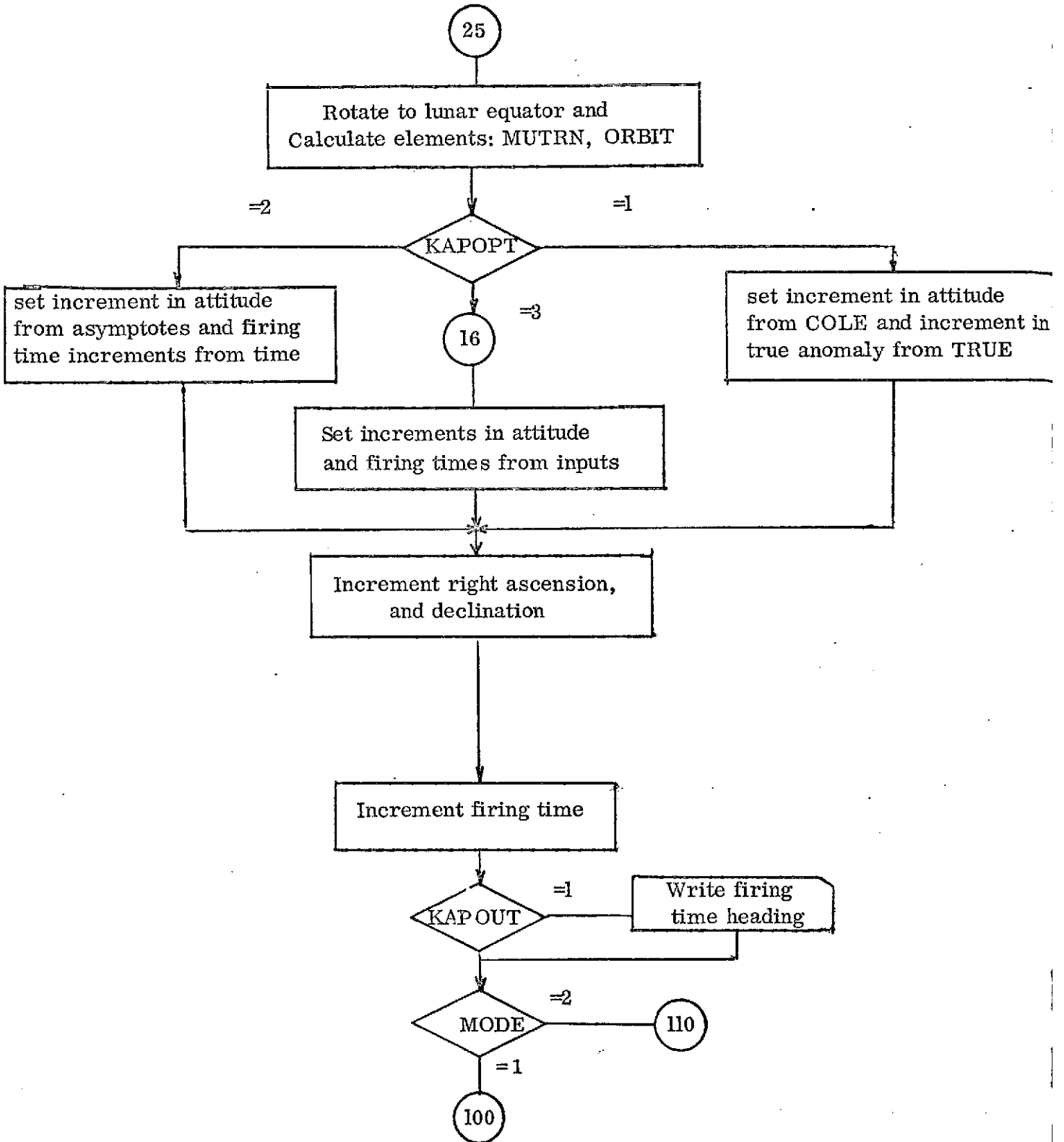
The logic flow for the precise case, MODE=1, differs slightly from the case discussed above. Instead of flying to closest approach to the target planet, the final time is set to the first firing time, TFIRE1. The KAPOPT flag must be set to 3 for this mode; thus, flow transfers to Statement 11.

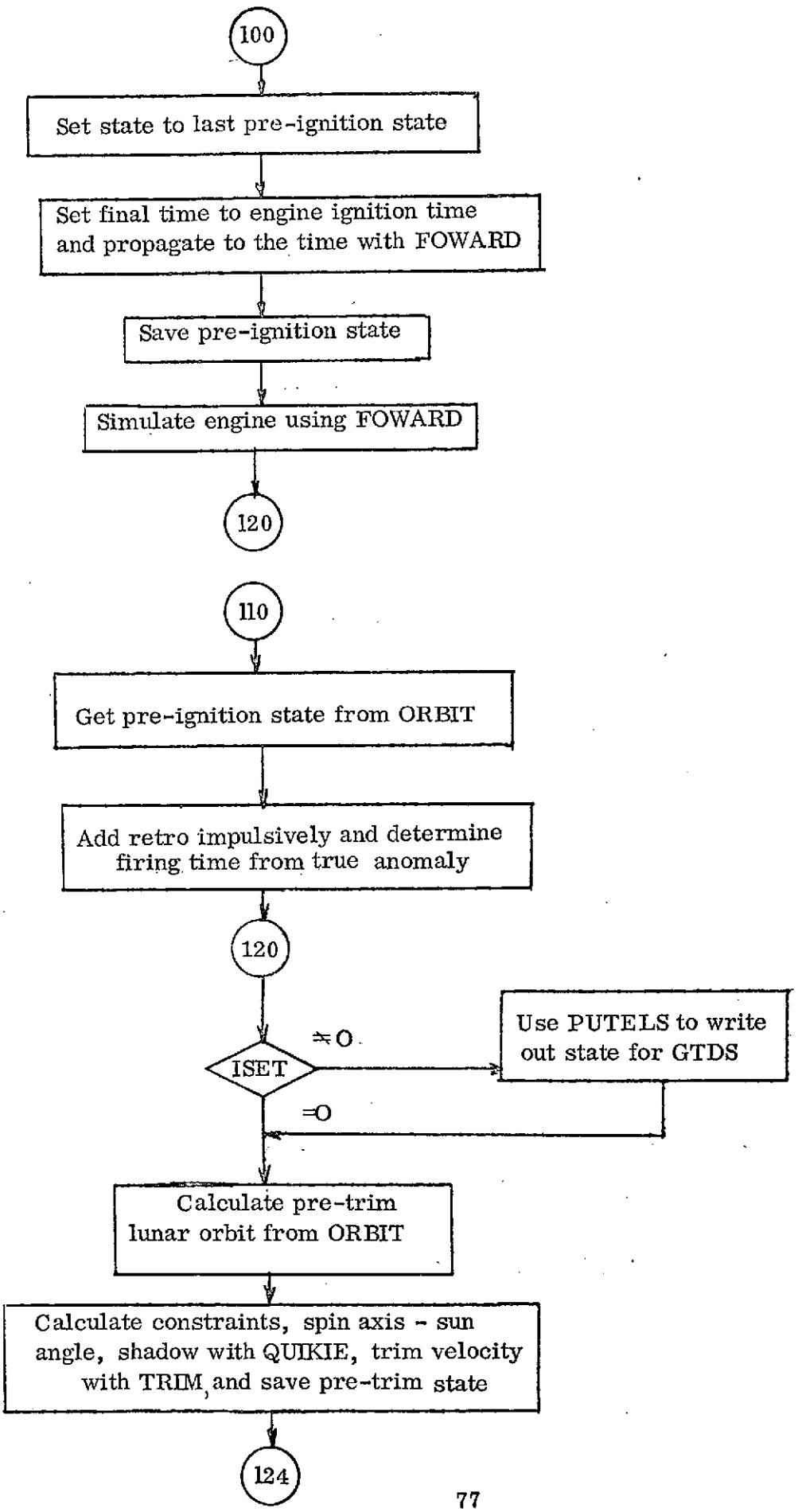
At this point, DO LOOPS are established to increment the right ascension, declination and firing time. The range of these loops extends to almost the end of the subroutine. The mission constraints are calculated for each attitude and firing time and saved in arrays for output or output inside the loops. When the MODE=2 is used, the position and velocity vectors are determined from subroutine ORBIT where the true anomaly is determined by the DO 10 loop. The impulsive velocity of the retro is added to the spacecraft's velocity in the direction specified by the attitude. This calculation is performed at statement 13. Next, the post retro orbit is determined and desired mission constraints calculated using subroutines QUIKIE and TRIM. The firing time display is output if the KAPOUT is set.

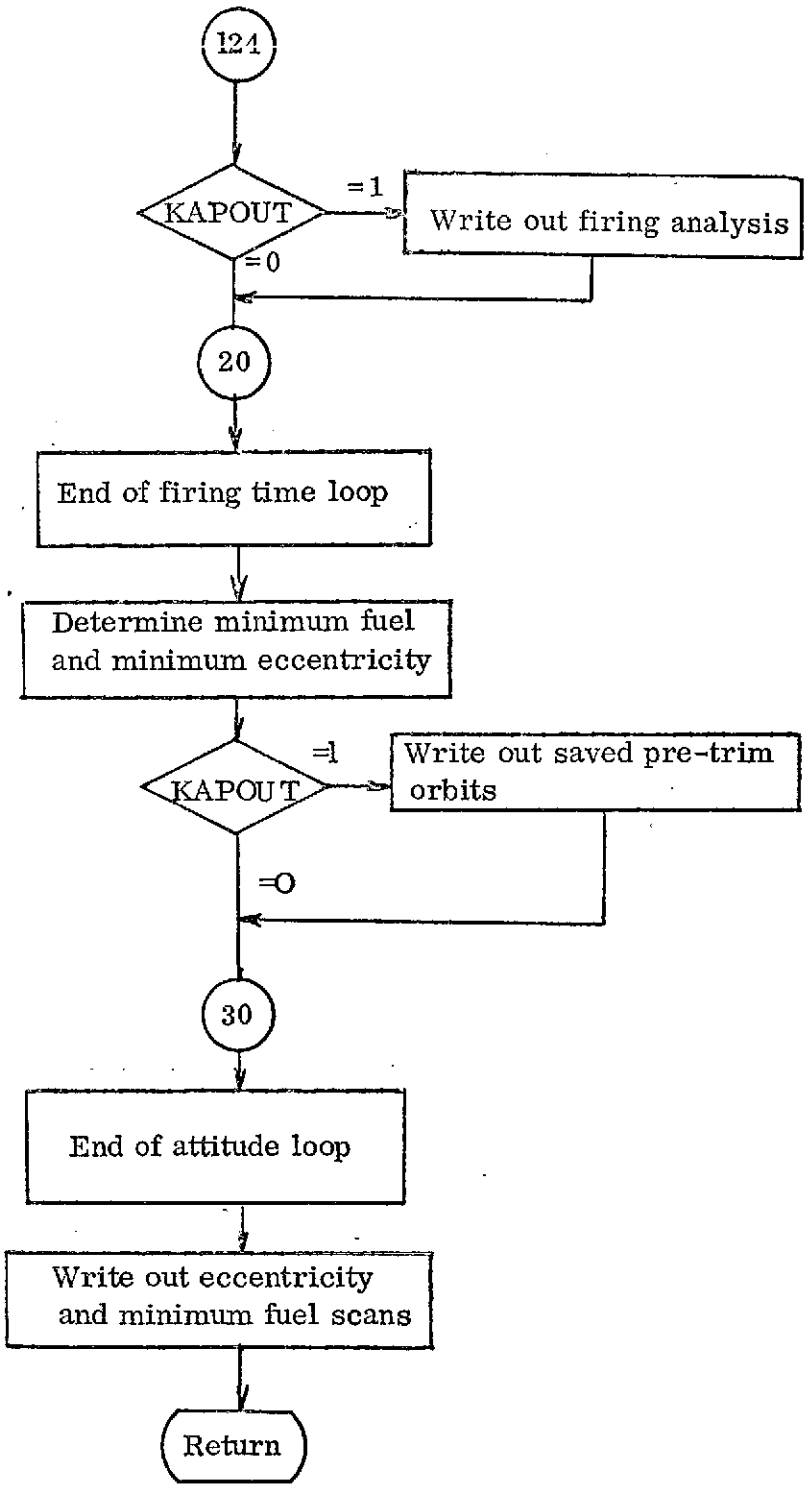
Subroutine FOWARD is used to simulate the retro motor when MODE=1. When this case is desired, subroutine FOWARD is first called to propagate the spacecraft to the ignition time, next, the compute interval is adjusted and the engine ignition and burnout times set. Finally, FOWARD is used to propagate through the burn. This logic is located between statements 100 and 110. Flow then transfers to the same place where the post retro orbit is determined when MODE=2.

The above logic is repeated until all attitudes and firing times are simulated. When the DO 30 loop is completed, the minimum eccentricity and trim fuel grids are output. This completes the subroutine.


ENTER
KREAD
GET INITIAL CONDITIONS FROM MIDCOURSE TAPE
NREV
≠ 0
Set closest approach test start time
Set initial weights and constants
MODE
=1
TF = TFIRE1
KCRASH = 0
KCRASH = 2
Propagate state with FOWARD
Get rotation matrices from mean 50 to lunar equator NUTATE, M50MDT
Get sun vector, XS, in lunar system
MODE
=1
=2
Set times and weight
16
25


25
Rotate to lunar equator and
Calculate elements: MUTRN, ORBIT
=2
=1
KAPOPT
set increment in attitude
from asymptotes and firing
time increments from time
=3
16
set increment in attitude
from COLE and increment in
true anomaly from TRUE
Set increments in attitude
and firing times from inputs
Increment right ascension,
and declination
Increment firing time
=1
KAP OUT
Write firing
time heading
=2
MODE
110
=1
100


100
Set state to last pre-ignition state
Set final time to engine ignition time and propagate to the time with FOWARD
Save pre-ignition state
Simulate engine using FOWARD
120
110
Get pre-ignition state from ORBIT
Add retro impulsively and determine firing time from true anomaly
120
ISET
≠0
=0
Use PUTELS to write out state for GTDS
Calculate pre-trim lunar orbit from ORBIT
Calculate constraints, spin axis - sun angle, shadow with QUIKIE, trim velocity with TRIM, and save pre-trim state
124


124
KAPOUT
=1
Write out firing analysis
=0
20
End of firing time loop
Determine minimum fuel and minimum eccentricity
KAPOUT
=1
Write out saved pre-trim orbits
=0
30
End of attitude loop
Write out eccentricity and minimum fuel scans
Return

SUBROUTINE ARMPIT

Calling Sequence: CALL ARMPIT

Purpose: This subroutine determines the post injection time requirements and outputs the post injection trim displays.

Common Block Required: CONST, INPUT, INTVAR, PIT, PLNET, STATE

Subroutines Required: M50LEQ, ORBIT, PUTELS, ROTATE, TRIM, TRMN, TWOPIT

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	CID	1	INPUT(446)	Inclination of desired final orbit w.r.t. target planet equator.
I	DJL	1	INPUT(37)	Modified Julian launch date
I	DJO	1	INPUT(46)	Modified Julian date of epoch
O	CIF	1	PIT (1)	Cosine of desired inclination
I	DV1	3	PIT (3)	First trim velocity maneuver
I	DV2	3	PIT (6)	Second trim velocity maneuver
I	IDSAT	1	INPUT(1089)	Satellite ID number
I	ISET	1	INPUT(1092)	Element set number when state is saved from GTDS
I	JT	1	INPUT(1031)	Target planet number
I	RD	1	INPUT(444)	Radius of final orbit
I	TRINC	1	INPUT(449)	Inclination of the transfer orbit
O	SIF	1	PIT(2)	Sine of desired inclination

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	UJT	1	STATE(32)	Current modified Julian date
I	X	6	STATE(1)	Position and velocity of spacecraft
I	XP	6, 12	PLNET(1)	Positions & velocities of the planets
I	XS	6	PIT(12)	Position and velocity of spacecraft just prior to first maneuver

Description:

This subroutine determines the post injection maneuver required to achieve a circular orbit with a desired radius and inclination. Two maneuvers are determined. First, a two impulse maneuver with a plane change and, second, a Hohmann transfer maneuver with no plane change. The maneuvers are determined in subroutines TWOPIT and TRIM, respectively.

The initial orbital elements are determined using subroutine ORBIT and supplied as inputs to subroutines TRIM and TWOPIT. The calculated maneuver is passed back to ARMPIT via DV1 and DV2 of PIT common. Also the state just prior to the first maneuver is passed back in XS of PIT common. The first maneuver is added to the state in XS to determine the transfer orbit. Subroutine ORBIT is used to determine the transfer orbit's elements, ELMT. A 180 degree transfer is specified. Thus, the true anomaly of the second maneuver on the transfer orbit is,

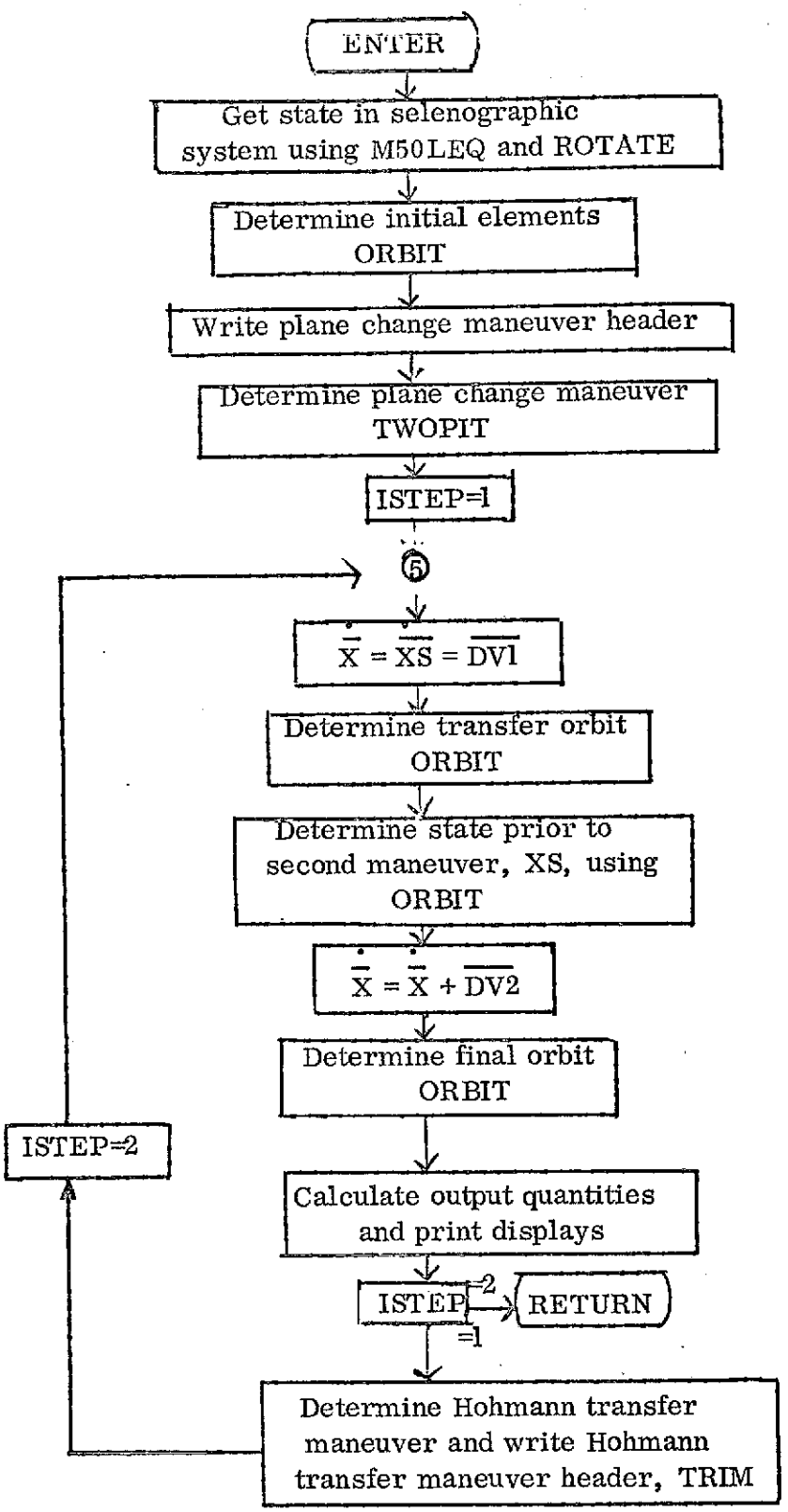
$$F = ELMT(3) + \pi$$

The cartesian state is determined at this true anomaly, the second maneuver applied, and the final orbit, ELMF, determined using ORBIT. The desired output quantities are determined and the appropriate displays presented.

The above logic is surrounded by a loop using the flag ISTEP. The purpose of the loop is to pass through the calculation of the orbits once for the plane change maneuver and a second time for the Hohmann transfer maneuver.

The state after the first trim can be written as a hold file for retrieval by the GTDS Program. The ISET flag is used to key this option. Subroutine PUTELS is used to write this file.

SUBROUTINE ARMPIT



FUNCTION ATMO

Calling Sequence: CALL ATMO(JC, N, H)

Purpose: To compute the atmospheric density for drag calculations.

Common Blocks Required: None

Subroutines Required: None

Inputs/Outputs

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	JC	1	Call List	Central body number
I	N	1	Call List	Atmospheric model number
I	H	1	Call List	Altitude

Discussion:

Atmospheric density is computed as an exponential function of altitude for Mercury, Venus, Earth, or Mars when altitude is below 50,000 km. The density data is tabulated as a function of altitude.

$$\rho = \exp\left[\rho_{k-1} + s\left(\ell n\, h - \ell n\, h_{k-1}\right)\right]$$

where

$$h_{k-1} \leq h < h_k$$

and

$$s = \left(\rho_k - \rho_{k-1}\right)/\left(\ell n\, h_k - \ell n\, h_{k-1}\right)$$

SUBROUTINE AVEQNS

Calling Sequence: CALL AVEQNS

Purpose: To average, by Gaussian quadrature integration, the planetary equations over one revolution of the satellite in its orbit.

Common Blocks Required: INPUT, CNTRL, STATE, AVG, INTVAR, PERT, CONST, GRAVTY

Subroutines Required: TRMN, ORBIT, ACCEL, ROTATE

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	METH	1	INPUT(1013)	Trajectory propagation method
I	NORD	1	INPUT(1068)	Number of ordinates to be used for each interval of Gaussian quadrature integration
I	INT	1	INPUT(1069)	Number of intervals into which the orbit is divided for Gaussian quadrature integration
I	DJ 0	1	INPUT(46)	Modified Julian date at liftoff epoch
I	EJ 0	1	STATE(26)	Modified ephemeris date at liftoff epoch
I	JC	1	CNTRL(7)	Central planet number
I	PI2	1	CONST(3)	Twice pi
I	GM	12	CONST(5-16)	Gravitational constants of the planets
I	X	1	INTVAR(1)	The current time (independent integration variable)
I	Y	6	INTVAR(2-7)	The current array of dependent integration variables
I	WEIGHT	78	AVG (1-78)	Weights for Gaussian quadrature formula
I	ABSCIS	78	AVG (9-156)	Abscissa for Gaussian quadrature formula
Ø	RATES	6	INTVAR(8-13)	The time derivatives of the current dependent integration variables

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	PI	1	CONST(2)	Pi
I	F	1	STATE(27)	Mean or true anomaly plus argument of perigee
O	EJT	1	STATE(28)	Current ephemeris date
O	UJT	1	STATE(32)	Current modified julian date

Theory:

Reference 1 gives a description of the theoretical basis for the numerical averaging. The planetary equations are written in Gauss's form where the perturbations enter directly as accelerations rather than as the partial derivatives of a potential function. The planetary equations exist in several forms in the program. They are symbolically represented by

$$\dot{E}_i = f_i (E_j, t) \qquad i, j = 1, 6, \tag{1}$$

where the E_i represent a set of 6 independent osculating orbital elements.

The averaging operation is accomplished numerically rather than analytically and is represented by

$$\bar{\dot{E}}_i = \frac{1}{\tau} \int_{t-\tau/2}^{t+\tau/2} f_i \, dt \tag{2}$$

where τ is the orbital period. The mean values of the orbital elements are defined by

$$\bar{E} = \frac{1}{\tau} \int_{t-\tau/2}^{t+\tau/2} E(t) \, dt . \tag{3}$$

The important thing to note is that the numerical averaging method essentially trades the set of 6 differential equations for the osculating orbital elements for a set of 6 differential equations for the mean orbital elements.

The mechanical averaging is done with respect to true anomaly rather than with the time directly. The Keplerian relation

$$\dot{f} = \frac{\sqrt{\mu p}}{r^2}$$

is used to transform the integrals in (2) from time to true anomaly giving

$$\bar{\dot{E}}_i = \frac{n p^2}{2\pi \sqrt{\mu p}} \int_{f(t-\tau/2)}^{f(t+\tau/2)} \frac{f_j(E_j, f)}{(1+e\cos f)^2} \, df \tag{4}$$

where f is the true anomaly and the other symbols represent Keplerian elements and constants as given in Reference 1.

Description:

This subroutine is a modified version of a variable-order, variable interval Gaussian quadrature integrator. The weights and abscissae for the quadrature formula are stored sequentially in AVG common, in the arrays called WEIGHT and ABSCIS. The values stored correspond to quadrature formulae of order 2 - 16 as given in reference 2. The single weight required for a two-point quadrature is stored in WEIGHT(1), the two weights required for a three point quadrature in WEIGHT(2) and WEIGHT(3), and the two weights required for a four point quadrature in WEIGHT(4) and WEIGHT(5). The pattern continues up through and including the eight weights required for a 16 point quadrature stored in WEIGHT (64-71). The corresponding abscissae are stored in the corresponding locations of the ABSCIS array.

The order (NØRD) of the quadrature is specified by the user as well as the number of subintervals (INT) to be used in the integration. The full range of the independent variable (true anomaly) is divided into INT intervals and the current (mean) value of the true anomaly is calculated from the mean mean anomaly and stored. The coding near statements 50 and 51 establishes the proper limits of integration and performs the transformation from mean anomaly to true anomaly for either the Method 5 or Method 8 variables. Throughout the subroutine, appropriate tests are made to distinguish between the two methods.

Certain quantities that are calculated only once are established before the quadrature summation begins at the DO 507 loop. This outer loop is for the summation of the sub-interval quadrature formula. The actual Gaussian quadrature algorithm starts at statement

150 where the time and the sine and cosine of the true anomaly are calculated for each required evaluation of the integrands for the 6 differential equations to be numerically averaged. The values of the integrands are calculated as shown in reference 1, multiplied by the appropriate weight, and the results are stored in the array SUM.

The results of each subinterval quadrature are accumulated in the array called ANS until all subintervals are completed at statement 507. The 6 stored integrals are then multiplied by the constants outside the integral signs of the averaged equations and the results are loaded into the RATES array of INTVAR common for use in the numerical solution of the averaged differential equations of motion.

References

1. Uphoff, C., Numerical Averaging in Orbit Prediction, Presented at AIAA/AAS Astrodynamics Conference, Palo Alto, Calif., AIAA preprint No. 72-934, September 1972.

2. Abramowitz, Milton and Stegun, Irene A., Handbook of Mathematical Functions, published by The National Bureau of Standards, Applied Mathematics Series No. 55, May 1968.

SUBROUTINE AVERGE

Calling Sequence: CALL AVERGE (KSET)

Purpose: AVERGE calculates the average perturbing acceleration due to the Sun and the Moon's indirect term over one step of the multiconic trajectory scheme.

Common Block Required: DUM, STATE, PLNET, CONST

Subroutines Required: DVMAG

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	A1S	3	DUM(1)	Earth-Moon acceleration at beginning of step
I/O	A2S	3	DUM(4)	Sun acceleration at beginning of step
I	DT	1	DUM(10)	Step time interval
I	GM	12	CONST(5)	Gravitational constants of planets
I	KSET	1	CALLING OPERAND	Flag for adding or subtracting accelerations
I/O	X	6	STATE(1)	Position and velocity of spacecraft
I	XP	72	PLNET(1)	Positions and velocities of planets wrt central planet

Theory:

AVERGE corrects the position and velocity of the spacecraft during each step in order to account for perturbing accelerations that are not included in the conic assumptions.

These perturbing accelerations are caused by the Sun and also by the effect of the Moon on the planet. Let $\bar{A1}$ be the acceleration vector due to the Earth-Moon term at the present time, and $\bar{A1S}$ be the same vector from the previous time (T-DT). Similarly let $\bar{A2}$ and $\bar{A2S}$ be the Sun term accelerations corresponding to T and T-DT.

Then $$\bar{A1} = G M_m * \bar{R}_{em} / |\bar{R}_{em}|^3$$

And $$\bar{A2} = G M_s * (\bar{R}_{es} / |\bar{R}_{es}|^3 + \bar{R}_{sv} / |\bar{R}_{sv}|^3)$$

Where subscripts m, e, s, sv refer to Moon, Earth, Sun and space vehicle respectively and $\bar{R}_{es}$ is the vector from the Earth to the Sun, etc.

The average accelerations on the spacecraft with respect to the Earth over the interval DT are:

$$\bar{A1}_a = -(\bar{A1} + \bar{A1S})/2.$$

$$\bar{A2}_a = -(\bar{A2} + \bar{A2S})/2.$$

If $\bar{X}$ and $\bar{DX}$ represent the position and velocity of the spacecraft respectively, then the corrected position and velocity are

$$\bar{X} = \bar{X} + 1/2 * (\bar{A1} + \bar{A2}) * DT^2$$

$$\bar{DX} = \bar{DX} + (\bar{A1} + \bar{A2}) * DT$$

If KSET .LT. 0,then the orbit is to be retraced, in which case the following equations are substituted for those above:

$$\bar{X} = \bar{X} - 1/2 * (\bar{A1} + \bar{A2}) * DT^2$$

$$\bar{DX} = \bar{DX} - (A1 + A2) * DT$$

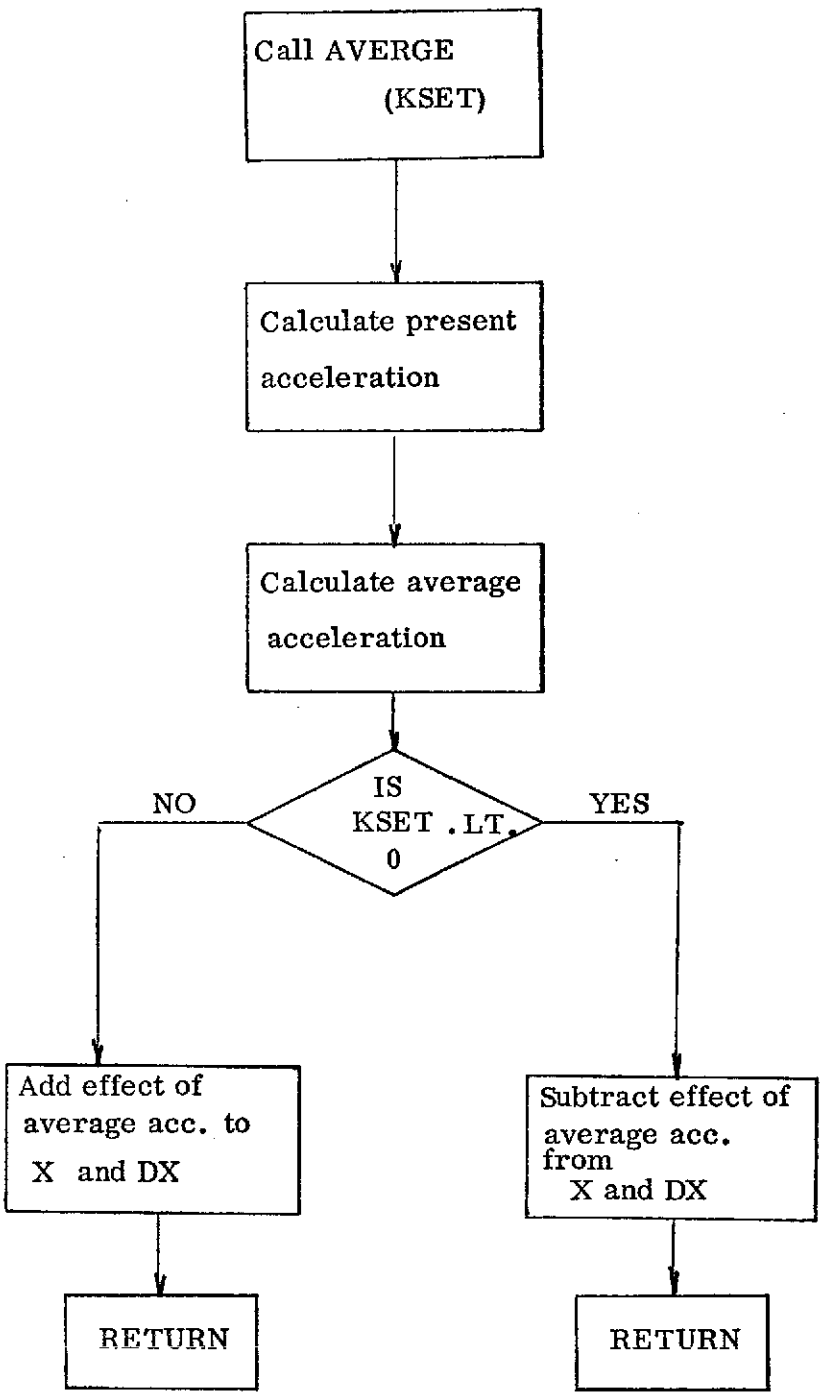
Description:

AVERGE is used in connection with the multiconic scheme. It calculates the acceleration of the Earth due to the Moon at the present time and adds it to the acceleration at the last time and divides by two for the average acceleration over the step. The present acceleration is then saved for future use.

The same procedure is followed for the calculation of the acceleration due to the Sun on the Earth and on the spacecraft.

Once these two average acceleration vectors have been calculated, the position and velocity of the spacecraft are updated to reflect the accelerations. The changes in position and velocity are either added or subtracted depending on the flag KSET. If KSET .GE. 0, then forward propagation is desired and the terms are added. Otherwise, the terms are subtracted and the orbit path is retraced.

SUBROUTINE AVERGE



SUBROUTINE AVSTRT

Calling Sequence: CALL AVSTRT

Purpose: To average the input osculating elements over one revolution of the satellite in its orbit. This procedure yields the mean orbital elements at a time halfway through the first revolution and these elements are used to start the numerical averaging techniques for long-term orbit prediction.

Common Blocks Required: CONST, INTVRX, STATE, INPUT, INTVAR, CNTRL

Subroutines Required: INTEG, ORBIT, EQNS (indirectly)

Reference: Uphoff, C., Numerical Averaging in Orbit Prediction, AAS/AIAA Preprint No. 72-934, Palo Alto, California, September 1972

Input / Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSIONS	COMMON BLOCK	DEFINITION
I/O	XX	6	STATE(1)	Position and velocity (osculating on entry, mean on exit)
I/O	TT	1	STATE(10)	TIME (advanced by one-half revolution on exit)

Theory:

In the reference, it was suggested that the numerical averaging method should be started with mean orbital elements and an algorithm for performing the start-up was presented. The averaging technique is properly started from osculating elements by performing a one-revolution integration of the actual equations of motion and, at the same time, forming the integrals

$$\bar{E}_i = \frac{1}{t} \int_o^t E_i \, dt \, .$$

At each step of the integration, the period $\bar{\tau}$ given by the running mean semi-major axis is calculated. The averaging start-up is terminated when the time t is equal to this running mean value of the period. The procedure is implemented by a simple linear search when t is found to be greater than $\bar{\tau}$. The mean values of the elements are then calculated from

$$\bar{E}_i = \frac{1}{\tau} \int_0^{\tau} E_i \, dt$$

and are associated with the mean value of the time $\bar{\tau}/2$.

Description:

The averaging startup is invoked by setting KAVST (location 1098) to 1. This single value of the flag will cause a call to AVSTRT at the beginning of subroutine FOWARD, the position and velocity will be changed to values corresponding to the mean elements at time $\bar{\tau}/2$ and the time is advanced accordingly.

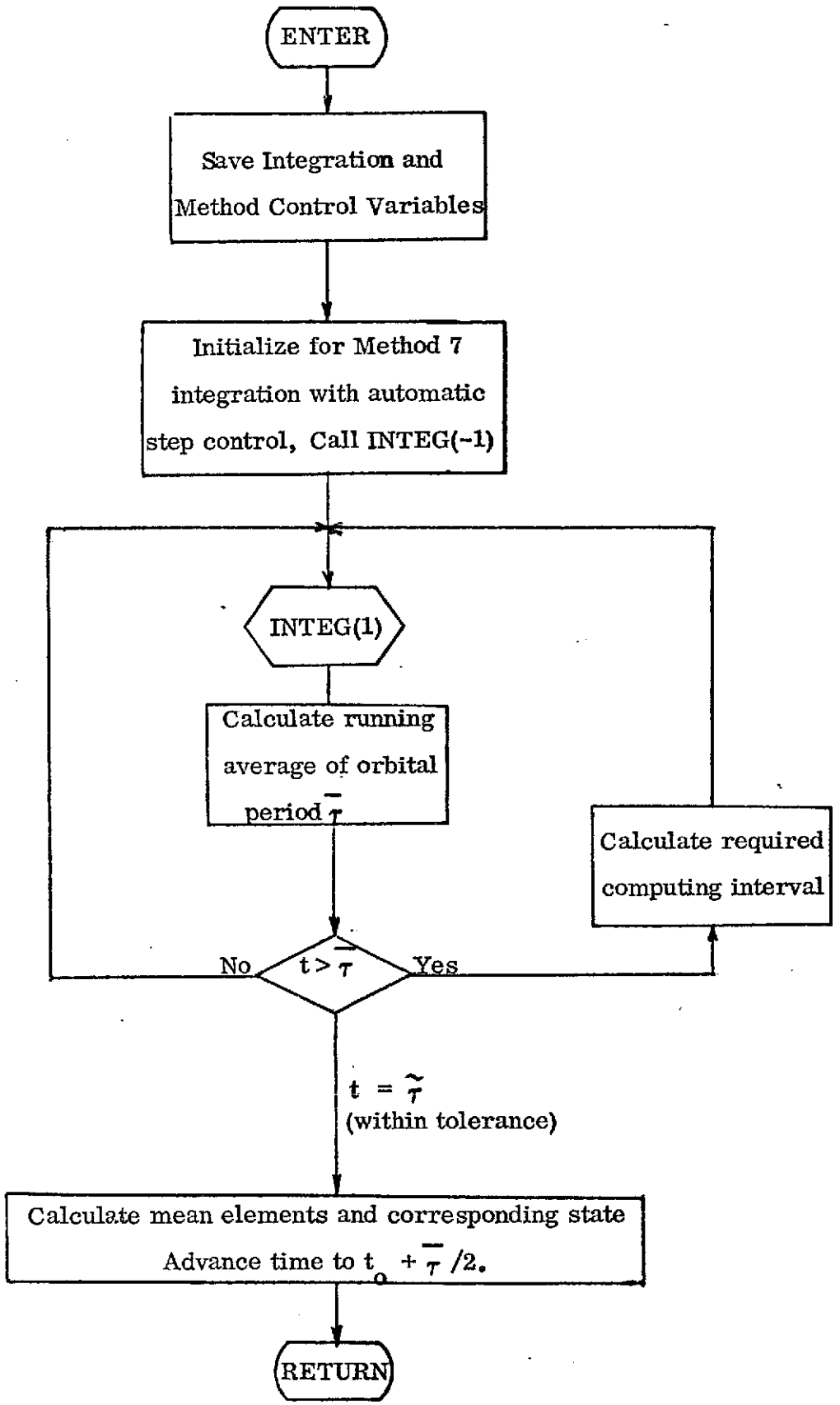
The startup will be performed in FOWARD only if KAVST is equal to 1. The associated logic in subroutine EQNS, however, will be in effect for any non-zero value of KAVST so that the startup may be called from other subroutines before entrance to FOWARD.

To use the subroutine, put the osculating state (position and velocity) into STATE common and the corresponding time into STATE(10). Set KAVST non-zero and call AVSTRT. The averaging integration will be performed using METH=7 with an internally set automatic computing interval. On return from AVSTRT, the position and velocity are changed to represent averaged values and the time is advanced by 1/2 revolution. All other quantities except ELM (STATE(14-19)) are restored to the values they had before the call to AVSTRT.

The integration of the six additional equations is accomplished simultaneously with the one revolution propagation of the state. Subroutine INTEG is used to integrate forward in time using Method 7 with the automatic computing interval control logic. The

additional integrals are accumulated in INTVRX common and, at the end of each step, the current time is tested against the running mean orbital period. When $t \geq \tau$ the computing interval is changed to take the state from its value at the previous step to the desired time when $t = \tau$, as obtained by linear interpolation on the current and previous values of t and τ. The tolerance on this simple hunting procedure is set at 10 seconds.

AVSTRT


ENTER
Save Integration and Method Control Variables
Initialize for Method 7 integration with automatic step control, Call INTEG(-1)
INTEG(1)
Calculate running average of orbital period $\bar{\tau}$
Calculate required computing interval
No
$t > \bar{\tau}$
Yes
$t = \tilde{\tau}$
(within tolerance)
Calculate mean elements and corresponding state
Advance time to $t_o + \bar{\tau}/2$.
RETURN

SUBROUTINE BCONIC

Calling Sequence: CALL BCONIC (UIN, X1, X2, TIN, ØUT, VØ, DVØ)

Purpose: BCONIC uses Lambert's Theorem to solve for the Keplerian conic trajectory which connects two given radius vectors in a specified transfer time.

Common Blocks Required: None

Subroutines Called: DOT, CROSS

Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	UIN	1	Call List	GM (km^3/sec^2) of the central body
I	X1	6	Call List	Position (km) and velocity (km/sec) vectors at the first terminal
I	X2	6	Call List	Position (km) and velocity (km/sec) vectors at the second terminal
I	TIN	1	Call List	Transfer time (sec)
I	ØUT	1	Call List	Iteration's tolerance on transfer time error (fraction of transfer time)
Ø	VØ	6	Call List	Terminal velocity vectors (km/sec)
Ø	DVØ	6	Call List	Terminal excess velocity vectors (km/sec)

Theory:

BCONIC implements the Schmidt-Bjorkman method to solve Lambert's problem, which is to find the conic section connecting two terminal radii in a specified transfer time. The implemented method bears similarities to several of the methods described in Ref. 1.

The conic parameters are not explicit functions of transfer time, although the converse is true, setting up the requirement for iteration to solve the problem. The conic sections connecting the two terminal radii can be parameterized by a single independent variable (in which flight time is a monotonic function). The independent variable of the S/B method is flight path angle, γ, at the first terminal, R_1.

The necessary equations for iterating to a solution are derived as follows. The distance from the force center is given by

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

where p is semi-latus rectum, e is eccentricity, and f is true anomaly — none of which is known a priori. We presume to know the two terminal radii, r_1 and r_2, and the transfer angle, ψ. Since r_1 and r_2 are points on the same conic section,

$$r_1 = \frac{p}{1 + e \cos f_1} \tag{2}$$

$$r_2 = \frac{p}{1 + e \cos (f_1 + \psi)} = \frac{p}{1 + e \cos f_1 \cos \psi - e \sin f_1 \sin \psi}. \tag{3}$$

The relationship of f_1 to γ at r_1 is shown by

$$e \sin f_1 = \frac{p}{r_1} \tan \gamma \tag{4}$$

and we can eliminate e and f_1 from equations (2) and (3) and solve for p as a function of γ and the geometry.

$$p = \frac{r_1 r_2 (1 - \cos\psi)}{(r_1 - r_2 \cos\psi) + r_2 \sin\psi \tan\gamma} = \frac{d_3}{d_1 + d_2 \tan\gamma} \tag{5}$$

The potential problems in computation of Eq. (5) are eliminated by properly limiting the range of γ. This range will be discussed later. The velocity at r_1 can be calculated by

$$v = \frac{\sqrt{\mu p}}{r_1 \cos\gamma} \tag{6}$$

where μ is the force center's gravitational constant. The semi-major axis of the transfer is given by

$$a = \mu \Big/ \left(\frac{2\mu}{r_1} - v^2\right) \tag{7}$$

One might worry about parabolic transfers when studying Eq. (7); however, the special treatment of parabolic transfers will be described later.

At this point, we launch into a discussion of the calculation of transfer time. BCONIC's formulation of transfer time is written in terms of incremental eccentric anomaly, φ, on the conic section connecting r_1 and r_2.

$$nt = \sigma\left[(\varphi - S\varphi) + \frac{r_1}{a} S\varphi + \frac{r_1 v \sin\gamma}{\sqrt{\mu\sigma a}} (1 - C\varphi)\right] \tag{8}$$

In this equation, n is the mean motion $\left(\sqrt{\mu/(\sigma a)^3}\right)$ while σ, $S\varphi$, and $C\varphi$ are defined by the following table.

	Elliptical	Hyperbolic
$S\varphi$	$\sin\varphi$	$\sinh\varphi$
$C\varphi$	$\cos\varphi$	$\cosh\varphi$
σ	+1	-1

It has been shown in Ref. 2 that:

$$\sin\psi = \frac{\sigma\sqrt{\mu p}}{n\, r_1\, r_2}\left[\beta(1-C\varphi) - \alpha S\varphi\right] \tag{9}$$

$$\cos\psi = \frac{r_1}{r_2} + \frac{\sigma a^2}{r_1\, r_2}\left[(\alpha^2-\sigma\beta)(1-C\varphi) + \alpha\beta S\varphi\right] \tag{10}$$

where $\beta = \frac{r_1}{a}$ and $\alpha = \frac{r_1\, v \sin\gamma}{\sqrt{\mu\sigma a}}$. It is easy to solve these equations for $S\varphi$ and $(1-C\varphi)$, rendering Eqs. (11) and (12) in which $d_1 = r_1 - r_2\cos\psi$ and $d_2 = r_2 \sin\psi$,

$$S\varphi = -\frac{1}{r_1}\left[\sqrt{\frac{\sigma a}{p}}(\alpha^2-\sigma\beta)d_2 + \alpha d_1\right] \tag{11}$$

$$1-C\varphi = \frac{1}{a}\left[\sqrt{\frac{\sigma a}{p}}\,\alpha d_2 + d_1\right] \tag{12}$$

If the transfer orbit is elliptical, $\varphi = 2\tan^{-1}\left(\frac{S\varphi}{1-C\varphi}\right)$, while if it is hyperbolic, $\varphi = \ell n\,(S\varphi + C\varphi)$. It should be noted that φ is the only transcendental function to be evaluated in computing the transfer time.

Iteration:

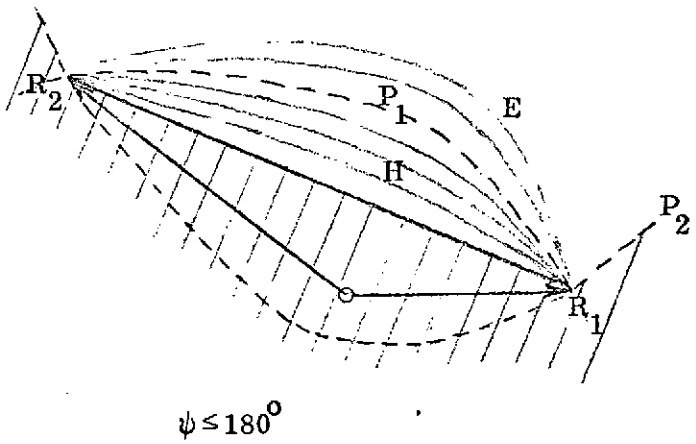
Equations (5) through (12) provide the recipe for computing transfer time as a function of flight path angle at the first terminal. The discussion which follows is concerned with the iteration to compute the unique flight path angle which renders the "desired" transfer time between the two terminals. BCONIC uses a simple Newton-Raphson iteration which invariably provides rapid convergence.

$$\gamma_{n+1} = \gamma_n + (t_d - t_n)\left(\frac{\gamma_n - \gamma_{n-1}}{t_n - t_{n-1}}\right) \tag{13}$$

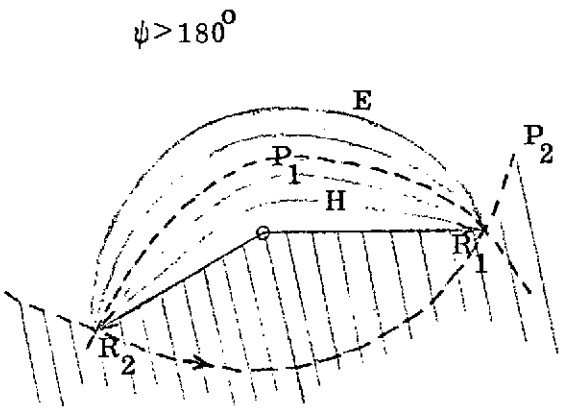
Flight time is a monotonic-increasing function over the allowable range of γ.

Limits:

The allowable transfer orbit regions are sketched in Fig. 1 for $\psi < 180^{\circ}$ and for $\psi > 180^{\circ}$.



$\psi \leq 180^{\circ}$



No orbits traveling <u>from</u> R_1 <u>to</u> R_2 are permitted in the shaded regions of the figure. The upper limit on flight path angle is, in either case, the path angle of the escape parabola P_2 at R_1. This flight path angle is derivable in terms of the geometry if we solve Eqs. (2), (3), and (4) for γ with the conditions that $e = 1$ and $f_1 = 2\gamma$.

$$d_1 \cos 2\gamma + d_2 \sin 2\gamma = r_2 - r_1 \tag{14}$$

$$\gamma_p = \tfrac{1}{2}\left[\cos^{-1}\left(\frac{r_2 - r_1}{\sqrt{d_1^2 + d_2^2}}\right) + \tan^{-1}\left(\frac{d_2}{d_1}\right)\right] \tag{15}$$

If the second term in Eq. (15) is evaluated with regard for quadrant in the range of 0° to 360°, the arccosine ambiguity still leads to two solutions for γ_p. The larger solution is the flight path angle of the escape parabola. The smaller solution is the flight path angle of the prograde transfer parabola connecting R_1 and R_2. The transfer time for this prograde parabola is computed according to

$$p_p = 2 r_1 \cos^2 \gamma_{p1} \tag{16}$$

$$t_p = \tfrac{1}{2}\sqrt{\frac{p_p^3}{\mu}}\left\{\tan\left(\gamma_{p1}+\frac{\psi}{2}\right)+\frac{\tan^3\left(\gamma_{p1}+\frac{\psi}{2}\right)}{3}-\tan\gamma_{p1}-\frac{\tan^3\gamma_{p1}}{3}\right\} \tag{17}$$

If the desired transfer time is equal to the parabolic transfer time (within tolerance), no iteration is required and the potential parabolic singularity in Eq. (7) is avoided. Otherwise, the desired transfer time will be less than t_p (hyperbolic solution) or greater than t_p (elliptical solution). Thus, the flight path angle limits for the ensuing iteration may be set as follows:

Elliptical

Lower: γ_{p1} Upper: γ_{p2}

Hyperbolic

Lower: $\begin{cases} -\tan^{-1}\left(\frac{d_1}{d_2}\right) & \text{if } \psi \le 180^\circ \\ -\pi/2 & \text{if } \psi > 180^\circ \end{cases}$ Upper: γ_{p1}

The lower limits on the hyperbolic solution correspond to the straight-line path if $\psi \le 180^\circ$ and to the path along $-R_1$ and then along R_2 if $\psi > 180^\circ$. Equation (5) has no singularities within the above limits.

Starting:

If the solution is expected to be hyperbolic, the initial γ is taken to be a little less than γ_{p1}, the prograde parabolic value. The amount less is chosen simply as 1% of the allowed range of γ. The starting value for γ when the solution is expected to be elliptical is the first-terminal flight path angle of the minimum-period ellipse joining R_1 and R_2. The minimum-period ellipse is characterized by the following semi-major axis and semi-latus rectum (Ref. 3).

$$a_{min} = \frac{1}{4}\left(r_1 + r_2 + \sqrt{d_1^2 + d_2^2}\right) \tag{18}$$

$$p_{min} = 2(2a_{min} - r_1)(2a_{min} - r_2)\Big/\sqrt{d_1^2 + d_2^2} \tag{19}$$

The flight path angle is then found by substituting p_{min} into Eq. (5).

$$\gamma = \tan^{-1}\left(\frac{d_3 - p_{min} d_1}{p_{min} d_2}\right) \tag{20}$$

I/O Computations:

BCONIC is entered with a 6-vector of Cartesian position and velocity components for each terminal. The magnitudes of the position vectors, R_1 and R_2, are r_1 and r_2, respectively. The velocity vector, V_1, at the first terminal is used in determining whether ψ is less or greater than 180°. The cosine of ψ is given by

$$\cos\psi = \frac{R_1 \cdot R_2}{r_1 r_2} \tag{21}$$

but the sine of ψ, computed as $\sqrt{1-\cos^2\psi}$, has a sign ambiguity. This ambiguity is resolved by comparing the cross-product, $R_1 \times V_1$, with the cross-product $R_1 \times R_2$. If these cross-products have a negative dot product, $\sin\psi$ must be negative to exclude retrograde orbits with respect to V_1 at R_1.

$$(R_1 \times V_1) \cdot (R_1 \times R_2) > 0 \quad : \quad \sin\psi = \sqrt{1-\cos^2\psi}$$

$$(R_1 \times V_1) \cdot (R_1 \times R_2) \leq 0 \quad : \quad \sin\psi = -\sqrt{1-\cos^2\psi}$$

The transfer angle is then defined by

$$\psi = \tan^{-1}\left(\frac{\sin\psi}{\cos\psi}\right), \quad 0 < \psi \leq 360^\circ \tag{22}$$

At the conclusion of the iteration, BCONIC computes the vector velocity, V*, of the transfer orbit at each terminal. If the transfer angle is not 180° or 360°, R_1 and R_2 define a unique plane for the transfer orbit. Then the velocities are

$$V_1^* = c_1 R_1 + c_2 (R_1 \times R_2) \times R_1 \tag{23}$$

and

$$V_2^* = c_3 R_1 + c_4 V_1^* \tag{24}$$

where

$$c_1 = v \sin\gamma \Big/ r_1 \tag{25}$$

$$c_2 = v \cos\gamma \Big/ (d_2 r_1^2) \tag{26}$$

$$c_3 = -\sqrt{\mu\sigma a}\ S\varphi \Big/ (r_1 r_2) \tag{27}$$

and

$$c_4 = (r_1 C\varphi + \sigma\alpha a S\varphi) \Big/ r_2 \tag{28}$$

If $|\sin\psi| < 10^{-6}$, the transfer orbit plane is ill-defined by R_1 and R_2, so the plane of R_1 and V_1 is chosen. In this case,

$$V_1^* = c_1 R_1 + c_2 (R_1 \times V_1) \times R_1 \tag{29}$$

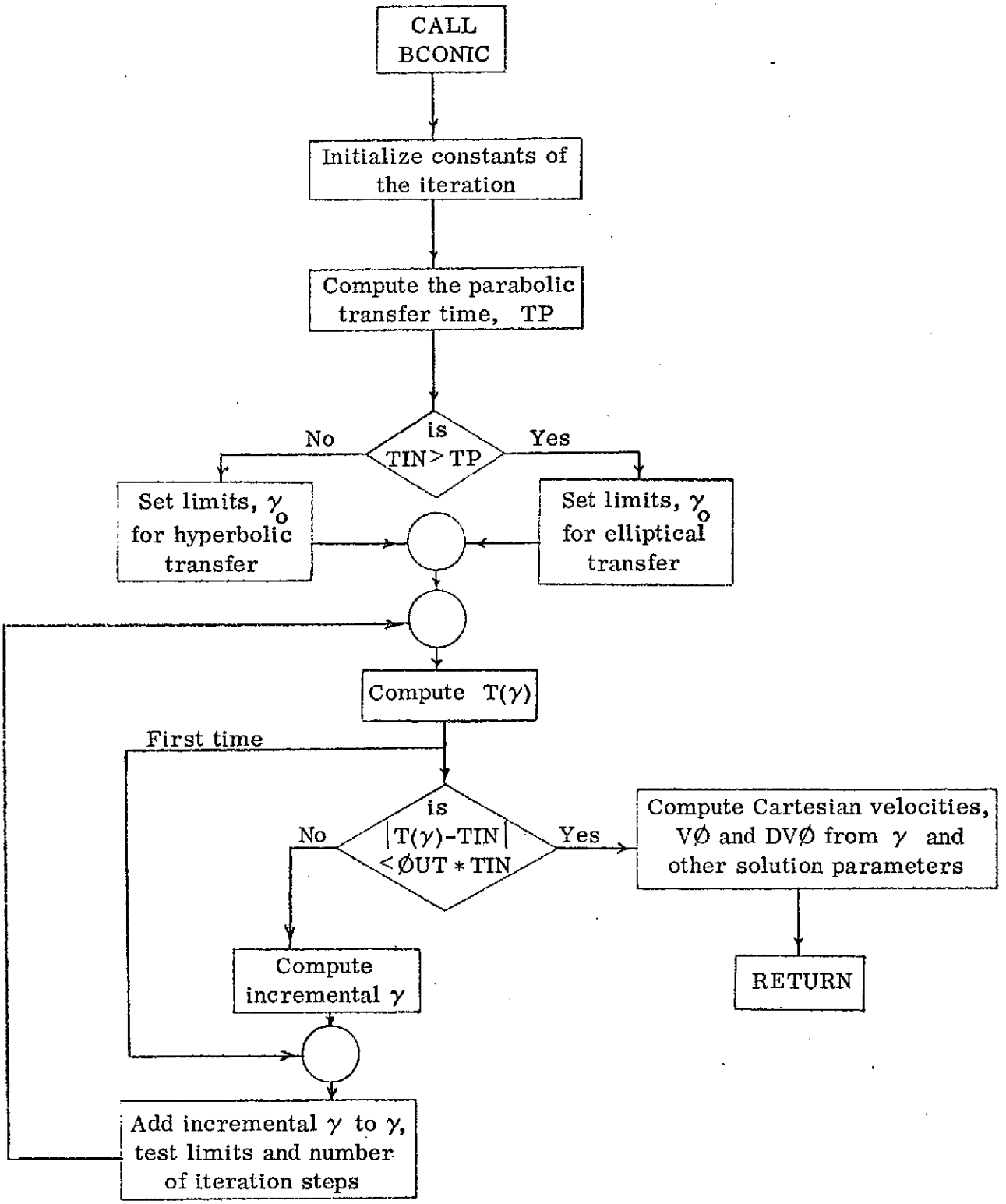
where

$$c_2 = v \cos\gamma \Big/ |(R_1 \times V_1) \times R_1| \tag{30}$$

The other calculations are unchanged. BCONIC also computes the terminal velocity differences for output.

$$\Delta V_1 = V_1^* - V_1 \tag{31}$$

$$\Delta V_2 = V_2^* - V_2 \tag{32}$$


CALL BCONIC
Initialize constants of the iteration
Compute the parabolic transfer time, TP
is TIN > TP
No
Yes
Set limits, γ_o for hyperbolic transfer
Set limits, γ_o for elliptical transfer
Compute T(γ)
First time
is |T(γ)-TIN| < ØUT * TIN
No
Yes
Compute Cartesian velocities, VØ and DVØ from γ and other solution parameters
RETURN
Compute incremental γ
Add incremental γ to γ, test limits and number of iteration steps

REFERENCES

1. Battin, R. H.; Astronautical Guidance, McGraw-Hill, 1964.

2. "Programer's Manual for Quick-Look Mission Analysis Program," Philco-Ford WDL TR-2217, January 1964.

3. "Design Parameters for Ballistic Interplanetary Trajectories — Part I: One-Way Transfers to Mars and Venus," JPL TR-32-77, January 1963.

SUBROUTINE BELL

Calling Sequence: CALL BELL

Purpose: BELL computes statistical estimates of end constraint errors and second-midcourse requirements based on estimated first-midcourse execution error statistics.

Common Blocks Required: CNTRL, INPUT, MCCOM, STATE

Subroutines Required: CROSS, DOT, DVMAG, FOWARD, MCBURN, MVTRN, RETDV, SENSO, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	JC	1	CNTRL(7)	Central body number of state
I	TFINAL	1	INPUT(4)	Trajectory stop-time (sec). Input value matters not.
I	WTO	1	INPUT(38)	Initial spacecraft weight (kg)
I	SIGAT	1	INPUT(435)	Attitude execution error statistic (rad)
I	SIGDV	1	INPUT(436)	Proportional velocity execution error statistic (km/sec)
I	PRX	1	MCCOM(3)	Scale factor for printed statistics
I	XMC	6	MCCOM(6)	Pre-midcourse state (km, km/sec)
I	DV	3	MCCOM(12)	Midcourse velocity impulse (km/sec)
I	DVMG	1	MCCOM(15)	Magnitude of DV (km/sec)
I	TMCS	1	MCCOM(18)	Midcourse time (seconds from anchor epoch)
O	DVB4	1	MCCOM(24)	Velocity impulse magnitude expended prior to current maneuver (km/sec)
O	EXFUEL	1	MCCOM(26)	Expected fuel required at next maneuver to correct errors of this one (kg)
O	SIGOUT	6	MCCOM(40)	Expected end constraint error statistics (^{t}f-sec, $^{v}\infty$-m/sec, ^{v}c-m/sec, f-kg, ^{r}p-km)

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	WTF	1	MCCOM(47)	Weight (kg) after the first midcourse maneuver
I	TMC2	1	MCCOM(48)	Time(sec) of the next midcourse maneuver relative to anchor epoch
O	EXV2	1	MCCOM(49)	Expected next-maneuver velocity magnitude (m/sec)
I	DPT	3,10	MCCOM(50)	Sensitivity matrix (transposed) of end constraints to midcourse velocity variations
O	KBURN	1	MCCOM(154)	Midcourse burn option key
O	IR	1	MCCOM(158)	Sensitivity option key
I	KGLAW	1	MCCOM(164)	Guidance law selection key
O	ICB	1	MCCOM(165)	Central body number at midcourse
O	X	6	STATE(1)	State (km, km/sec) relative to JC
O	T	1	STATE(10)	Time (sec) to which X corresponds

Theory:

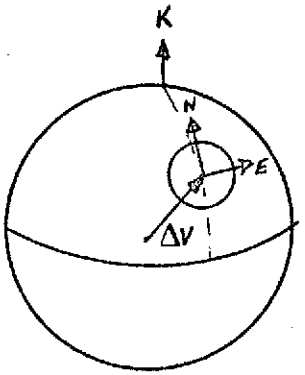
Errors in the miss vector, B, are assumed to be entirely attributable to errors in executing the required midcourse maneuver. Deviations in B are assumed linearly related to deviations in the corrective velocity impulse, ΔV, by

$$\delta B = \frac{\partial B}{\partial \Delta V} \delta(\Delta V) \tag{1}$$

Deviations in ΔV are, in turn, related to errors along ΔV (cut off error, ϵ_v) and normal to ΔV (pointing errors, ϵ_e and ϵ_n) by

$$\delta(\Delta V) = \frac{\partial \Delta V}{\partial(\epsilon_v, \epsilon_e, \epsilon_n)} \begin{Bmatrix} \epsilon_v \\ \epsilon_e \\ \epsilon_n \end{Bmatrix} \tag{2}$$

The pointing errors ϵ_e and ϵ_n, are components of the error vector eastward and northward, respectively, from ΔV on the celestial sphere as shown in the figure.



The partial derivative in Equation (2) is therefore written as

$$\frac{\partial \Delta V}{\Delta(\epsilon_v, \epsilon_e, \epsilon_n)} = \begin{bmatrix} V & E & N \end{bmatrix} \tag{3}$$

V, E, and N are unit column vectors defined by

$$V = \Delta V / |\Delta V| \tag{4a}$$

$$E = K \times V / |K \times V| \qquad \left(K = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right) \tag{4b}$$

$$N = V \times E \tag{4c}$$

The partial derivative in Equation (1) is the gradient needed for computing the required midcourse maneuvers and is therefore available. It is computed by the secant method in the vicinity of the solution. Because B has six components (time of flight, v_∞, v_c, total fuel, r_p, i) $\partial B / \partial \Delta V$ is a 6 x 3 matrix.

Defining P by

$$P = \frac{\partial B}{\partial \Delta V} \quad \frac{\partial \Delta V}{\partial(\epsilon_v, \epsilon_e, \epsilon_n)} \tag{5}$$

we can combine Equations (1) and (2) to obtain

$$\delta B = P \begin{Bmatrix} \epsilon_v \\ \epsilon_e \\ \epsilon_n \end{Bmatrix} \tag{6}$$

The covariance matrix of miss vector errors, C, is given by

$$C = E\left[\delta B\, \delta B^T\right] = P\, E\left[\begin{Bmatrix} \epsilon_v \\ \epsilon_e \\ \epsilon_n \end{Bmatrix} \begin{pmatrix} \epsilon_v & \epsilon_e & \epsilon_n \end{pmatrix}\right] P^T \tag{7}$$

where E is the expectation operator and where P, being deterministic, is not affected by E. Assuming no correlation among ϵ_v, ϵ_e, ϵ_n,

$$E\left[\begin{Bmatrix} \epsilon_v \\ \epsilon_e \\ \epsilon_n \end{Bmatrix} (\epsilon_v \quad \epsilon_e \quad \epsilon_n)\right] = \begin{bmatrix} \sigma_v^2 & 0 & 0 \\ 0 & \sigma_e^2 & 0 \\ 0 & 0 & \sigma_n^2 \end{bmatrix} \tag{8}$$

If we input values for σ_v, σ_e, and σ_n, we can then calculate the covariance matrix of miss vector errors by Equation (7).

Then the standard deviations of the miss-vector components are computed as uncorrelated errors by simply square-rooting the appropriate diagonal element of C. These are then multiplied by PRX to be displayed as "Probability P" values.

The problem of appropriately supplying execution error statistics now arises. If we assume cut-off error to be comprised of independent resolution and proportional errors,

$$\epsilon_v = \epsilon_{res} + \epsilon_{prop} \left|\Delta V\right| \tag{10}$$

$$\text{then} \quad \sigma_v^2 = \sigma_{res}^2 + \sigma_{prop}^2 \left|\Delta V\right|^2 \tag{11}$$

The resolution error is coded into the program as a uniformly distributed random variable on a .1 m/s interval centered at zero. The proportional error is normally

distributed with input standard deviations.

Pointing errors are conveniently specified as normally-distributed along orthogonal axes with the same variance on each axis. The distributions and axis orientations would more appropriately be related to the plane of the attitude maneuver, but this is more difficult to simulate. The pointing error is input as a single number which represents the standard deviation of equal normal distributions along the E and N axes.

Second Maneuver Requirements

We now consider the requirements of a second maneuver, executed at t_2, to correct the errors made in executing the first maneuver at t_1, $t_1 < t_2$. We must develop a mapping, M, of first midcourse execution errors, $\delta(\Delta V_1)$; into a second-midcourse velocity impulse, ΔV_2. The end constraint errors, δB related to $\delta(\Delta V)$ by Equation (1) are the very ones to be corrected by ΔV_2, so

$$\delta B = \frac{\partial B}{\partial \Delta V_1} \delta(\Delta V_1) = - \frac{\partial B}{\partial \Delta V_2} \Delta V_2 \tag{12}$$

or

$$\Delta V_2 = - \left(\frac{\partial B}{\partial \Delta V_2} \right)^{-1} \frac{\partial B}{\partial \Delta V_1} \delta(\Delta V_1) = M \delta(\Delta V_1) \tag{13}$$

The particular constraint error set to be nulled by ΔV_2 depends on the guidance law invoked. We assume that the guidance law for the second maneuver is the same as for the first. If the guidance law is FTA, FTE, VTE, or MTF, B has three components and, assuming the independence of their gradients with respect to ΔV_2, the indicated inverse exists. For the MFG law where there are only two end constraints, $\partial B/\partial \Delta V_2$ is made invertible by adjoining a row perpendicular to the two rows it has. The third column of the inverse is then zeroed before the multiplication, $\frac{\partial B}{\partial \Delta V}^{-1} \frac{\partial B}{\partial \Delta V_1}$, takes place because a third-constraint error must have no influence on the MFG solution. The expected value of the second-midcourse velocity is computed as the square root of the trace of the following matrix.

$$E\ (\Delta V_2\ \Delta V_2{}^T) = M\,E\ \left(\delta(\Delta V_1)\,\delta(\Delta V_1)^T\right)\ M^T$$

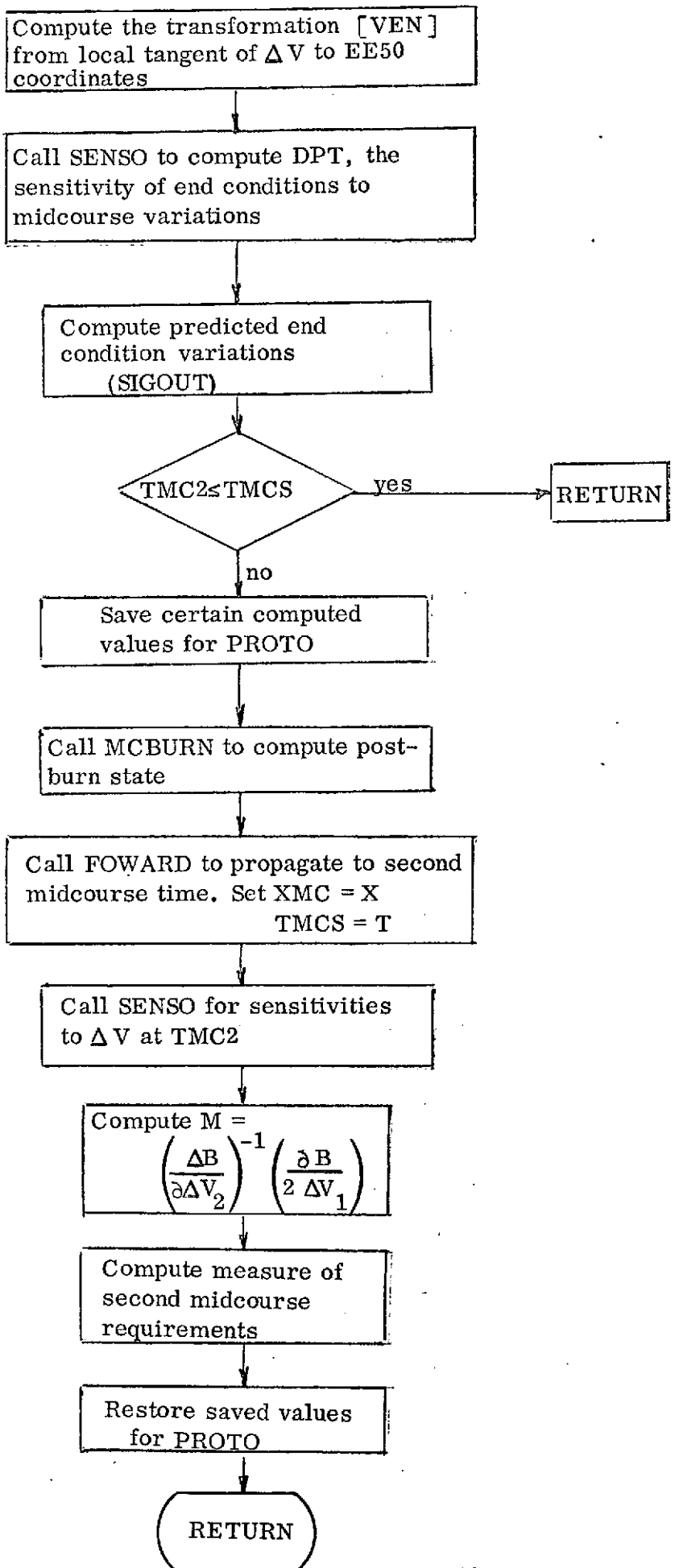
$$= M\,P \begin{bmatrix} \sigma_v^2 & 0 & 0 \\ 0 & \sigma_e^2 & 0 \\ 0 & 0 & \sigma_n^2 \end{bmatrix} \tag{14}$$

This expected value is then scaled by PRX to bring it up to a specified probability (univariate Gaussian) level for printout.

It would be desirable, perhaps, to present the expected spherical probable second-midcourse velocity, but the computation of this measure is too complex and time-consuming to be merited. The expected fuel is computed from the expected velocity by means of the rocket equation.

The sensitivity matrix, $\partial B/\partial \Delta V_1$, is usually available, since BELL is called right after the first-midcourse maneuver has been calculated. If not, however, it is computed by finite differences by calling SENSO. The second-midcourse sensitivity matrix, $\frac{\partial B}{\partial \Delta V_2}$, is computed at t_2 by finite differences about the trajectory which would result from a perfectly-executed maneuver at t_1.

SUBROUTINE BELL



SUBROUTINE BIGMAT

Calling Sequence: CALL BIGMAT (A, VALU, EVE, NN, NEIG, NVEC)

Purpose: BIGMAT calculates eigenvalues and eigenvectors of a symmetric array using Householder's method.

Common Blocks required: None

Subroutines required: None

Reference: Householder, A. S., and Bauer, F. L., "On Certain Methods for Expanding the Characteristic Polynomial", Numerische Mathematik 1. Bank, 1. Heft, p. 29. (1959).

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	A	21	Calling Operand	Symmetric Array *
O	EVE	6, 6	Calling Operand	Eigenvectors
I	NEIG	1	Calling Operand	Number of eigenvalues desired
I	NN	1	Calling Operand	Dimension of array *
I	NVEC	1	Calling Operand	Number of eigenvectors desired
O	VALU	6	Calling Operand	Eigenvalues

* The square symmetric array B has dimensions NN x NN, however, it is loaded into BIGMAT as A, a column vector that includes only diagonal and lower diagonal elements. Thus A(N) corresponds to B ((N-1) /NN + 1, MOD (N-1, NN) + 1).

Theory:

BIGMAT uses the Householder method of calculation. This method involves a codiagonalization of the original matrix before the roots are calculated. See the above reference for details of the method.

Description:

Given the column vector A, its dimension NN (see *), the number of eigenvalues desired

NEIG, and the number of eigenvectors desired NVEC, BIGMAT will calculate the corresponding number of eigenvalues and eigenvectors. The eigenvalues will be output largest first in the vector VALU. The eigenvectors will be output in the matrix EVE so that the elements in the ith column correspond to the ith eigenvalue. Since the eigenvalue is needed for the calculation of the corresponding eigenvector, NVEC must be less than or equal to NEIG. It is inportant to note that the elements in A are changed in the subroutine and the values output are in general not the same as those input. If NVEC is not equal to zero, then there will be output written on Unit 6 as follows: first a value "EPS" will be printed and then several lines indicating the eigenvalues and their corresponding eigenvectors. The value of EPS represents the accuracy of the eigenvalue calculation. For good calculations of the eigenvectors, EPS must be small compared to the difference of any two eigenvalues.

BLOCK DATA

Calling Sequence: Preloaded

Purpose: BLOCK DATA initializes various constants in the program.

Common Blocks Required: AVG, CONST, INPUT

Subroutines Required: None

Reference: Scarborough, James B., NUMERICAL MATHEMATICAL ANALYSIS, The Johns Hopkins Press, 1930.

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	DEFINITION
O	A	1000	Real portion of input array
O	ABSCIS	78	Abscissa for Gaussian quadrature formula
O	CONST	50	Program constants
O	KOPT	100	Integer portion of input array
O	WEIGHT	78	Weights for Gaussian quadrature formula

Description:

The arrays A and KOPT contain initialized values for variables which control program operations and may be changed through input. These initialized values are listed in Tables I and II. For a detailed description of these variables, see write-up of INPUT common.

The arrays WEIGHT and ABSCIS contain the weights and corresponding abscissa for the Gaussian quadrature formula. The values stored in these arrays are listed in Tables III and IV. For a description of the Gaussian quadrature and the use of these values, see the above reference.

The array CONST contains general mathematical constants in addition to constants pertaining to the planets. BLOCK DATA initializes only the mathematical constants. These constants and their preset values are as follows:

LOCATION	SYMBOLIC NAME	PRESET VALUE	DESCRIPTION
1	RAD	57.29577951308232	Degrees per radian
2	PI	3.141592653589793	π
3	PI2	6.283185307179586	2π
4	AU	149597893.	Kilometers per astronomical unit.
41	THS	3600.	Seconds per hour
42	TSH	2.7777777777778D-4	Hours per second
43	TDS	86400.	Seconds per day
44	TSD	1.1574074074074D-5	Days per second
45	G	.0098066	Average surface gravity on Earth (KM/sec^2)

TABLE I
REAL PORTION OF INPUT ARRAY

LOCATION	VALUE	LOCATION	VALUE	LOCATION	VALUE
4	540000.000	5	1.00000000	6	1.00000000
8	0.100000000D-10	10	0.100000000D 21	38	331.400000
100	2.16855300	101	324335.400	102	399600.800
103	42915.5150	104	126710600.	105	37918690.0
109	0.132715450D 12	110	4902.77800	112	2339.81600
113	6098.63600	114	6378.14000	115	3409.53000
116	71422.0000	117	57505.0000	118	25484.0000
119	24983.0000	120	6345.00000	121	706000.000
122	1738.09000	126	0.729212361D-04	127	0.798820000D-04
128	0.175854900D-03	134	0.266169950D-05	197	0.450000000D-04
198	0.200000000	201	900.000000	202	20000.0000
221	20.0000000	222	25.0000000	260	48.5000000
261	26.6892000	262	26.6892000	280	-9867.12000
281	-9867.12000	321	900.000000	322	20000.0000
331	20.0000000	332	25.0000000	350	0.220400000D-01
351	0.120200000D-01	352	0.120200000D-01	360	3.55550000
361	3.55550000	380	0.100000000D 21	381	0.100000000D 21
382	0.100000000D 21	383	0.100000000D 21	394	0.100000000D 21
385	0.100000000D 21	407	0.600000000	408	136000000.
409	400000000.	410	27.7072300	411	47.2231680
412	113.716372	413	148.956961	414	-147.511732
415	-82.8760930	416	-76.8429650	417	-70.6665190
420	2838.00000	421	-59.0000000	422	396000.000
423	0.620000000	433	13560.0000	434	7200.00000
435	0.700000000	436	0.200000000D-01	437	0.700000000D-01
438	0.300000000D-03	439	36000.0000	440	259200.000
441	226.000000	442	282.500000	443	71.1100000
444	2838.00000	446	59.0000000	447	6000.00000
448	6000.00000	450	0.100000000D 21	460	18000.0000
470	6.00000000	471	0.167800000	472	20.4000000
473	13.7700000	474	5.00000000	475	20.0000000
478	7200.00000	479	0.300000000D-03	480	-25.7357820
481	-18.9032240	482	-24.7576070	483	-35.4480090
484	64.8236430	485	35.0138150	486	38.8103660
487	-32.9759380	490	10.0000000	491	10.0000000
492	10.0000000	493	0.100000000D-03	494	0.100000000D-03
495	0.200000000D-01	496	5.00000000	497	0.200000000
501	616000.000	502	925000.000	503	565000.000
504	48000000.0	505	54000000.0	506	51000000.0
507	86000000.0	508	33000000.0	509	0.100000000D 11
510	110000.000	512	20.0000000	514	2.00000000
515	90.0000000	516	12.0000000	517	238.000000
518	20.0000000	519	20.4100000	520	1.00000000
521	1.00000000				

TABLE II

INTEGER PORTION OF INPUT ARRAY

LOCATION	VALUE	LOCATION	VALUE	LOCATION	VALUE
3	2	10	1	11	2
14	3	15	3	17	5
18	1	19	1	31	11
32	2	35	1	36	4
39	3	40	2	41	10
42	15	43	21	48	1
51	10	52	2	53	50
54	17	55	1	60	1
61	11	62	2	63	2
64	1	65	10	66	1
67	100	68	6	69	3
70	95	71	6	75	6
83	11	88	12	89	1234567

TABLE III

WEIGHT ARRAY

LOCATION	VALUE	LOCATION	VALUE	LOCATION	VALUE
1	1.00000000	2	0.888888889	3	0.555555556
4	0.652145155	5	0.347854845	6	0.568888889
7	0.473628670	8	0.236926385	9	0.467913935
10	0.360761573	11	0.171324492	12	0.417959194
13	0.381830051	14	0.279705291	15	0.220484966
16	0.362683783	17	0.313706645	18	0.222381034
19	0.101228536	20	0.330239355	21	0.312347077
22	0.260610696	23	0.180643161	24	0.813743894D-01
25	0.295524225	26	0.269266719	27	0.219086363
28	0.149451349	29	0.666713443D-01	30	0.272925087
31	0.262804545	32	0.233193765	33	0.196290211
34	0.125580369	35	0.556635671D-01	36	0.249147046
37	0.233492537	38	0.203167427	39	0.160978329
40	0.106939326	41	0.471753364D-01	42	0.232551553
43	0.226283180	44	0.207816048	45	0.172145981
46	0.138873510	47	0.921214992D-01	48	0.404840048D-01
49	0.215263853	50	0.205193464	51	0.135538397
52	0.157203167	53	0.121519571	54	0.201582872D-01
55	0.351194603D-01	56	0.202578242	57	0.190431495
58	0.196161000	59	0.166269206	60	0.139570678
61	0.107159220	62	0.703660475D-01	63	0.307532420D-01
64	0.189450610	65	0.182603415	66	0.169156519
67	0.149595989	68	0.124629971	69	0.951585117D-01
70	0.622535239D-01	71	0.271524594D-01	72	0.176140971D-01
73	0.406914298D-01	74	0.626720483D-01	75	0.933767416D-01
76	0.101930120	77	0.118194532	78	0.131688639

TABLE IV

ABSCISSA ARRAY

LOCATION	VALUE	LOCATION	VALUE	LOCATION	VALUE
1	0.577350269	2	0.0	3	0.774596669
4	0.339981044	5	0.861136312	6	0.0
7	0.538469310	8	0.906179846	9	0.238619186
10	0.661209386	11	0.932469514	12	0.0
13	0.405845151	14	0.741531186	15	0.949107912
16	0.183434642	17	0.525532410	18	0.796666477
19	0.960289856	20	0.0	21	0.324253423
22	0.613371433	23	0.836031107	24	0.968160240
25	0.148874339	26	0.433395394	27	0.679409568
28	0.865063367	29	0.973906529	30	0.0
31	0.269543156	32	0.519096129	33	0.730152006
34	0.887062600	35	0.978228658	36	0.125233409
37	0.367831499	38	0.587317954	39	0.769902674
40	0.904117256	41	0.981560634	42	0.0
43	0.230458316	44	0.448492751	45	0.642349339
46	0.801578091	47	0.917598399	48	0.984183055
49	0.108054949	50	0.319112369	51	0.515248636
52	0.687292905	53	0.827201315	54	0.928434884
55	0.986283809	56	0.0	57	0.201194094
58	0.394151347	59	0.570972173	60	0.724417731
61	0.848206583	62	0.937273392	63	0.987992518
64	0.950125098D-01	65	0.281603551	66	0.458016778
67	0.617876244	68	0.755404408	69	0.865631202
70	0.944575023	71	0.989400935	72	0.993128599
73	0.963971927	74	0.912234428	75	0.939116972
76	0.746331906	77	0.636053681	78	0.510867002

FUNCTION BURND

Calling Sequence: T = BURND(DVMG)

Purpose: BURND computes burn duration as a function of incremental velocity.

Common Blocks Required: CØNST, INPUT, MCCØM

Subroutines Called: None

Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	DVMG	1	Argument List	Midcourse correction velocity (km/sec)
I	G	1	CØNST(45)	Earth's surface gravity (km/sec^2)
I	WTØ	1	INPUT(38)	Initial spacecraft weight (kg)
I	TWD	10	INPUT(320)	Times for weight flow rate table (sec)
I	WD	10	INPUT(350)	Weight flow rate tabular values (kg/sec)
I	ASPMC	1	INPUT(441)	Specific impulse (sec)
I	DW	10	MCCØM (110)	Weight expenditure at TWD (kg)
Ø	BURND	1	Call	Burn duration (sec)

Method:

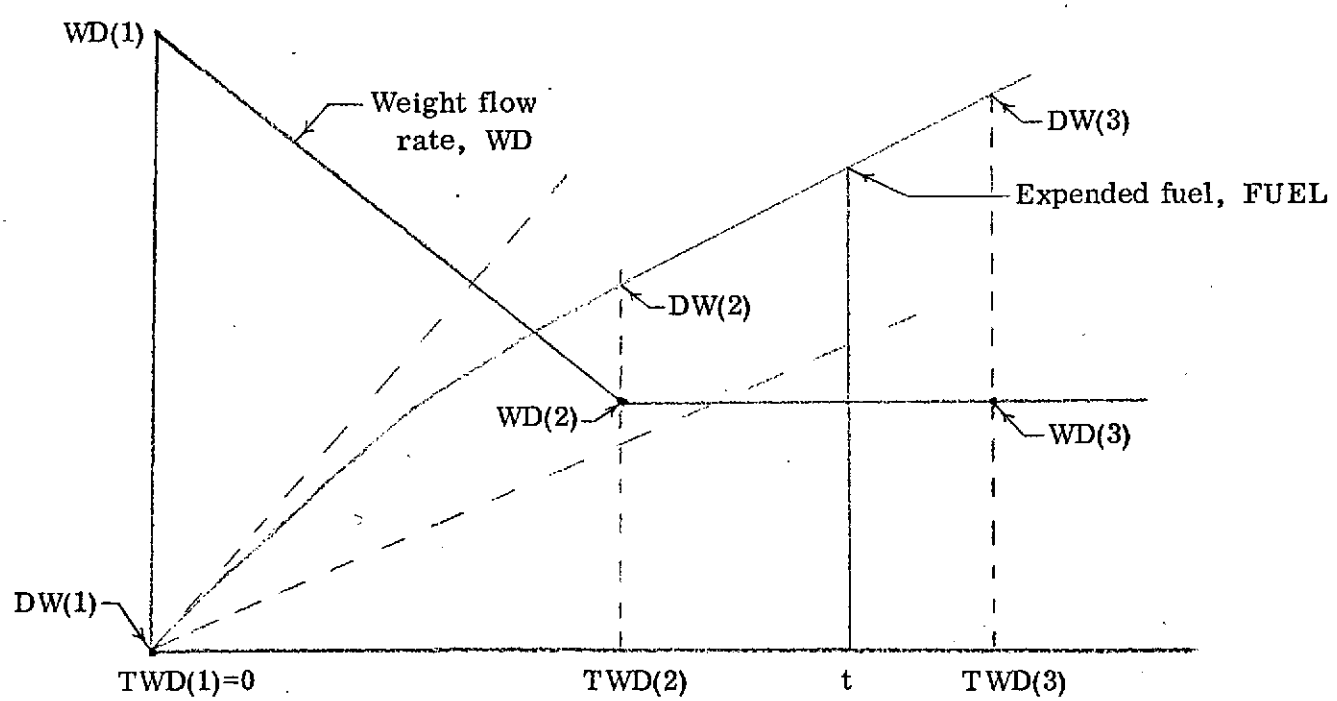
Validity of the rocket equation is assumed. The fuel weight, FUEL, corresponding to the velocity increment, DVMG, is

$$\text{FUEL} = \text{WT}\emptyset\left(1 - e^{-\frac{\text{DVMG}}{G \cdot \text{ASPMC}}}\right) .$$

A piecewise-linear weight flow rate is assumed. The fuel expended as a function of time is computed under this assumption in MCSET and stored in DW. That is,

$$\text{DW(I)} = \text{fuel expended at TWD(I)} .$$

FUEL is tested against DW until FUEL $\leq$ DW(I+1). Then the burn duration is TWD(I) plus the time since TWD(I), which would cause the fuel to increase by FUEL - DW(I).



Weight flow rate and expended weight versus burn duration

On any segment of the burn history where the weight flow rate, $\dot{w}$, is linear, the slope of the weight flow rate, $\ddot{w}$, is constant. We define the following:

$$\delta t = t - TWD(I)$$

$$\delta f = FUEL - DW(I)$$

$$\dot{w}_o = WD(I)$$

$$\ddot{w} = \frac{WD(I+1) - WD(I)}{TWD(I+1) - TWD(I)}$$

The fuel expenditure is a quadratic function of time.

$$\delta f = \dot{w}_o \, \delta t + \tfrac{1}{2} \ddot{w} \, \delta t^2$$

The usual solution of the quadratic,

$$\delta t = \frac{-\dot{w}_o \pm \sqrt{\dot{w}_o^2 + 2 \ddot{w} \, \delta f}}{\ddot{w}}$$

is ill-defined when $\ddot{w} = 0$, so we multiply numerator and denominator by $-\dot{w}_o \mp \sqrt{\dot{w}_o^2 + 2 \ddot{w} \, \delta f}$. The result is

$$\delta t = \frac{2 \, \delta f}{\dot{w}_o + \sqrt{\dot{w}_o^2 + 2 \ddot{w} \, \delta f}} \quad .$$

The sign ambiguity on the radical is removed by observing that for $\ddot{w} = 0$, the solution must reduce to

$$\delta t = \frac{\delta f}{\dot{w}_o} \quad .$$

It may be noted that

$$\overline{\dot{w}} = \tfrac{1}{2}\left(\dot{w}_o + \sqrt{\dot{w}_o^2 + 2 \ddot{w} \, \delta f} \right)$$

is the constant weight flow rate which also would render δf in the time interval, δt.

SUBROUTINE BVE

Calling Sequence: CALL BVE(X, V, U, B, BTR, C3, STHET, CTHET, E, S, RP, T, R, KK)

Purpose: To compute the miss-vector components of the orbit, relative to the target body, from the Cartesian state.

Common Blocks Required: None

Subroutines Called: CRØSS, RØTAIT

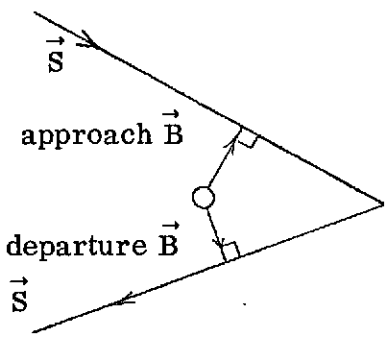
Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	X	3	Call List	Position vector (km)
I	V	3	Call List	Velocity vector (km/sec)
I	U	1	Call List	Gravitational constant (km^3/sec^2)
Ø	B	3	Call List	Miss vector (km)
Ø	BTR	2	Call List	Miss vector (miss-plane components (km)
Ø	C3	1	Call List	Energy $\left(V^2 - \frac{2\mu}{r}\right)$ (km^2/sec^2)
Ø	STHET	1	Call List	Sine of the true anomaly
Ø	CTHET	1	Call List	Cosine of the true anomaly
Ø	E	1	Call List	Eccentricity
Ø	S	3	Call List	Unit asymptote vector
Ø	RP	1	Call List	Radius of periapsis (km)
Ø	T	3	Call List	Equatorial miss-axis
Ø	R	3	Call List	Zenith miss-axis
I	KK	1	Call List	Asymptote indicator

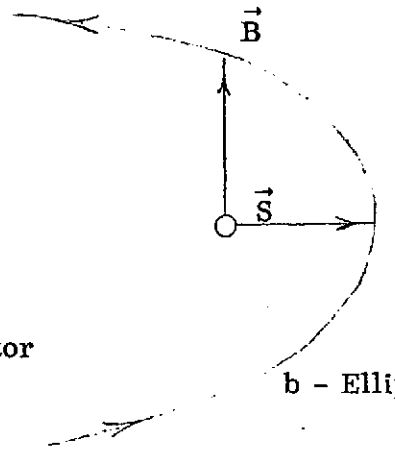
Method:

Subroutine BVE computes the target miss vector, B, from Cartesian input vectors X, position, and V, velocity.

For hyperbolic orbits, B is the vector from the body center to either the approach or departure asymptote depending on a key set by the user, (see Fig. 1a). For elliptic orbits, the B vector is always the semi-latus rectum for the departing orbit, (see Fig. 1b).



a - Hyperbolic orbit



b - Elliptic orbit

Figure 1 - B Vector

S is a unit vector along the asymptote for the hyperbolic case, and is directed toward periapsis from the body center in the elliptic case.

T is a unit vector taken as

$$T = \frac{\vec{k} \times \vec{S}}{|k \times S|}$$

where i, j, k are the orthogonal unit vectors of the Cartesian input position and velocity vectors X and V.

R is the unit vector given by

$$R = \vec{S} \times \vec{T}$$

The vectors B, R, and T lie in a plane normal to the unit vector S (see Fig. 2).

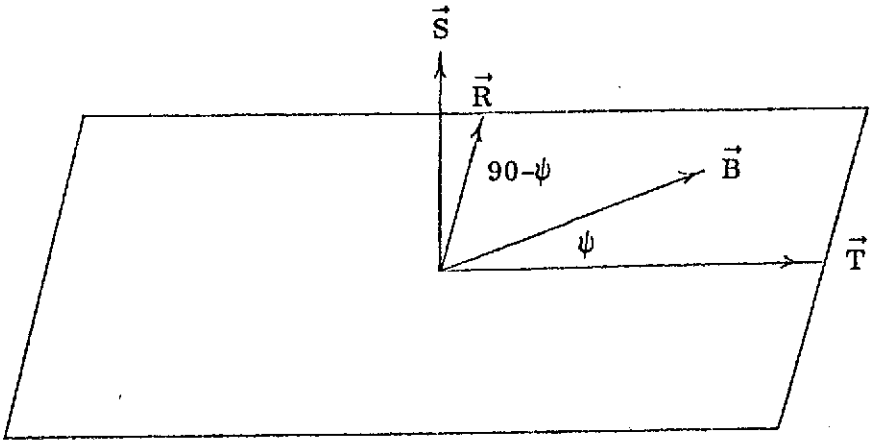


Figure 2 - S Normal to Plane Containing $\vec{R}$, $\vec{B}$, $\vec{T}$

For visualization purposes, if $\vec{X}$ and $\vec{V}$ were given to the subroutine with the i and j vectors in the moon's equatorial plane, then the $\vec{T}$ and $\vec{R}$ vectors would be as shown in Fig. 3. The $\vec{R}$ vector doen not lie along the polar axis but, rather, normal to $\vec{S}$ and $\vec{T}$. For lunar approach from Earth, the $\vec{R}$ vector is roughly in the direction of the polar axis, so if one had a trajectory such that $B \cdot R = 0$, the plane of the trajectory around the moon should lie close to the equator.

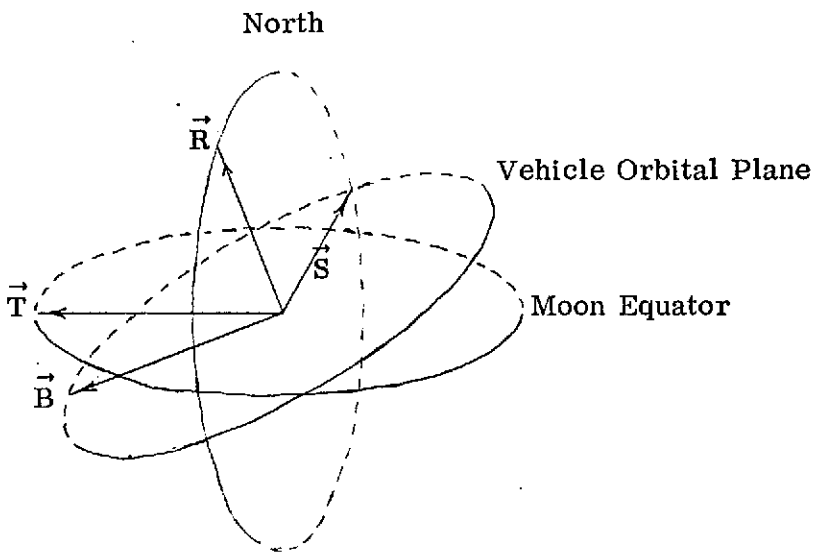


Figure 3 - Miss-Plane Geometry

Theory:

Given $\vec{X}$ and $\vec{V}$, the position and velocity vectors of the vehicle relative to the central body, and U, the gravitational constant for the central body, compute the following:

The semi-latus rectum, PP

$$PP = |\vec{X} \times \vec{V}|^2 / U \tag{1}$$

The magnitude of the position vector, $\vec{X}$

$$RMAG = |\vec{X}| \tag{2}$$

The energy, C3

$$C3 = \vec{V} \cdot \vec{V} - 2U/RMAG \tag{3}$$

The eccentricity, E

$$E = [1 + (C3)PP/U]^{\frac{1}{2}} \tag{4}$$

The radial component of velocity

$$RDOT = (\vec{X} \cdot \vec{V})/RMAG \tag{5}$$

The sine and cosine of the true anomaly, θ

$$CTHET = \cos\theta = (PP - RMAG)/(E\ RMAG) \tag{6}$$

$$STHET = \sin\theta = RDOT(PP/U)^{\frac{1}{2}} / E \tag{7}$$

Define a unit vector $\overrightarrow{UX}$ by

$$\overrightarrow{UX} = \vec{X}/RMAG \tag{8}$$

and the unit angular momentum vector by

$$\overrightarrow{UW} = \frac{\vec{X} \times \vec{V}}{|\vec{X} \times \vec{V}|} \tag{9}$$

As shown in Fig. 4, a unit vector $\vec{UY}$ normal to $\vec{UX}$ and $\vec{UW}$ lying in the orbital plane, and in the direction of rotation is

$$\vec{UY} = \vec{UW} \times \vec{UX} \tag{10}$$

The rotation of $\vec{UX}$ and $\vec{UY}$ through $(-\theta)$ yields $\vec{P}$ and $\vec{Q}$, where $\vec{P}$ is a unit vector toward periapsis, and $\vec{Q}$ is a unit vector in the same direction as the semi-latus rectum of the orbit.

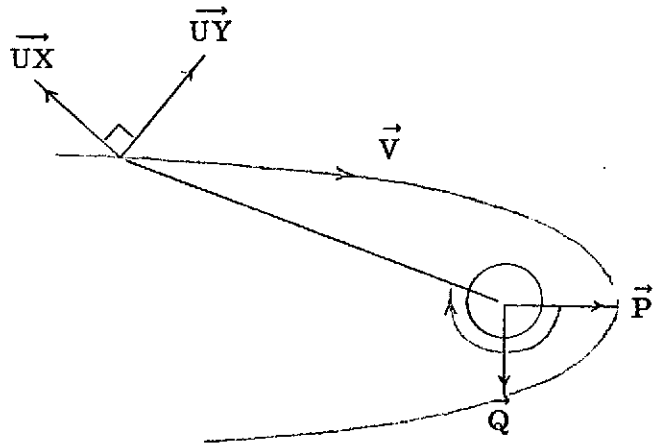


Figure 4 - In-Plane Orbit Direction Vectors

Rotate $\vec{UX}$ and $\vec{UY}$ through $(-\theta)$ to $\vec{P}$ and $\vec{Q}$

$$\begin{aligned} \vec{P} &= \vec{UX} \cos\theta - \vec{UY} \sin\theta \\ \vec{Q} &= \vec{UX} \sin\theta + \vec{UY} \cos\theta \end{aligned} \tag{11}$$

The $\vec{S}$ and $\vec{B}$ vectors are computed as follows:

For elliptical orbits

$$\begin{aligned} \vec{S} &= \vec{P} \\ \vec{B} &= \vec{Q} \left| \vec{B} \right| = \vec{Q}\, \mathrm{BMAG} = \vec{Q}\, \mathrm{PP} \end{aligned} \tag{12}$$

For hyperbolic orbits, $\vec{Q}$ and $\vec{P}$ are rotated through an angle of magnitude $(90 - \alpha)$ to obtain $\vec{S}$ and $\vec{UB}$, a unit vector in the direction of the $\vec{B}$ vector. The angle α is the half-angle between asymptotes and is computed by

$$\text{CALP} = \cos\alpha = 1/E = \sin(90-\alpha)$$
$$\text{SALP} = \sin\alpha = (1-1/E^2)^{\frac{1}{2}} = \cos(90-\alpha) \tag{13}$$

In the case of the approach hyperbola, $\vec{Q}$ and $\vec{B}$ must be rotated counter-clockwise through $(90-\alpha)$, while for departure, clockwise. Thus, for approach, with STHET negative (see Fig. 5)

$$\vec{S} = \vec{Q}\sin\alpha + \vec{P}\cos\alpha$$
$$\overrightarrow{UB} = -\vec{Q}\cos\alpha + \vec{P}\sin\alpha \tag{14}$$

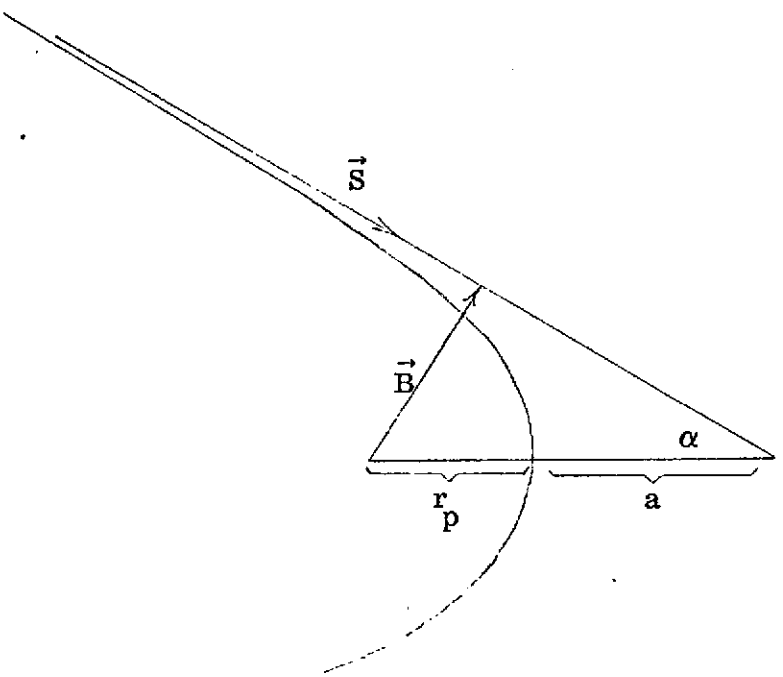


Figure 5 - Hyperbola and Approach Asymptote

For departure, with STHET positive

$$\vec{S} = \sin\alpha\,\vec{Q} - \cos\alpha\,\vec{P}$$
$$\overrightarrow{UB} = \cos\alpha\,\vec{Q} + \sin\alpha\,\vec{P} \tag{15}$$

Finally, the $\vec{B}$ vector is

$$\vec{B} = \text{BMAG}\,\overrightarrow{UB} \tag{16}$$

where

$$\text{BMAG} = (\text{U/C3} + \text{RP}) \sin \alpha$$

$$\text{U/C3} = \text{the semi-major axis}$$

$$\text{RP} = \text{PP}/(1 + \text{E}) = \text{radius of periapsis}$$

$\vec{B} \cdot \vec{T}$ and $\vec{B} \cdot \vec{R}$ are computed and returned by the subroutine as BTR(1) and BTR(2) respectively. As seen in Fig. 2, BTR(1) represents BMAG cos ψ and BTR(2) represents BMAG cos $(90 - \psi)$.

SUBROUTINE CALEND

Calling Sequence: CALL CALEND(T, JYR, JDY, MO, NHR, MIN, SEC)

Purpose: This subroutine converts the current MAESTRO time to calendar date.

Common Blocks Required: INPUT

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DJO	1	INPUT(46)	Modified julian date of state epoch
O	JDY	1	Calling Operand	Day
O	JYR	1	Calling Operand	Years since 1900.
O	MIN	1	Calling Operand	Minutes
O	MO	1	Calling Operand	Month
O	NHR	1	Calling Operand	Hour
O	SEC	1	Calling Operand	Seconds
I	T	1	Calling Operand	Seconds since state epoch

Description:

The calendar date is obtained from the modified julian date of state epoch and time since state epoch. Output is year, day, month, hour, minute and seconds.

SUBROUTINE CLOSE

Calling Sequence: CALL CLOSE

Purpose: Subroutine CLOSE determines the spacecraft's central planet and transforms the state to the central planet if a new planet becomes central.

Common Blocks Required: CNTRL, CONST, INPUT, INTER, PLNET, STATE

Subroutines Required: DVMAG, INTEG, ORBIT, TRMN, UPDATE

Input / Output

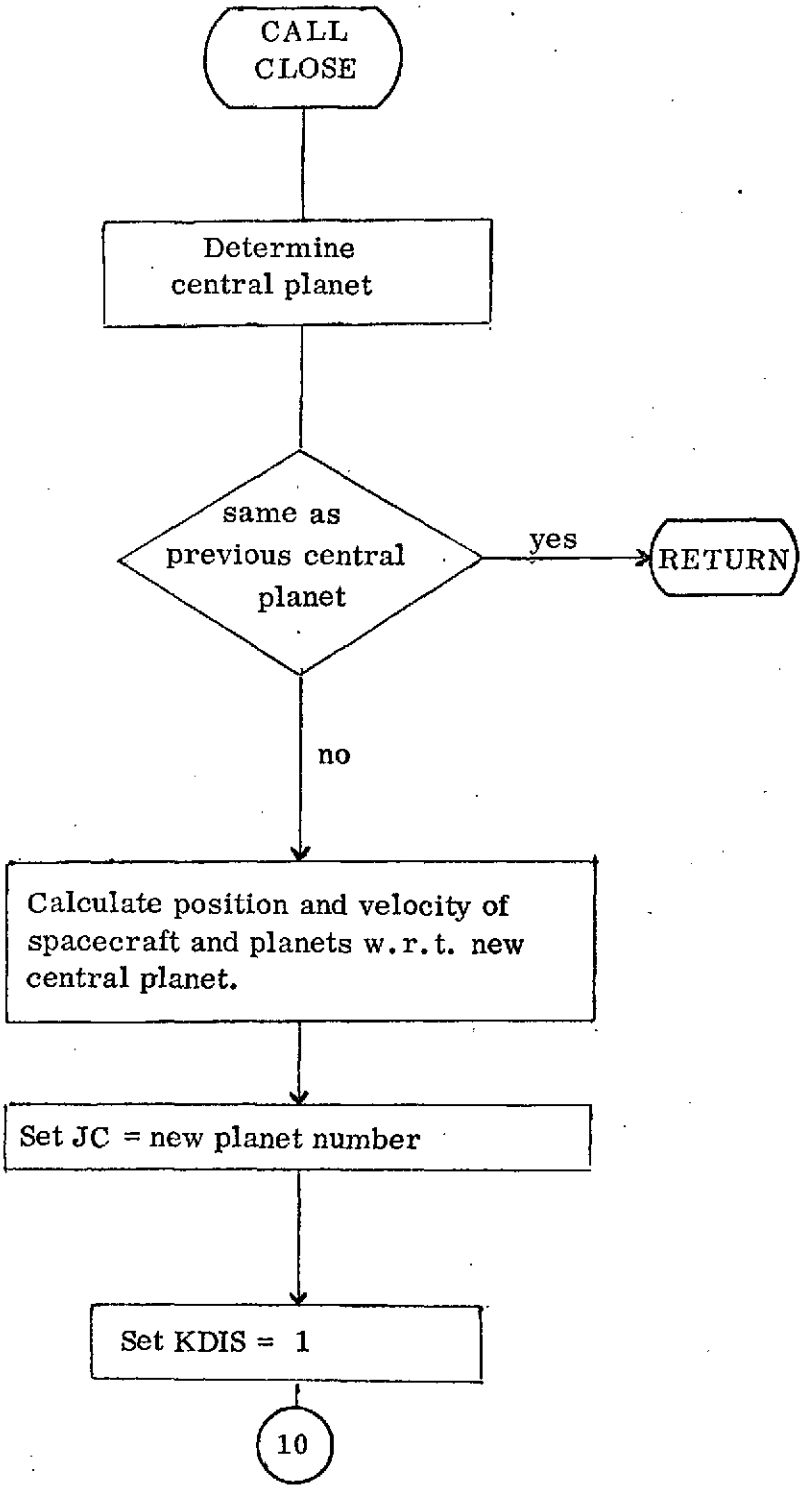
I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	DST	12	PLNET(73)	Distances from s/c to planets
I/O	DX	3	STATE(4)	Velocity of s/c
O	ELM	6	STATE(14)	Orbital elements of s/c
I	GM	12	CONST(5)	Planet gravitational constants
O	INT	1	INTER(131)	Interpolation counter
I/O	JC	1	CNTRL(7)	Central planet number
O	KDIS	1	CNTRL(5)	Discontinuity flag
I	KP	12	INPUT(1001)	Planets in the system
I	METH	1	INPUT(1013)	Propagator method
I	RSWTCH	12	INPUT(500)	Distances when planets become central
I/O	X	3	STATE(1)	Position of s/c
I/O	XP	6, 12	PLNET(1)	Positions and velocities of planets

Description:

There are two major parts to this subroutine. In the first part CLOSE determines which planet's sphere of influence the spacecraft resides in. The sphere of influence is denoted by RSWTCH. If the spacecraft's distance from planet J is less than RSWTCH(J), then planet J is central. If that planet is already the central planet, then CLOSE returns. If that planet is different than the central planet, then that planet becomes the central planet.

The second part of the subroutine takes care of the changes in the system due to the change in central planet. The position and velocity vectors of the planets and the spacecraft are changed to correspond to position and velocity relative to the new central planet. KDIS is set to one to denote a discontinuity in the system. Depending on the propagation scheme (METH), new orbital elements are calculated if needed, and INTEG is called to set up new values and derivatives of integration variables. INT is set to zero and then UPDATE is called so that the interpolation table is cleared and a new one is started.

SUBROUTINE CLOSE



SUBROUTINE CONTRL

Calling Sequence: CALL CONTRL

Purpose: This subroutine initializes the compute interval table and other constants before initiation of the program options

Common Blocks Required: ANKOR, CONST, INPUT, STATE

Subroutines Required: DVMAG, PRINT, SETUP2

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	ANKVEC	6	ANKOR(1)	Initial anchor vector position and velocity vectors
O	ATT	3	STATE(11)	Unit vector along the spacecraft's centerline
O	DECO	1	INPUT(48)	Initial declination
O	DELT	10	INPUT(180)	Table of compute intervals
I	KMETH	1	INPUT(1036)	Initial trajectory propagation indicator
I	RAD	1	CONST(1)	Radian-degree conversion factor
O	RAO	1	INPUT(47)	Initial right ascension
O	TCOMP	10	INPUT(170)	Array of switching times used in compute interval table.
I	X	6	STATE(1)	Initial position and velocity vectors

Description:

This subroutine initializes program constants and flags before initiation of the program options. Subroutine SETUP2 is called at the beginning of this routine. Most of the initialization is performed in SETUP2. The compute interval table is also established if one is not already input. The compute interval table is established according to the

initial spacecraft radius, R, as shown in the following table.

$R < 20,000$

DELT(1) = 300.	TCOMP(1) = 3600.
DELT(2) = 1800.	TCOMP(2) = 18000.
DELT(3) = 18000.	TCOMP(3) = $1\ (10)^{20}$

$20000 < R < 40000$

DELT(1) = 1800.	TCOMP(1) = 18000
DELT(2) = 18000.	TCOMP(2) = $1\ (10)^{20}$

$40000 < R$

DELT(1) = 18000.	TCOMP(1) = $1(10)^{20}$

If the initial spacecraft attitude is not input, the attitude is set in the same direction as the initial velocity vector. The subroutine next calls subroutine PRINT in order to obtain a list of the initial input array.

SUBROUTINE COVERT

Calling Sequence: CALL COVERT (COV, X, Q, KCOV)

Purpose: This subroutine transforms a covariance matrix from a local tangent plane coordinate system to the coordinate system of the input state which defines the local system.

Common Blocks Required: None

Subroutines Required: CROSS, VNORM

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	COV	6, 6	Calling Operand	Input covariance matrix
I	KCOV	1	Calling Operand	Transformation flag. If KCOV is not zero, the covariance matrix will be transformed
O	Q	6, 6	Calling Operand	Output covariance matrix
I	X	6	Calling Operand	Position and velocity vectors of the state

Description:

The local tangent plane coordinate system is defined as follows:

X-axis Positive along the position vector

Y-axis Lies in the plane formed by the position and velocity vectors and normal to the position vector positive in the direction of the velocity. It is represented vectorially by

$$(\bar{X} \times \bar{V}) \times \bar{X}$$

Z-axis Normal to the plane formed by the position and velocity vectors. It is represented vectorially by

$$\bar{X} \times \bar{V}$$

This subroutine determines the rotation matrix from the local tangent plane to the coordinate system of the state $\bar{X}$ and $\bar{V}$. To calculate the transformation matrix, T, it is first necessary to establish the vectors

$$\hat{S} = \bar{X} \times \bar{V} / (|\bar{X}| \, |\bar{V}|)$$

$$\hat{u} = \bar{S} \times \bar{X} / |\bar{X}|$$

Then the first column of the matrix is a unit vector along $\bar{X}$. The second column is composed of vector $\hat{u}$, while the last column consists of vector $\hat{S}$.

The covariance matrix is transformed to the new coordinate system using the following relationship,

$$[Q] = [T] [COV] [T]^t$$

If KCOV is zero, no transformation is performed and the output matrix, Q, is set equal to the input matrix, COV.

SUBROUTINE CRASH

Calling Sequence: CALL CRASH

Purpose: Subroutine CRASH determines the time of closest approach to the target planet.

Common Blocks Required: CNTRL, INPUT, INTVAR, PLNET, SAVE, STATE

Subroutines Required: DVMAG, INTERP, PLANET

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	JC	1	CNTRL(7)	Central planet number
I	JT	1	INPUT(1031)	Target planet number
I	KCA	1	CNTRL(11)	Counter used in the closest approach iteration
I	KHIGH	1	INPUT(1079)	If set to 1. Subroutine will determine the time of apoapis.
I	RSAV	1	SAVE (41)	Flight path angle on last step
I	T	1	STATE(10)	Current time since state epoch
O	TCA	1	STATE (29)	Time of closest approach
I	X	6	STATE(1)	Current spacecraft position and velocity vectors
I	XP	6, 12	PLNET(1)	Planet's position and velocity vectors.

Theory:

The time of closest approach is determined by an iterative process using the flight path angle as the dependent variable and time as the independent variable. The time is adjusted in order to drive the flight path angle to zero. A Newton-Raphson type iteration is employed to determine the time. The sine of the flight path angle at time, t, is determined from:

$$R3 = \left(\bar{X} \cdot \bar{V}\right) \Big/ |\bar{X}| \quad |\bar{V}| \tag{1}$$

where $\bar{X}$ and $\bar{V}$ are the position and velocity vectors at time t determined from subroutine INTERP.

The derivative of this sine with respect to time is numerically determined by calculating the sine at two times as described in equation (1) and dividing their difference by the difference in time as,

$$DERIV = (R2 - R3) \; / \; DELT \tag{2}$$

where R2 is the sine of the flight path angle at T + DELT. The change in time to drive the sine of the flight path angle to zero is calculated from:

$$DEL = R3 \; / \; DERIV \tag{3}$$

The sine of the flight path angle is determined at time T = DEL. The iteration is assumed to converge if the flight path angle is within a small tolerance around zero. If not, the derivative is recalculated in equation (2) with T = T2 + DEL and the process repeated. A limit of 20 iterations are allowed. A limit to the size of the step, DEL, is also employed to help assure convergence.

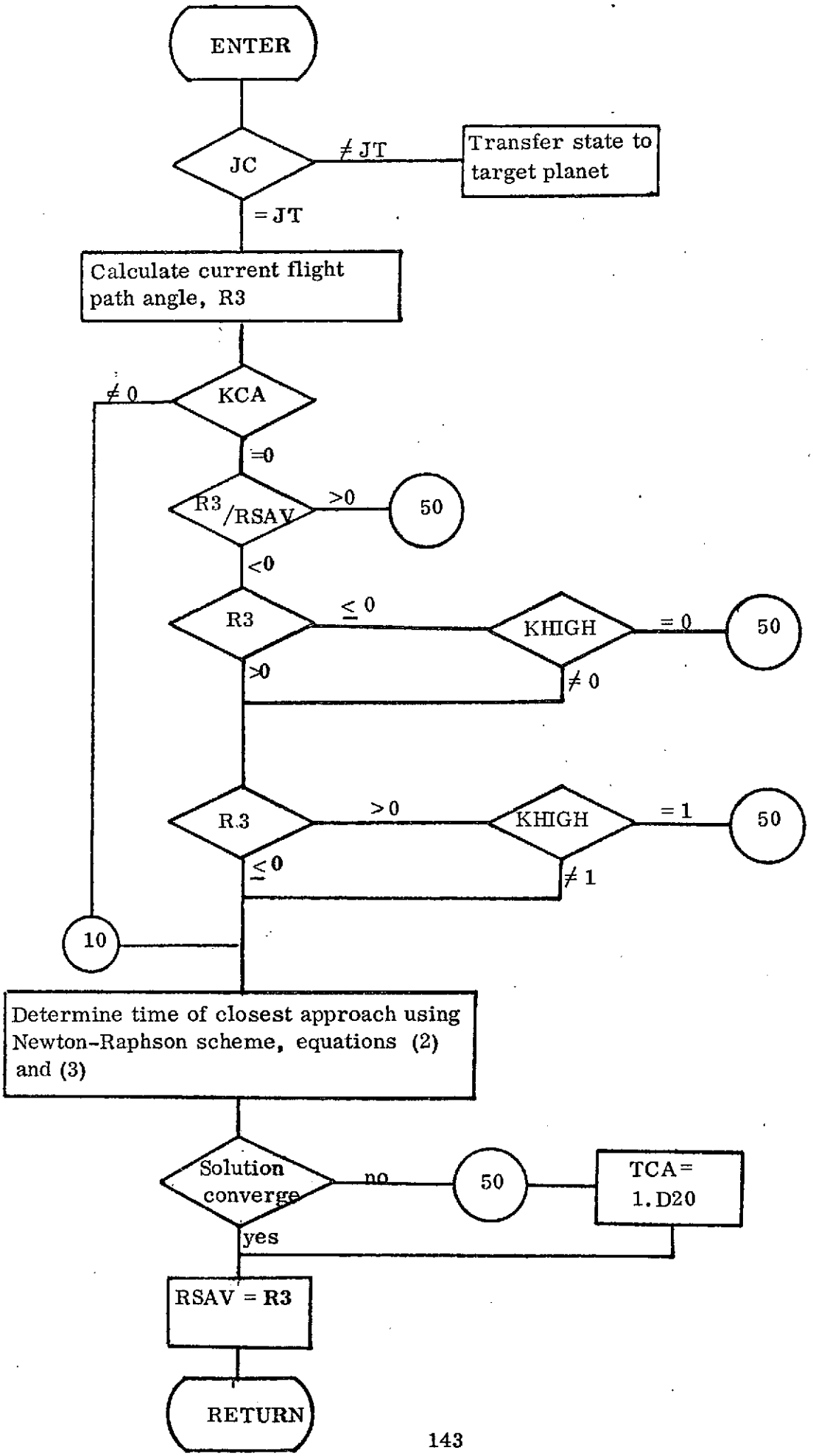
Description:

The time of closest approach is determined through a double iteration using this routine along with subroutines INTERP and TIMEC. There is an iteration inside subroutine CRASH to determine the time of closest approach using subroutine

INTERP. In this iteration the spacecraft's state at time, T, is determined using the interpolation logic in subroutine INTERP. T is adjusted until the spacecraft's flight path angle is within a specified tolerance. A Newton -Raphson type iteration discussed in the theory is employed to determine this time. The second iteration loop is associated with subroutines TIMEC and CRASH.

The time of closest approach is transfered to TIMEC. The time of closest approach is considered a discontinuity time in subroutine TIMEC. Thus, if the current time is greater than the time of closest approach,the state is restored to the values at the last step and the compute interval is adjusted to integrate to the time of closest approach. The time of closest approach is recalculated in subroutine CRASH at the discontinuity time. If the new time of closest approach is within a tolerance of the last time of closest approach, it is assumed that convergence has been achieved on the time of closest approach. If not, the new time of closest approach is used as a discontinuity time in TIMEC and the iteration repeated. The iteration between TIMEC and CRASH usually converges in one or two iterations. This iteration would not be necessary if the interpolation logic exactly matched the numerical integrator. KCA is a counter used to determine the number of TIMEC-CRASH iterations. A limit of 7 iterations are allowed.

SUBROUTINE CRASH



SUBROUTINE CROSS

Calling Sequence: CALL CROSS (X, Y, Z)

Purpose: This subroutine calculates the vector cross product

Common Blocks Required: None

Subroutines Required: None

Inputs / Outputs

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	X	3	Calling Argument	Input vector X
I	Y	3	Calling Argument	Input vector Y
O	Z	3	Calling Argument	Output cross product

Description:

The vector cross product is determined by this subroutine. The cross product is determined by

$$Y_1 = X_2 Y_3 - X_3 Y_2$$

$$Y_2 = X_3 Y_1 - X_1 Y_3$$

$$Y_3 = X_1 Y_2 - X_2 Y_1$$

SUBROUTINE DATE

Calling Sequence: CALL DATE (YEAR, DAY, QM, HR, DJO)

Purpose: DATE converts a calendar date to its julian date.

Common Blocks Required: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DAY	1	CALLING OPERAND	Day of the month
O	DJO	1	CALLING OPERAND	Julian date
I	HR	1	CALLING OPERAND	Hour of the day
I	QM	1	CALLING OPERAND	Month
I	YEAR	1	CALLING OPERAND	Year

Description:

Date calculates the number of days since 1900 and adds it to the modified julian date of 1900 to get the actual modified julian date.

SUBROUTINE DOPLER

Calling Sequence: CALL DOPLER

Purpose: This subroutine determines the velocity away from the visible tracking sites and determines the doppler frequency shift.

Common Blocks Required: CNTRL, CONST, INPUT, OBSIT, STATE

Subroutines Required: DVMAG, M50EPM, ROTATE, ORBIT, INTEG, UPDATE

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	CAR1	1	INPUT(408)	Spacecraft's primary carrier frequency
I	CAR2	1	INPUT(409)	Spacecraft's secondary carrier frequency
I	DJ0	1	INPUT(46)	Modified julian date of state epoch
I	DJ1	1	INPUT(37)	Modified julian date of liftoff epoch
I	DOBS	10,2	OBSIT(1)	Velocity of the tracking stations due to the Earth's rotation
I	DX	3	STATE(4)	Spacecraft's velocity vector
I	HR	1	INPUT(53)	Hour of state epoch
I	KWTDOP	1	CNTRL(10)	Doppler first pass flag. Zero on first pass
I	OBSLAT	10	INPUT(480)	Latitudes of the tracking stations
I	OBSLON	10	INPUT(410)	Longitudes of the tracking stations
I	SEC	1	INPUT(55)	Seconds of state epoch
I	T	1	STATE(10)	Seconds since state epoch
I	TBO	1	INPUT(383)	Burnout time of engine 1.
I	THRUST	1	STATE(33)	Engine thrust

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	TIG	1	INPUT(380)	Ignition time of engine 1.
I	X	3	STATE(1)	Spacecraft's position vector
I	XMIN	1	INPUT(54)	Minutes of state epoch
I	XOBS	10,3	OBSIT(21)	Vectors from center of Earth to tracking stations in Earth equator and Greenwich
I	UJT	1	STATE(32)	Current modified julian date
I	W	1	STATE(35)	Current spacecraft mass

Description:

The spacecraft's velocity with respect to a rotating Earth must be determined in order to calculate the velocity away from a tracking station. Subroutine M50EPM is used to determine the transformation to the Earth equator and Greenwich. The vector from the observation site to the spacecraft is established from

$$\overline{XOB} = \overline{XE} - \overline{XOBS} \tag{1}$$

where $\overline{XE}$ is the position vector of the spacecraft in the Earth equator and Greenwich, and

$\overline{XOBS}$ is the vector to the tracking site obtained from OBSIT common.

The spacecraft is visible from the tracking station if

$$\overline{XOB} \cdot \overline{XOB} > 0 \tag{2}$$

On the first pass through this subroutine (KWTDOP = 0), KSTAT (I), where I corresponds to the tracking station, is set to one if the spacecraft is visible. This array is used to output the information only for the visible tracking stations. The velocity relative to the tracking station is obtained from

$$\bar{V} = \overline{DXE} - \overline{DOBS} \tag{3}$$

where $\overline{DXE}$ is the velocity of the spacecraft in the Earth equator and Greenwich, and

$\overline{DOBS}$ is the velocity of the tracking station from OBSIT common.

Note: The z-component of $\bar{V}$ is equal to z-component of $\overline{DXE}$ since the z-component of $\overline{DOBS}$ is zero.

Finally, the velocity away from the tracking station is determined from

$$\overline{RDOT} = (\bar{V} \cdot \overline{XOB}) \; / \; |\overline{XOB}| \tag{4}$$

and the doppler shift is obtained from

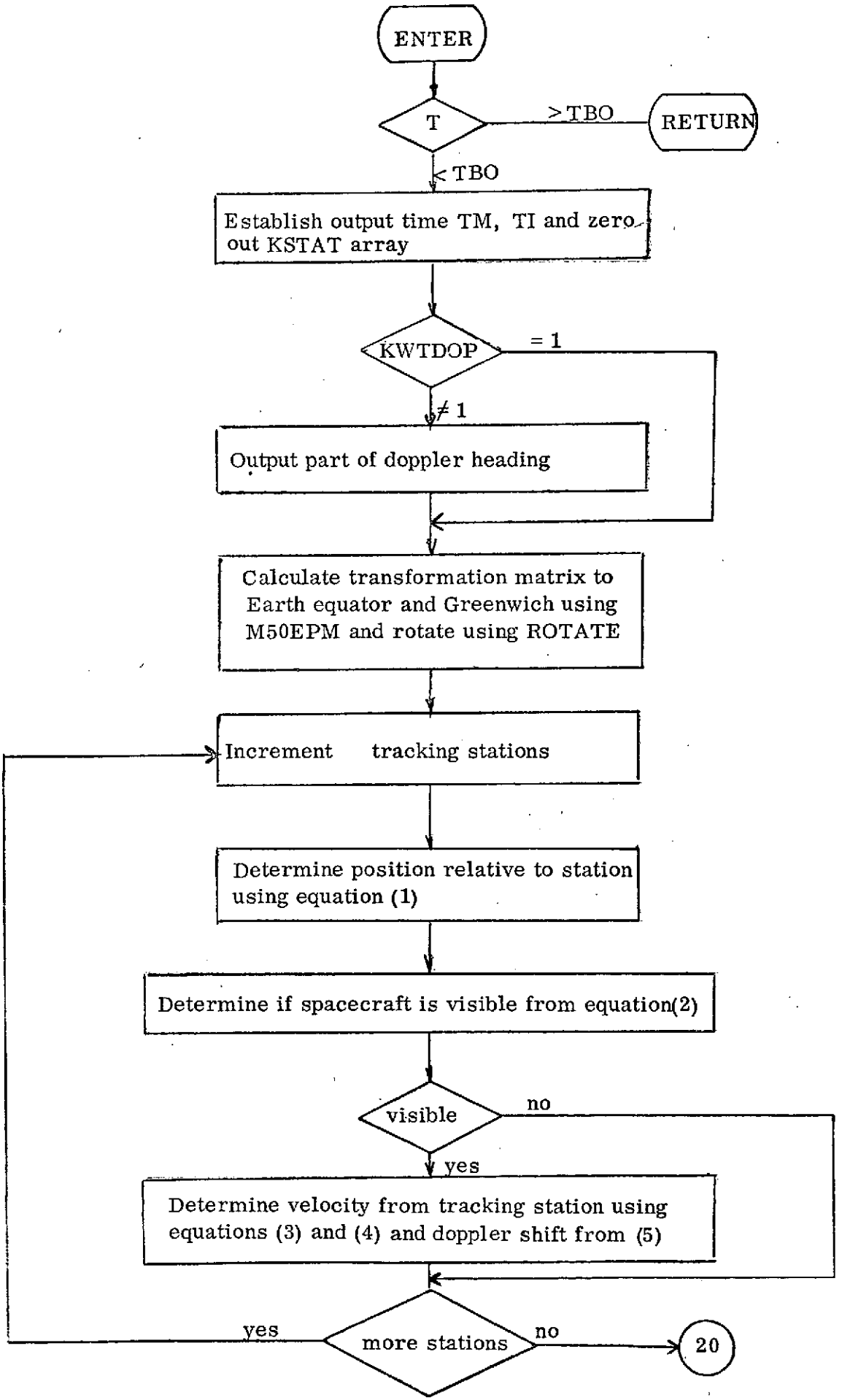
$$DOP = (CAR2^2 - CAR1^2) \; RDOT \; / \; C \tag{5}$$

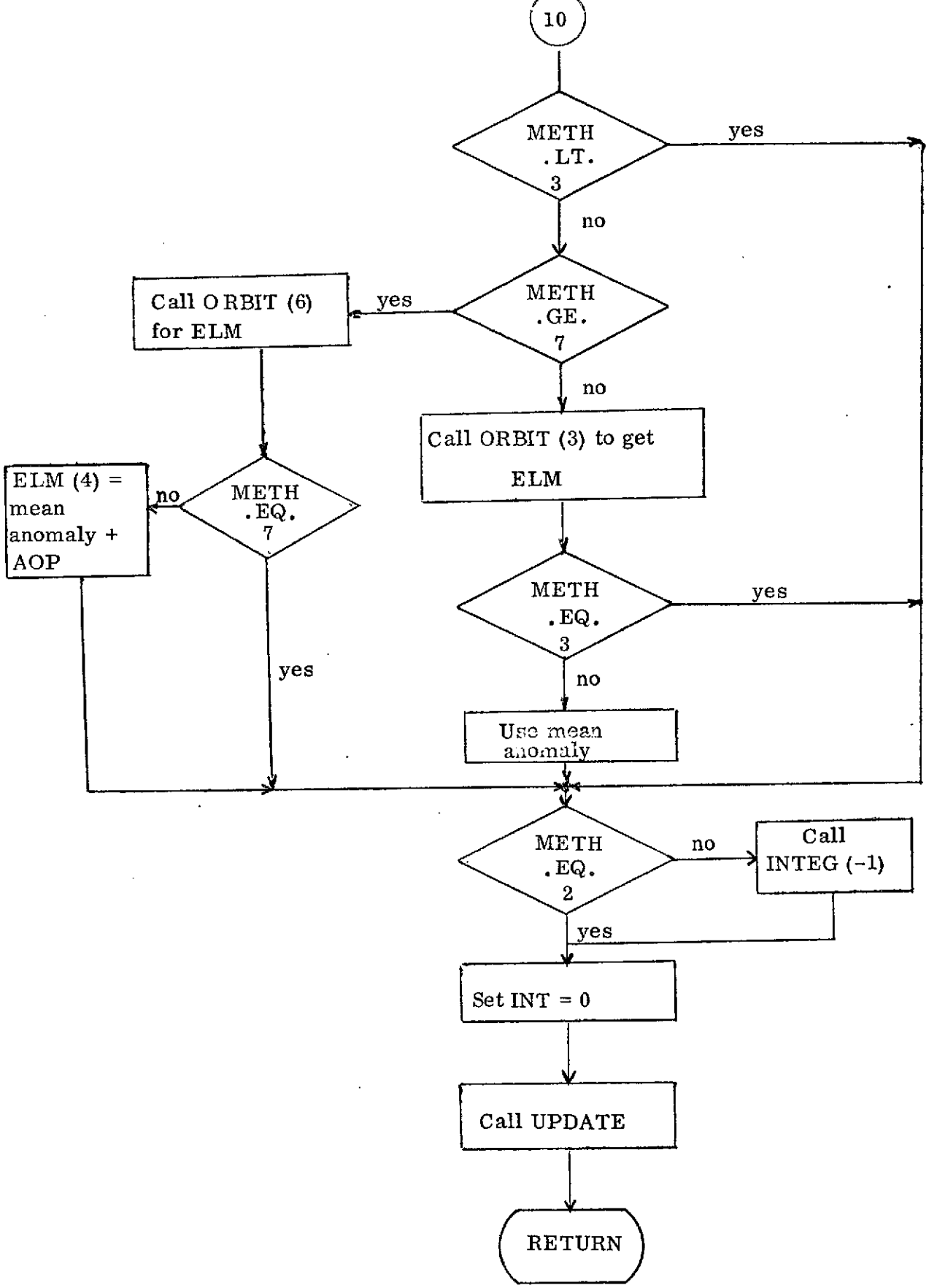
where CAR1 and CAR2 are the spacecraft carrier frequencies, and

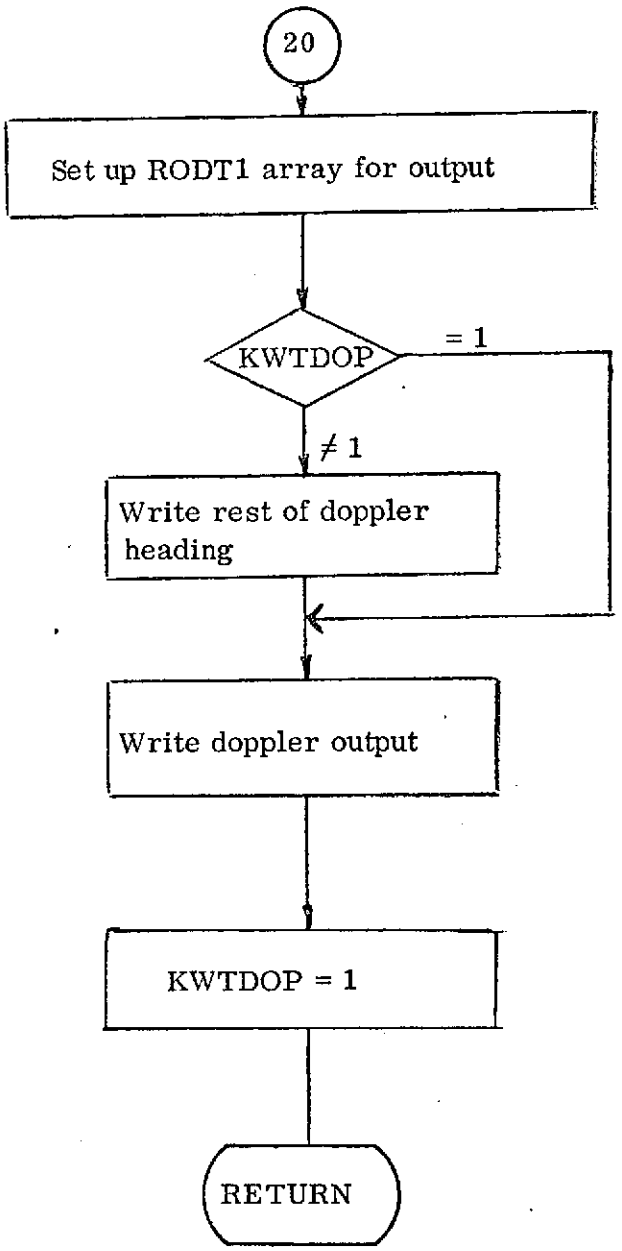
C is the velocity of light

The spacecraft's velocity away from each of the visible tracking stations and the corresponding doppler shift along with the current thrust and mass are output on unit 6. The KWTDOP flag is used to control the writing of the heading for the writes.

SUBROUTINE DOPLER




10
METH .LT. 3
yes
no
METH .GE. 7
yes
Call ORBIT (6) for ELM
no
Call ORBIT (3) to get ELM
METH .EQ. 7
no
ELM (4) = mean anomaly + AOP
yes
METH .EQ. 3
yes
no
Use mean anomaly
METH .EQ. 2
no
Call INTEG (-1)
yes
Set INT = 0
Call UPDATE
RETURN


20
Set up RODT1 array for output
KWTDOP
= 1
≠ 1
Write rest of doppler heading
Write doppler output
KWTDOP = 1
RETURN

FUNCTION DOT

Calling Sequence: Z = DOT (X, Y)

Purpose: This function performs the vector dot product

Common Blocks Required: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	DOT	1	FUNCTION NAME	Vector dot product
I	X	3	CALLING OPERAND	Input vector X
I	Y	3	CALLING OPERAND	Input vector Y

Description:

The vector dot product is obtained from

$$DOT = X_1 Y_1 + X_2 Y_2 + X_3 Y_3$$

where

X_i, Y_i $i = 1,3$ are the components of the X and Y vectors, respectively

SUBROUTINE DRAG

Calling Sequence: CALL DRAG

Purpose: This subroutine calculates the spacecraft acceleration due to atmospheric drag.

Common Blocks Required: CNTRL, CONST, GRAVTY, INPUT, PERT, PLNET.

Subroutines Required: ATMO, VNORM.

Inputs/Outputs

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DST	12	PLNET (73)	The JC element is the spacecraft distance from the central planet.
I	JC	1	CNTRL (07)	Central planet number
I	KATMOS	1	INPUT (1097)	Drag flag
I	POS	3	GRAVTY (1)	Position vector from central planet.
I/O	RCART	3	PERT (1)	Spacecraft perturbing acceleration
	RE	12	CONST (17)	Equatorial radius of the planet

Description:

This subroutine determines the acceleration due to atmospheric drag. The magnitude of the acceleration is obtained from:

$$a = \frac{1}{2} \rho V^2 C_D (10)^6$$

where ρ is the density in g/CM^3

V is spacecraft velocity in km/sec

C_D is the drag-area-mass coefficient

and the multiplier $(10)^6$ is required to convert units. The acceleration is applied along the velocity vector. The density is obtained from the function ATMOS while the drag coefficient is set at .03.

FUNCTION DVMAG

Calling Sequence: Y = DVMAG (X)

Purpose: This function determines the magnitude of an input vector

Common Blocks Required: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	DVMAG	1	FUNCTION NAME	Magnitude of the input vector
I	X	3	CALLING OPERAND	Input vector

Description:

The magnitude of a vector is determined from

$$DVMAG = \sqrt{X_1^2 + X_2^2 + X_3^2}$$

where

X_i $i = 1, 3$ are the components of the input vector X

SUBROUTINE EQNS

Calling Sequence: CALL EQNS

Purpose: This subroutine calculates the derivatives of the variables being numerically integrated.

Common Blocks Required: CNTRL, CONST, GRAVTY, INPUT, INTVAR, INTVRX, PERT, STATE.

Subroutines Required: ACCEL, AVEQNS, ORBIT, GRAV

Input / Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	DJO	1	INPUT(46)	Julian date of state epoch
I	GM	12	CONST(5)	Gravitational constants
I	KP	12	INPUT(1001)	Planets in the system
I	METH	1	INPUT(1013)	Trajectory propagator indicator
O	RATES	6	INTVAR(8)	Derivatives of integration variables
I	RCART	3	PERT(1)	Disturbing acceleration
I	X	1	INTVAR(1)	Current independent variable
I	Y	6	INTVAR(2)	Current dependent or integration variables
I	UJT	1	STATE(32)	Current julian date

Theory:

This subroutine calculates the derivatives of the integration variables at the current time when propagating the state using any of the following trajectory propagation methods,

1. Cowell
2. Encke
3. NICE/True
4. NICE/Mean
5. NICE/e cos ω, e sin ω.

The equations which define the derivatives of each of these methods are presented below.

Cowell

Let the origin of the cartesian coordinate system be located at the central planet of mass M. The disturbing planets are denoted by M_i, and the spacecraft by m. Then the perturbing acceleration becomes,

$$\vec{R} = \sum_{i=1}^{n} GM_i \left[\frac{\bar{X}'_i - \bar{X}_i}{\rho^3} - \frac{\bar{X}'}{|\vec{X}'|^3} \right] + \bar{a}_{SP} + \bar{a}_T + \bar{a}_{OB} \qquad (1)$$

where

$\bar{X}'$ is the vector from the central planet to the disturbing planet,

$\bar{X}$ is the vector from the central planet to the spacecraft,

$\bar{a}_{SP}$ is the acceleration due to solar pressure,

$\bar{a}_T$ is the acceleration due to engine thrusting,

$\bar{a}_{OB}$ is the acceleration due to an oblate planet, and

ρ is $|\bar{X}' - X|$

The total acceleration acting on the spacecraft is the sum of the perturbing acceleration and the acceleration due to the central planet; thus the total acceleration becomes,

$$\bar{a} = \frac{GM\bar{X}}{r^3} + \bar{R} \qquad (2)$$

The above vector equation denotes the acceleration, which is numerically integrated to obtain the velocity. The velocity is numerically integrated to obtain the position.

Encke's Method

In this method an attempt is made to utilize the knowledge that the motion is very nearly two-body with respect to the central planet. Thus, only the motion which deviates from the two-body motion is integrated. This motion is added to the two-body motion to obtain the position and velocity of the spacecraft.

Let $\bar{X}_0$ be the position vector of the spacecraft obtained from two-body motion, and $\bar{\xi}$ be a vector describing the deviation from the two-body orbit. The position vector of the spacecraft is then obtained by,

$$\bar{X} = \bar{X}_0 + \bar{\xi} \, . \tag{3}$$

The acceleration of the disturbing vector, $\bar{\xi}$, is obtained as follows:
If the quantity f is defined as

$$f = 1 - (1 + 2q)^{-3/2} \tag{4}$$

where

$$q = \left[\sum_{i=1}^{3} \left(\vec{X}_{0_i} + \frac{1}{2} \vec{\xi}_i \right) \vec{\xi}_i \right] \left(\frac{1}{|\vec{X}_0|^2} \right) \tag{5}$$

then the disturbing acceleration is

$$\vec{\xi} = \frac{\mu}{|\vec{X}_0|^3} \left(f \bar{X} - \bar{\xi} \right) + \bar{R} \tag{6}$$

where $\vec{R}$ is obtained from Equation (1). This vector is numerically integrated to obtain the deviation from the two-body orbit. The reference orbit is updated (rectified) whenever

$$\frac{|\vec{X}|^2}{|\vec{X}_0|^2} > 0.001 \tag{7}$$

These equations are much more complicated than Cowell's equations. Also, to obtain the position along the reference orbit, one must calculate the true anomaly from the mean anomaly. This involves an iterative solution and is time consuming.

NICE Methods

The three NICE methods involve the numerical integration of the classical orbital elements to obtain the orbit of the spacecraft as a function of time.

The orbital elements integrated are,

1. semilatus rectum, p
2. eccentricity, e
3. true or mean anomaly, f or M
4. argument of perigee, ω
5. inclination, i
6. longitude of ascending node, Ω

or, alternately

1. p
2. e cos ω
3. e sin ω
4. f + ω
5. i
6. Ω

The derivatives of the above quantities are determined and numerically integrated to determine the instantaneous orbital elements. The derivatives of the orbital elements are derived in many texts and reports and only the results will be presented here. The derivatives of the orbital elements are

$$\dot{p} = \left(2\, r \sqrt{\frac{p}{\mu}}\right) C \tag{8}$$

$$\dot{e} = \sqrt{\frac{p}{\mu}} \left\{R \sin f + \frac{r}{p}\left[2 \cos f + e\,(1 + \cos^2 f)\right] C\right\}$$

$$\dot{\omega} = \sqrt{\frac{p}{\mu}} \left\{ \frac{\sin f}{e} \left(1 + \frac{r}{p}\right) C - \left(\frac{\cos f}{e}\right) R - \left(\frac{r}{p} \sin u \cot i\right) W \right\}$$

$$\dot{\Omega} = r \sin u \, W \Big/ \left(\sin i \sqrt{\mu p}\right)$$

$$\dot{i} = \left(\frac{r}{\sqrt{p\mu}} \cos u\right) W$$

$$\dot{f} = \frac{\sqrt{\mu p}}{r^2} + \frac{1}{e} \sqrt{\frac{p}{\mu}} \left\{ \cos f \, R - \sin f \left(1 + \frac{r}{p}\right) C \right\}$$

$$\dot{M} = n + \sqrt{p \frac{(1 + e^2)}{\mu}} \left\{ \left(\frac{\cos f}{e} - \frac{2r}{p}\right) - R \frac{\sin f}{C} \left(1 + \frac{r}{p}\right) C \right\} \quad (- \text{ for } e >$$

$$(\dot{e} \sin\omega) = \sqrt{\frac{p}{\mu}} \left\{ - \cos\mu \, R + \left[\left(1 + \frac{r}{p}\right) \sin u + e \frac{r}{p}\right] C + \frac{r}{p} \, e \cos\omega \, \sin u \cot i \, W \right\}$$

$$(\dot{e} \cos) = \sqrt{\frac{p}{\mu}} \left\{ \sin u \, R + \left[\left(1 + \frac{r}{p}\right) \cos u + e \frac{r}{p} \sin\omega\right] C + e \frac{r}{p} \sin u \cot i \, W \right\}$$

$$\dot{u} = \frac{\mu}{r^2} \sqrt{\frac{p}{\mu}} - \frac{r}{p} \sqrt{\frac{p}{\mu}} \sin u \cot i \, W$$

where n is the mean motion

μ is the gravitational potential

$u = \omega + f$

r is the radius

R, C, W are the perturbing accelerations

The perturbing accelerations are written with respect to the orbit plane. They are in the radial direction, circumferential direction and normal to the orbit plane. These accelerations are obtained from the perturbing acceleration derived in Equation (1) by rotation to the orbit plane, as

$$\begin{Bmatrix} R \\ C \\ W \end{Bmatrix} = [A] \begin{Bmatrix} R_1 \\ R_2 \\ R_3 \end{Bmatrix} \tag{9}$$

where

$$[A] \begin{bmatrix} \cos\Omega\cos u - \sin\Omega\sin u\cos i & \sin\Omega\cos u + \cos\Omega\sin u\cos i & \sin u\sin i \\ -\cos\Omega\sin u - \sin\Omega\cos u\cos i & -\sin\Omega\sin u + \cos\Omega\cos u\cos i & \cos u\sin i \\ \sin\Omega\sin i & -\cos\Omega\sin i & \cos i \end{bmatrix} \tag{10}$$

The difference between the NICE/Mean and NICE/True methods is that the NICE/Mean method uses the mean anomaly equation of (8) while in the NICE/TRUE method, the true anomaly is used. When the last of the NICE methods are used, the last 3 equations of equation (8) are used instead of the equations for $\dot{e}$, $\dot{\omega}$ and $\dot{f}$.

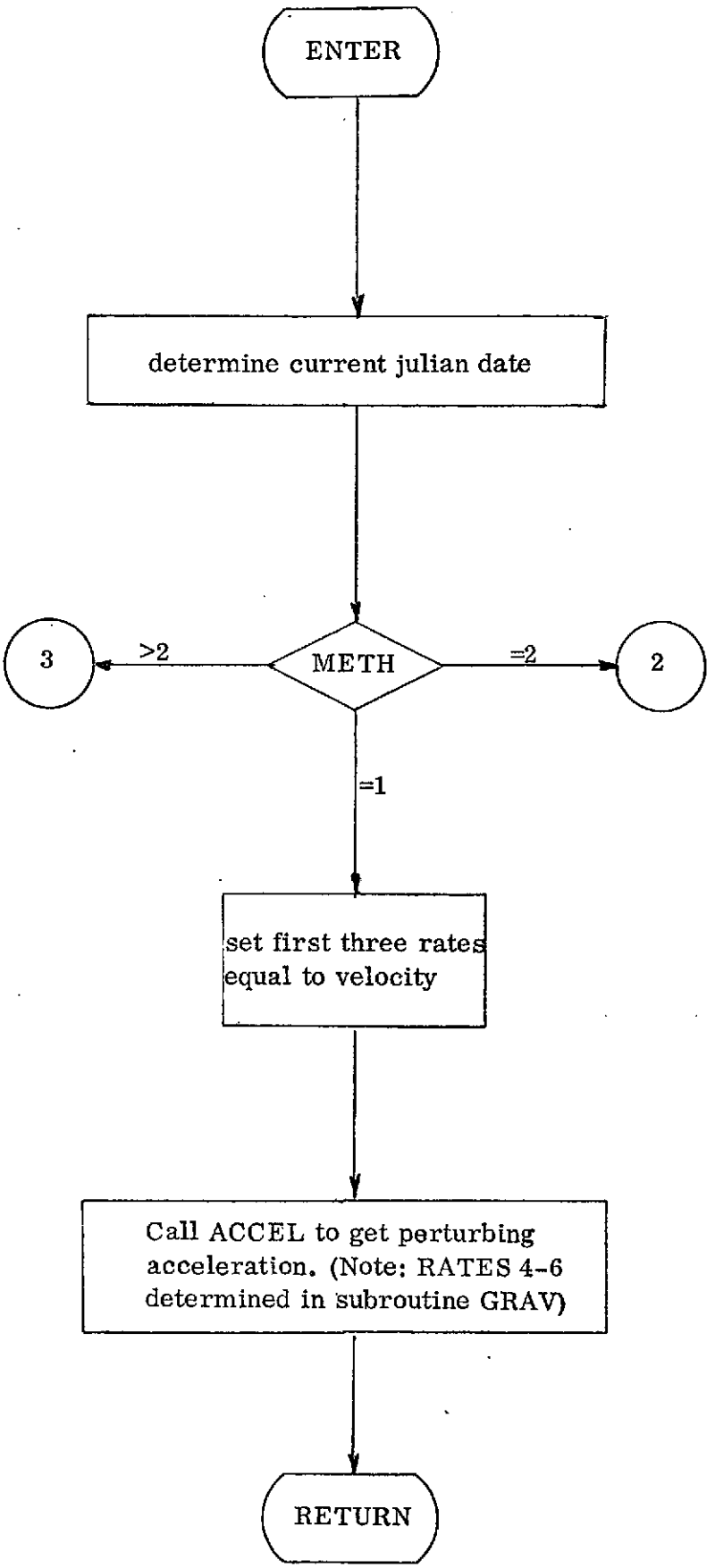
Description:

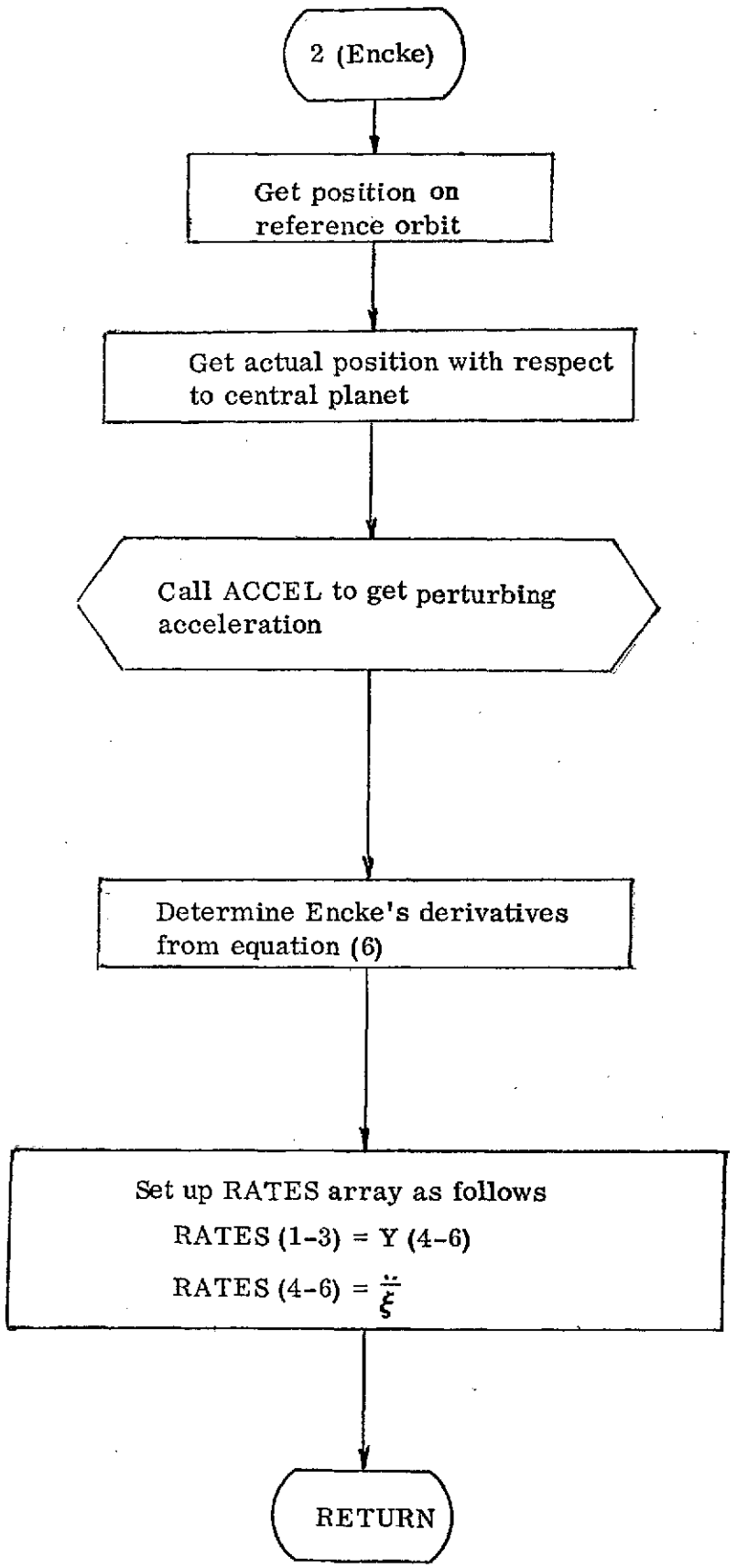
The derivatives of the integration variables are as described in the above equations. The set of equations used is determined by the METH flag as follows:

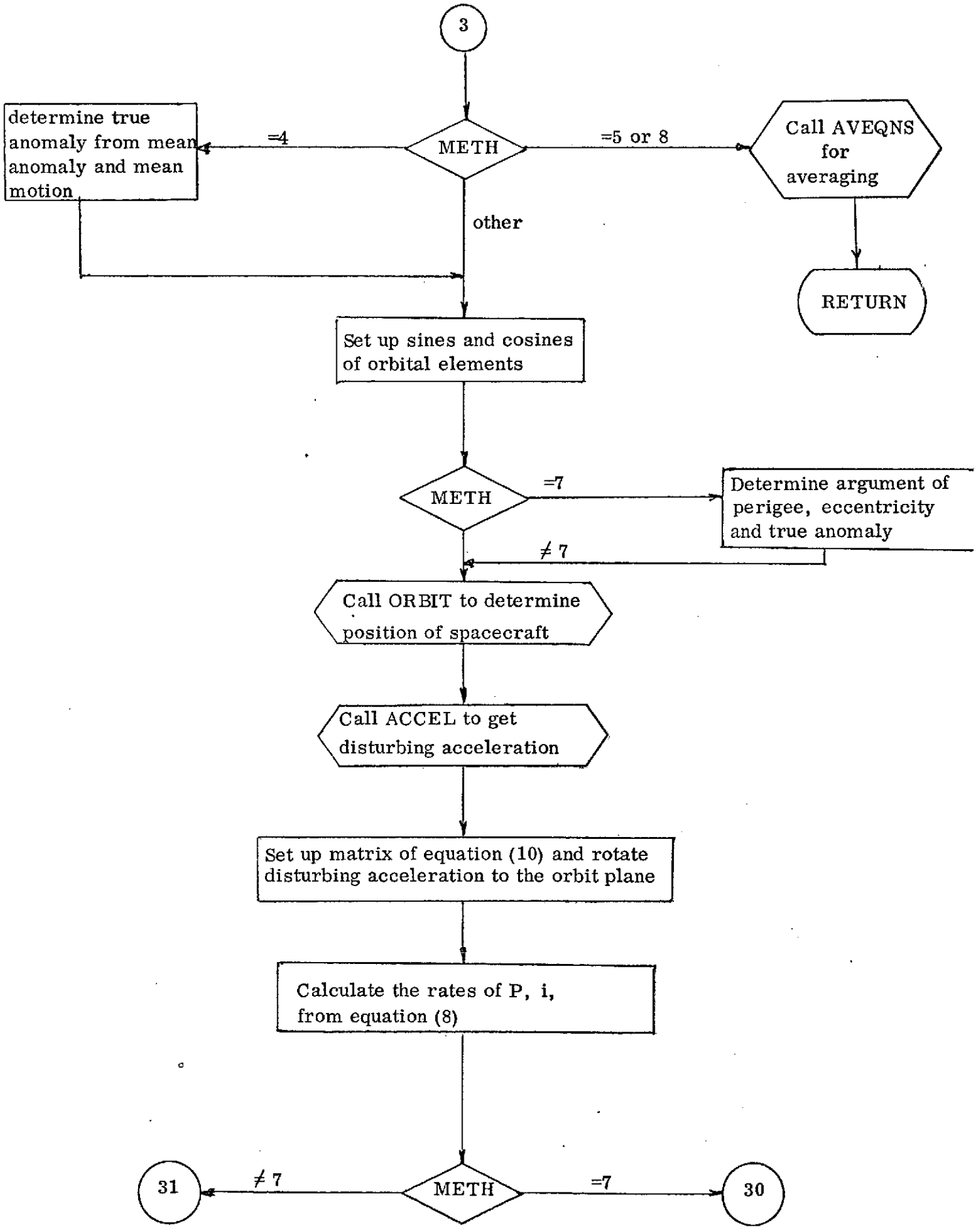
METH = 1. Cowell
2. Encke
3. NICE/True
4. NICE/Mean
7. NICE/ e sin ω, e cos ω.

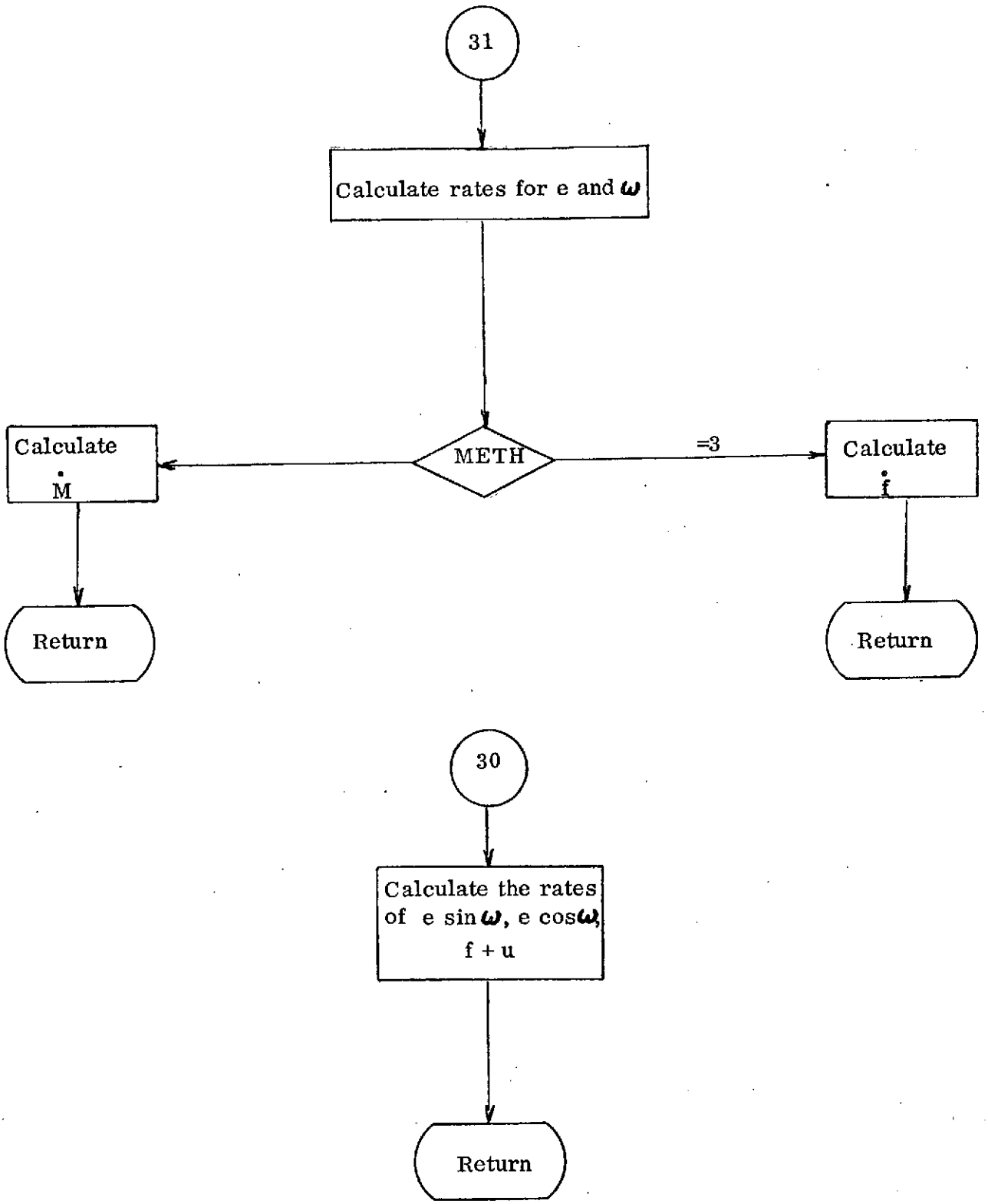
The current set of integration variables and the independent variable, time, is brought into the subroutine via INTVAR common. This common block is initiated in subroutine INTEG. The Encke reference orbit is input via PERT common. The disturbing acceleration is calculated in subroutine ACCEL and transferred to EQNS via PERT common.

The logic flow consists of testing the METH flag to determine which set of equations to calculate. The derivatives are determined in a straightforward manner using the equations described in the theory.


ENTER
determine current julian date
3
>2
METH
=2
2
=1
set first three rates equal to velocity
Call ACCEL to get perturbing acceleration. (Note: RATES 4-6 determined in subroutine GRAV)
RETURN


2 (Encke)
Get position on reference orbit
Get actual position with respect to central planet
Call ACCEL to get perturbing acceleration
Determine Encke's derivatives from equation (6)
Set up RATES array as follows
RATES (1-3) = Y (4-6)
RATES (4-6) = $\ddot{\xi}$
RETURN


3
determine true anomaly from mean anomaly and mean motion
=4
METH
=5 or 8
Call AVEQNS for averaging
other
RETURN
Set up sines and cosines of orbital elements
METH
=7
Determine argument of perigee, eccentricity and true anomaly
≠ 7
Call ORBIT to determine position of spacecraft
Call ACCEL to get disturbing acceleration
Set up matrix of equation (10) and rotate disturbing acceleration to the orbit plane
Calculate the rates of P, i, from equation (8)
31
≠ 7
METH
=7
30


31
Calculate rates for e and ω
METH
=3
Calculate
Ṁ
Calculate
ḟ
Return
Return
30
Calculate the rates
of e sin ω, e cos ω,
f + u
Return

SUBROUTINE FIELD2

Calling Sequence: CALL FIELD2

Purpose: FIELD2 uses the chain rule to calculate the partial derivatives of the gravitational potential.

Common Blocks Required: CNTRL, CONST, FIELDM, GRAVTY, INPUT, INTVAR, PERT, STATE

Subroutines Required: M50JPM, ROTATE, SPNM

Reference: Gulick, L.J., "A Comparison of Methods for Computing Gravitational Potential Derivative," ESSA TECHNICAL REPORT C & GS 40, 1970.

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	CC	16,17	FIELDM(17)	Tesseral coefficients
I	GM	12	CONST(5)	Gravitational constants
I	JC	1	CNTRL(7)	Central planet number
I	KVAR	1	INPUT(1096)	Gradient flag
I	NMOD	1	FIELDM(298)	Number of zonals
I	POS	3	GRAVTY(1)	Position of S/C
I	RE	12	CONST(17)	Equatorial radii
I	RCART	3	PERT(1)	Acceleration of S/C
I	SELNEQ	3,3	FIELDM(289)	Transformation matrix
I	TIME	1	INTVAR(1)	Time since epoch

I	UJT	1	STATE(32)	Modified Julian date
I	WP	12	CONST(29)	Planet spin rates
I	XJ	16	FIELDM(1)	Zonal coefficients

Theory:

The gravitational potential of the central planet in terms of spherical harmonics may be expressed as

$$V = \frac{GM}{r} \left\{ 1 + \sum_{n=1}^{\infty} \left(\frac{R}{r}\right)^n \sum_{m=o}^{n} P_n^m (\sin\beta)\, x \right.$$

$$\left. \left(C_n^m \cos(m\lambda) + S_n^m \sin(m\lambda)\right) \right\}$$

where GM is the gravitational constant, R is the equatorial radius of the central planet, C_n^m and S_n^m are the coefficients representing the mass distribution, $P_n^m(\sin\beta)$ is the associated Legendre polynomial of degree n and order m, and β, λ, and r are the body-fixed latitude, longitude, and radius of the point where the disturbing force is to be evaluated.

The force due to this potential is the partial derivative of V with respect to inertial cartesian coordinates. One method of evaluating this force is by using the chain rule to evaluate the derivatives with respect to the body cartesian coordinates and then rotating to the integration (inertial) frame. This chain rule method incorporates the following recursion relationship for $P'^{m}_{n}(x)$

$$\frac{d\,P^{m}_{n}(x)}{dx} = -\frac{mxP^{m}_{n}(x)}{1-x^2} + \frac{P^{m+1}_{n}(x)}{(1-x^2)^{\frac{1}{2}}}$$

See Gulick (referenced above) for the derivation of this relationship and for further notes on the chain rule method.

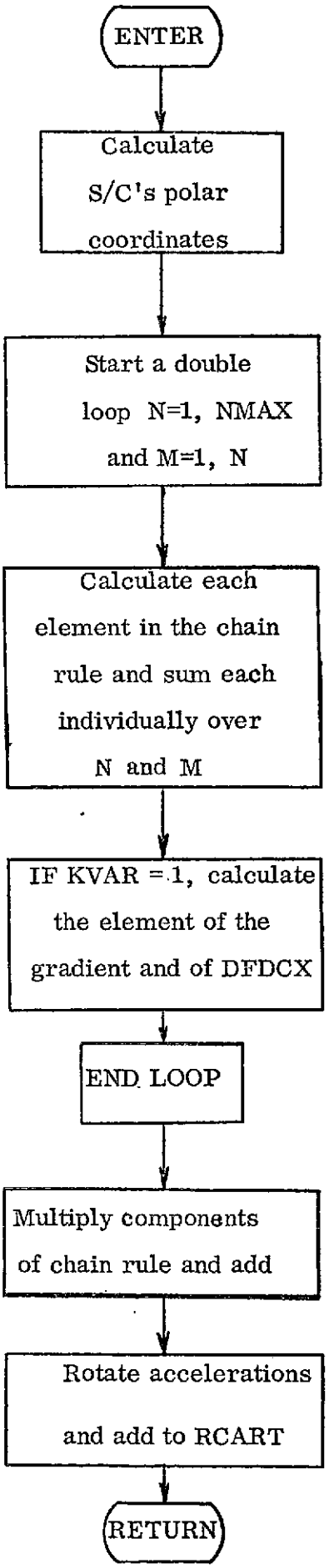
Description:

The main purpose of FIELD2 is to calculate the partial derivatives of the gravitational potential. These derivatives are used to evaluate the acceleration of the S/C at any given position. To use FIELD2 for this purpose, KVAR should be set to zero. FIELD2 will calculate the acceleration of the S/C due to the central planet using the input field (CC and XJ), the position of the S/C, and the time. The resulting acceleration vector will be rotated to the integration frame using the transformation matrix SELNEQ, and then added to RCART.

In addition to this function, FIELD2 may also be used to calculate the gradient of the force field and the partial derivatives of the force with respect to the spherical harmonic coefficients of the field. To use this mode of FIELD2 , KVAR must be set to one, in which case the acceleration (described above) will still be output, along with the matrices D2VDX2 and DFDCX.

The symmetric matrix D^2VDX^2 contains the second partials of the potential with respect to the body-centered cartesian coordinates. For a description of the use of this Jacobian matrix in linear variational theory, see subroutine SHIMMY.

The matrix DFDCX contains the explicit partial derivatives of the force with respect to the NB different harmonic coefficients to be estimated. The maximum value of NB is 100. The input vector IX determines which coefficients are to be studied, and must be set up in the following way. The i^{th} element of IX indicates the i^{th} coefficient to be studied, and IX(i) must be equal to the position of the i^{th} coefficient in FIELDM common. The order of the values in IX is crucial since the derivatives are calculated in the same order as the coefficients are used in FIELD2. This order is $((C_{ij}, S_{ij}, j = 0, i), \quad i=1, NMOD)$. The element C_{ij} occupies position $16 \times j + i$ of FIELDM common and S_{ij} occupies position $(i+1) \times 16 + j$. Thus, if C_{20}, C_{22}, C_{31}, C_{32}, C_{41}, S_{21}, S_{32}, S_{33} are the coefficients to be studied, then NB = 8 and IX would be 2, 49, 34, 19, 35, 66, 67, 20 since the calculations would be made in the order C_{20}, S_{21}, C_{22}, C_{31}, C_{32}, S_{32}, S_{33} and C_{41}. If C_{33} were to be studied also,then NB would be 9 and 51 would have to be inserted between 66 and 67 in the vector IX. The order that the derivatives are output in DFDCX is the same as the order in IX.


ENTER
Calculate S/C's polar coordinates
Start a double loop N=1, NMAX and M=1, N
Calculate each element in the chain rule and sum each individually over N and M
IF KVAR =1, calculate the element of the gradient and of DFDCX
END LOOP
Multiply components of chain rule and add
Rotate accelerations and add to RCART
RETURN

SUBROUTINE FIND

Calling Sequence: CALL FIND (IDSAT, ISET, $)

Purpose: This subroutine reads a file from the GTDS 24-hour hold file to retrieve the state from the GTDS program.

Common Blocks Required: ELMNT

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	IDSAT	1	CALLING OPERAND	Satellite identification number
I	ISET	1	CALLING OPERAND	Element set number of desired data
O	$	-	-	Statement number to transfer to if error return.

Description:

This subroutine is used to retrieve the state and covariance matrix from the 24-hour hold file written by the GTDS program. The data is read from unit 26 using a direct read. The record number used in the direct read is determined from the element set number brought in via the argument list and variables defining the 24-hour hold file obtained from a read to unit 1.

The data is read into a working array and later transferred to the SET array of ELMNT common. ELMNT common is used to transfer the data to other subroutines in MAESTRO. The SET array is defined as follows:

Location	Definition
1	Date of state in year, month and day written as YYMMDD.
2	Time of state in hours, minutes and seconds written as HHMMSS.SSS.

Location	Definition
3-5	Cartesian position vector.
6-8	Cartesian velocity vector.
9-14	Keplerian orbital elements.
15-35	Upper triangle of the state covariance matrix.
36	Start time of fitted data, (year, month, day)
37	Start time of fitted data, (hour, minute, second)
38	End time of fitted data, (year, month, day)
39	End time of fitted data, (hour, minute, second)
40	Root mean square of fit
41	Satellite identification number
42	Reference coordinate system of state
43	Central body indicator
44	Element set number

SUBROUTINE FIXATG

Calling Sequence: CALL FIXATG

Purpose: FIXATG controls the fixed-attitude guidance logic.

Common Blocks Required: CNTRL, CONST, INPUT, MCCOM, STATE

Subroutines Required: FOWARD, SENSO, POST

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	JC	1	CNTRL(7)	Central body number
I	UT	1	CONST(15)	GM of the Moon (km^3 / sec^2)
I	TFINAL	1	INPUT(4)	Trajectory stop time (sec)
I	WO	1	INPUT(38)	Initial weight of the spacecraft (kg)
I	RAI	1	INPUT(47)	Central value of right ascension, (deg)
I	DECI	1	INPUT(48)	Central value of declination (deg)
I	ASPMC	1	INPUT(441)	Specific impulse of the midcourse engine (sec)
I	WRETRO	1	INPUT(443)	Weight of retro-fuel (kg)
I	WDROP	1	INPUT(473)	Drop-weight of retro (kg)
I	CONE	1	INPUT(474)	Step-size for attitude (deg)
I	DINK	1	INPUT(479)	Midcourse velocity step (km/sec)
O	KRASH	1	INPUT(1032)	Trajectory stop-type key
I	JRA	1	INPUT(1041)	Number of right ascension steps
I	JDEC	1	INPUT(1042)	Number of declination steps
I	KOUT9	1	INPUT(1058)	Logical unit number for scope output
I	KTF	1	INPUT(1077)	Number of velocity steps (negative)
O	XMC	6	MCCOM(6)	Midcourse pre-ignition state (km, km/sec)

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	DV	3	MCCOM(12)	Midcourse velocity impulse (km/sec)
I	TMCS	1	MCCOM(18)	Midcourse time (sec after anchor epoch)
I	DVRET	1	MCCOM(25)	Retro velocity impulse (km/sec)
I	DJDIF	1	MCCOM(39)	Anchor-launch epoch difference (sec)
I	WTF	1	MCCOM(47)	Weight after midcourse burn (kg)
I	PSID	10	MCCOM(80)	Desired end constraints, except PSID(7) is the central value of impulse magnitude (km/sec)
I	PSI	10	MCCOM(100)	Constraint error vector
O	IR	1	MCCOM(158)	Return key for SENSO
O	KDV	1	MCCOM(161)	Counter for delta-V steps taken
O	ICB	1	MCCOM(165)	Midcourse central body number
I	X	6	STATE(1)	State vector (km, km/sec)
I	T	1	STATE(10)	Time (sec)
I	ATT	3	STATE(11)	Unit thrust (ΔV) vector (equator, equinox of 1950.0)

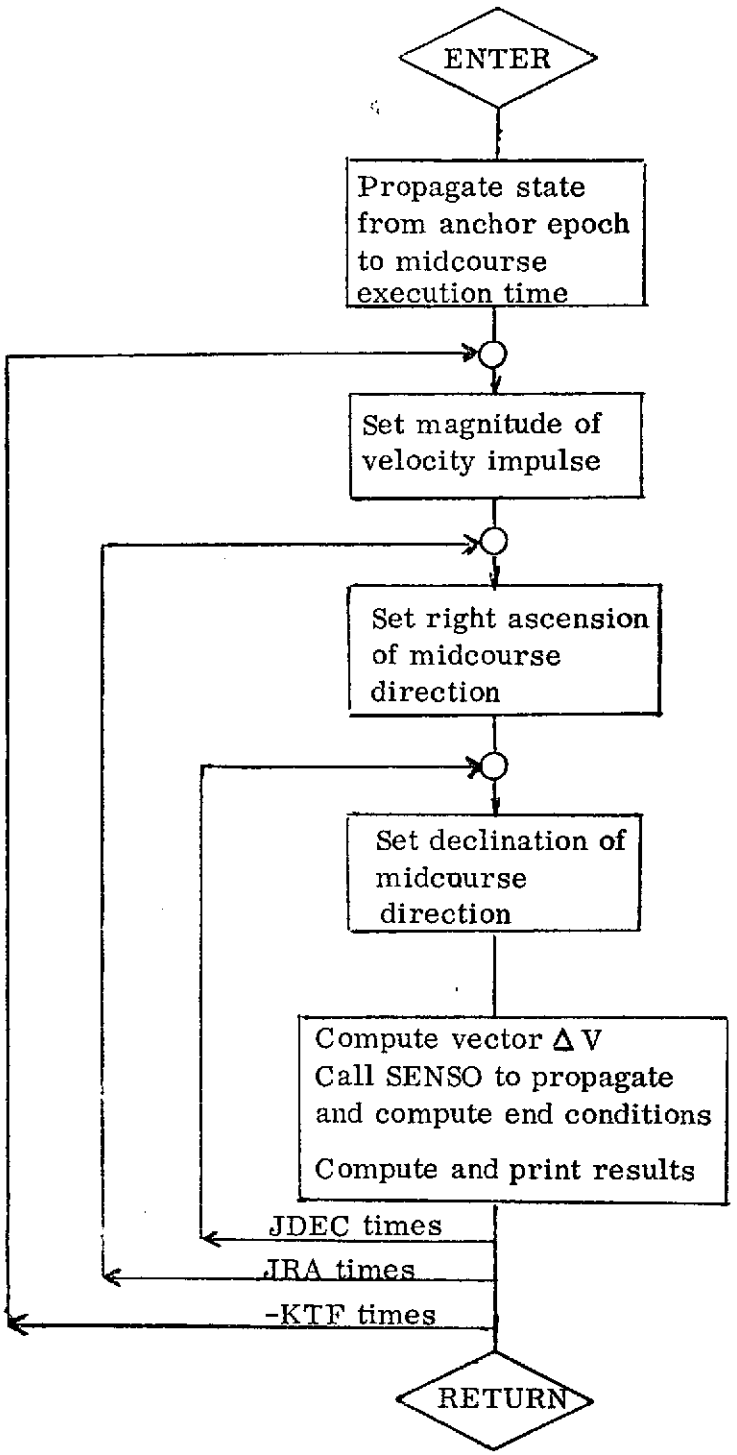
Description:

FIXATG varies the velocity impulse magnitude to scan the end conditions achievable with a midcourse burn of fixed thrust direction. The direction may also be systematically varied to ascertain the effects of attitude errors. FIXATG is called by PROTO within a loop in which midcourse execution (ignition) time is varying, but, for any particular entry, ignition time is fixed. The anchor vector state is first propagated to midcourse time by a call to FOWARD. The pre-midcourse state is saved in XMC. SENSO is then called to propagate the state through the burn and on to the target, then to compute end conditions (TARGET) from the arrival state. The following quantities are printed out at each step of the scan.

DVM	Midcourse velocity impulse magnitude (m/sec)
RTA	Right ascension of the thrust (deg)
DEC	Declination of the thrust (deg)
RCA	Radius at closest approach (km)
INC	Inclination (deg)
TFLT	Time of flight to closest approach (hours past launch)
ROPA	Radius at opposite apsis (km, opposite RCA after retro)
FCP	Fuel to circularize at periapsis (kg)
TCF	Total correction fuel (kg)

Computation of the last three quantities assumes variable attitude for the retro and trim maneuvers.

SUBROUTINE FIXATG



SUBROUTINE FOWARD

Calling Sequence: CALL FOWARD (KSET)

Purpose: This subroutine establishes certain constants to propagate the state forward in time. Calls are made to subroutines which propagate the state

Common Blocks Required: CNTRL, CONST, INPUT, INTVAR, SAVE, SHAD, STATE

Subroutines Required: INTEG, MULCON, ORBIT, OUTPUT, PLANET, PRINT, TIMEC, TRMN

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	DSAD	3,5	SHAD(1)	Array of back distances to the shadow cone
I/O	DX	3	STATE(4)	Spacecraft's velocity vector
I	EJO	1	STATE(26)	Ephemeris date corresponding to state epoch
O	ELM	6	STATE(14)	Osculating orbital elements to be integrated
O	KDIS	1	CNTRL(5)	Discontinuity flag
O	KFIRST	1	CNTRL(12)	First pass flag
O	KHALT	1	CNTRL(6)	Error return flag
I	KMETH	3	INPUT(1036)	Trajectory propagation indicator table
I	KOUT	1	INPUT(1030)	Output frequency flag
I	KSET	1	Calling Operand	If non-zero, more constants are initialized
O	RSAV	1	SAVE(41)	Last flight path angle used in closest approach calculation.
I	T	1	STATE(10)	Seconds since state epoch
O	TCA	1	STATE(29)	Time of closest approach

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	TOUTL	1	SAVE(40)	Last output time
O	TSAV	1	SAVE(7)	Time of saved state used to restore in TIMEC
O	TSAD	3	SHAD(3)	Times corresponding to the shadow distances in PSAD
I/O	X	3	STATE(1)	Initial position vector

Description:

This subroutine sets up constants to propagate the state in time, establishes the integration variables and calls the proper subroutine to propagate the state. The KHALT and KWTDOP flags are initialized to zero and METH set to KMETH (1). If the KSET flag is non-zero, the following constants are initialized:

1. TOUTL = T
2. DSAD and TSAD arrays to zero
3. KNTRL (1-6 and 8-10) to zero
4. KDIS and KFIRST to one
5. TCA to a large number

The integration array is established according to the trajectory propagation technique. If Cowell is to be used, METH=1, the position and velocity arrays in STATE are used as the integration variables. However, if any other method is used, the ELM array is set to the integration variables. Subroutines ORBIT and TRMN are used to establish the proper set of orbital elements when any of the "NICE" methods or averaging is used. After the integration array is established, subroutine INTEG is used to determine initial derivatives of the state. The initial state is output using subroutine OUTPUT if the output frequency flag is greater than 0. Finally, the state is propagated in time using subroutine TIMEC when numerical integration is desired at MULCON when the multi-conic algorithm is used.

SUBROUTINE GETTAP

Calling Sequence: CALL GETTAP

Purpose: This subroutine reads the ephemeris tape and sets up CETBL3 common for use in subroutine READE

Common Blocks Required: CETBL2, CETBL3, CETBL9, CNTRL

Subroutines Required: None

Input / Output

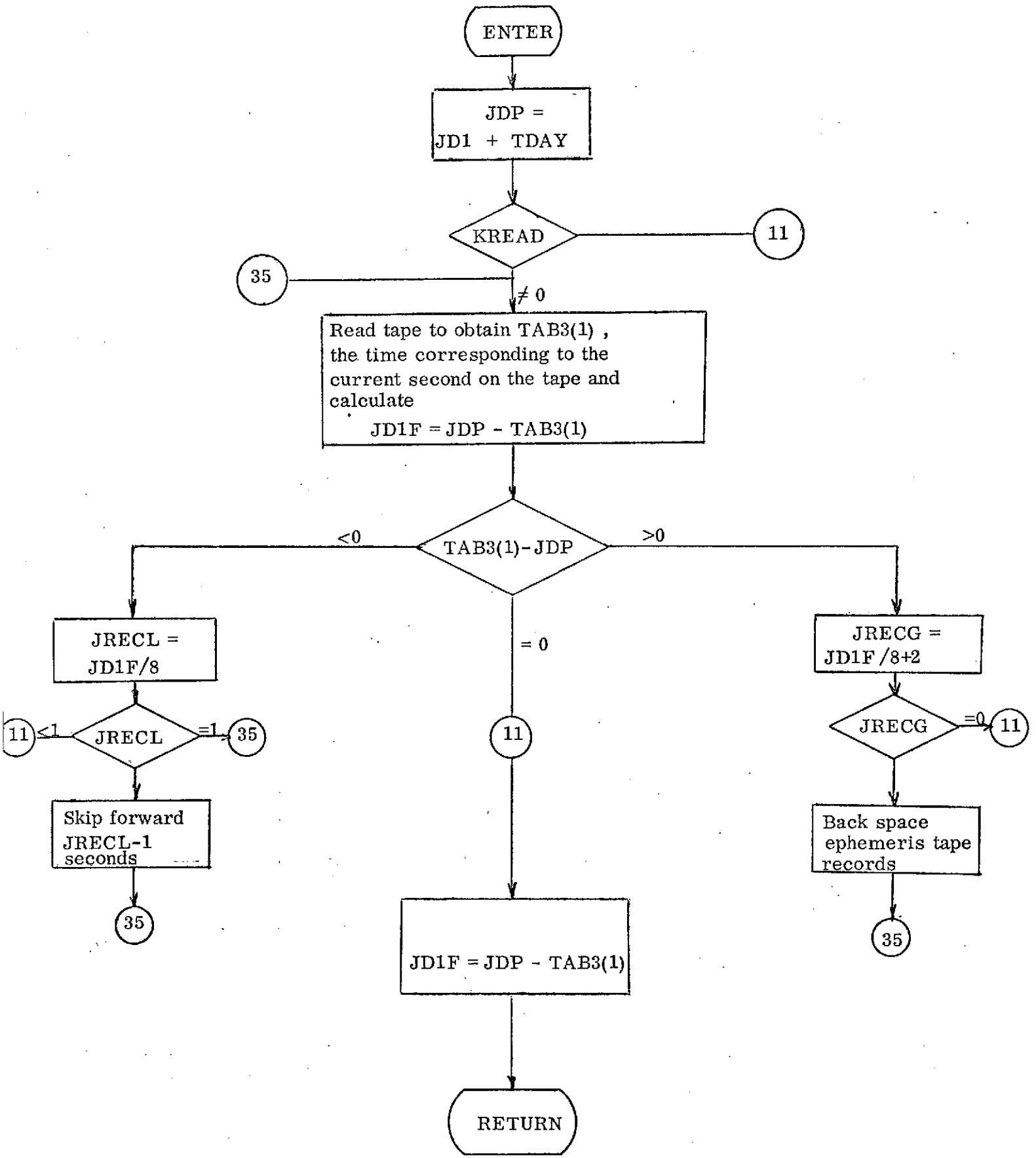
I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	ICW	1	CETBL2(1)	Flag indicating status of common block CETBL3
O	IERR1	1	CETBL9(4)	Error flag
O	JDIF	1	CETBL9(3)	Time from beginning of ephemeric block of data to current time
I	JD1	1	CETBL9(1)	Reference ephemeris julian date
I	KREAD	1	CNTRL(8)	Ephemeris tape read flag
O	NUT	204	CETBL3(830)	Nutation output
O	TAB3	829	CETBL3(1)	Planetary and Lunar ephemeris raw data from tape
I	TDAY	1	CETBL.9 (2)	Time from reference ephemeris Julian date

Description:

This subroutine reads the ephemeris tape and sets up the TAB3 and NUT arrays for use in subroutine READE. If the KREAD flag is zero, these arrays are already established and no read is performed. Most of the logic in this subroutine is involved in searching through the ephemeris tape to find the desired record. The time of the desired ephemeris data, JDP, is determined from the sum of TDAY and JD1. The next record of the ephemeris tape is read to establish its current time, TAB3(1). The difference between the current time and desired time is determined. If the difference is negative the tape

must be backspaced,while the tape is advanced when the difference is positive. The number of records to advance or backspace the tape is determined by dividing the difference by eight,since eight days of data are stored in each record. When the proper record is determined,JDIF and ICW are set and the subroutine terminates. Another version of GETTAP is available for use at Goddard Space Flight Center. This version uses the ephemeris data stored in disk form and the direct read feature is used to retrieve the data.

SUBROUTINE GETTAP



SUBROUTINE GRAV

Calling Sequence: CALL GRAV

Purpose: GRAV calculates the disturbing accelerations due to external bodies.

Common Blocks Required: CNTRL, CONST, GRAVTY, INPUT, INTVAR, PLNET, PERT

Subroutines Required: DVMAG

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
O	DST	12	PLNET (73)	Distance from central planet to other planets
I	GM	12	CONST(5)	Planet gravitational constants
I	JC	1	CNTRL(7)	Central planet number
I	JMN	1	INPUT(1017)	Ephemeris flag
I	KP	12	INPUT(1001)	Planets in system
I	METH	1	INPUT(1013)	Method of integration
I	POS	3	GRAVTY(1)	Spacecraft position
I/O	RATES	6	INTVAR(8)	Derivatives of integration variables
I/O	RCART	3	PERT(1)	Accelerations of S/C
I	XP	XP(6, 12)	PLNET(1)	Positions and velocities of planets in system

C 3

Theory:

The acceleration of a body in space due to the presence of another body in space is given by

$$\vec{A} = \frac{GM\,\vec{R}}{|\vec{R}|^3}$$

where GM is the gravitational constant and R is the vector from the first body to the second body.

The net acceleration of a S/C with respect to a central planet is given by the difference between the spacecraft's acceleration and the planet's acceleration.

$$\vec{A}_{s/c} = GM_j \left[\frac{\left(\vec{R}_j - \vec{R}_{s/c}\right)}{\left|\vec{R}_j - \vec{R}_{s/c}\right|^3} - \frac{\vec{R}_j}{\left|\vec{R}_j\right|^3} \right]$$

where j represents the jth planet, S/C represents spacecraft, and all vectors are with respect to the central planet.

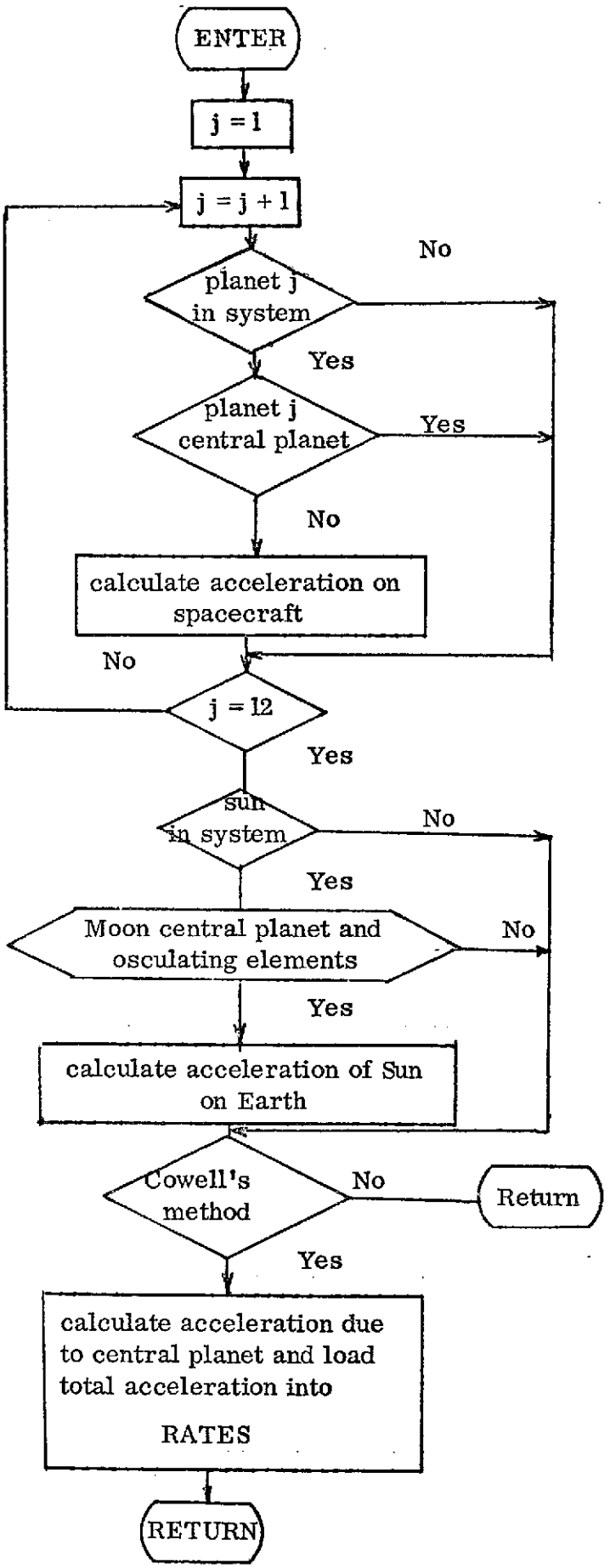
Description:

GRAV calculates the disturbing acceleration on the spacecraft due to all external planets in the system. The planets in the system are determined by the vector KP, i.e., if KP(i) is not equal to zero, then the net acceleration due to Planet i will be calculated. The accelerations are added and stored in the vector RCART.

If the moon is the central planet, and osculating elements are used for the moon (JMN = 4), then the net acceleration on the spacecraft due to the Sun is the acceleration on the S/C due to the Sun minus the acceleration on the Earth due to the Sun.

If Cowell's method of integration is to be used, then the acceleration due to the central planet is added to the other acceleration and the total is loaded into RATES (4-6) for use in the integration step.

SUBROUTINE GRAV



SUBROUTINE HSDTHR

This subroutine determines the thrust and weight characteristics of the midcourse motor used on the RAE-B spacecraft. This subroutine was supplied by Hamilton-Standard Corp., who is the builder of the motor. Any questions concerning this subroutine should be directed to Mr. Charles Newman of NASA Goddard Space Flight Center.

SUBROUTINE INPUTF

Calling Sequence: CALL INPUTF

Purpose: This subroutine reads the input data cards and stores the information in the proper common blocks.

Common Block Required: FIELDM, INPUT, INPUTS

Subroutines Required: OBSET

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	A	1000	INPUT(1)	Real part of the input array
O	AS	1000	INPUTS(1)	Real part of the saved input array
O	C	16,16	FIELDM(1)	Cosine coefficient of spherical harmonic potential term.
O	KMOD	16,16	FIELDM(525)	Array of flags used to indicate if a harmonic is input
O	KOPT	100	INPUT(1001)	Integer portion of the input array
O	KOPTS	100	INPUTS(1001)	Integer portion of the saved input array
O	MMOD	1	FIELDM(514)	Number of tesserals used
I	MODLEM	1	INPUT(1035)	Flag used to determine the type of gravitational field.
O	NMMOD	1	FIELD(515)	Highest zonal for which a tesseral is desired
O	NMON	1	FIELDM(513)	Number of zonals used
O	S	16,16	FIELD(257)	Sine coefficient of the spherical harmonic potential term

Description:

This subroutine reads the input data cards and establishes the working input arrays. These arrays consist of the A and the KOPT arrays of INPUT common. These arrays are set equal to the saved input arrays of INPUTS common before the case is initiated. The saved input arrays consist of the accumulation of all previous input including inputs from previous cases. Thus, only the inputs which differ from case to case need be input. The cases are separated by a blank card.

The saved input array is initialized to preset values in the BLOCKDATA subroutine. However, some of the values of preset inputs are dependent on the program MODE flag input via location 1044. Hence, logic is incorporated in this subroutine to preset those inputs dependent on the program mode. The inputs preset and their respective values are presented in Table I.

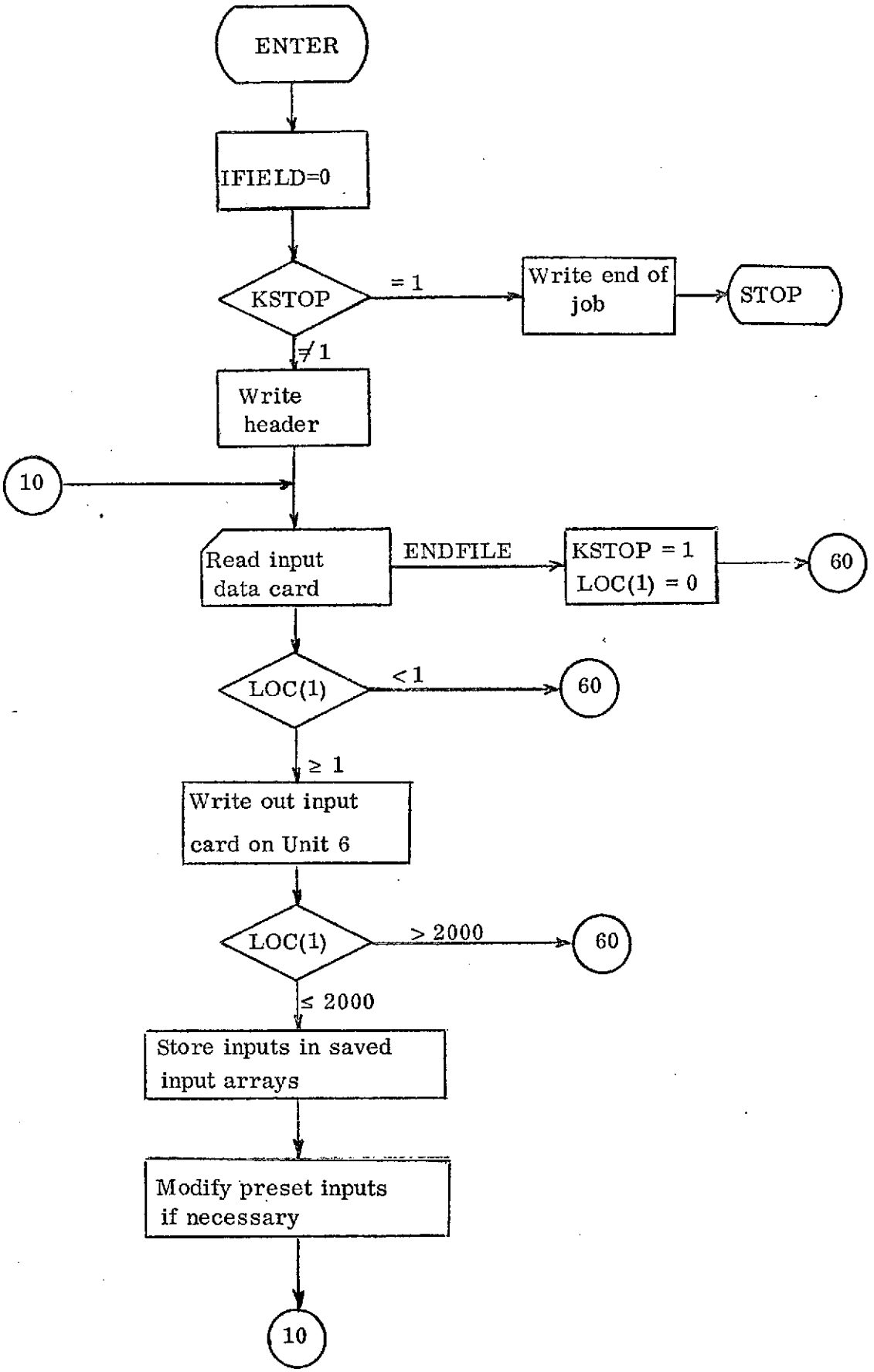
A special set of inputs are also included to set up the gravitational field. These inputs are designated by input locations greater than 2000. The data cards with input locations in the 2000's must be placed after the normal 1000 series inputs.

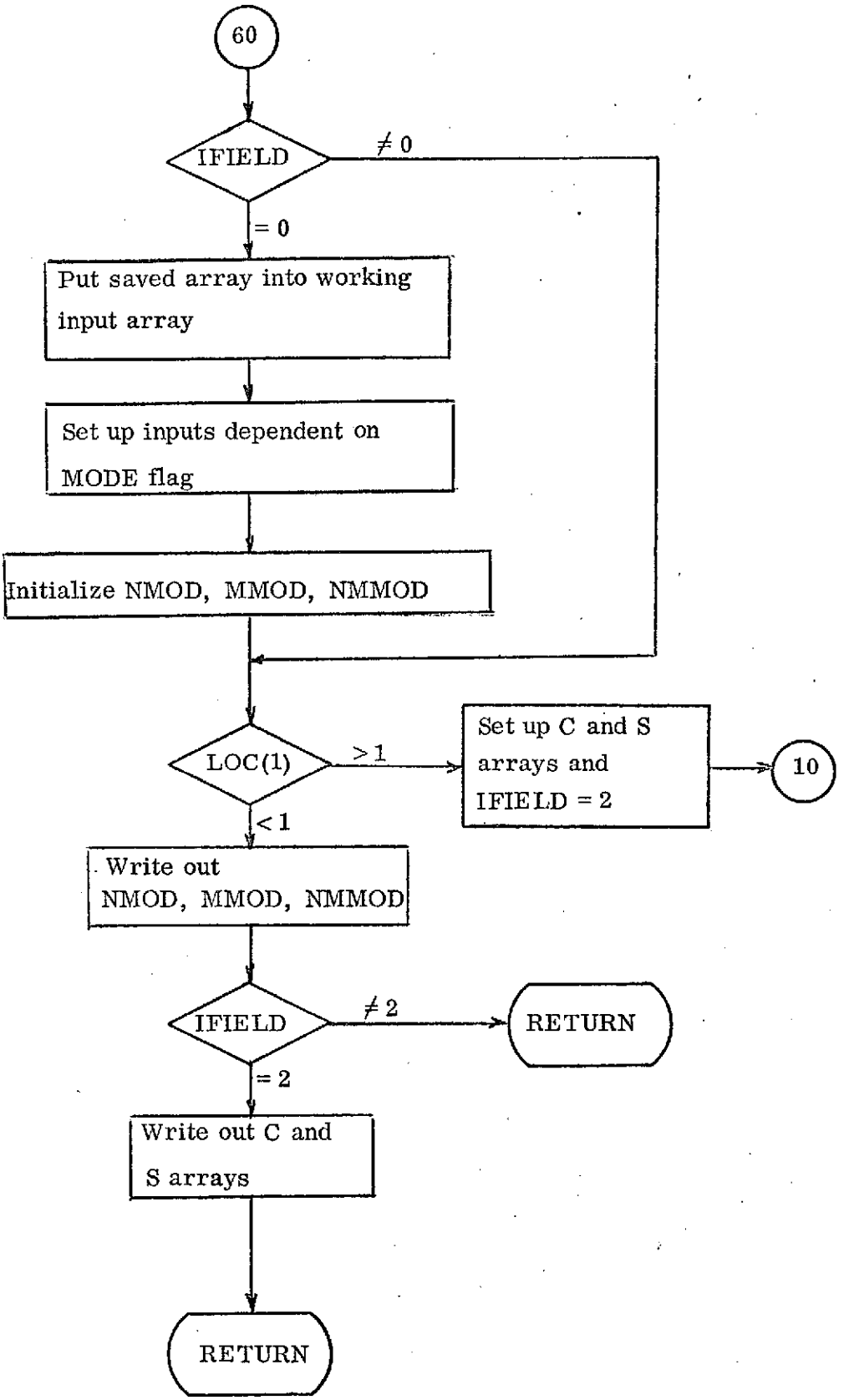
The inputs for the gravity field can either define the entire field, modify one of the preset fields, or use the field set up in the previous case. The MODLEM flag is used to determine the field as follows:

MODLEM =	1, 2, 3	Modify or use the L1, Earth J_2, or JPL 15 x 8 field.
=	5	New field input. Field is comprised of only the 2000 series inputs of the current case.
=	10	Use the field from the last case.

Subroutine OBSET is used to initialize the field when MODLEM = 1, 2 or 3.

SUBROUTINE INPUT F




60
IFIELD
≠ 0
= 0
Put saved array into working input array
Set up inputs dependent on MODE flag
Initialize NMOD, MMOD, NMMOD
LOC(1)
> 1
Set up C and S arrays and IFIELD = 2
10
< 1
Write out NMOD, MMOD, NMMOD
IFIELD
≠ 2
RETURN
= 2
Write out C and S arrays
RETURN

SUBROUTINE INTEG

Calling sequence: CALL INTEG (LOPT)

Purpose: This subroutine sets up arrays for use in subroutine EQNS where the derivatives of the integration variables are calculated. This subroutine also calls EQNS and / or the numerical integration routines.

Common blocks required: CONST, CNTRL, INPUT
INTVAR, PERT, STATE

Subroutines required: EQNS, ORBIT,
RKSEVN, TWELVE

I/O	SYMBOLIC NAME	PROGRAM DIMENSIONS	COMMON BLOCK	DESCRIPTION
I	DELTO	1	INPUT (2)	Initial compute interval in automatic compute interval mode.
I	DOM	3	PERT (25)	Position and velocity vectors on reference orbit when using Encke.
I/O	DX	3	STATE (4)	Spacecraft velocity
O	D2X	3	STATE (7)	Spacecraft accelerations
I/O	ELM	6	STATE (14)	Spacecraft osculating orbital elements on Encke variables
I	GM	12	CONST (5)	Gravitational constants
I/O	H	1	INTVAR (14)	Compute interval
I	JC	1	CNTRL (7)	Central planet
I/O	KDIS	1	CNTRL (5)	Discontinuity flag
I	KHALT	1	CNTRL (6)	Error return flag
I	KINT	1	INPUT (1014)	Numerical integration scheme indicator
I	KORECT	1	INPUT (1073)	Flag not to calculate derivatives at the end of the step
I	LOPT	1	Calling Argument	Positive to integrate a step Negative to calculate derivatives only
I	METH	1	INPUT (1013)	Trajectory propagation method indicator
I	RATES	6	INTVAR (8)	Derivatives of state
I/O	T	1	STATE (10)	Time corresponding to state in STATE common
O	TIME	1	INTVAR (1)	Time corresponding to variables to be integrated in INTVAR common
I	TRU	1	STATE (27)	True anomaly of state when using METH = 4 or 5, or true anomaly plus argument of perigee when METH = 8
I/O	X	3	STATE (1)	Spacecraft position
I/O	Y	6	INTVAR (2)	Variables to be integrated

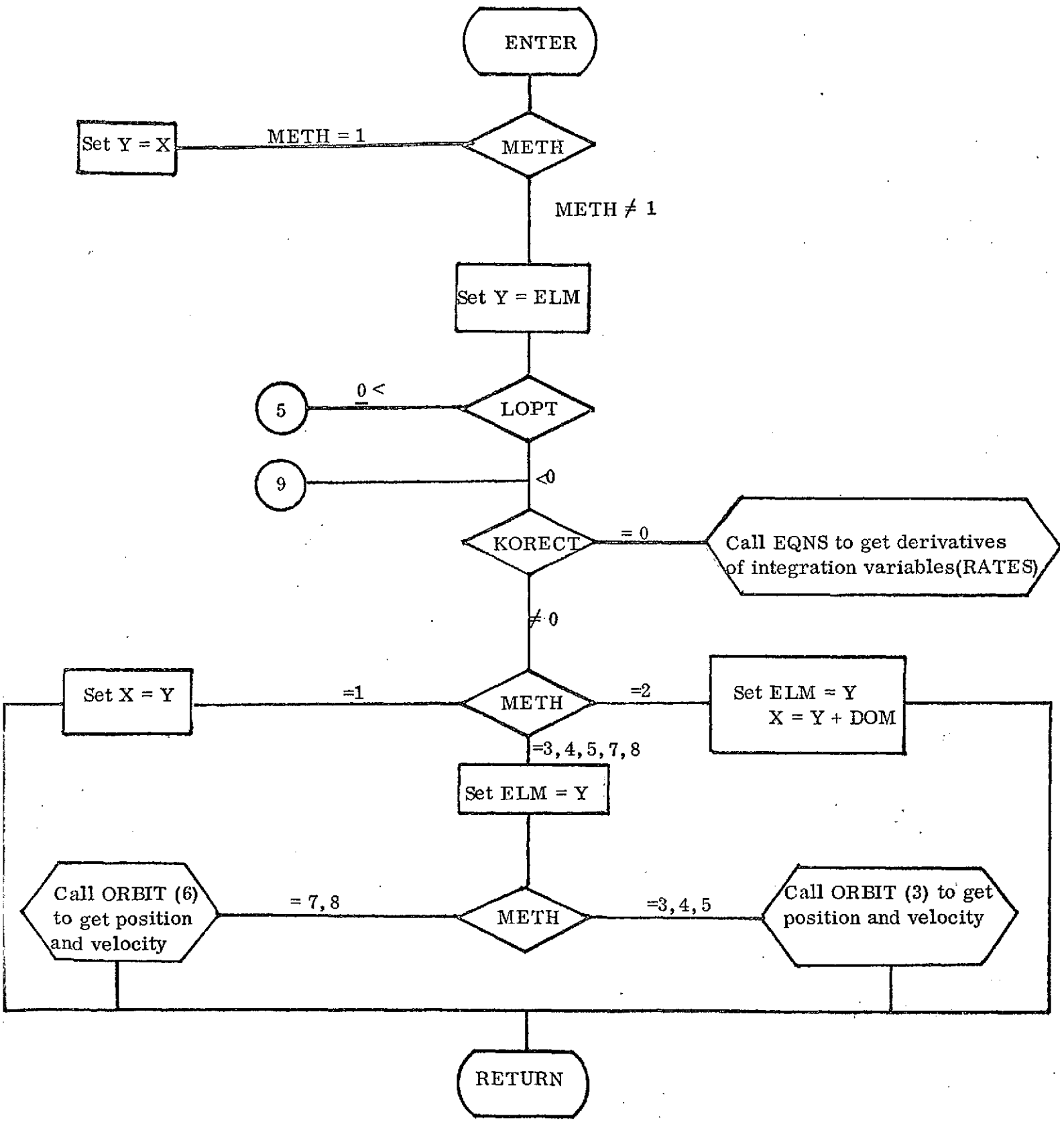
Description:

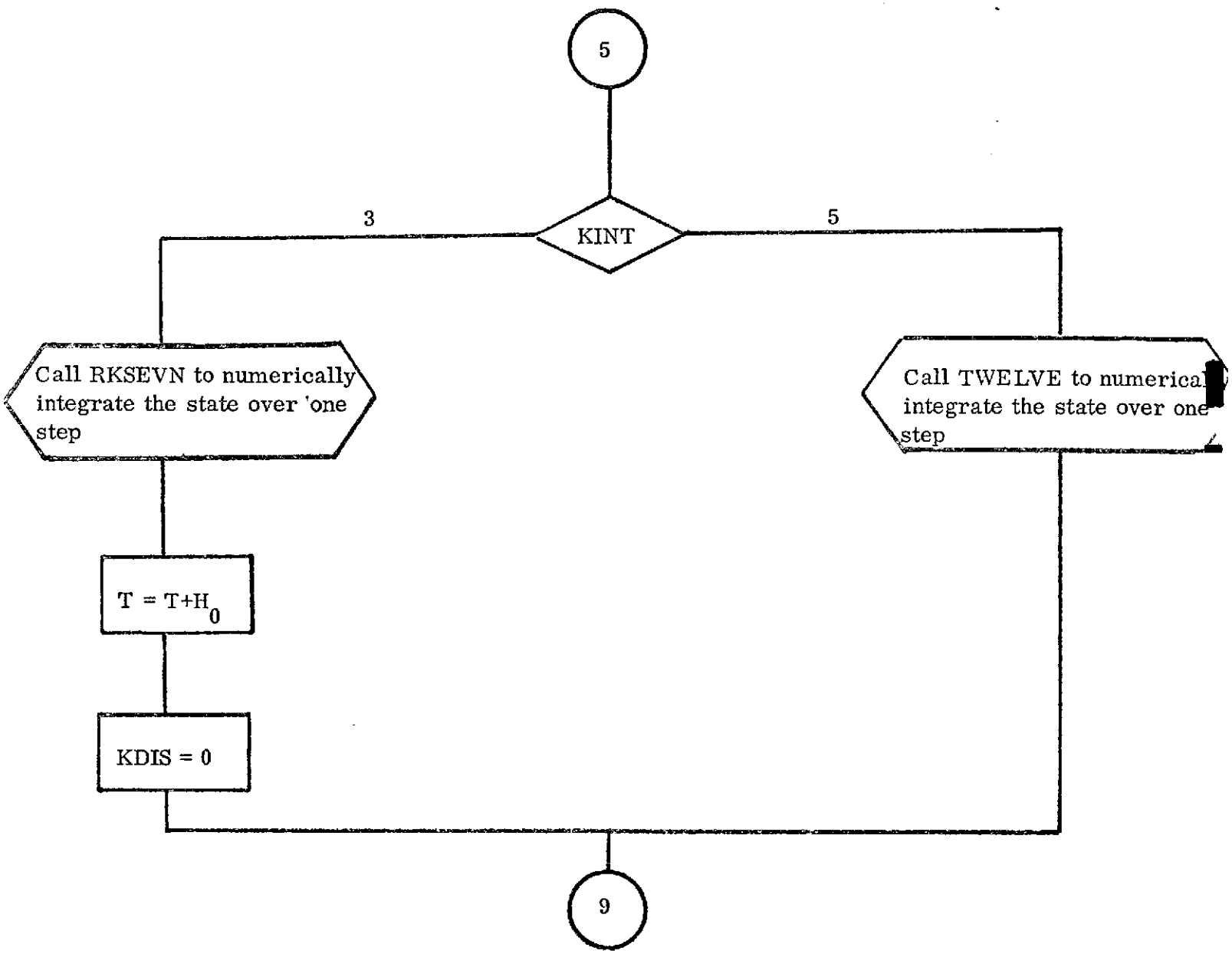
The primary function of this subroutine is to set up the integration array before integration and restore the new state after integration is completed. If LOPT is negative, numerical integration is not executed and this routine is only used to determine the derivatives at a specified time. The initial state is brought into the routine via X and DX or ELM in STATE common. When Cowell is used (METH = 1), the position and velocity vectors are in X and DX. When the other trajectory propagation methods are used, the various sets of orbital elements or Encke's variables are in ELM. Whichever method is used, the state is loaded into the integration array, Y.

If LOPT is zero or positive, the appropriate numerical integration subroutine is called to propagate the state over the time step H.

After the state is propagated, the state array is restored to the integration array in order to set the state to the values after integration. The position and velocity vectors are also calculated at the end of integrations for use in other subroutines.

SUBROUTINE INTEG




5
KINT
3
5
Call RKSEVN to numerically integrate the state over 'one step
Call TWELVE to numerical integrate the state over one step
T = T+H0
KDIS = 0
9

SUBROUTINE INTERP

Calling Sequence: CALL INTERP (T, Z)

Purpose: To determine the position and velocity of the spacecraft at times other than compute times.

Common Blocks Required: CNTRL, CONST, INPUT, INTER, PERT

Subroutines Required: ORBIT, TRMN

Input / Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	ACL	6, 10	INTER(81)	Back values of the derivatives of the interpolation quantities
I	GM	12	CONST(5)	Gravitational constant
I	JC	1	CNTRL(7)	Central planet
I	INT	1	INTER(131)	Number of back values stored
I	METH	1	INPUT(1013)	Trajectory propagation indicator
I	PI 2	1	CONST(3)	Twice pi
I	POS	6, 12	INTER(11)	Back values of the interpolation quantities
I	T	1	CALLING OPERAND	Independent variable used in interpolation
I	X	10	INTER(1)	Back values of the independent variable.
O	Z	6	CALLING OPERAND	Interpolated position and velocity vectors

Theory:

The interpolation formula used is of the form

$$P(X) = \sum_{j=0}^{n} h_{0,j}(X)\, f_i + h_{1,j}(X)\, f_i' + h_{2,j}(X)\, f_i'' \qquad (1$$

where

$P(X)$ is the interpolated value

X is the independent variable

$h_{0,j}$, $h_{1,j}$, $h_{2,j}$ are functions in X

f, f', f'' are the back values of the quantity to be interpolated and its first and second derivatives.

For the cases where only the first derivatives are available, the interpolating polynomial shown in equation (1) reduces to a Hermitian polynomial and the functions $h_{i,j}$ become,

$$h_{0,j}(X) = \left[1-2(X-X_j)\,\lambda_j'(X_j)\right]\,\lambda_j^2(X) \qquad (2)$$

$$h_{i,j}(X) = (X-X_j)\,\lambda_j^2(X)$$

where λ_j is the Lagrange interpolating coefficient defined by

$$\lambda_j(X) = \prod_{\substack{i=0 \\ i\neq j}}^{n} \left[(X-X_i) / (X_j - X_i)\right]$$

also

$$\lambda_j'(X_j) = \sum_{\substack{i=0 \\ i\neq j}}^{n} \frac{1}{X-X_i}$$

When both the first and second derivatives of the interpolation quantities are available, the functions in the interpolating polynomial become

$$h_{k,j}(X) = \phi_{k,j}(X)\,\lambda_j^3(X) \qquad K=0,2 \qquad (3$$

The functions $\emptyset_{k,j}$ are defined by

$$\emptyset_{0,j}(X) = 1 + 6\,(X-X_j)^2 \left[\left(\lambda_j^1(X_j)\right)^2 - \frac{1}{4}\lambda_j''(X_j)\right] - 3(X-X_j)\,\lambda_j'(X_j)$$

$$\emptyset_{1,j}(X) = (X-X_j) - 3(X-X_j)\,\lambda_j'(X_j) \qquad (4)$$

$$\emptyset_{2,j}(X) = \frac{1}{2}(X - X_j)^2$$

Where the second derivative of the Lagrange interpolating coefficient is given by

$$\lambda_j''(X_j) = \sum_{\substack{i=1 \\ i\neq j}}^{n} \left(\frac{1}{X_j - X_i}\right)^2 - \sum_{\substack{i=1 \\ i\neq j}}^{n} \frac{1}{(X_j - X_i)^2}$$

The first derivative of equation (1) defines an interpolating polynomial for the first derivative of the interpolation quantity. Differentiating equation (1) yields

$$P'(X) = \sum_{j=0}^{k} h'_{0,j}(X)\,f_j + h_{1,j}(X) f'_j + h'_{2,j}(X)\,f''_j \qquad (5)$$

where

$$h'_{i,j}(X) = \gamma_{i,j}(X)\,\lambda_j^3(X_j) \qquad i=0,2$$

and

$$\gamma_{i,j}(X) = 3\,\emptyset_{i,j}(X) \sum_{\substack{i=1 \\ i\neq j}}^{n} \frac{1}{X - X_i} + \emptyset_{i,j}(X)$$

The functions $\emptyset'$ are given by

$$\emptyset'_{0,j}(X) = 12\,(X-X_j)\left[\left(\lambda_j'(X_j)\right)^2 - \frac{1}{4}\lambda_j''(X_j)\right] - 3\lambda_j'(X_j)$$

$$\emptyset'_{1,j}(X) = 1-6\,(X-X_j)\,\lambda_j'(X_j) \qquad (6)$$

$$\emptyset_{2,j}(X_j) = (X-X_j)$$

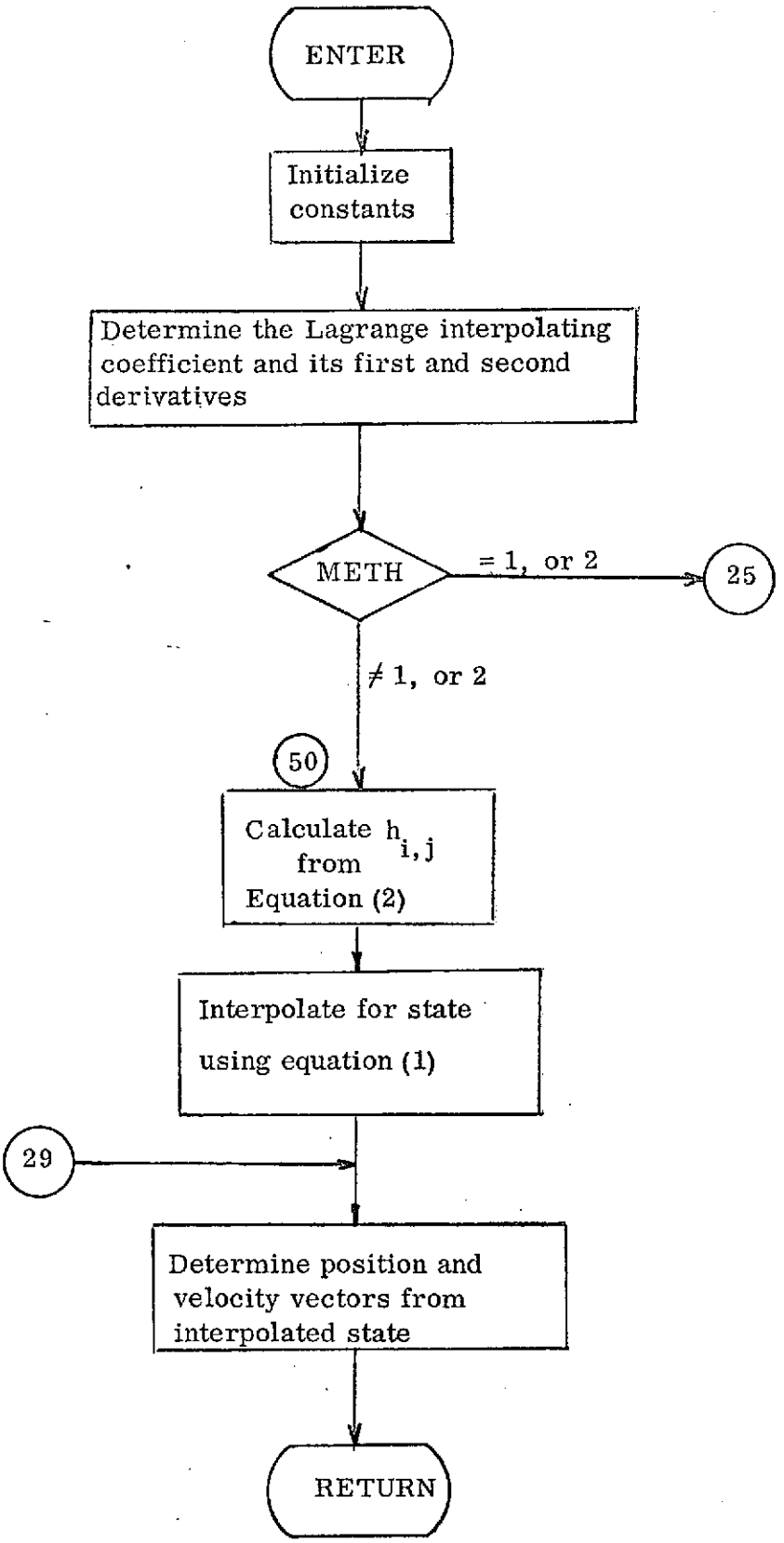
Description:

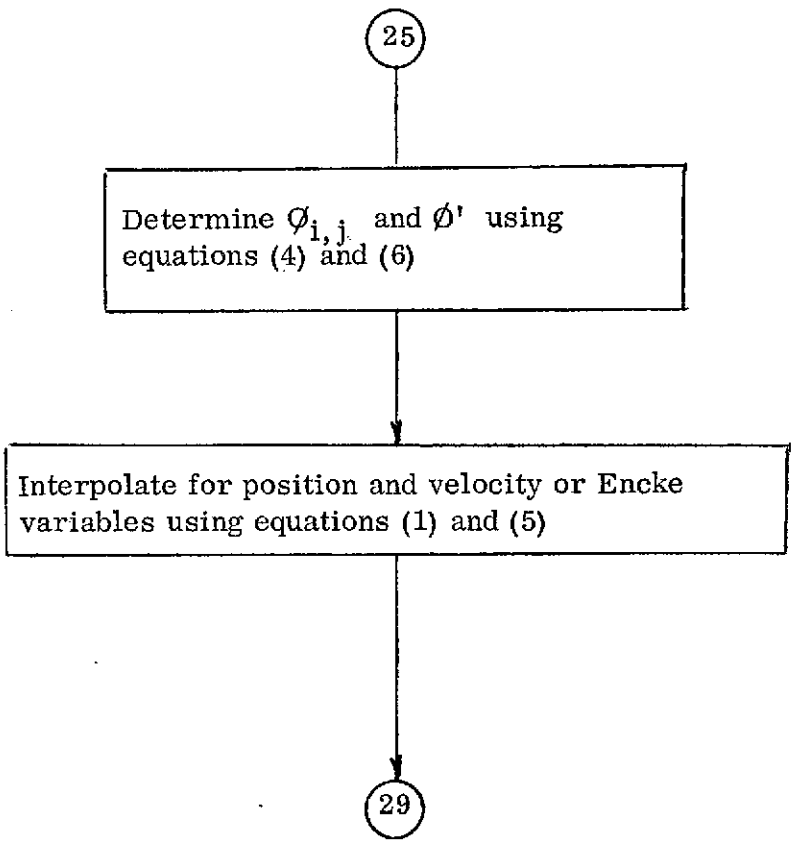
This subroutine determines the position and velocity vectors at any arbitrary time by interpolating on saved values of the state and its derivatives. The saved values of the state and its derivatives are brought into the subroutine via INTER common. This array is set up by subroutine UPDATE. The values in this array are the integration parameters. Thus if METH = 1,this array contains the position and velocity vectors, whereas it contains the orbital elements if METH=3. If METH = 1 or 2, the state and its first and second derivatives are contained in INTER. Only the state and its first derivative are available when METH is other than 1 or 2.

When METH equals 1 or 2,the position (or position from reference orbit) is obtained using equations (1), (3) and (4). The velocity (or velocity from reference orbit) is obtained using equations (5) and (6). When METH is other than 1 or 2, the state is obtained from equations (1) and (2).

The position and velocity vectors are determined from the interpolated state. These quantities are set in the Z array and returned through the argument list.

SUBROUTINE INTERP




25
Determine $\emptyset_{i,j}$ and $\emptyset'$ using equations (4) and (6)
Interpolate for position and velocity or Encke variables using equations (1) and (5)
29

SUBROUTINE JET

Calling Sequence: CALL JET

Purpose: JET computes a first-guess translunar midcourse velocity impulse using patched-conic assumptions.

Common Blocks Required: CONST, INPUT, INTVAR, MCCOM, PLNET, STATE

Subroutines Called: BCONIC, CROSS, DOT, DVMAG, MVTRN, M50LEQ, ORIENT, PLANET, RETDV, ROTAIT, TRMN, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	RTD	1	CONST(1)	Radians to degrees conversion factor
I	GME	1	CONST(7)	GM(km^3/sec^2) for the Earth
I	GMM	1	CONST(15)	GM(km^3/sec^2) for the Moon
I	WTO	1	INPUT(38)	Initial spacecraft weight (kg)
I	DJO	1	INPUT(46)	Julian date (days) of anchor vector epoch
I	JT	1	INPUT(1031)	Target body number (11)
I	MCOUT	1	INPUT(1050)	Extra output key (prints if .GE. 3)
O	TINT	1	INTVAR(1)	Time for anchor epoch (sec)
O	DVMC	3	MCCOM(12)	Midcourse velocity impulse (km/sec)
I	BVD	2	MCCOM(19)	Desired miss-vector (km) if IBTR = 1
I	DVB4	1	MCCOM(24)	Velocity expended previously (km/sec)
I	PRD	1	MCCOM(80)	Desired radius of closest approach (km)
I	OINC	1	MCCOM(81)	Desired selenographic approach inclination (deg)
I	DTFLS	1	MCCOM(82)	Desired time to closest approach (sec) measured from anchor epoch, DJO

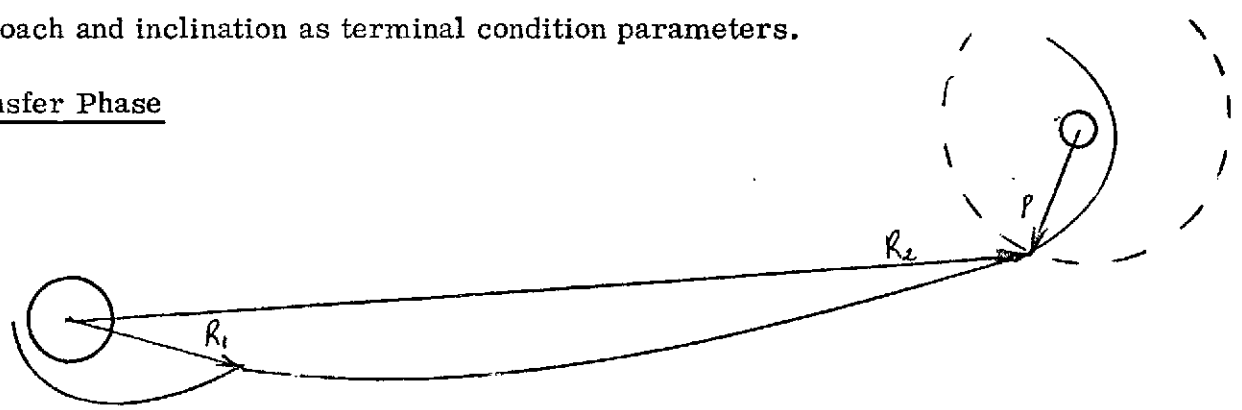
I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	VINFD	1	MCCOM(83)	Desired hyperbolic excess velocity(km/sec)
I	DCEV	1	MCCOM(84)	Desired circular excess velocity (km/sec) after retro
I/O	NGROPT	1	MCCOM(163)	Gradient re-computation key
I	KGLAW	1	MCCOM(164)	Guidance law key
I	IBTR	1	MCCOM(167)	End constraint type key (1 for BVD, 2 for PRD and OINC)
O	XP	6,12	PLNET(1)	Ephemeris state of body I (km, km/sec)
I	X1	6	STATE(1)	Pre-maneuver state (km, km/sec)
I	TIME	1	STATE(10)	Time(sec) of X1 referred to anchor epoch
I	EJO	1	STATE(26)	Ephemeris time (days) at anchor epoch
O	EJT	1	STATE(28)	Ephemeris time (days)

Introduction:

The method to be described has been fruitfully applied to the determination of midcourse correction maneuvers for the RAE-B mission. It has been used as a pre-targeting device to provide "first-guess" midcourse corrections to a precise differential-correction-type targeting scheme. In this role, the method has eliminated the need for non-linear targeting measures such as control limiting and gradient re-computations and greatly reduced the number of trajectory calculations required.

A gross description of the method would be "patched-conic targeting with constrained end conditions." Refinements which contribute to the success of the method are primarily found in the use of the Jacobian energy for targeting and in the use of radius of closest approach and inclination as terminal condition parameters.

Transfer Phase



Schematic for a Transfer Trajectory

The transfer phase is treated as a single conic section relative to the central body. The first terminal of this phase is the point of the midcourse correction on the uncorrected transfer trajectory at radius R_1, velocity V_1, at time t_1. The second terminal is on the sphere of influence of the target at a radius R_2. The second terminal's radius vector is computed from

$$R_2 = R_t + P \tag{1}$$

where R_t is the position of the target at t_2 relative to the central body and where P is a vector from the central body to the point of entry of the target's sphere of influence. The time, t_2, of arrival at the sphere of influence is computed from

$$t_2 = t_f - \tau \tag{2}$$

where t_f is the time of arrival at closest approach (specified a priori) and where τ is the time required to trsvel on the approach hyperbola from the sphere of influence to closest approach.

The arrival phase parameters, P and τ, will be described later. An iterative solution of Lambert's Problem (i.e., "Find the conic section passing from R_1 to R_2 in time $t_2 - t_1$") is employed. This solution provides V_1^*, the velocity on the transfer conic at the first terminal, and V_2^*, the transfer conic's velocity at the second terminal. The midcourse correction impulse, ΔV, is computed from

$$\Delta V = V_1^* - V_1 \tag{3}$$

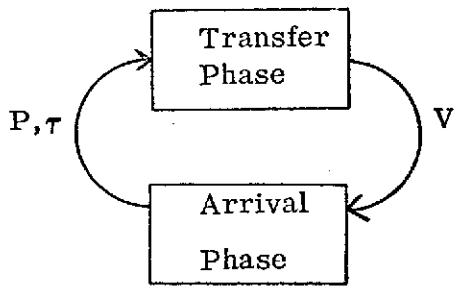
and the target-relative velocity, V, at t_2 is

$$V = V_2^* - V_t \tag{4}$$

where V_t is the target's velocity at t_2 relative to the central body.

The first transfer conic is computed with P in the target's orbital plane at 20^oearthward from tangential. The target relative transfer time, τ, is fixed at 66,000 seconds.

The target-relative velocity derived from this first conic is used to initiate the arrival phase calculations. A two-or-three-step iteration (transferring V to arrival phase calculations and P and τ to transfer phase calculations) will



converge to a steady-state value for ΔV. This answer is the impulsive patched-conic, fixed-time-of-arrival midcourse correction.

Arrival Phase

The arrival phase computations use the target-relative approach velocity and desired arrival conditions to develop (1) the time of passage from the sphere of influence to closest approach and (2) the point of entry into the sphere of influence. The desired arrival conditions are specific values of radius of closest approach and inclination. We develop characteristics which the approach hyperbola must possess in order to satisfy the desired arrival conditions. We assume that the target-relative approach velocity vector defines the direction of the arrival asymptote of the approach hyperbola. Furthermore, we assume that the point of entry into the sphere of influence can be computed for the next transfer phase using the direction of the arrival asymptote and the target-relative energy from the current arrival phase in its calculation. The energy of the approach orbit as used in arrival phase computation is defined by equation (5),

$$C_3 = V \cdot V - \frac{2\mu_t}{a'} \tag{5}$$

where V is the approach velocity. This (Jacobian) energy is adjusted for perturbation by the target body on the transfer trajectory.

Figure (1) shows an arrival hyperbola whose no-plane character is described by a closest approach radius, r_p, and half-angle, α. Given r_p, we can use the energy, C_3,

to solve

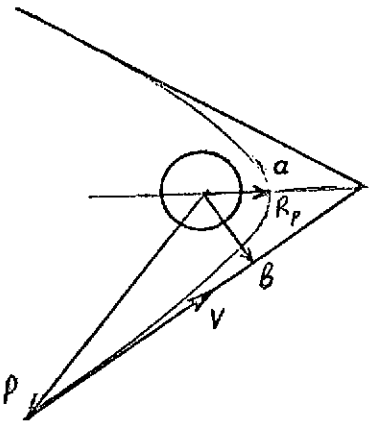


Figure 1

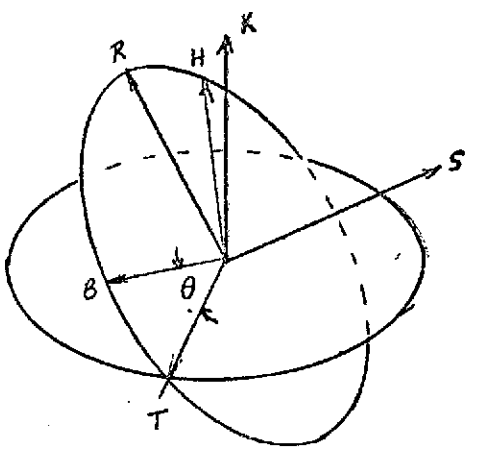


Figure 2

for the various parameters of the hyperbola

$$v_h = \sqrt{C_3} \qquad \text{(hyperbolic excess speed)} \qquad (6)$$

$$v_p = \sqrt{C_3 + \frac{2\mu}{r_p}} \qquad \text{(closest approach speed)} \qquad (7)$$

$$\alpha = \frac{-1\ v_p\ v_h}{\left(\mu / r_p\right)} \qquad \text{(half-angle between asymptotes)} \qquad (8)$$

$$a = -\mu / C_3 \qquad \text{(semi-major axis)} \qquad (9)$$

$$b = (r_p - a) \sin \alpha \qquad \text{(asymptotic miss distance)} \qquad (10)$$

$$e = \sqrt{1 + \left(\frac{b}{a}\right)^2} \qquad \text{(eccentricity)} \qquad (11)$$

$$p = -a\left(\frac{b}{a}\right)^2 \qquad \text{(semi-latus rectum)} \tag{12}$$

$$\frac{1}{n} = |a|\sqrt{\frac{|a|}{\mu}} \qquad \text{(reciprocal mean motion)} \tag{13}$$

The true anomaly, f, at "patch" distance, r, is computed from

$$f = \cos^{-1}\left\{\left(\frac{p}{r} - 1\right)/e\right\} \tag{14}$$

and the time of passage, τ, from the sphere of influence to the point of closest approach is

$$\tau = \frac{1}{n}\left\{e \sinh f' - f'\right\} \tag{15}$$

where $$f' = \ln\left\{1 + \frac{\sqrt{e-1}}{e+1}\tan\frac{f}{2}\right\} - \ln\left\{1 - \frac{e-1}{e+1}\tan\frac{f}{2}\right\} . \tag{16}$$

We next calculate the point of entry into the sphere of influence in such a way that the hyperbola has the specified inclination, i_d. We first define the arrival asymptote, S, as V normalized to unity. Then, if K is a unit vector normal to the target's equatorial plane at time of closest approach, we can define vectors T and R normal to S as follows. (See Figure 2)

$$T = \frac{S \times K}{|S \times K|} \tag{17}$$

$$R = T \times S \tag{18}$$

The miss-vector, B, lies in the orbital plane and in the plane of T and R at an angle θ measured from T towards R.

$$B = T\cos\theta + R\sin\theta \tag{19}$$

The plane of the hyperbola is defined by a unit vector, H, in the direction of the angular momentum.

$$H = B \times S = R \cos\theta - T \sin\theta \tag{20}$$

The condition that the orbit's inclination is i_d is

$$H \cdot K = \cos i_d = R \cdot K \cos\theta \tag{21}$$

which can be solved for $\cos\theta$ if $|R \cdot K| \leq |\cos i_d|$. If $|R \cdot K| > |\cos i_d|$, it means i_d cannot be attained. In that case, the best that can be done is

$$\cos\theta = 1 \cdot \text{sign}\left(\frac{\cos i_d}{R \cdot K}\right) \tag{22}$$

The sign on $\sin\theta = \pm\sqrt{1 - \cos^2\theta}$ can be chosen to make the miss-vector lie above or below the equator in the miss plane. Having now calculated B, we can form the vector, P, from the target's center toward the point of entry into the sphere of influence.

$$P = r\left\{S \cos(f+\alpha) + B \sin(f+\alpha)\right\} \tag{23}$$

It has been found beneficial for convergence of the arrival-transfer iteration to introduce a "gain" of .7 on the change of P between iteration steps.

The computations above provide the point of entry of the sphere of influence and the time of passage from the sphere of influence to closest approach. The point of entry is then used to establish a new second terminal position for the transfer trajectory.

Guidance Laws

The solution provided by the above process is the fixed-time-of-arrival (FTA) guidance solution. The FTA guidance law constrains arrival (at closest approach) time, t_f, to be a specific value while satisfying the desired end conditions of radius of closest approach and inclination. Other guidance laws of interest relative to the RAE-B mission are:

1) minimum midcourse fuel,

2) fixed target energy, and

3) variable target energy.

Each of the other guidance laws constrain radius of closest approach and inclination just as the FTA law does, but do not specifically constrain arrival time. The minimum midcourse fuel law embraces the "critical plane" solution, the fixed target energy law constrains hyperbolic excess speed at the target, and the variable target energy law constrains post-retro velocity at the target subject to a prescribed de-boost strategy. The solution for each of these laws, however, corresponds to a particular arrival time (or, flight time), so flight time is used as the independent variable in seeking solutions for each law.

The MFG and MTF laws are pre-targeted by means of a Newton-Raphson-type iteration with flight time as the independent variable. The iteration seeks to null the dot product of the difference of two successive midcourse correction impulses with the impulse itself. That is, it seeks to find the flight time for which the magnitude of the correction velocity doesn't change (i.e.,is minimum).

Equation 24 defines the condition for minimizing the magnitude of ΔV.

$$\Delta V \cdot \frac{d(\Delta V)}{dt} = 0 \tag{24}$$

where t signifies flight time in this case. We define t_n to be the flight time for the n-th trial and ΔV_n to be the velocity impulse for that trial. Then, approximately,

$$\left(\frac{d(\Delta V)}{dt}\right)_n = \frac{\Delta V_n - \Delta V_{n-1}}{t_n - t_{n-1}} \tag{25}$$

and more approximately,

$$\Delta V_{n+1} = \Delta V_n + (t_{n+1} - t_n) \left(\frac{d(\Delta V)}{dt}\right)_n \tag{26}$$

We look for t_{n+1} such that equation 27 holds.

$$\Delta V_{n+1} \cdot \left(\frac{d(\Delta V)}{dt} \right)_{n+1} = 0 \tag{27}$$

By making liberal use of equations 25 and 26, equation 27 may be solved to render $t_{n+1} - t_n$.

$$t_{n+1} - t_n = -(t_n - t_{n-1}) \quad \frac{\Delta V_n \cdot (\Delta V_n - \Delta V_{n-1})}{|\Delta V_n - \Delta V_{n-1}|^2} \tag{28}$$

This process converges well in all cases tested. The resultant ΔV changes very little with precise targeting, although the corresponding flight time shifts by an hour or two.

The fixed time of arrival guidance law is pre-targeted to 1.015 times the desired flight time. This empirical factor tends to compensate for the difference between patched-conic and integrated-perturbed lunar transfer trajectory flight times. No iteration is required as for the other guidance laws, since flight time is both the third constraint of the FTA law and the independent variable for pre-targeting.

The pre-targeting process for the FTE and VTE guidance laws include a regular Newton-Raphson iteration to null the third constraint by varying flight time. For the FTE law, the third constraint error function is

$$\psi_3 = v_\infty (\text{desired}) - \sqrt{C_{3T}} \tag{29}$$

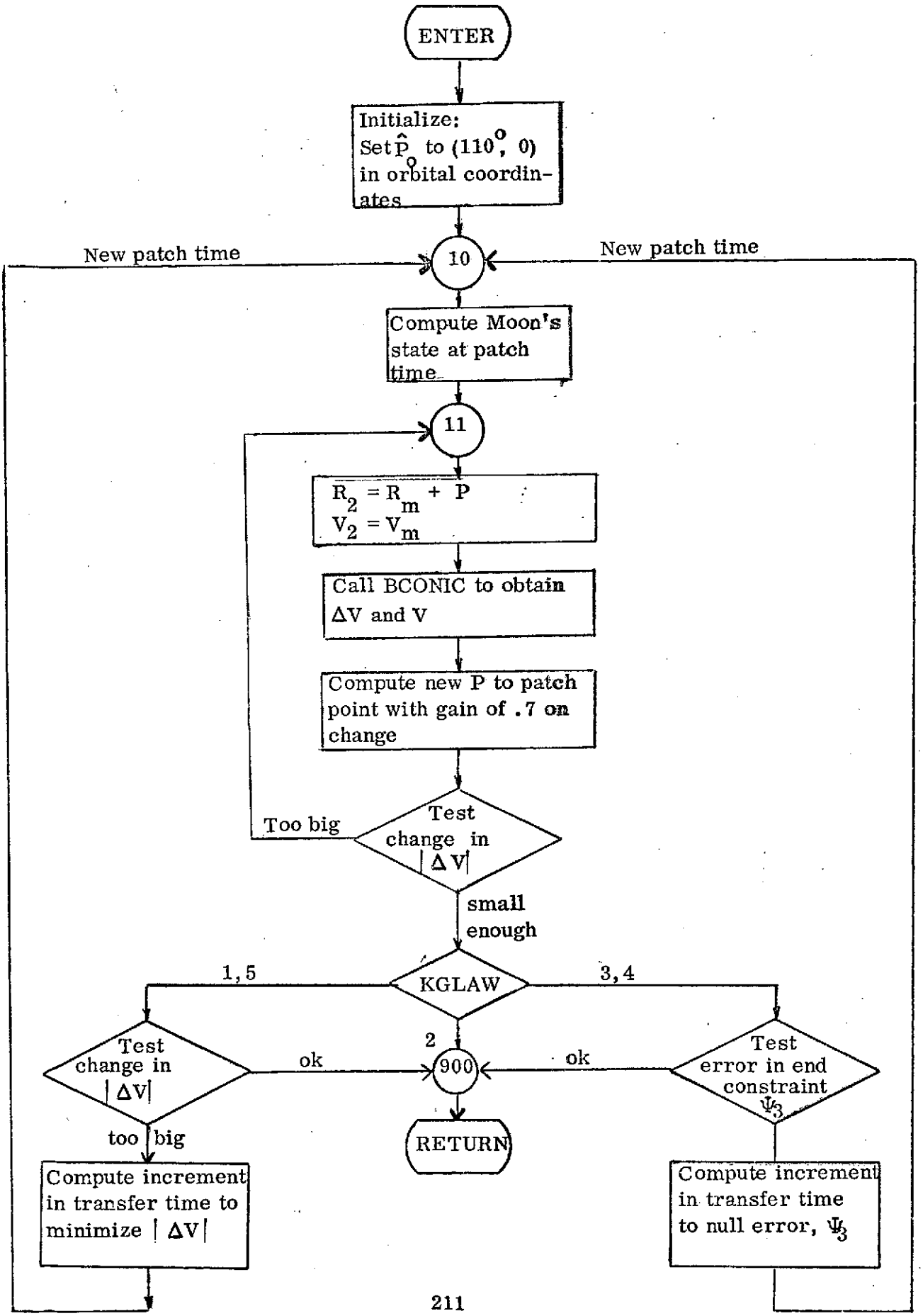
where C_{3T} is defined by equation 5. For the VTE law, the function to be nulled is

$$\psi_3 = \sqrt{\frac{\mu_m}{r_{pd}}} - \sqrt{C_{3T} + \frac{2\mu_m}{r_{pd}}} + \delta v_r \tag{30}$$

where r_{pd} is the desired distance of closest approach and δv_r is the velocity impulse imparted by the retro motor. The VTE law usually has two solutions, i.e., two flight

times for which $\psi_3 = 0$. The iteration is constrained to find the solution with positive slope, which is the solution of the longer flight time.

SUBROUTINE JET



SUBROUTINE LUNA

Calling Sequence: CALL LUNA

Purpose: To determine the position of the Moon with respect to the Earth.

Common Blocks Required: CONST, CNTRL, INPUT, INTVAR, MOON, PLNET, STATE

Subroutines Required: MVTRN, M50MDT, OBLTY, ORBIT, ROTATE, TRMN

Reference:

1. Supplement to the American Ephemeris and Nautical Almanac, U.S. Naval Observatory, U.S. Government Printing office.

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DJO	1	INPUT(46)	Modified julian date of state epoch
I	EJO	1	STATE(26)	Modified ephemeris epoch
I	ELMMN	12	MOON(1)	Osculating elements of Moon
I	FOM	1	MOON(15)	Mean anomaly at epoch of Moon's osculating elements
I	GM	12	CONST(5)	Gravitational constants of the planets
I	JC	1	CNTRL(7)	Central planet number
I	JMN	1	INPUT(1017)	Ephemeris type flag
I	PM	1	MOON(14)	Moon's mean motion from osculating elements
I	RAD	1	CONST(1)	Radian-degree conversion factor
I	T	1	INTVAR(1)	Time since state epoch ephemeris desired
I	TVPI	1	CONST(3)	2π
I	TMOON	1	MOON(13)	Epoch of osculating lunar elements
O	XP	6,12	PLNET(1)	Positions and velocities of the planets

Theory:

This subroutine calculates the position and velocity of the Moon from the mean element, mean elements plus extra terms from the lunar theory or from input osculating elements. The mean elements are obtained from the supplement to the nautical ephemeris. The mean longitude of the Moon , the mean longitude of perigee and the longitude of the ascending node are obtained from the following polynomials in time:

$$\begin{aligned} \lambda &= 4.3853720 + 8399.0912\ C - 1.97746D\text{-}5\ C^2 \\ \gamma &= 1.730894 + 71.017994\ C - 3.6267D\text{-}5\ C^2 \\ \Omega &= 1.4312588 + 33.757099\ C + 3.6263D\text{-}5\ C^2 \end{aligned} \qquad (1)$$

where

λ = longitude of the Moon

γ = longitude of perigee

Ω = longitude of the ascending node

C = number of julian centuries since 1965.

These quantities are in the mean equinox and ecliptic of date. The argument of perigee, ω, and the mean anomaly, AM, of the lunar orbit is obtained from

$$\begin{aligned} AM &= \lambda - \gamma \\ \omega &= \gamma - \Omega \end{aligned} \qquad (2)$$

Next,the eccentric anomaly of the lunar orbit is determined from Kepler's equation for small eccentricities, or

$$E = AM + e \sin AM + \frac{e^2}{2} + \frac{e^3}{8}(3 \sin 3AM - \sin AM) \qquad (3)$$

where e is the eccentricity of the lunar orbit (.054900489)

The position and velocity of the Moon in its orbit plane can now be obtained from

$$\begin{aligned} X &= a(\cos E - e) \\ Y &= a\sqrt{1-e^2} \sin E \\ \dot{X} &= -V \sin E / S \\ \dot{Y} &= V \sqrt{1-e^2} \cos E / S \end{aligned} \qquad (4)$$

where

$$S = \sqrt{(1-e^2)\cos^2 E + \sin^2 E}$$

$$V = \sqrt{GMN\left(\frac{2}{\sqrt{x^2+y^2}} - \frac{1}{a}\right)} \quad \text{(the velocity of the Moon)}$$

a is the semi-major axis of the lunar orbit (384750.8998)

The X and Y components are in the lunar orbit plane with the X axis pointing to perigee and the Y axis in the direction of motion.

The cartesian coordinates of the Moon are next transformed to the mean equinox and ecliptic of date using the following rotation matrix:

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{bmatrix} \cos\omega\cos\Omega & -\sin\omega\cos\Omega \\ -\cos i\sin\Omega\sin\omega & -\cos i\sin\Omega\cos\omega \\ \cos\omega\sin\Omega & -\sin\omega\sin\Omega \\ +\cos i\cos\Omega\sin\omega & +\cos i\cos\Omega\cos\omega \\ \sin i\sin\omega & \sin i\cos\omega \end{bmatrix} \begin{Bmatrix} x \\ y \end{Bmatrix} \quad (5)$$

where i is the inclination of the lunar orbit w.r.t. the ecliptic (.08980414 radians)

The Moon's position and velocity vectors in the Earth's mean equinox and equator of 1950 are obtained from standard rotations.

Extra terms are used in the calculation of the inclination, longitude of the ascending node of the lunar orbit, mean anomaly and semi-major axis of the lunar orbit when the proper input flag is set. The terms include the effects of evection, variation, and other periodic terms due to the Sun's force on the Moon.

The longitude of the Sun and the longitude of the perigee of the Sun's orbit is determined from mean elements similar to the lunar terms. The polynomials used are:

$$\begin{aligned} \lambda s &= 4.8860536 + 6.28331958\ C + 5.2796\ D\text{-}6C^2 \\ \gamma s &= 4.9082294 + .030005264C + 7.90246D\text{-}6\ C^2 \end{aligned} \quad (6)$$

The changes in the above mentioned quantities are:

$$\begin{aligned} i &= i\ \text{mean} + .002515665\cos(2\lambda s - 2\,\Omega\text{mean}) \\ \Omega &= \Omega\text{mean} + .02805487\sin(2\lambda s - 2\,\Omega\text{mean}) \\ a &= a\ \text{mean}\,(1 + .0090714046\cos(2\lambda - 2\lambda s) \end{aligned} \quad (7)$$

and the true anomaly of the Moon is determined from

$$\begin{aligned} u = {} & AM + .10975961\sin AM + .0037634149\sin 2AM \\ & + .00017926303\sin 3AM - .00054493\sin(\lambda - \lambda s) \\ & + .022238412\sin(2\lambda - 2\lambda s - AM) + .011493967\sin(2\lambda - 2\lambda s) \\ & - .00324292\sin(\lambda s - \gamma s) \end{aligned} \quad (8)$$

The osculating lunar position and velocity vectors are determined in a completely different manner. The mean anomaly at the desired time is determined from the mean motion and mean anomaly at epoch from

AM = PM (DELT) + FOM

where PM is the mean motion

DELT is the time since epoch (9)

FOM is the mean anomaly at epoch.

The true anomaly is determined from subroutine TRMN,and the position and velocity vectors obtained from the osculating elements from subroutine ORBIT.

Description:

The ephemeris epoch and time since the epoch that the ephemeris is to be determined are brought into the routine via common blocks. The JMN flag is used to determine the type of ephemeris as follows

JM = 1 mean elements

= 2 mean elements plus extra terms

= 3 osculating elements.

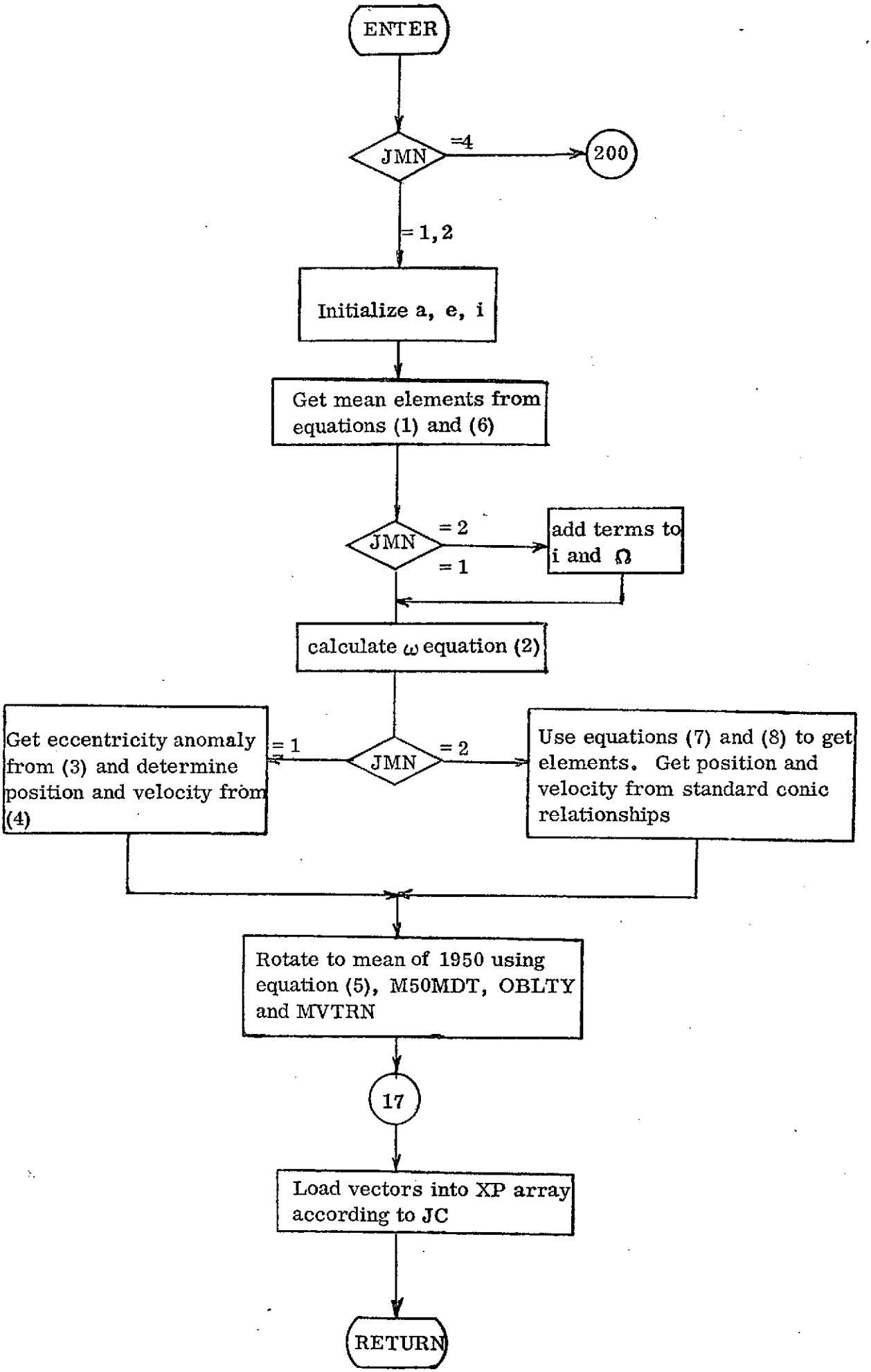
If JMN equals one or two, the mean elements of the lunar orbit are determined from the polynomials in equation (1). If the extra terms are to be included, they are determined from equation (6) and the eccentric anomaly from equation (7). Otherwise, the extra terms are not included and equations (2) and (3) are used. Finally, equation (4) is used to determine the coordinates and they are rotated to the mean equinox and equator of date using the rotation matrix shown in equation (5). Subroutines M50MDT and OBLTY are used to obtain the vectors in Earth mean equinox and equator of date.

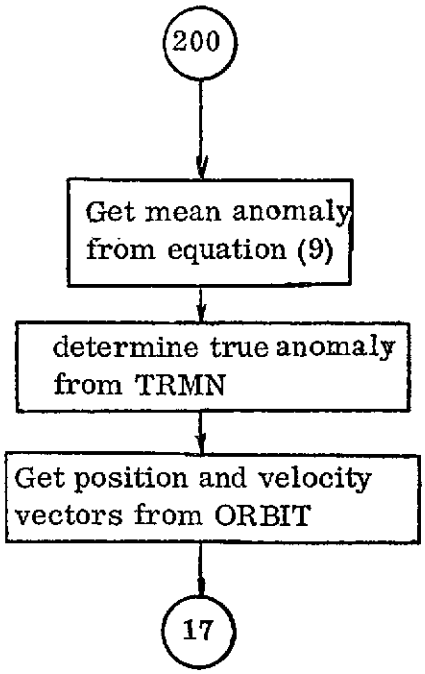
The osculating position and velocity vectors are obtained from TRMN and equation (9). The osculating elements are brought into the subroutine via Moon common and describe the Moon's orbit with respect to the Earth in the Earth's mean equator and equinox of 1950.

The position and velocity vectors determined are loaded into the XP array according to the value of the current central planet number in JC.

JC = 3 Earth central planet, thus position and velocity of Moon w.r.t. Earth loaded into XP array in Moon's position

JC	=	11	Moon central planet, thus position and velocity of Earth w.r.t. Moon loaded into XP array in Earth's position. The negative of the values determined for the Moon w.r.t. Earth are loaded into XP(I, 3).
JC	≠	3 or 11	Some other planet central, thus the position of the Earth w.r.t. central planet is added to Moon's position and velocity and loaded into the Moon's position in the XP array.


ENTER
JMN
=4
200
= 1, 2
Initialize a, e, i
Get mean elements from equations (1) and (6)
JMN
= 2
= 1
add terms to i and Ω
calculate ω equation (2)
Get eccentricity anomaly from (3) and determine position and velocity from (4)
= 1
JMN
= 2
Use equations (7) and (8) to get elements. Get position and velocity from standard conic relationships
Rotate to mean of 1950 using equation (5), M50MDT, OBLTY and MVTRN
17
Load vectors into XP array according to JC
RETURN


200
Get mean anomaly from equation (9)
determine true anomaly from TRMN
Get position and velocity vectors from ORBIT
17

SUBROUTINE MATMPY

Calling Sequence: CALL MATMPY (KRL, A, KCLRR, B, KCR, C)

Purpose: MATMPY multiplies the matrices A (KRL x KCLRR) by B(KCLRR x KCR) to get C(KRL x KCR)

Common Blocks: None

Subroutines None

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	A	Variable	Calling Operand	Matrix on the left
I	B	Variable	Calling Operand	Matrix on the right
O	C	Variable	Calling Operand	[A] x [B]
I	KCLRR	1	Calling Operand	Number of columns in A and rows in B
I	KCR	1	Calling Operand	Number of columns in C and B
I	KRL	1	Calling Operand	Number of rows in A and C

Theory:

General matrix multiplication is defined by [C] = [A] x [B]

where

$$C_{ij} = \sum_{k=1}^{m} A_{ik} B_{kj}$$

m is the number of columns in A and rows in B.

Description:

MATMPY first initializes the matrix C to zero and then forms the sum of the products above for i = 1, KRL and j = 1, KCR. The arrays are singly-dimensioned to avoid variable dimensioning.

SUBROUTINE MCBURN

Calling Sequence: CALL MCBURN

Purpose: MCBURN accepts the pre-midcourse state and time plus a velocity impulse, computes the post-midcourse state and time.

Common Blocks Required: CNTRL, CØNST, INPUT, MCCØM, STATE

Subroutines Called: DVMAG, FØWARD, DØT, BURND

Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
Ø	JC	1	CNTRL(7)	Post-maneuver central body number
I	GM	12	CØNST(5)	GM gravitational array (km^3/sec^2)
I	G	1	CØNST(45)	Earth's surface gravity (km/sec^2)
Ø	TFINAL	1	INPUT(4)	Integration stop time (sec)
I	WTØ	1	INPUT(38)	Initial spacecraft weight (kg)
Ø	TCØMP	10	INPUT(170)	Time (sec) to change integration step size
Ø	DELT	10	INPUT(180)	Integration step size (sec)
I	TWD1	3	INPUT(320)	Times (sec) for weight flow rate changes
I	WDØT1	3	INPUT(350)	Weight flow rate changes (kg/sec)
Ø	TIG	1	INPUT(380)	Ignition time (sec)
Ø	TBØ	1	INPUT(383)	Burnout time (sec)
I	ASPMC	1	INPUT(441)	Specific impulse of the midcourse motor (sec)
Ø	KRASH	1	INPUT (1032)	Trajectory stopping key
Ø	KMETH	1	INPUT (1036)	Trajectory computation method
I	IBURN	1	INPUT (1071)	Burn computation method key

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	XMC	6	MCCØM (6)	Pre-maneuver state (km, km/sec)
I	DV	3	MCCØM (12)	Midcourse velocity impulse (km/sec)
I	TMCS	1	MCCØM (18)	Midcourse initiation time (sec)
I	KBURN	1	MCCØM (154)	Burn maneuver computation key
I	ICB	1	MCCØM (164)	Central body of XMC
Ø	X	6	STATE(1)	Post-maneuver state (km, km/sec)
Ø	T	1	STATE(10)	Post-maneuver time (sec)
Ø	ATT	3	STATE(11)	Thrust attitude (unit vector)
Ø	WEIGHT	1	STATE(34)	Spacecraft weight (kg)

Method:

MCBURN's method for computing the post-midcourse state and time is determined by the input key, KBURN.

Impulsive (KBURN.LE.0)

In this case, the input velocity impulse, DV, is simply added to the pre-midcourse velocity to form the post-midcourse state.

$$X = XMC + \begin{bmatrix} 0 \\ DV \end{bmatrix} \tag{1}$$

$$T = TMCS \tag{2}$$

Finite Burn (KBURN.GT.0)

In this case, the input velocity impulse, DV, is used to calculate the burn duration and thrust attitude. The computation of burn duration assumes that the thrust magnitude is proportional to mass flow rate, that the mass flow rate decreases linearly in time, and that the velocity impulse is entirely attributable to thrust. Defining the characteristic velocity, c, of the velocity correction package as the product of gravity, G, and specific impulse, ASPMC, we write thrust as follows:

$$|\vec{T}| = c\dot{m} = -c\frac{dm}{dt} \tag{3}$$

The minus sign is necessary here if we define $\dot{m}$ to be positive.

The acceleration due to thrust is thrust/mass, so the velocity impulse, δv, due to thrusting for time t_b is:

$$\delta v = \int_o^{t_b} \frac{|\vec{T}|}{m}\, dt = -c\,\ell n\left[\frac{m(t_b)}{m(o)}\right] . \tag{4}$$

Equation (4) may be solved (given $\delta v = |DV|$) for the mass expenditure, δm.

$$\delta m = m(o) - m(t_b) = m_o(1 - e^{-\delta v/c}) \tag{5}$$

Under the assumption of a linear mass flow rate, we can solve for burn duration as follows:

$$\dot{m}(t) = \dot{m}(o) + \ddot{m}\,t \qquad (\ddot{m} \le 0) \tag{6}$$

$$\delta m = \int_o^{t_b} \dot{m}\,dt = \dot{m}(o)\,t_b + \frac{\ddot{m}}{2}\,t_b^2 \tag{7}$$

$$\dot{m}_{avg} = \tfrac{1}{2}\left[\dot{m}(o) + \sqrt{\dot{m}(o)^2 + 2\ddot{m}\,\delta m}\,\right] \tag{8}$$

$$t_b = \frac{\delta m}{\dot{m}_{avg}} \tag{9}$$

This form (9) of the solution to (7) is preferable to the more-standard quadratic-equation solution because it does not require division by $\ddot{m}$ and reduces immediately to the linear-equation solution if $\ddot{m}=0$. The thrust direction, $\hat{T}$, is computed parallel to DV.

$$\hat{T} = \frac{DV}{\delta v} \tag{10}$$

The method (KBURN.NE.6) for computing the post-midcourse state for a finite burn is to integrate the equations of motion over the burn duration.

$$X = XMC + \int_o^{t_b} \dot{X}\,(\text{gravity} + \text{thrust})\,dt \tag{11}$$

$$T = TMCS + t_b \tag{12}$$

Closed-Form Approximation (KBURN.EQ.6)

The double integration of the thrusting acceleration can be done in closed form if a mass flow rate which varies linearly with burn time is assumed. The superposition of this closed-form solution with the thrust-free state-change solution during the burn provides an excellent approximation to the numerically-integrated solution. In MCBURN, the closed-form solution is superimposed only on the

multi-conic thrust-free solution. The closed-form development follows. The position (δR) and velocity (δV) increments are presented.

$$\delta V = \int_o^{t_b} \frac{T}{m}\, dt = - c \hat{T} \ell n \left[\frac{m(t_b)}{m(o)} \right] \tag{13}$$

$$\delta R = \int_o^{t_b} \delta V\, dt = - c \hat{T} \int_o^{t_b} \ell n \frac{m}{m_o}\, dt \qquad [m = m(t), m_o = m(o)] \tag{14}$$

To evaluate the integral in (14), we note from Eqs. (6) and (7) that

$$d\dot{m} = \ddot{m}\, dt \tag{15}$$

and

$$m = m_o - \frac{1}{2\ddot{m}} (\dot{m}^2 - \dot{m}_o^2) \quad . \tag{16}$$

The integral is

$$\int_o^{t_b} \ell n \frac{m}{m_o}\, dt = \frac{1}{\ddot{m}} \int_{\dot{m}_o}^{\dot{m}(t_b)} \ell n \left[1 - \frac{1}{2\ddot{m} m_o} (\dot{m}^2 - \dot{m}_o^2) \right] d\dot{m}$$

$$= \frac{1}{\ddot{m}} \int_{\dot{m}_o}^{\dot{m}(t_b)} \ell n \left[\dot{m}^2 - 2\ddot{m} m_o - \dot{m}_o^2 \right] d\dot{m} - \ddot{m} t_b \ell n (-2\ddot{m} m_o) \quad . \tag{17}$$

The integral in (17) is of the form $\int \ell n (x^2 + a^2)\, dx$, where

$$a^2 = - 2\ddot{m} m_o - \dot{m}_o^2$$

is positive if $\ddot{m}$ is less than $(-\dot{m}^2 / 2 m_o)$.

$$\int \ell n (x^2 + a^2)\, dx = x \ell n (x^2 + a^2) - 2x + 2a \tan^{-1} \frac{x}{a} \tag{18}$$

Omitting the intermediate algebra, the resultant position change is

$$\delta R = \left\{ \frac{\dot{m}(t_b)}{\ddot{m}} + 2 \frac{c}{\delta v} \left[t_b + \frac{a}{\ddot{m}} \tan^{-1} \left(\frac{a t_b}{2 m_o - \dot{m}_o t_b} \right) \right] \right\} \delta V \tag{19}$$

It may be shown (but not simply) that the limit of (19) as $\ddot{m}$ approaches zero is the constant mass flow rate solution,

$$\delta R = \left\{ \frac{c}{\delta v} t_b - \frac{m(t_b)}{\dot{m}} \right\} \delta V \tag{20}$$

The only reason why the closed-form solutions, (13) and (19), do not provide exact results (under model assumptions) when added to the thrust-free state at burnout time is that the gravitational acceleration depends on $\delta R(t)$. A simple extension is included in MCBURN's approximate burn to modify the velocity for gravitational accelerations resulting from $\delta R(t)$. It is assumed for this extension that the gravitational acceleration is $(-\mu R/r^3)$. The additional velocity, $\delta\delta V$, is

$$\delta\delta V = -\frac{\mu}{r^3} (I - 3\hat{R}\hat{R}^T) \int_o^{t_b} \delta R(t)\, dt \tag{21}$$

The integral of (21) can be evaluated as follows.

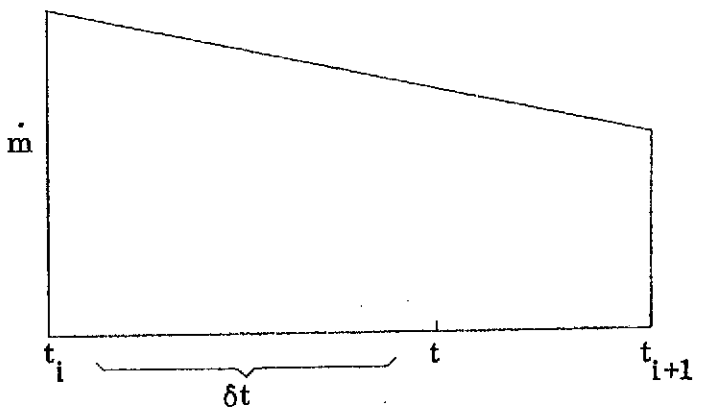
$$\int_o^{t_b} \delta R(t)\, dt = \int_o^{t_b} \left\{ -\frac{\dot{m}(t)}{\ddot{m}} c\, \ell n \frac{m(t)}{m_o} + 2c\left[t + \frac{a}{\ddot{m}} \tan^{-1}\left(\frac{a t}{2 m_o - \dot{m}_o t} \right) \right] \right\} dt\, \hat{\delta V}$$

$$= \left\{ \dot{m}(t_b) \frac{\delta r(t_b)}{\delta v} + \left(m(t_b) - \frac{c\, \delta m}{\delta v} \right) \delta V \right\} \Big/ \ddot{m} \tag{22}$$

$$\delta\delta V = -\frac{\mu}{r^3} (\delta V - 3\hat{R}\hat{R}^T \cdot \delta V) \left\{ \dot{m} \frac{\delta r}{\delta v} + m - \frac{c\, \delta m}{\delta v} \right\} \Big/ \ddot{m} \tag{23}$$

This velocity correction improves the closed-form solution. MCBURN is used in midcourse targeting. When entered with DV, it provides post-burn state where the burn time is computed from Eq. (9). The integrated state differences give rise to end-constraint errors which, through targeting, change DV. The changed DV does not then represent a velocity impulse, but rather an intermediate variable set which gives rise to a changed burn time and thrust direction through MCBURN's formulas. Such a procedure has programing advantages over a procedure which would switch targeting control parameters to time and thrust direction when passing from an impulsive burn to a finite burn. Logic of this subroutine is straightforward and requires no flowchart or block diagram.

Consider the time-segment $\{t_i : t\}$ on which $\dot{m}$ is linear



$$m_o = m(t_i)$$

$$\dot{m}_o = \dot{m}(t_i)$$

$$\delta R_i(t) = \int_{t_i}^{t} \left(- c \, \ell n \, \frac{m(\tau)}{m_o} \right) d\tau$$

$$\delta\delta V_i(t) = \int_{t_i}^{t} \delta R_i(\tau) \, d\tau$$

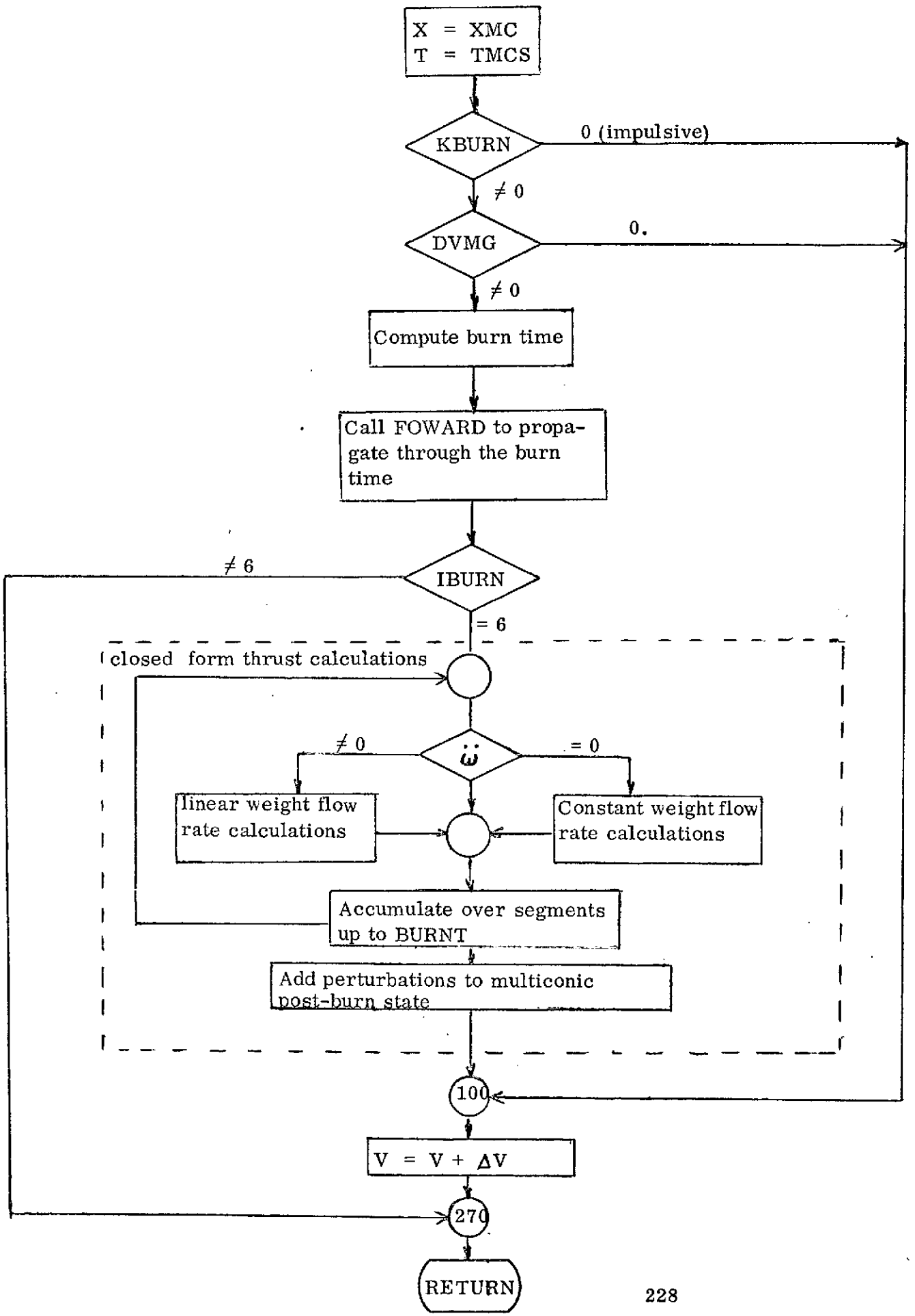
Case 1: $\underline{\dot{m} = + \text{const.}}$, $m = m_o - \dot{m}\,\delta t$, $d\left(\frac{m}{m_o}\right) = - \frac{\dot{m}}{m_o} d\tau$

$$\delta R_i(t) = - c \int_0^{t-t_i} \ell n \left(1 - \frac{\dot{m}}{m_o} \tau \right) d\tau \qquad (\tau = t - t_i = \delta t)$$

$$= - c \left[\frac{1 - \frac{\dot{m}}{m_o}\tau}{-\frac{\dot{m}}{m_o}} \ell n \left(1 - \frac{\dot{m}\tau}{m_o} \right) - \tau \right]_0^{t-t_i}$$

$$= + \frac{c}{\dot{m}} \left[m_o - \dot{m}\tau \; \ell n \left(\frac{m_o - \dot{m}\tau}{m_o} \right) + \dot{m}\tau \right]_0^{t-t_i}$$

$$= \frac{c}{\dot{m}} \left[m \, \ell n \frac{m}{m_o} + \dot{m}\,\delta t \right] = - \frac{m}{\dot{m}} \delta v + c\,\delta t = \underline{\underline{\delta v \left[\frac{c\,\delta t}{\delta v} - \frac{m}{\dot{m}} \right]}}$$


X = XMC
T = TMCS
KBURN
0 (impulsive)
≠ 0
DVMG
0.
≠ 0
Compute burn time
Call FOWARD to propagate through the burn time
≠ 6
IBURN
= 6
closed form thrust calculations
≠ 0
ω̈
= 0
linear weight flow rate calculations
Constant weight flow rate calculations
Accumulate over segments up to BURNT
Add perturbations to multiconic post-burn state
100
V = V + ΔV
270
RETURN

SUBROUTINE MCSET

Calling Sequence: CALL MCSET

Purpose: MCSET performs initializing calculations for midcourse and Monte Carlo analyses.

Common Blocks Required: CONST, INPUT, MCCOM

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	RAD	1	CONST(1)	Radian-to-degrees conversion factor
I	GM	12	CONST(5)	Gravitational constants (km^3/sec^2)
I	DJL	1	INPUT(37)	Julian date at launch (days)
I	DJO	1	INPUT(46)	Julian date at anchor epoch (days)
I	COV	6, 6	INPUT(56)	Tracking error covariance matrix
I	TWD	10	INPUT(320)	Times for weight flow rate table (sec)
I	WD	10	INPUT(350)	Weight flow rate table (kg/sec)
I	PSIDIN	10	INPUT(420)	Desired end constraint values (km, deg, km/sec)
I/O	SIGATM	1	INPUT(435)	Midcourse pointing error (deg→ rad)
I/O	SIGATR	1	INPUT(437)	Retro pointing error (deg→ rad)
I	TMCZIN	1	INPUT(440)	Second midcourse time (sec past DJL)
I	ASPMC	1	INPUT(441)	Specific impulse - midcourse motor (sec)
I	ASPR	1	INPUT(442)	Specific impulse - retro motor (sec)
I	WRETRO	1	INPUT(443)	Weight of retro fuel (kg)
I	RO	1	INPUT(444)	Desired circular orbit radius (km)

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	BVDIN	2	INPUT(447)	Desired miss vector components (km)
I	DINK	1	INPUT(479)	Velocity step for partial generation (km/sec)
I	TOLIN	10	INPUT(490)	Constraint error tolerances
I	JT	1	INPUT(1031)	Target body number
I	MODE	1	INPUT(1044)	Mode switch (3 midcourse, 4 Monte Carlo)
I	IBTRIN	1	INPUT(1062)	Target constraint type key
I	KGLAWI	1	INPUT(1063)	Guidance law selector
I	NGRPTI	1	INPUT(1064)	Number of trials to re-compute gradient
I	NTIN	1	INPUT(1065)	Number of targeting trials allowed
I	MCLIM	1	INPUT(1067)	Factor of DINK for limiting
I	IPROB	1	INPUT(1070)	Probability for scaling purposes (%)
O	ALIMIT	1	MCCOM(1)	Velocity step limit (km/sec)
O	PRX	1	MCCOM(3)	Probability scale factor
O	DV	3	MCCOM(12)	Initial-guess midcourse velocity impulse (km/sec)
O	BVD	2	MCCOM(19)	Desired miss vector (km)
O	DVB4	1	MCCOM(24)	Expended midcourse velocity (km/sec)
O	PFAC	1	MCCOM(38)	Factor used in VTE law calculations (kg)
O	DJDIF	1	MCCOM(39)	Julian date difference, DJO-DJL (sec)
O	TMC2	1	MCCOM(48)	Second midcourse time (sec past DJO)
O	PSID	10	MCCOM(80)	Desired end constraints
O	TOL	10	MCCOM(90)	End constraint tolerances
O	DW	10	MCCOM(110)	Expended fuel table (kg) wrt TWD
O	NT	1	MCCOM(151)	Number of targeting trials allowed
O	NP	1	MCCOM(160)	Number of constraints to test for convergence

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	NGROPT	1	MCCOM(163)	Number of trials to re-compute gradient
O	KGLAW	1	MCCOM(164)	Guidance law indicator
O	IBTR	1	MCCOM(167)	Target constraint-type key
O	IPD	3	MCCOM(169)	Indicators for constraints to test for convergence

Description:

This subroutine is straightforward and requires only a little explanation. First, the computation of the desired 5th constraint is normally circular velocity at desired circular radius, r_d = RO, which is also the desired radius of closest approach, r_p = PSID(1).

$$\text{PSID(5)} = \sqrt{\frac{\mu}{r_p}} + \text{desired overburn velocity}$$

The user may choose to target the VTE law to $r_p > r_d$ in just such a way that the apsidal radius opposite r_p on the post-retro orbit is r_d.

In this case,

$$\text{PSID(5)} = \sqrt{\frac{\mu}{r_p}} \quad \sqrt{\frac{2\, r_d}{r_p + r_d}} \quad .$$

The second calculation worthy of mention is that of the expended fuel weight table, DW. It is computed by integrating the piecewise-linear mass flow rate table, WD.

$$\delta W_{i+1} = \delta W_i + \int_{t_i}^{t_i + 1} \dot{W}_i \, d\tau = \sum_{j=1}^{j=i} \left(\dot{W}_j \, \delta t_j + \frac{1}{2} \ddot{W}_j \, \delta t_j^2 \right)$$

The third calculation is of PRX, used in BELL for scaling the propagated ensemble error statistics. The input IPROB is interpreted as the desired Gaussian probability, P(x).

$$P(x) = \int_{-\infty}^{x} p\,(x)\, dx = \text{erf}\left(\frac{x}{\sqrt{2}}\right)$$

The equation, $\text{erf}\left(\frac{x}{\sqrt{2}}\right) = \text{IPROB}$, is solved iteratively with a Newton-Raphson predictor, beginning with the first guess, x=1.

SUBROUTINE MCVERF

Calling Sequence: CALL MCVERF

Purpose: This subroutine controls the logic flow during Midcourse Verification Analysis

Common Blocks Required: CNTRL, CONST, ELMNT, INPUT, MCCOM, OBSIT, PLNET, STATE

Subroutines Required: CROSS, DVMAG, FOWARD, MVTRN, M50EPM, M50LEQ, PUTELS, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ASPMC	1	INPUT(441)	Specific impulse of the midcourse motor.
I/O	ATT	3	STATE(10)	Attitude unit vector
I	BURNT	1	INPUT(476)	Midcourse motor burn time
I	DAY	1	INPUT(51)	Day of state epoch
O	DELT	10	INPUT(180)	Compute intervals
O	DELTO	1	INPUT(2)	Initial compute interval when in automatic compute interval mode
I	DJO	1	INPUT(46)	Modified julian date of state epoch
I	DJL	1	INPUT(37)	Modified julian date of liftoff epoch
I	DV	3	MCCOM(12)	Midcourse impulsive velocity increment
I	DX	3	STATE(4)	Spacecraft's velocity
O	ERRC	1	INPUT(1)	Error control limit of automatic compute interval
I	HR	1	INPUT(53)	Hour of state epoch
I	HRL	1	INPUT(23)	Hour of launch epoch
I	IDSAT	1	INPUT(1084)	Satellite identification number

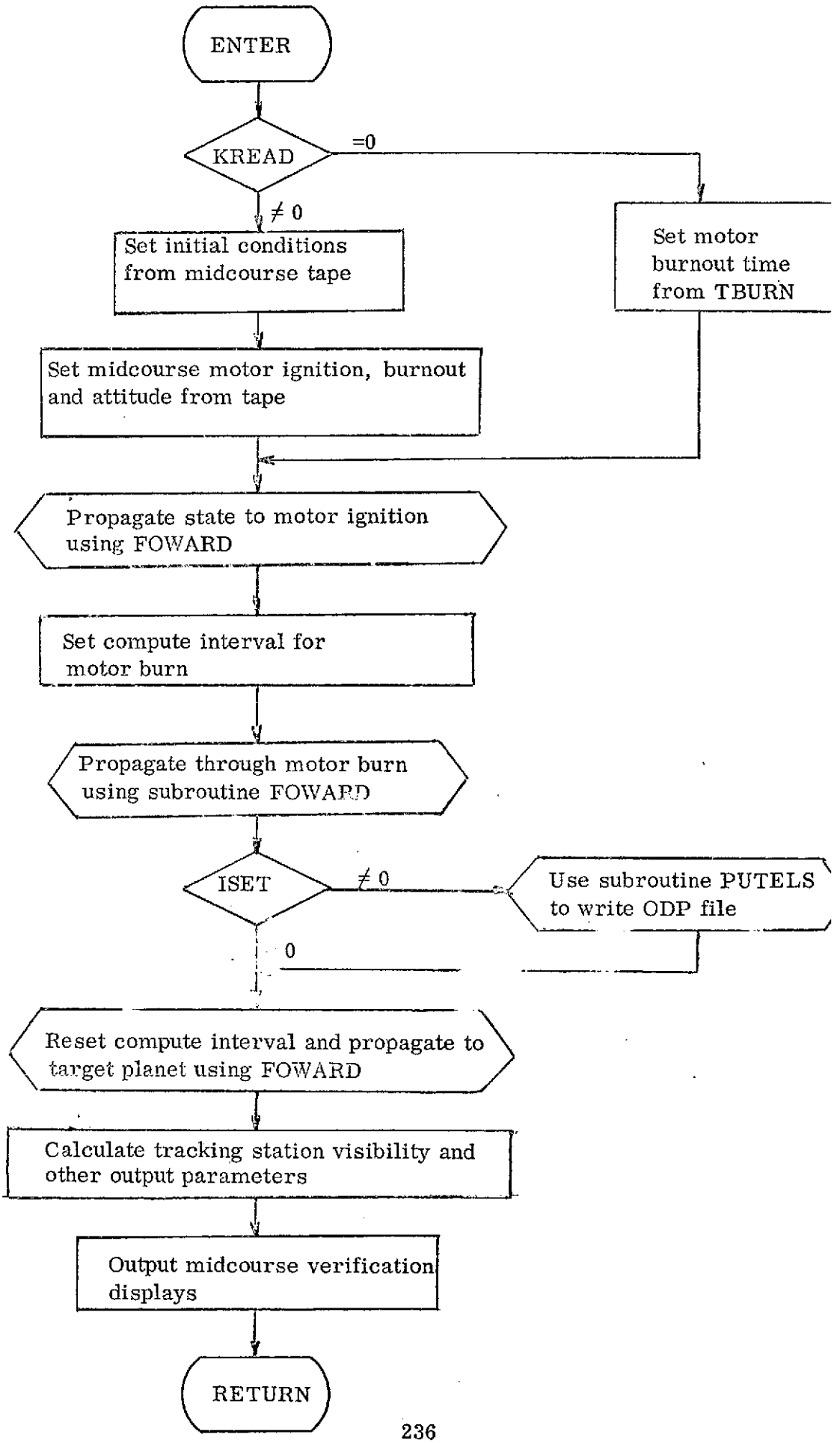
I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ISET	1	INPUT(1092)	Flag used to write out state for the orbit determination program
O	KMETH	1	INPUT(1036)	Initial trajectory propagation indicator
O	KOUT	1	INPUT(1030)	Output frequency flag
I	KREAD	1	INPUT(1057)	Flag to read initial state from midcourse tape
I	MCUNIT	1	INPUT(61)	Midcourse unit number
I	OBSLON	10	INPUT(410)	Longitudes of the tracking stations
I	SEC	1	INPUT(55)	Seconds of state epoch
I	SECL	1	INPUT(25)	Seconds of launch epoch
I	T	1	STATE(10)	Seconds since state epoch
O	TBO	1	INPUT(383)	Burnout time of midcourse engine (engine 1)
O	TCOMP	10	INPUT(170)	Switching times of compute interval table
O	TF	1	INPUT(4)	Final time
O	TIG	1	INPUT(380)	Ignition time of midcourse engine (engine 1)
I	W	1	INPUT(38)	Initial spacecraft mass
I	WT	1	STATE(35)	Current spacecraft mass
I	X	3	STATE(1)	Spacecraft's position vector
I	XMC	6	MCCOM(6)	Spacecraft's position and velocity vectors at midcourse motor ignition
I	XMIN	1	INPUT(54)	Minutes of state epoch
I	XMINL	1	INPUT(24)	Minutes of launch epoch
I	XMON	1	INPUT(50)	Month of state epoch
I	XOBS	10, 3	OBSIT(21)	Position vectors of tracking stations
I	XP	6, 12	PLNET(1)	Position and velocity vectors of the planets
I	YEAR	1	INPUT(52)	Year of state epoch

Description:

The midcourse verification analysis is used to numerically integrate the midcourse motor and present output which describe conditions at motor ignition and at closest approach to the Moon. The conditions at motor ignition, burntime and burn attitude can be input via a tape generated in a previous midcourse analysis. The KREAD flag is used to determine if the tape option is to be used. If the tape is not read, the initial conditions are in X and the burntime and ignition time must be input via TBURN and TIG, respectively. The burn attitude is brought into the subroutine via ATT.

The initial conditions are propagated to midcourse motor ignition time using subroutine FOWARD. The compute interval is adjusted for the motor burn and the midcourse motor is numerically integrated using subroutine FOWARD. A flag is set to obtain the doppler output during the motor burn. This output is obtained from subroutine DOPLER. The state is written on a file to be used by the Orbit Determination Program if ISET is one. Subroutine PUTELS is used to write this file. Next the state is propagated to closest approach to the target planet using FOWARD. Information about the motor burn and conditions at the target planet are printed before the subroutine terminates. The information printed includes midcourse fuel expended, tracking station elevation, azimuth angles at motor ignition,and the state at the target planet.

SUBROUTINE MCVERF



SUBROUTINE MDCORS

Calling Sequence: CALL MDCORS

Purpose: MDCORS is the driver for the midcourse guidance targeting procedure

Common Blocks Required: CNTRL, CONST, INPUT, MCCOM, PLNET, STATE

Subroutines Required: CROSS, DOT, DVMAG, FOWARD, JET, MVTRN, MCBURN, RETDV, SENSO, SPER, VNORM, TARGET

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	JC	1	CNTRL(7)	Central body number
I	RTD	1	CONST(1)	Radian to degree conversion factor
I	TFINAL	1	INPUT(4)	Integration stop time (sec) from anchor epoch
I	WTO	1	INPUT(38)	Initial spacecraft weight (kg)
I	JTARG	1	INPUT(1031)	Target body number
O	KRASH	1	INPUT(1032)	Trajectory stop-type key
I	KMETH	1	INPUT(1036)	Trajectory computation method key
I	MCOUT	1	INPUT(1050)	Midcourse extra output key
I	MCKLUG	1	INPUT(1066)	Pre-targeting option key
I	IBURN	1	INPUT(1071)	Midcourse burn computation option key
I	KMETHP	1	INPUT(1075)	Initial trajectory computation method key
I	KTF	1	INPUT(1077)	Number of extra points in flight time scan
I	ALIMIT	1	MCCOM(1)	Control step limit (km/sec)
O	XMC	6	MCCOM(6)	Pre-ignition midcourse state (km, km/sec)

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	DV	3	MCCOM(12)	Midcourse velocity impulse(km/sec)
O	DVS	3	MCCOM(15)	Spherical DV (mag-km/sec, dec-deg, ra-deg)
I	TMCS	1	MCCOM(18)	Midcourse ignition time (sec) from anchor epoch
O	XSUN	3	MCCOM(21)	Spacecraft-to-sun vector at ignition (unit)
I	DVB4	1	MCCOM(24)	Previous midcourse velocity used (km/sec)
O	DVRET	1	MCCOM(25)	Retro-velocity impulse magnitude (km/sec)
I	PFAC	1	MCCOM(38)	Propulsion factor (kg, km/sec)
O	WTF	1	MCCOM(47)	Spacecraft weight (kg) after midcourse burn
I	DPT	3,10	MCCOM(50)	Constraint/control sensitivity matrix transposed
I	PSID	10	MCCOM(80)	Desired end condition vector
I	TOL	10	MCCOM(90)	Tolerances on end constraint errors
I	PSI	10	MCCOM(100)	End constraint error vector
I	NT	1	MCCOM(151)	Number of trials allowed in targeting
O	KBURN	1	MCCOM(154)	Burn computation method indicator
O	IT	1	MCCOM(157)	Running number of trials
O	IR	1	MCCOM(158)	Return key for targeting status
I	NP	1	MCCOM(160)	Number of constraints in targeting process
I	NGROPT	1	MCCOM(163)	Number of trials to re-compute gradients
I	KGLAW	1	MCCOM(164)	Guidance law indicator
O	ICB	1	MCCOM(165)	Midcourse central body number
O	ISP	1	MCCOM(166)	Gradient-computed indicator
I	IPD	3	MCCOM(169)	Constraint indicator vector
I	X	6	STATE(1)	Anchor vector state
I	T	1	STATE(10)	Anchor time (sec) anchor epoch, DJO

Description:

MDCORS is best described with reference to its accompanying flow charts. The first of these depicts the gross targeting logic.

The first step in MDCORS is to propagate the state, X_2, forward from T to the midcourse time, TMCS, by calling FOWARD. The resulting pre-ignition midcourse state, XMC, is then saved. If the pre-targeting option key, MCKLUG, is positive, JET is then called to furnish a first-guess value for DV. If MCKLUG is zero, the starting value for DV is obtained from common where it was placed either by input or by previous targeting. Having thusly initialized, targeting is begun.

Subroutine SENSO is called to transform XMC and DV into an end constraint error vector, PSI. SENSO effects this transformation in the following three steps:

1. XMC and DV are converted into a post-maneuver state, X, and time, T, by subroutine MCBURN.
2. The post-maneuver state is propagated to the point and time of target closest approach by subroutine FOWARD.
3. The end state is used to compute end constraint function values which are subtracted from desired values to render the constraint error vector, PSI.

PSI is dimensioned 10, although only 8 of its components are used. These components represent the following errors:

PSI(1)	B·T, miss-vector component
PSI(2)	B·R, miss-vector component
PSI(3)	Time of flight
PSI(4)	Hyperbolic excess velocity of arrival hyperbola.
PSI(5)	Circular excess velocity after retro at periapsis
PSI(6)	Total correction fuel expended (value, not error)
PSI(7)	Radius at periapsis of the arrival hyperbola
PSI(8)	Inclination of hyperbola to target's equator

The tolerance array, TOL, is similarly defined, so that the criterion for convergence is

$$PSI(I) \leq TOL(I)$$

for each of the end constraints associated with the particular guidance law in force. Each of the guidance laws available in MDCORS constrains the same first two functions: B·T and B·R if IBTR=1 or radius at periapsis and inclination if IBTR=2. (IBTR is an input quantity used by AUTO to set the array, IPD). The remaining function constrained by each law is:

1. Minimum (midcourse) fuel	None
2. Fixed time of arrival	PSI(3)
3. Fixed target energy	PSI(4)
4. Variable target energy	PSI(5)
5. Minimum total fuel	PSI(3) * See MTF procedure description

If one or more of the errors for the guidance law in force exceeds tolerance, a new estimate of DV is computed. Details of this computation will be described later. If a new gradient is to be generated (iteration trial number less than input, NGROPT), SENSO is called with IR=2 to generate it. SENSO uses the secant method (or finite increments of DV) in repeating the three steps described above to compute sensitivities of PSI to variations in DV. The result is interpreted as $\frac{\partial PSI}{\partial DV}$ evaluated at DV. Its transpose, a 3 x 10 matrix is stored in DPT. The last two columns of DPT are undefined. When the iteration trial number equals or exceeds NGROPT, no new gradient is computed. In this case, the last-computed DPT is used for succeeding iteration trials in computation of DV. By not computing new gradients at each trial, three trajectories per trial may be saved. The resulting deterioration in convergence is small if DV is "near" its final value when gradient computation is terminated. Results show that when pre-targeting is performed, a single evaluation of the gradient at the first guess DV is optimal in terms of total trajectories required and run time.

Before discussing the computation of DV, let us consider the post-targeting logic. This consists of tests for jumping back into the targeting logic. First, if a minimum midcourse

fuel iteration has converged on the 0th step, another step is forced. This is done because only the first two constraint errors are tested for the MFG law. Tne solution DV for any other law would satisfy these two constraints as well, without necessarily being a minimum DV or "critical plane" maneuver. The second reason for jumping back into the targeting logic is due to the minimum total fuel law implementation. The reason will become apparent when this implementation is described. The third jump-back is because of the approximate-intermediate-trajectory capability. The initial estimate of DV is obtained using trajectory method KMETHP, which should be set to 6 for multi-conic for translunar trajectories. If it is desired that DV be estimated for a more precise trajectory computation method, KMETH $\neq$ KMETHP, the initial estimate of DV is used to re-start the KMETH targeting procedure. It should never be necessary to re-generate gradients for this additional targeting.

Estimation of ΔV

The method of calculation of ΔV is dependent on the guidance law in force, although each guidance law is designed to constrain two common end conditions, i.e., B·T and B·R or radius at periapsis and inclination. It has been found that even when the latter conditions are to be constrained, B·T and B·R errors should be used to calculate ΔV. That is, desired values of radius at periapsis and inclination should be used, together with hyperbolic excess velocity (vector), to compute the "desired" B·T and B·R and their corresponding errors, Ψ_1 and Ψ_2. When these errors are nulled, the radius of periapsis and inclination errors will be nulled also. Formulae for computing "desired" B·T and B·R are to be found in Reference 1.

Minimum (midcourse) Fuel Guidance (MFG)

The MFG law minimizes the magnitude of ΔV while constraining the B·T and B·R errors, Ψ_1 and Ψ_2. It is desirable from a programing point of view to avoid numerical minimization procedures if possible - which it is in this case. We could null Ψ_1 and Ψ_2 (linearly speaking) with any ΔV such that

$$-\begin{pmatrix}\Psi_1\\ \Psi_2\end{pmatrix} = \begin{pmatrix}R_1\\ R_2\end{pmatrix} \Delta V$$

where R_1 and R_2 are rows of the 2 x 3 gradient matrix, $\frac{2\Psi}{2\Delta V}$. The family of such ΔV's is

$$\Delta V = - \frac{\begin{pmatrix} R_2^t \times R_3^t & R_3^t \times R_1^t & R_1^t \times R_2^t \end{pmatrix} \begin{pmatrix} \Psi_1 \\ \Psi_2 \\ \Psi_3 \end{pmatrix}}{(R_1 \times R_2)^t R_3}$$

where Ψ_3 is a free parameter and R_3 is any vector which is linearly independent of R_1 and R_2. That is, if we adjoin a third row, R_3, to the gradient, we can invert the results and solve for ΔV. Making the substitution C_i for columns of the inverse, we can write

$$\Delta V = -(C_1\ C_2\ C_3) \begin{pmatrix} \Psi_1 \\ \Psi_2 \\ \Psi_3 \end{pmatrix} = - C\Psi$$

and $$\delta v^2 = \psi^t C^t C \psi .$$

We now minimize δv^2 with respect to the free parameter, Ψ_3, by setting the derivative to zero and solving for Ψ_3.

$$\frac{d\,\delta v^2}{d\Psi_3} = 2\,\Psi^t C^t C \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 2\,(\Psi_1 C_1 + \Psi_2 C_2 + \Psi_3 C_3)^t C_3 = 0$$

$$\Psi_3 = - \frac{(\Psi_1 C_1 + \Psi_2 C_2)^t C_3}{C_3^t C_3}$$

This Ψ_3 leads to the following ΔV,

$$\Delta V = - \left(I - \frac{C_3 C_3^t}{C_3^t C_3} \right) (C_1\ C_2\ C_3) \begin{pmatrix} \Psi_1 \\ \Psi_2 \\ \Psi_3 \end{pmatrix}$$

where Ψ_3 can have any value, since $\left(I - \frac{C_3 C_3^t}{C_3^t C_3} \right) C_3 = 0$. The vector, C_3, is normal to the "critical plane" defined by R_1 and R_2, and the minimum-magnitude ΔV lies in this plane. If the adjoined row, R_3, is defined as the cross-product of R_1 and R_2, the implemented solution is identical to the steepest ascent solution. The minimum fuel

guidance solution is computed by first computing

$$\Delta V^{*} = -(C_1\ C_2\ C_3)\begin{pmatrix}\psi_1\\ \psi_2\\ 0\end{pmatrix}$$

and then projecting the result onto the critical plane.

$$\Delta V = \Delta V^{*} - \left(\frac{C_3^{t}\ \Delta V^{*}}{C_3^{t}\ C_3}\right) C_3$$

Since the constraint functions are not strictly linear functions of ΔV, an iteration is required to find the solution, and the gradient varies during the iteration. And since the "critical plane" may change somewhat during an iteration, it is important to project the whole ΔV solution onto the current critical plane rather than simply to project the incremental ΔV due to residual constraint errors.

Fixed Time of Arrival (FTA) Guidance

The end constraints for this law are $B\cdot T$, $B\cdot R$ and time of flight. Representing errors in these constraints by Ψ_1, Ψ_2 and Ψ_3, respectively, we need a ΔV such that

$$-\begin{pmatrix}\psi_1\\ \psi_2\\ \psi_3\end{pmatrix} = \begin{pmatrix}R_1\\ R_2\\ R_3\end{pmatrix}\Delta V$$

where $R_i = \dfrac{\partial \Psi_i}{2\ \Delta V}$ is a row of the gradient, $\dfrac{\partial \Psi}{2\ \Delta V}$. The solution is

$$\Delta V = -(C_1\ C_2\ C_3)\begin{pmatrix}\psi_1\\ \psi_2\\ \psi_3\end{pmatrix}$$

where

$$(C_1\ C_2\ C_3) = \frac{1}{(R_1 \times R_2)^{t} R_3}\left((R_2 \times R_3)^{t}\ (R_3 \times R_1)^{t}\ (R_1 \times R_2)^{t}\right).$$

Fixed Target Energy (FTE) Guidance

The only difference between the FTE and FTA laws is that the third constraint for the FTE law is hyperbolic excess speed with respect to the target at arrival.

Variable Target Energy (VTE) Guidance

Application of the VTE law is peculiar to missions where the retro-burn at the target is an attitude-controllable fixed impulse. The VTE law is set up to calculate the midcourse maneuver ΔV for which the post-retro velocity is a desired value (especially circular) while constraining B·T and B·R or closest approach distance and inclination.

The post-retro velocity is dependent upon the weight of fuel expended at midcourse and the arrival energy. Implementation of this law assumes that the retro-burn is executed at periapsis of the arrival hyperbola with the thrust deflected parallel to the periapsis velocity. The constraint error, Ψ_3, for this law is formulated as "circular excess" velocity.

$$\Psi_3 = \sqrt{\frac{\mu}{r_d}} + \epsilon - \sqrt{v_\infty^2 + \frac{2\mu}{r_d}} - \delta v_r$$

In this formulation, r_d is the desired final orbit radius, ϵ is an input desired circular excess velocity, and δv_r is the velocity impulse magnitude imparted by the retro-burn. Calculation of an appropriate midcourse ΔV for the VTE law can be formally identical to that for the FTA and FTE laws, given a good first-guess ΔV to start the iteration. An "assistance" procedure is beneficial, however, to reduce convergence difficulties which may arise from nonlinearities of $\Psi_3(\Delta V)$. This procedure consists of an iteration loop imbedded in the Newton-Raphson iteration loop to predict changes in δv_r due to next-step changes in ΔV. Reference 1 contains a detailed description of this procedure.

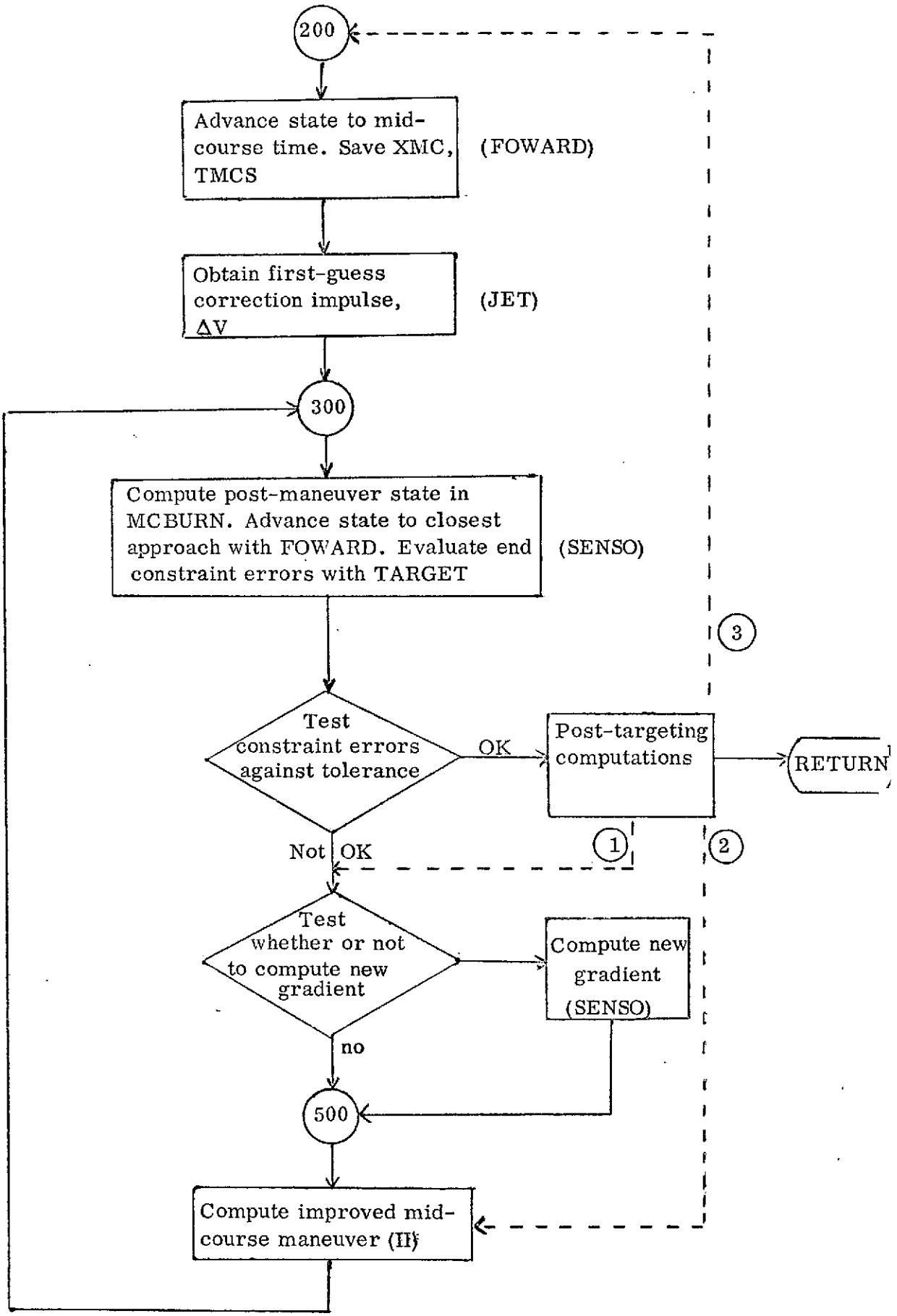
Minimum Total Fuel (MTF) Guidance

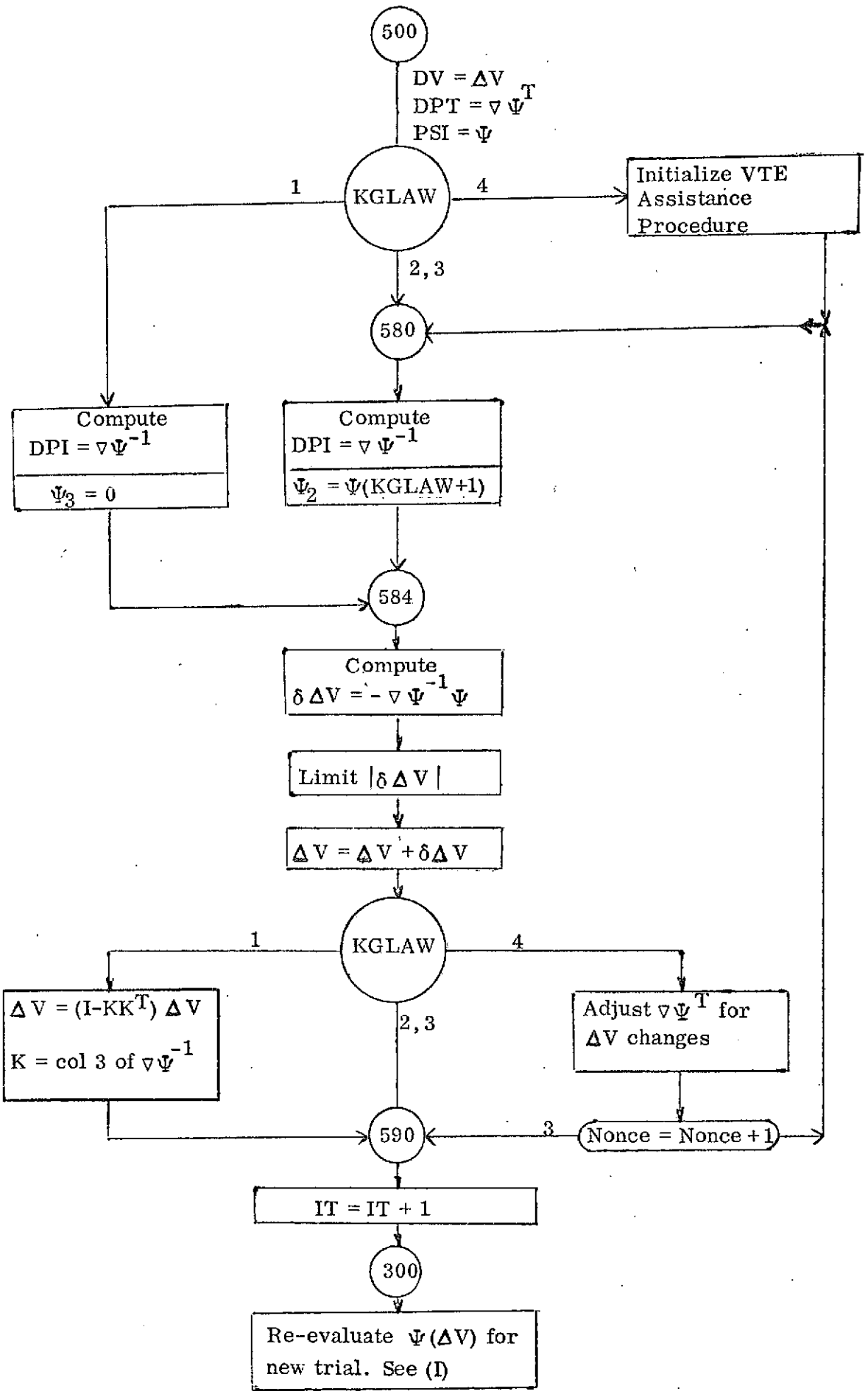
The implemented MTF guidance solution is the restricted solution which constrains approach conditions, B·T and B·R or radius at periapsis and inclination. In this respect, the implemented solution is neither general nor optimal. Results of tests indicate,

however, that for missions of the RAE-B type, the implemented solution will differ from the optimal by only a few hundredths of a kilogram in total fuel. The total correction fuel (when arrival conditions are constrained) is conveniently parameterized by flight time to target periapsis. The MTF solution is therefore obtained by solving for the FTA guidance solution within a loop which minimizes total fuel as a function of flight time. The minimization procedure will be found in the description of subroutine MINTF. The procedure is initiated at the MFG flight time, since the MFG solution is never far from the MTF solution. It is unnecessary to recompute gradients for obtaining rapid FTA convergence within the MTF procedure.

Reference 1 Bjorkman, W.S., Midcourse Guidance for Lunar and Planetary Orbiting Missions, AMA 71-16, March, 1971.

SUBROUTINE MDCORS I (Gross Targeting Logic)




500
DV = ΔV
DPT = ∇Ψ^T
PSI = Ψ
KGLAW
1
4
2, 3
Initialize VTE Assistance Procedure
580
Compute DPI = ∇Ψ^-1
Ψ3 = 0
Compute DPI = ∇Ψ^-1
Ψ2 = Ψ(KGLAW+1)
584
Compute δΔV = - ∇Ψ^-1 Ψ
Limit |δΔV|
ΔV = ΔV + δΔV
KGLAW
1
4
2, 3
ΔV = (I-KK^T) ΔV
K = col 3 of ∇Ψ^-1
Adjust ∇Ψ^T for ΔV changes
590
3
Nonce = Nonce + 1
IT = IT + 1
300
Re-evaluate Ψ(ΔV) for new trial. See (I)

SUBROUTINE MINTF

Calling Sequence: CALL MINTF

Purpose: MINTF controls the gross logic for targeting the minimum total fuel guidance law.

Common Blocks Required: MCCOM

Subroutines Called: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	PSID	10	MCCOM(80)	Desired constraint vector - esp. PSID(3) = time of flight (sec)
I	TOL	10	MCCOM(90)	TOL(3)=flight time tolerance(sec) TOL(6)=total fuel tolerance (kg)
I/O	PSI	10	MCCOM(100)	PSI(3) = flight time error (sec) PSI(6)=total fuel (kg)
O	KM	1	MCCOM(152)	Output key for MDCORS logic
I/O	JUMPTF	1	MCCOM(153)	Logic key for MDCORS & MINTF
O	I3	1	MCCOM(155)	Third constraint indicator
O	IT	1	MCCOM(157)	Trial counter for MDCORS
O	NP	1	MCCOM(160)	Number of constraints
O	KGLAW	1	MCCOM(164)	Guidance law indicator
O	IPD	3	MCCOM(169)	Constraint indicator array

Description:

The minimum total fuel guidance law solution minimizes total correction fuel subject to equality constraints on closest approach distance and inclination. To effect this solution, the fixed time of arrival guidance (FTA) law is invoked with systematic variation of flight time. MINTF contains the logic for varying flight time to find the minimum total fuel solution.

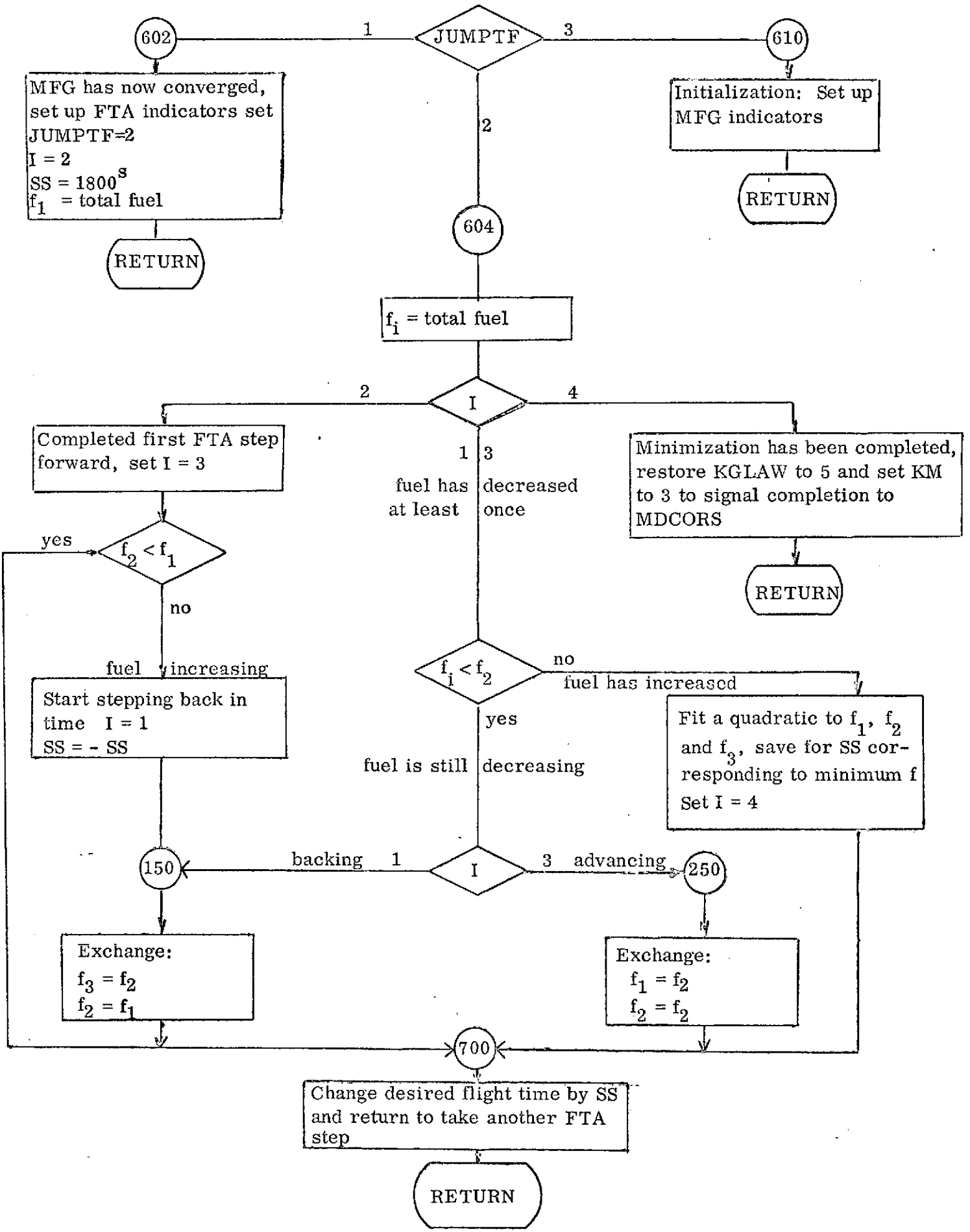
The first time MDCORS calls MINTF, MINTF sets up indicators for the MFG (minimum midcourse fuel) law. When MINTF is called again, the MFG solution has been found, with its corresponding values of total fuel and time of flight. MINTF initializes the FTA law at this time and requests a flight time increase of 1800 seconds on return to MDCORS. On subsequent entries, MINTF seeks the minimum total fuel flight time by the following procedure:

1. If, on return from targeting the FTA law to $t_{MFG} + 1800^{s}$, it is found that the total fuel is,
   a. less than it was for the MFG solution, flight time is stepped forward until the total fuel increases.
   b. greater than it was for the MFG solution, flight time is stepped backward in 1800^{s} - steps until the total fuel increases.
2. The result of (1) is three values of total fuel, f(t), corresponding to three different flight times. These obey the following inequality.

$$f_{n-1} \geq f_n \geq f_{n+1} \quad , \quad f_n = f(t_{MFG} + n * 1800^{s})$$

The three values are used to determine the coefficients of a quadratic equation describing total fuel as a function of flight time. This equation is then solved for the flight time corresponding to the minimum of the quadratic equation. (See MINV for details of fitting a quadratic equation.) This flight time is then used for the final FTA targeting step and the resultant total fuel is assumed to be the minimum value.

SUBROUTINE MINTF



SUBROUTINE MINV

Calling Sequence: CALL MINV (ELM, RA, DEC, DVR, DVT, FFIRE)

Purpose: MINV finds the retro-firing true anomaly of minimum trim velocity.

Common Blocks Required: CONST, INPUT

Subroutines Called: CROSS, DOT, DVMAG, ORBIT, ORIENT, TRIM, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ELM	12	ARGUMENT LIST	Orbital elements of the hyperbola (LEQ)
I	RA	1	ARGUMENT LIST	Right ascension of the spin-axis (LEQ, rad)
I	DEC	1	ARGUMENT LIST	Declination of the spin-axis (LEQ, rad)
I	DVR	1	ARGUMENT LIST	Retro velocity impulse (km/sec)
O	DVT	1	ARGUMENT LIST	Trim velocity required (km/sec)
O	FFIRE	1	ARGUMENT LIST	Firing true anomaly (rad)
I	RAD	1	CONST(1)	Radians-to-degrees conversion factor (deg/rad)
I	GM	12	CONST(5)	Gravitational constant array (km^3/sec^2)
I	TRUE	1	INPUT(475)	Limiting true anomaly allowed (deg)
I	JT	1	INPUT(1031)	Target body number
I	KFIRE	1	INPUT(1043)	Number of true anomaly steps to try

Method

The method used by MINV in trying to establish the minimum trim velocity and corresponding firing true anomaly is:

1. Step along the hyperbola in KFIRE regular steps from -TRUE to +TRUE, computing and storing required trim velocity at each step.

2. Fit a quadratic (trim velocity as a function of true anomaly) about the point of least trim velocity as found in procedure 1. Solve for the minimum trim velocity and corresponding true anomaly as coordinates of the quadratic's minimum.

Subroutine TRIM is used by MINV to determine the trim velocity. TRIM, when supplied with the post-retro orbit (which must be elliptical), computes the trim velocity as a two-impulse in-plane Hohmann transfer plus a nodal inclination-trim impulse. A large part of MINV's coding is concerned with establishing true anomaly bounds within which the post-retro orbit will be elliptical. These bounds are determined from the condition,

$$V(f) \cdot \Delta V = \frac{\mu}{h} - \hat{P} \cdot \Delta V \sin f + \hat{Q} \cdot \Delta V (e + \cos f) < - \frac{C_{3h} + \delta v^2}{2}$$

which must hold in the elliptical range. $\hat{P}$ and $\hat{Q}$ are unit vectors toward periapsis and along periapsis velocity, respectively, C_{3h} is the hyperbola's C_3, and f is true anomaly. The other symbols are standard enough. The equation may be solved for f (two solutions usually) by replacing the inequality with an equality. If the input limits are outside the computed bounds, they are replaced by the computed bounds as limits for the stepping procedure (see 1 above).

If the least trim velocity from the stepping procedure occurs at the first or last step, this limit value is returned from MINV. Otherwise, the least trim velocity occurs at the i-th point and the quadratic fit coefficients, a, b and c are determined as follows,

$$v(f) = a (f-f_i)^2 + b (f-f_i) + c$$

$$v(f_{i+1}) = a (f_{i+1} - f_i)^2 + b (f_{i+1} - f_i) + c = v_{i+1}$$

$$v(f_i) = c = V_i$$

$$v(f_{i-1}) = a\,(f_{i-1} - f_i)^2 + b\,(f_{i-1} - f_i) + c = v_{i-1}$$

Since the true anomaly step-size is constant,

$$\delta f = f_{i+1} - f_i = f_i - f_{i-1} ,$$

the solution for the coefficients is

$$a = \frac{1}{2\delta f^2} \quad (v_{i+1} + v_{i-1}) - 2v_i$$

$$b = \frac{1}{2\delta f} \quad (v_{i+1} - v_i)$$

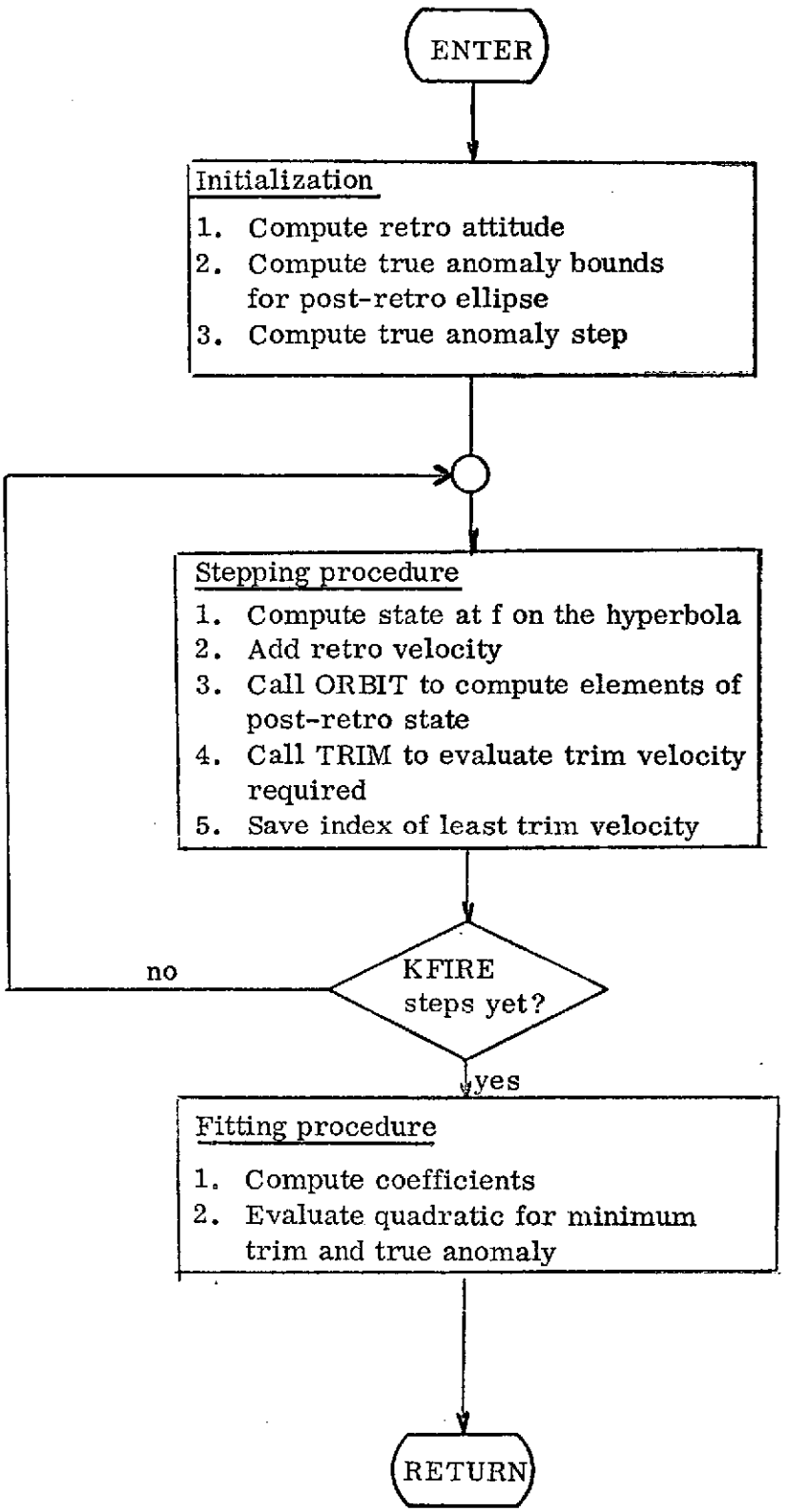
$$c = v_i$$

The minimum (extremum) occurs where

$$\frac{dv}{df} = 2\,a\,(f - f_i) + b = 0$$

or $$f_{min} = f_i - \frac{b}{2a} \quad .$$

The minimum trim velocity (returned by MINV) is computed by evaluating the quadratic for $v\,(f_{min})$.


ENTER
Initialization
1. Compute retro attitude
2. Compute true anomaly bounds for post-retro ellipse
3. Compute true anomaly step
Stepping procedure
1. Compute state at f on the hyperbola
2. Add retro velocity
3. Call ORBIT to compute elements of post-retro state
4. Call TRIM to evaluate trim velocity required
5. Save index of least trim velocity
KFIRE steps yet?
no
yes
Fitting procedure
1. Compute coefficients
2. Evaluate quadratic for minimum trim and true anomaly
RETURN

SUBROUTINE MONTE

Calling Sequence: CALL MONTE

Purpose: MONTE performs a Monte Carlo analysis of success probability for the RAE-B mission.

Common Blocks Required: ANKOR, CONST, INPUT, MCCOM, STATE

Subroutines Called: BIGMAT, COVERT, FOWARD, MCBURN, MDCORS, MVTRN, M50LEQ, ORBIT, RANDM1, RETRO, SENSO, TRIM, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ANKVEC	6	ANKOR(1)	Anchor vector state (km, km/sec)
I	RAD	1	CONST(1)	Degrees per radian
I	TDS	1	CONST(43)	Seconds per day
I	TSH	1	CONST(42)	Hours per second
I	GM	12	CONST(5)	Gravitational constants (km^3/sec^2)
I	G	1	CONST(45)	Earth's surface gravity (km/sec^2)
I	TF	1	INPUT(4)	Trajectory stop time (sec)
I	DJL	1	INPUT(37)	Julian date at launch (days)
I	WO	1	INPUT(38)	Initial spacecraft weight (kg)
I	DJO	1	INPUT(46)	Julian date at anchor epoch (days)
I	HRO	1	INPUT(53)	Hours of anchor epoch
I	XMINO	1	INPUT(54)	Minutes of anchor epoch
I	SECO	1	INPUT(55)	Seconds of anchor epoch
I	COV	6, 6	INPUT(56)	Tracking covariance matrix
I	DTFIN	1	INPUT(422)	Desired time of flight (seconds)

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	SIGATM	1	INPUT(435)	Midcourse pointing error (rad)
I	SIGDVM	1	INPUT(436)	Midcourse velocity error (fraction)
I	SIGATR	1	INPUT(437)	Retro pointing error (rad)
I	SIGDVR	1	INPUT(438)	Retro velocity error (fraction)
I	TMC1	1	INPUT(439)	First midcourse time (sec)
I	TMC2	1	INPUT(440)	Second midcourse time (sec)
I	ASPMC	1	INPUT(441)	Specific impulse of midcourse and trim motor (sec)
I	ASPR	1	INPUT(442)	Specific impulse of retro motor (sec)
I	WRETRO	1	INPUT(443)	Weight of retro fuel (kg)
I	ATFULA	1	INPUT(470)	Available attitude maneuver fuel (kg)
I	FTOT	1	INPUT(472)	Available correction fuel (kg)
I	WDROP	1	INPUT(473)	Post-retro drop-weight (kg)
I	JT	1	INPUT(1031)	Target body number (11)
O	KCRASH	1	INPUT(1032)	Trajectory stop-type indicator
I	KMONTE	1	INPUT(1052)	Monte Carlo logic key
I	KMAX	1	INPUT(1053)	Monte Carlo sample size
I/O	KSTART	1	INPUT(1054)	Random number kernel
O	MCKLUG	1	INPUT(1066)	Midcourse pre-targeting key
I	KMTOUT	1	INPUT(1074)	Extra output flag
I	KCOV	1	INPUT(1085)	Covariance matrix conversion key
O	XMC	6	MCCOM(6)	Pre-ignition midcourse state (km, km/sec)
I/O	DELV	3	MCCOM(12)	Midcourse velocity impulse (km/sec)
I/O	DVMG	1	MCCOM(15)	Midcourse velocity magnitude (km/sec)
O	TMC	1	MCCOM(18)	Midcourse ignition time (sec)

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	DV	1	MCCOM(24)	Accumulated correction velocity
I/O	DPT	30	MCCOM(50)	Constraint/control sensitivity matrix
O	DTF	1	MCCOM(82)	Desired time of flight from anchor epoch (sec)
O	KENTRY	1	MCCOM(156)	Entry key for MDCORS
O	IR	1	MCCOM(158)	Sensitivity generation key for SENSO
O	NGROPT	1	MCCOM(163)	Gradient re-computation key for MDCORS
I	ISP	1	MCCOM(166)	Gradient-calculated indicator from SENSO
O	X	6	STATE(1)	Trajectory state (km, km/sec)
O	T	1	STATE(10)	Time from epoch (sec)
I	ATT	3	STATE(11)	Spin-axis attitude (unit vector)

Description:

MONTE simulates the events of the RAE-B mission with sampled random errors to establish the probability of mission success. The events to be simulated for a given case are specified by KMONTE.

KMONTE = 1 Retro plus trim only - no midcourses

KMONTE = ±2 One midcourse correction plus retro and trim

KMONTE = ± 3 Two midcourse corrections plus retro and trim

If KMONTE is negative, the first (or only) midcourse correction maneuver is calculated for the estimated (anchor vector) trajectory and not re-calculated for sampled errant anchor vectors. If KMONTE is +2 or +3, the first (or only) midcourse correction is re-calculated for each sampled errant trajectory. The positive KMONTE case is applicable to pre-flight studies for which the "tracking" error covariance matrix in reality describes expected trajectory dispersions due to expected launch and injection errors.

The event logic as implemented is fairly straightforward as seen in the principal flow

chart. Some of the logic associated with midcourse maneuver computation requires some explanation, however. In order to avoid redundant computations in MDCORS, the first-maneuver correction impulse is computed and saved for the anchor vector trajectory. The constraint/control sensitivity matrix or gradient is also saved. If two midcourse maneuvers are simulated, the gradient at the second correction is also computed about the once-corrected anchor trajectory and stored. The stored gradients are good enough to bring about convergence for the midcourse maneuver calculations related to perturbed trajectories, so the gradient re-computation key, NGROPT, is set zero for the Monte Carlo process. The first-maneuver correction impulse computed for the anchor trajectory is a good first-guess for successive midcourse calculations, so the pre-targeting key, MCKLUG, is set zero as well. The first-guess value for the second maneuver can be taken as zero if the first maneuver execution errors are small. The logic shown in flow charts A and B avoids redundant and time-consuming calculations in MDCORS in the computation of midcourse correction maneuvers.

Error Models

The tracking error covariance matrix, P, is a 6 x 6 positive definite matrix of anchor vector estimation errors.

$$P = E \quad (x - \tilde{x})\,(x - \tilde{x})^t \qquad (\tilde{x} = \text{anchor vector, } x = \text{true state})$$

There is a preferred coordinate system in which components of the error vector, y, are uncorrelated. The (similarity) transformation between y and $x - \tilde{x}$ is S, an orthogonal matrix which diagonalizes P.

$$x - \tilde{x} = S_y$$

The diagonal matrix, D, defined by

$$D = S^t PS,$$

has as its diagonal elements the variances of the uncorrelated components of y. Scaling a white noise (uncorrelated random numbers) sampled by the standard deviations of y, we obtain an error vector in uncorrelated coordinates with components, yi.

$$yi = d_i \; n(0,1)$$

(d_i is the i-th diagonal element of D, n (0, 1) is a sample random number from a normal (Gaussian) distribution of mean 0 and variance 1.) A sampled errant trajectory state is then computed from

$$x = \tilde{x} + Sy.$$

Execution errors for the midcourse and retro maneuvers are computed from assigned standard deviations of pointing and velocity errors. The velocity error is treated as a proportional error normally distributed plus a resolution error uniformly distributed.

$$\epsilon_v = v\,\sigma_v\, n\,(0,1) + (.0001 \text{ km/sec})\, u$$

(v is the velocity impulse magnitude and u is a random number from a distribution uniform on the interval $\{-.5 < u < .5\}$). The retro velocity error is formulated without a resolution error. The pointing error is formulated as two independent errors normally distributed along mutually orthogonal axes which are both orthogonal to the maneuver impulse direction. Let the maneuver impulse be denoted by V and its direction by $\hat{V}$. If $\hat{K}$ is the unit polar axis of V's coordinate reference frame, we can construct unit vectors normal to V.

$$\hat{E} = \frac{\hat{K} \times \hat{V}}{\left|\hat{K} \times \hat{V}\right|} = \frac{\begin{matrix} -v_2 \\ v_1 \\ 0 \end{matrix}}{\sqrt{v_1^2 + v_2^2}} = \frac{\begin{matrix} -v_2 \\ v_1 \\ 0 \end{matrix}}{v \cos(\text{dec})}$$

$$\hat{N} = \hat{V} \times \hat{E} = \frac{\hat{K} - \hat{V}\,(\hat{K} \cdot \hat{V})}{\cos(\text{dec})}$$

The "eastward" and "northward" pointing errors, θ_e and θ_n, are computed by scaling random numbers with the input pointing erros, σ_a. They are then converted to velocity deviations using the small angle approximation.

$$\epsilon_e = v\,\sigma_a\, n_1\,(0,1)$$

$$\epsilon_n = v\,\sigma_a\, n_2\,(0,1)$$

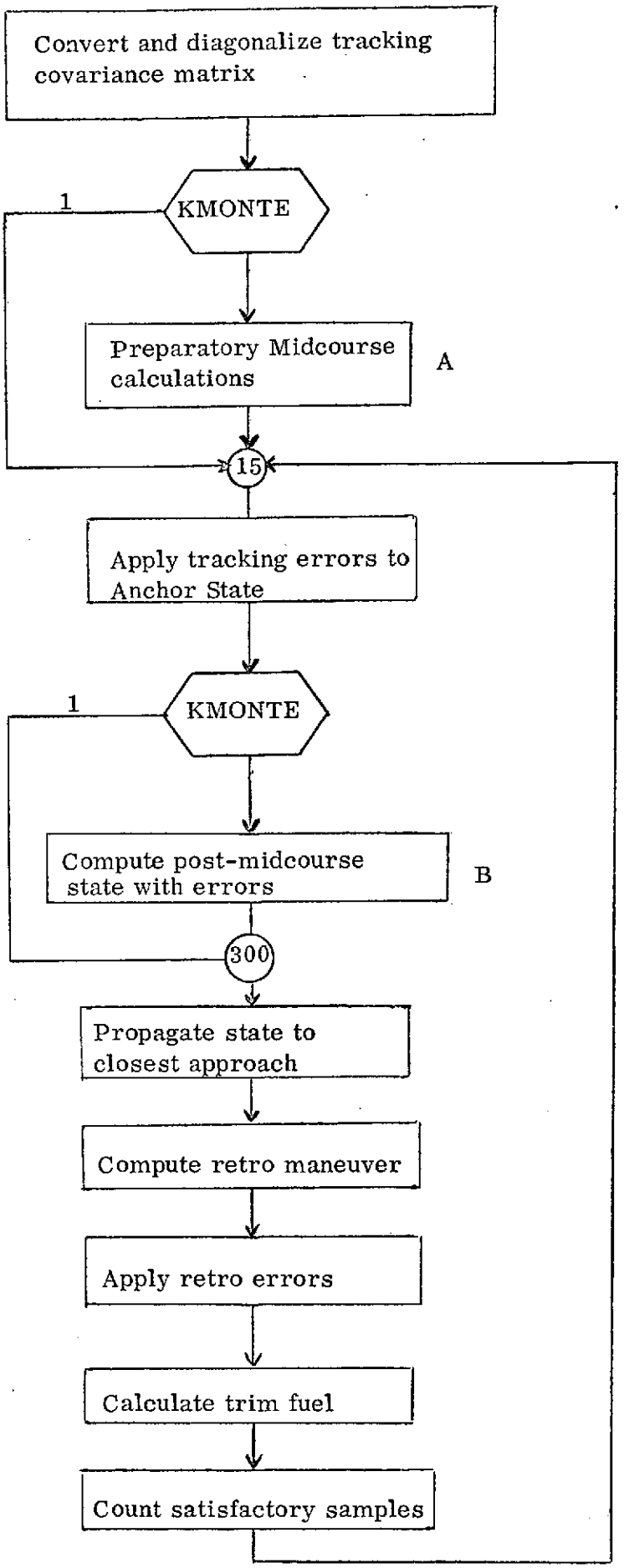
The misdirected velocity vector has the direction

$$\hat{V}' = (v\,\hat{V} + \epsilon_e\,\hat{E} + \epsilon_n\,\hat{N}) \,/\, \sqrt{v^2 + \epsilon_e^2 + \epsilon_n^2}$$

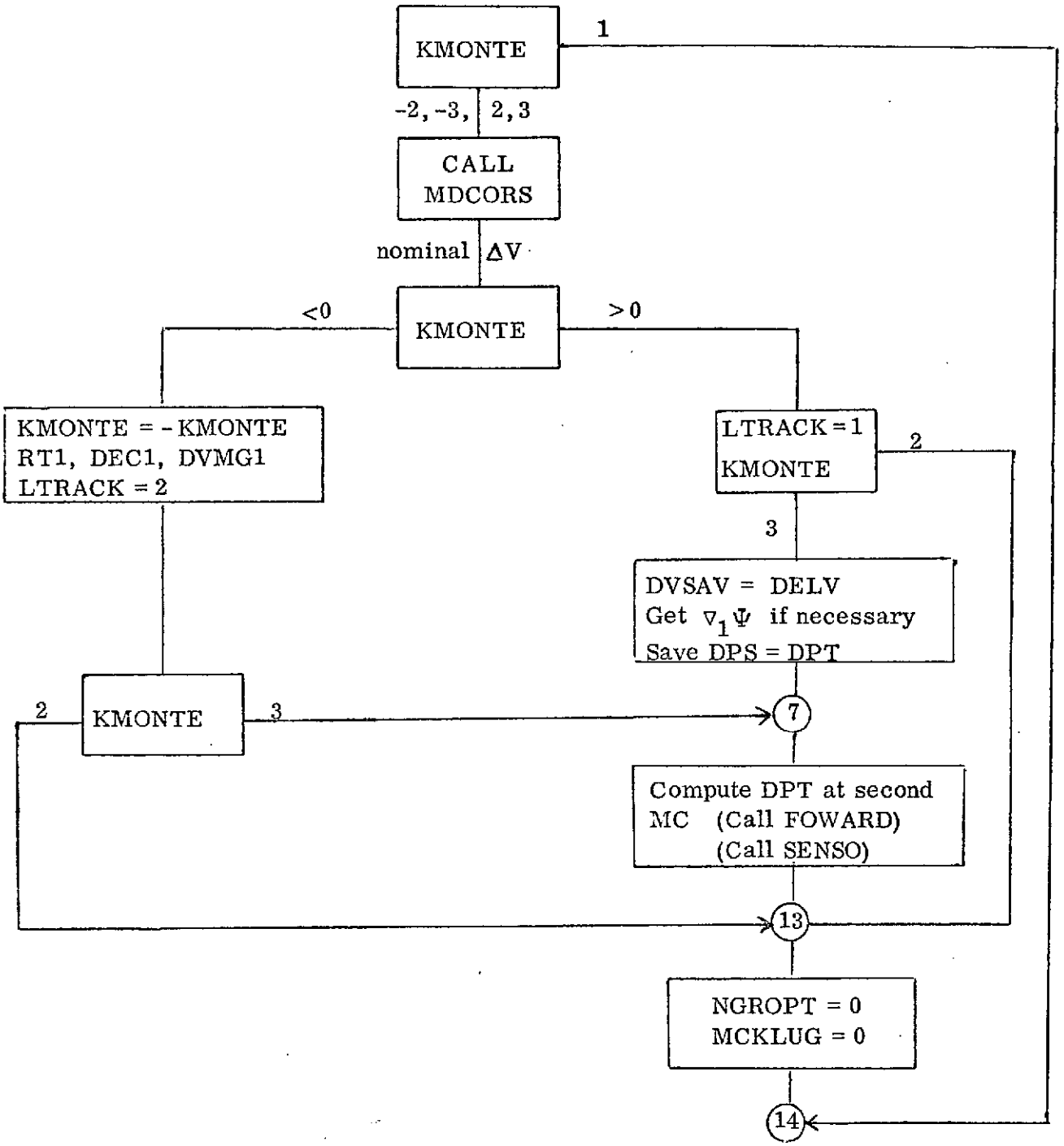
and the magnitude $v + \epsilon_v$, so that it may be programmed as follows,

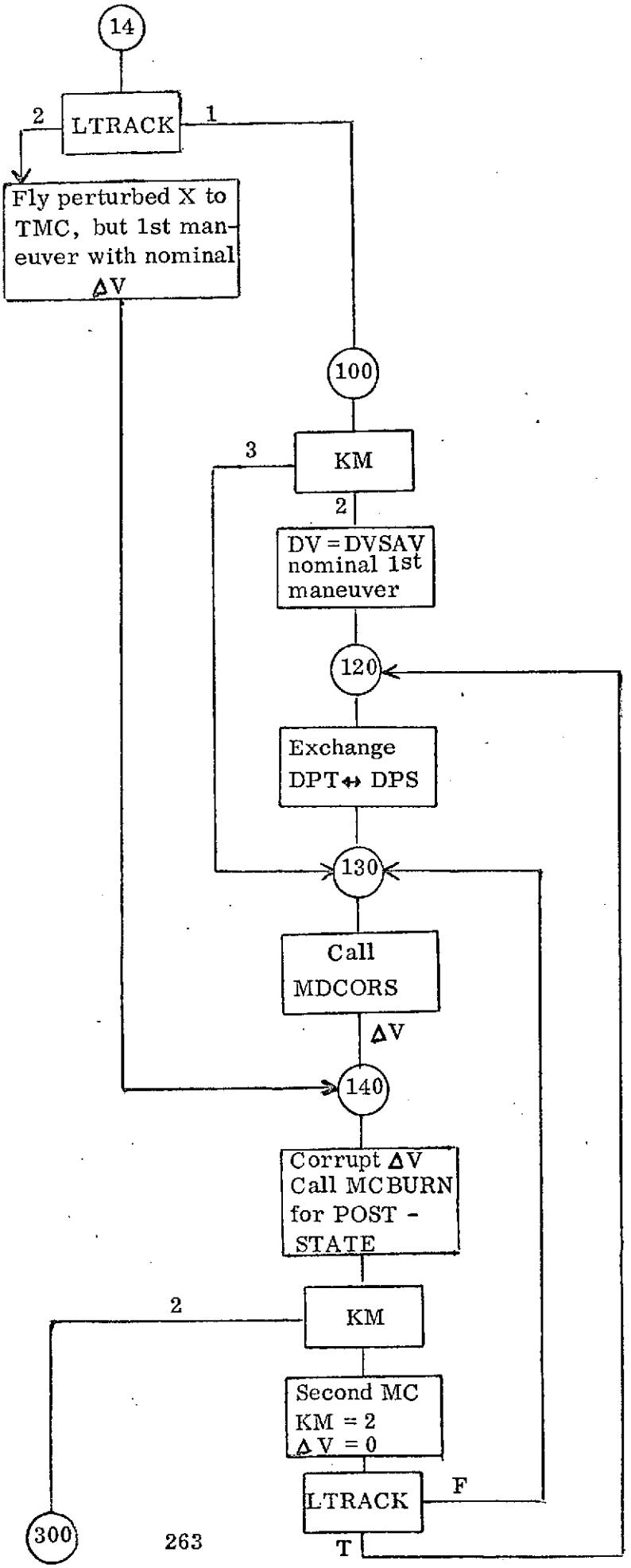
$$V' = \frac{v + \epsilon_v}{v\sqrt{1+\sigma_a^2\,(n_1^2 + n_2^2)}} \left\{ \left(1 - \frac{J_a\ N_2\ V_3}{v \cos(\text{dec})} \right) \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} + \frac{J_a}{\cos(\text{dec})} \begin{pmatrix} -n_1\ v_2 \\ n_1\ v_1 \\ n_2\ v \end{pmatrix} \right\}$$

SUBROUTINE MONTE



MDCORS initialization for MONTE




14
LTRACK
2
1
Fly perturbed X to TMC, but 1st maneuver with nominal ΔV
100
KM
3
2
DV = DVSAV nominal 1st maneuver
120
Exchange DPT↔ DPS
130
Call MDCORS
ΔV
140
Corrupt ΔV Call MCBURN for POST - STATE
KM
2
Second MC KM = 2 ΔV = 0
LTRACK
F
T
300

SUBROUTINE MOTORS

Calling Sequence: CALL MOTORS (KMOD)

Purpose: This subroutine determines thrusting status and increments velocity, or acceleration and weight accordingly.

Common Blocks Required: PERT, CNTRL, INTVAR, STATE, INPUT

Subroutines Called: TABINT

Input / Output

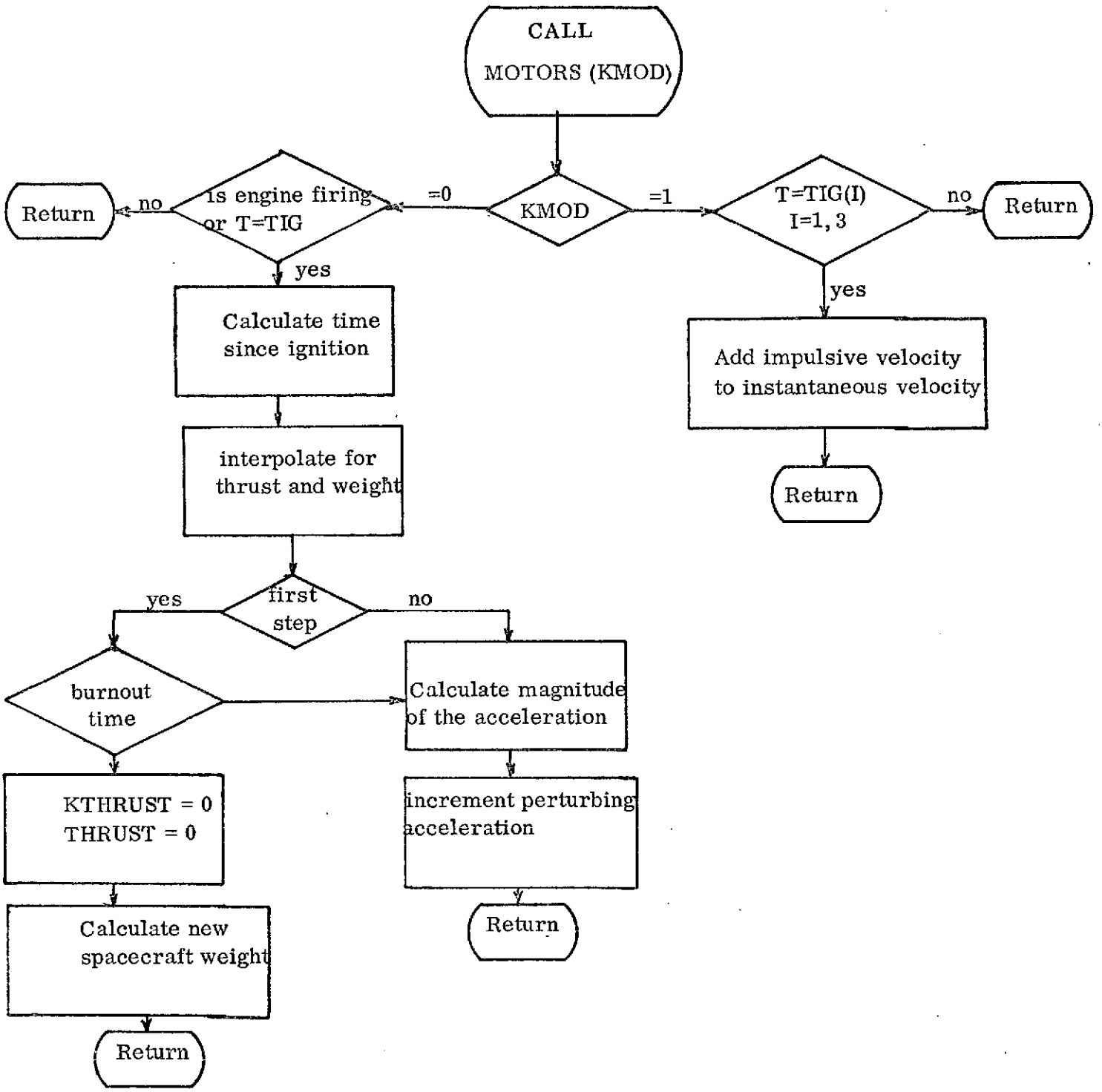
I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	KMOD	1	Calling Operand	Impulsive velocity flag
I/Ø	RCART	3	PERT(1)	Perturbing acceleration vector
I	KDIS	1	CNTRL(05)	Discontinuity Flag
I/Ø	KTHRST	1	CNTRL(02)	Thrusting Flag
I	KFIRST	1	CNTRL(12)	First step Flag
I	T	1	INTVAR(1)	Current time
I/Ø	THRUST	1	STATE(33)	Magnitude of thrust
I/Ø	WT	1	STATE(34)	Weight at engine ignition
I	ATT	3	STATE(11)	Attitude of spacecraft
Ø	W	1	STATE(35)	Current weight
I/Ø	DX	3	STATE(04)	Current velocity
I	DV	3	INPUT(430)	Impulsive velocity
I	TIG	3	INPUT(380)	Ignition times
I	TBO	3	INPUT(383)	Burnout times

Description: The engine characteristics can be simulated by either impulsive velocity or input thrust and weight flow histories. If KMOD is equal to 1, impulsive velocity is assumed. The impulsive velocity of engine I is applied along the current attitude when the current time, T, is equal to the ignition time, TIG.

When KMOD is not equal to one, the instantaneous thrust and weight are determined from linear interpolation between time points on the input tables. Next the magnitude of the acceleration is determined from:

$$a = .001 \text{ THRUST/WEIGHT.}$$

The acceleration is added to the total perturbing acceleration vector of the spacecraft (PERT) according to the current attitude (ATT).


CALL
MOTORS (KMOD)
KMOD
=0
=1
Is engine firing
or T=TIG
no
Return
yes
Calculate time
since ignition
interpolate for
thrust and weight
first
step
yes
no
burnout
time
KTHRUST = 0
THRUST = 0
Calculate new
spacecraft weight
Return
Calculate magnitude
of the acceleration
increment perturbing
acceleration
Return
T=TIG(I)
I=1, 3
no
Return
yes
Add impulsive velocity
to instantaneous velocity
Return

SUBROUTINE MULCON

Calling Sequence:	CALL MULCON
Purpose:	To propagate the state in time using the Multiconic technique.
Common Blocks Required:	CNTRL, CONST, DUM, INPUT, INTVAR, PLNET, STATE
Subroutines Required:	AVERGE, DVMAG, OBLATE, ORBIT, PLANET, OUTPUT, PRINT, TOBODY
Reference:	D. Byrnes and H. Hooper, Multi-conic: A Fast and Accurate Method of Computing Space Flight Trajectories, AIAA Paper No. 70-1062, 1970.

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DELT	10	INPUT(180)	Compute Intervals
I	DJO	1	INPUT(46)	Epoch of the initial state
I/O	DX	3	STATE(4)	Spacecraft velocity
O	ELM	6	STATE(14)	Orbital elements
I/O	JC	1	CNTRL(7)	Central Planet
I	JL	1	INPUT(1015)	Launch planet
I	KCRASH	1	INPUT(1032)	Closest approach flag
I	KOBLE	1	INPUT(1018)	Earth oblateness flag
I	KOUT	1	INPUT(1030)	Output frequency flag
I	KP	12	INPUT(1001)	Planets in the system
I/O	T	1	STATE(10)	Current time since DJO
I	TCOMP	10	INPUT(170)	Switching times of compute interval table
I	TF	1	INPUT(4)	Final time
O	UJT	1	STATE(32)	Current julian date
I/O	X	3	STATE(1)	Spacecraft position

Theory:

The Multiconic technique is an approximate technique to determine space trajectories. This technique differs from most approximate techniques and more resembles a numerical integration method as the independent variable, time, is stepped along and the state determined at the end of each step.

The geocentric equation governing the motion of a spacecraft perturbed by the Sun and Moon is as follows:

$$\ddot{\bar{R}} = \mu_e \frac{\bar{R}}{|\bar{R}|^3} - \mu_m \frac{\bar{r}_m}{|\bar{r}_m|^3} - \mu_m \frac{\bar{R}_m}{|\bar{R}_m|^3} - \mu_s \frac{\bar{r}_s}{|\bar{r}_s|^3} - \mu_s \frac{\bar{R}_s}{|\bar{R}_s|^3} \qquad (1)$$

where $\bar{R}$ is the geocentric position vector of the spacecraft,

$\bar{R}_m$, $\bar{R}_s$ are the geocentric position vectors of the Moon and Sun, respectively,

r_m, r_s are the vectors from the Moon and the Sun to the spacecraft, and

μ_e, μ_m, μ_s are gravitational constants of the Earth, Moon and Sun, respectively.

Each term of equations (1) describes an inverse square force and, taken separately, would yield simple conic motion. The Multiconic technique assumes that the trajectory can be simulated by sequentially summing these conics over several time intervals. However, the terms due to the Sun and the second term due to the Moon are slowly-varying functions. Therefore, without any significant loss in accuracy and with a substantial decrease in computation time, these terms are simplified. It is assumed that the acceleration due to these three terms can be approximated by a constant acceleration equal to the average accelerations at the beginning and at the end of the step applied over the entire step.

There is also the problem of separating the Keplerian orbit with respect to the Moon (second term of equation 1) from the other accelerations. Byrnes and Hooper solved this problem by flying backwards in a gravity free environment, an increment of time equal to the step, dt, and then flying forward along the selenocentric conic, a time increment dt.

The algorithm used to propagate the state from T_1 to T_2 is as follows:

1. The positions of the Moon and Sun are stored at time T_1.
2. The geocentric state is propagated along the Earth conic from T_1 to T_2 using the Keplerian orbit defined by the first term of equation (1).
3. The positions of the Moon and Sun are stored at time T_2.
4. The mean acceleration due to the Moon's indirect term (third term of equation 1) is calculated as:

$$\bar{a}_1 = \frac{\mu_m}{2} \left[\left(\frac{\bar{R}_m}{|R_m|^3} \right)_{T_2} + \left(\frac{\bar{R}_m}{|R_m|^3} \right)_{T_1} \right]$$

5. The mean acceleration due to the Sun is calculated as:

$$\bar{a}_2 = \frac{\mu_S}{2}\left[\left(\frac{\bar{r}_S}{|\bar{r}_S|^3} + \frac{\bar{R}_S}{|\bar{R}_S|^3}\right)_{T_2} + \left(\frac{\bar{r}_S}{|\bar{r}_S|^3} + \frac{\bar{R}_S}{|\bar{R}_S|^3}\right)_{T_1}\right]$$

6. The state at the completion of the Earth conic (Step 2) is adjusted to account for the accelerations determined in Steps 4 and 5. The corrections are obtained from,

$$\Delta\bar{V} = (\bar{a}_1 + \bar{a}_2)\ dt$$

$$\Delta\bar{r} = \frac{1}{2}(\bar{a}_1 + \bar{a}_2)\ dt^2$$

where $\Delta\bar{V}$ and $\Delta\bar{r}$ are the changes in the position and velocity vectors and, dt is the step $(T_2 - T_1)$

7. The corrected geocentric state is converted to selenocentric coordinates and propagated back in time along a straight line defined by the selenocentric velocity an amount dt.

8. The state is then propagated forward along the Keplerian selenocentric conic described by the second term in equation (1) from T_1 to T_2. This completes the algorithm and describes the state at the end of the step.

During the trans-Earth leg of a circumlunar trajectory, the algorithm is slightly modified. The new order is

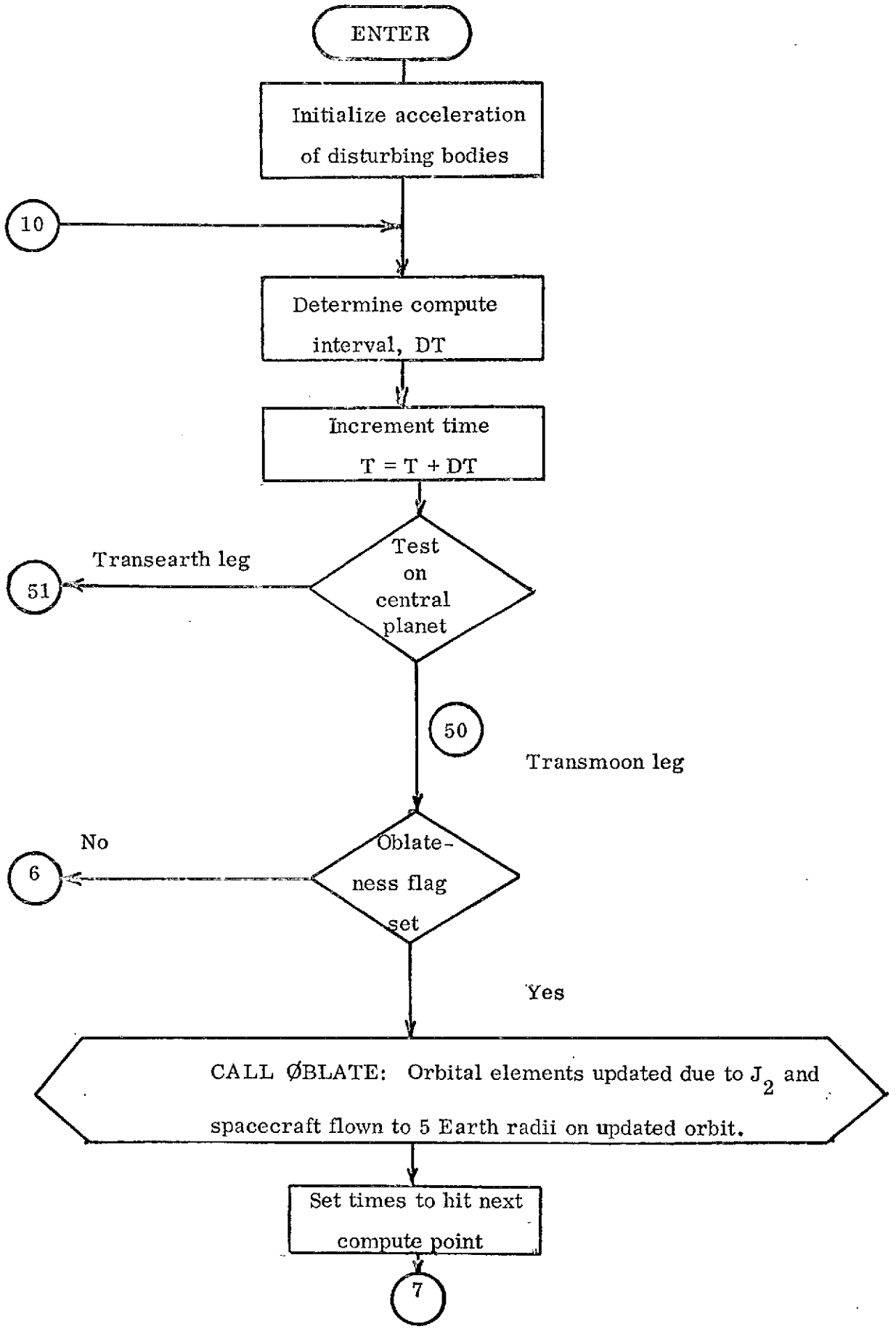
1. Forward along the selenocentric conic.
2. Backward along a straight line.
3. Apply averaged accelerations.
4. Forward along an Earth-centered conic.

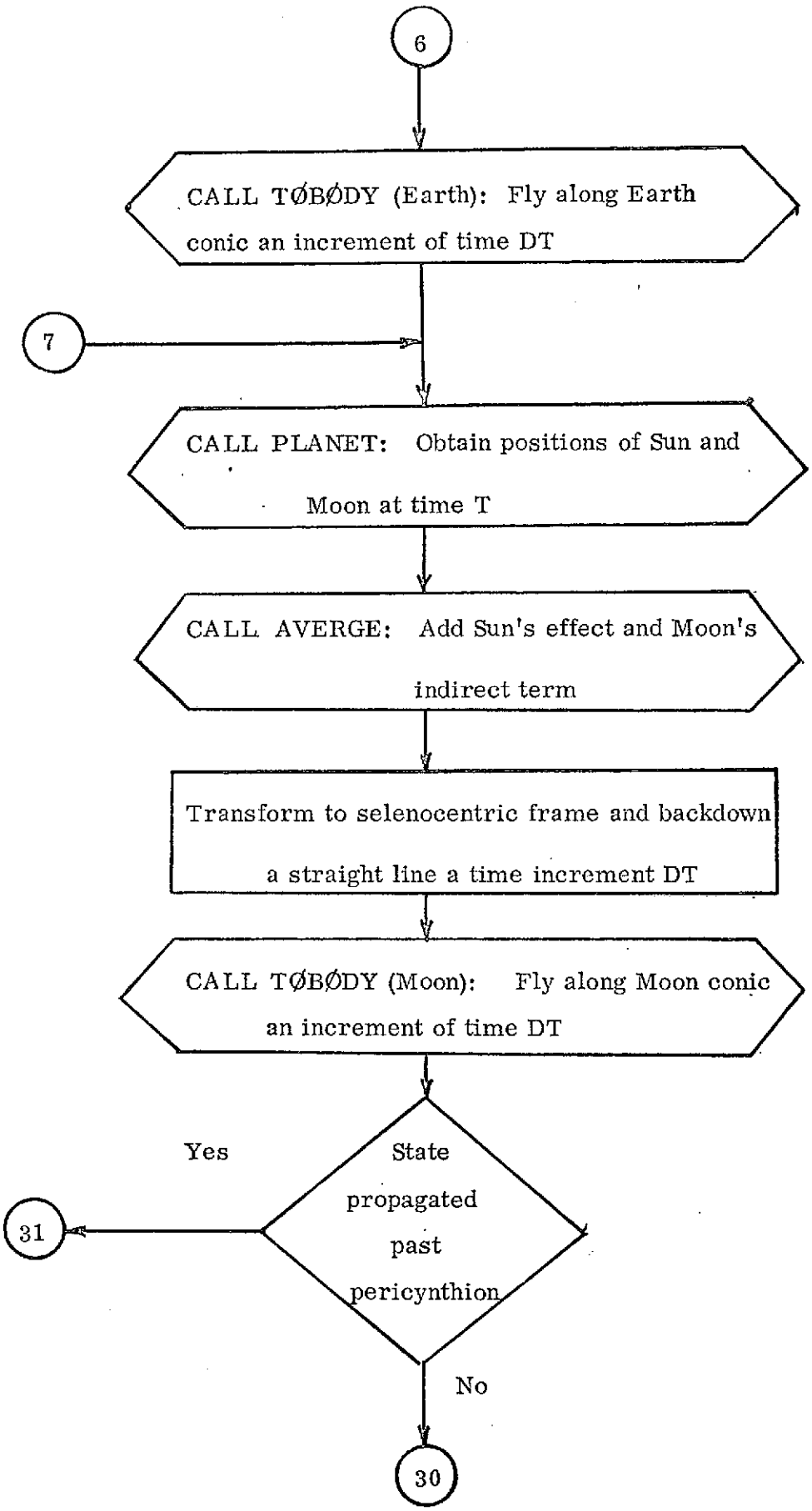
Description:

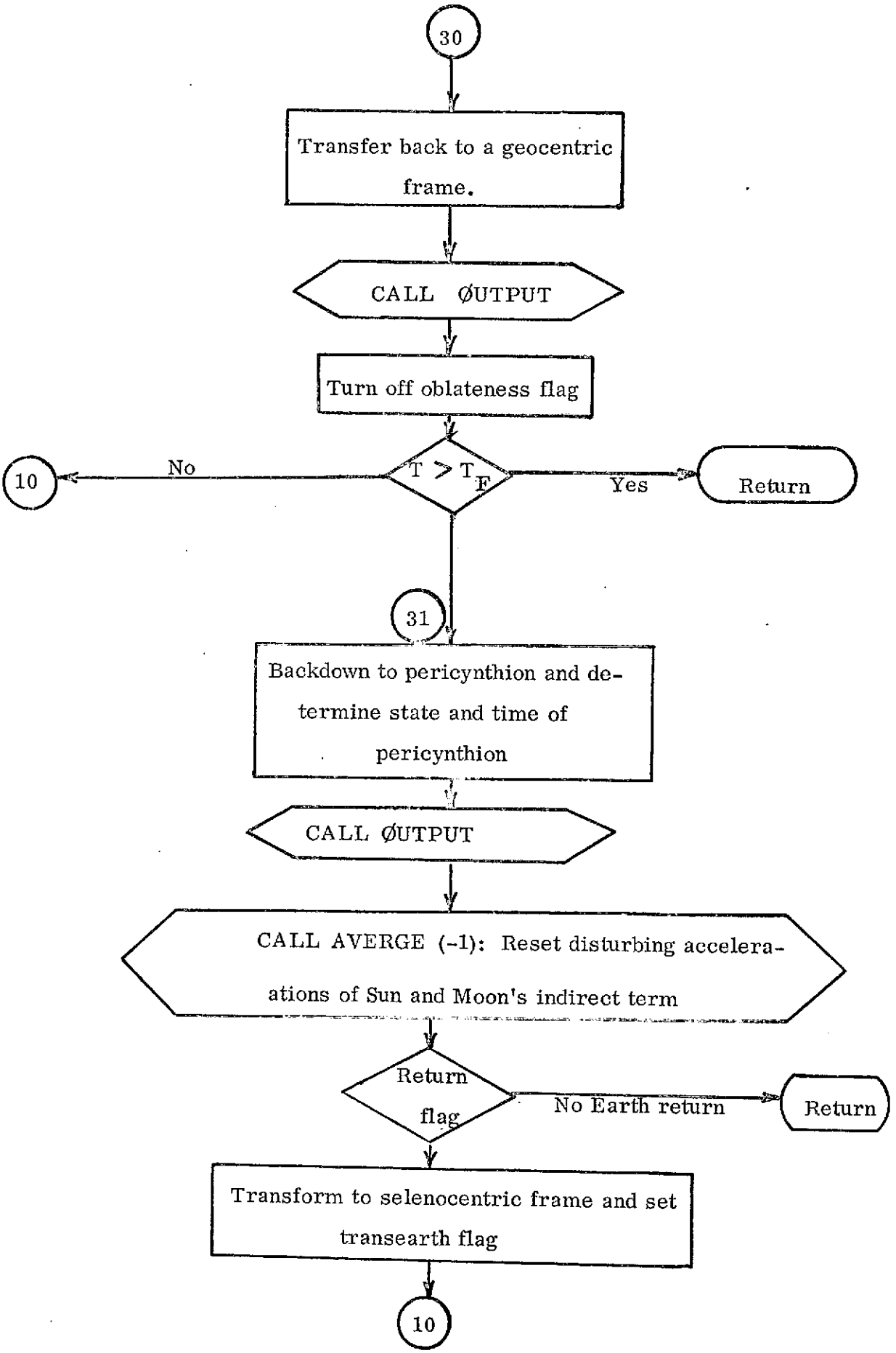
The Mulcon subroutine is divided into three parts. The first part initializes the necessary flags and constants and determines the compute interval. The second part controls the logic flow of the Multiconic algorithm as the translunar leg of a trajectory. The third leg controls the logic flow of the Multiconic algorithm on the transearth leg of the trajectory.

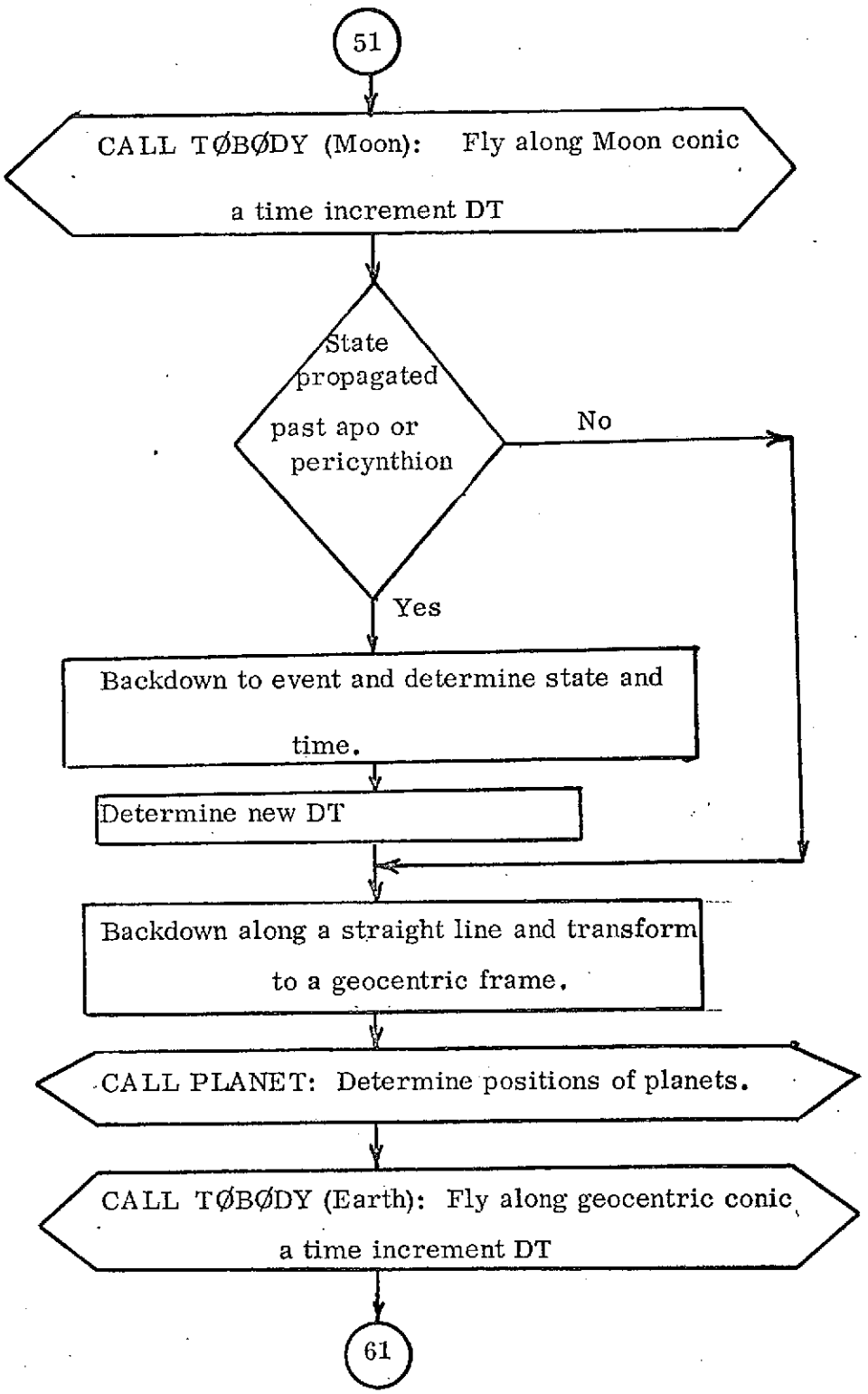
The logic flow of this subroutine can more easily be understood by examining the accompanying flow chart. The transmoon leg starts at statement 50. If Earth oblateness is to be used, the orbital elements are adjusted due to J_2 and the spacecraft flown to approximately 5 Earth radii. After that, the Multiconic algorithm is applied as described earlier. The test to determine if the spacecraft passed through pericynthion or if time is greater than the final time is made before statement 31. If these tests are not satisfied, the logic flow transfers to statement 10 where the next Multiconic interval begins. If pericynthion is passed, the time is adjusted to pericynthion and AVERGE called to reset the averaged accelerations. If the KCRASH flag is set to 1, the state is translated back to a selenocentric system and flags set to begin the transearth leg. The transearth leg is similar to the translunar leg. Logic for this leg begins at statement 51.

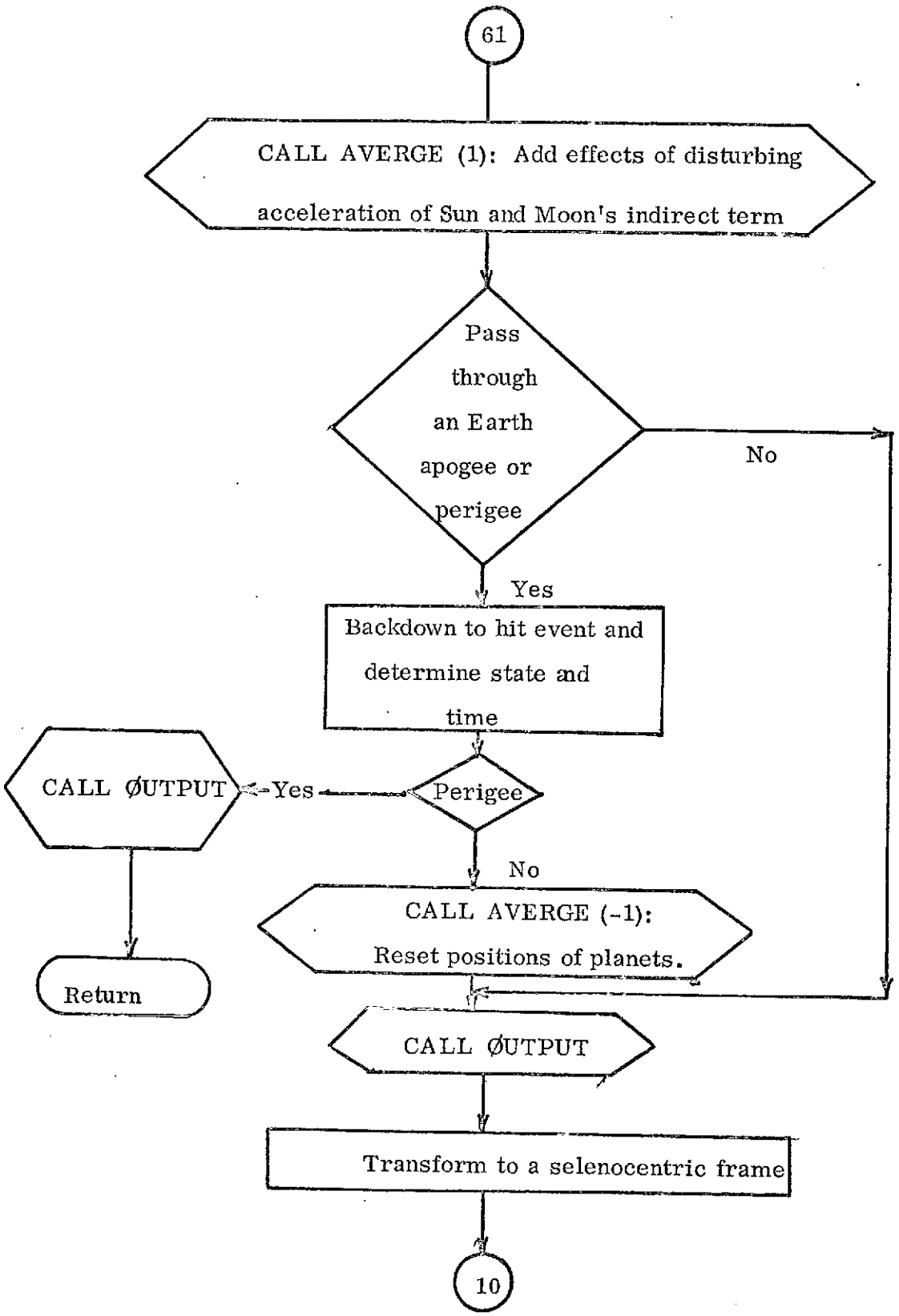
MULCON FLOWCHART




6
CALL TØBØDY (Earth): Fly along Earth conic an increment of time DT
7
CALL PLANET: Obtain positions of Sun and Moon at time T
CALL AVERGE: Add Sun's effect and Moon's indirect term
Transform to selenocentric frame and backdown a straight line a time increment DT
CALL TØBØDY (Moon): Fly along Moon conic an increment of time DT
State propagated past pericynthion
Yes
31
No
30


30
Transfer back to a geocentric frame.
CALL ØUTPUT
Turn off oblateness flag
T > T_F
No
10
Yes
Return
31
Backdown to pericynthion and determine state and time of pericynthion
CALL ØUTPUT
CALL AVERGE (-1): Reset disturbing accelerations of Sun and Moon's indirect term
Return flag
No Earth return
Return
Transform to selenocentric frame and set transearth flag
10


51
CALL TØBØDY (Moon): Fly along Moon conic a time increment DT
State propagated past apo or pericynthion
No
Yes
Backdown to event and determine state and time.
Determine new DT
Backdown along a straight line and transform to a geocentric frame.
CALL PLANET: Determine positions of planets.
CALL TØBØDY (Earth): Fly along geocentric conic a time increment DT
61


61
CALL AVERGE (1): Add effects of disturbing acceleration of Sun and Moon's indirect term
Pass through an Earth apogee or perigee
No
Yes
Backdown to hit event and determine state and time
CALL ØUTPUT
Yes
Perigee
No
CALL AVERGE (-1): Reset positions of planets.
Return
CALL ØUTPUT
Transform to a selenocentric frame
10

SUBROUTINE MVTRN

Calling Sequence: CALL MVTRN(A, B, C, M, N)
CALL RØTATE(M, A, B, C)

Purpose: To form the matrix product $C = AB$ or $C = A^T B$ where A is a 3x3 matrix and B and C are 3xN matrices (3x1 in RØTATE).

Common Blocks Required: None

Subroutines Called: None

Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	A	9	Call List	Matrix multiplier
I	B	3,N	Call List	Matrix multiplier
Ø	C	3,N	Call List	Product matrix
I	M	1	Call List	Indicator: $C = AB$ if $M = 1$ $C = A^T B$ otherwise
I	N	1	Call List	Number of columns of B and C

SUBROUTINE M50EPM

Calling Sequence: CALL M50EPM (UJT, C)

Purpose: M50EPM calculates the transformation matrix from the mean equinox and equator of 1950 to the Earth's true equator and prime meridian.

Common Blocks Required: CONST

Subroutines Required: NUTAIT, M50MDT

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	C	3,3	Calling Operand	Transformation matrix
I	RAD	1	CONST(1)	Degrees per radian
I	UJT	1	Calling Operand	Modified julian date

Theory:

The transformation from mean equator and equinox of 1950 to true equator and prime meridian can be accomplished by first transforming to the mean equator and equinox of date (Call the matrix [M]) and then from there to the true equator and equinox of date (Call the matrix [T]) and finally from equinox to the prime meridian ([P]).

Thus, if [C] is to be the transformation from mean of 1950 to true equator and prime meridian, then

$$[C] = [P]\,[T]\,[M]$$

Both [T] and [M] are calculated in other subroutines (NUTAIT and M50MDT respectively). The [P] matrix represents the rotation through the Greenwich hour angle about the Z-axis. In general, a rotation through an angle θ about the positive Z-axis is accomplished through the following matrix

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In this case, θ is the Greenwich hour angle, (GHA), which is given by .

$$GHA = 100^{\circ}.075542 + 0^{\circ}.98564735(d) + (2.9015)10^{-13}(d)^2$$

$$+ \omega(t) + (\text{Nutation in right ascension})$$

ω is the angular velocity (deg/sec) of the Earth and is given by

$$\omega = .0041780742 \,/\, \left(1. + (5.21)10^{-13}(d)\right)$$

d is the number of whole days since 1950.

t is the fraction of days (i.e. modified julian date - 33282.5 = d + t)

The nutation in right ascension is element (2, 1) of the matrix [T] from NUTAIT.

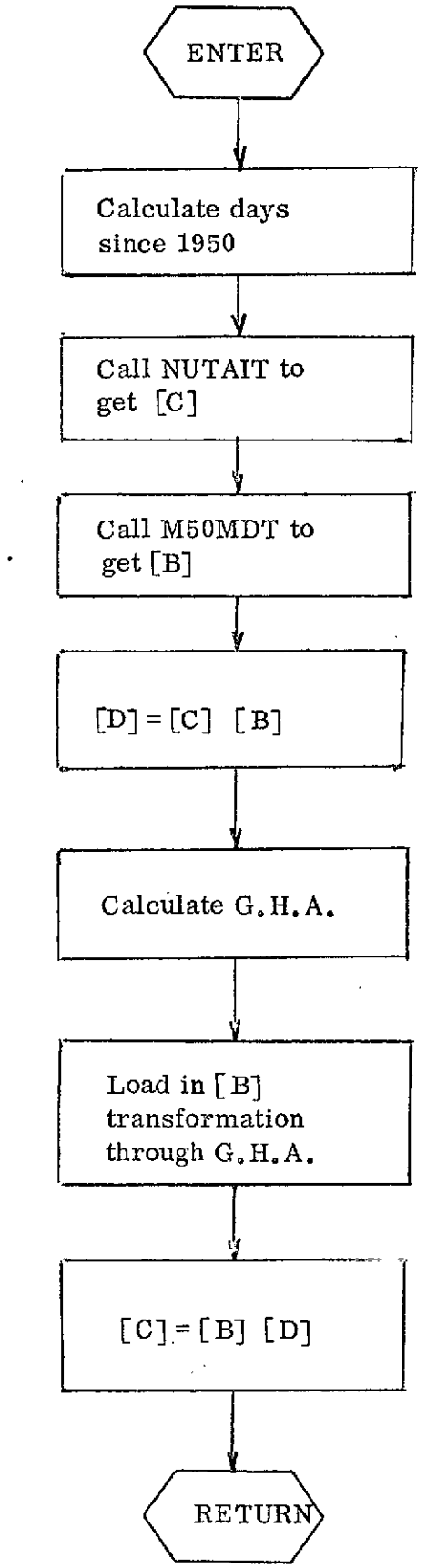
The three matrices are multiplied to give the total transformation matrix [C].

Description:

M50EPM calculates the transformation matrix from the mean equator and equinox of 1950 to the Earth's true equator and prime meridian. This transformation is composed of three separate transformations.

The first transformation is from mean of 1950 to mean of date. This matrix is calculated in M50MDT. The second transformation is from mean of date to true equator and equinox of date. This matrix is calculated in NUTAIT. The third transformation is from true equator and equinox of date to true equator and prime meridian. This matrix is set up in this subroutine and consists of a rotation about the z-axis (North Pole) through the Greenwich hour angle. Once these transformation matrices are found, they are multiplied together to form the total transformation.

SUBROUTINE M50EPM



SUBROUTINE M50JPM

Calling Sequence: CALL M50JPM (UJT, B, J)

Purpose: M50JPM calculates the transformation matrix from Earth mean equator and equinox of 1950 to true equator and prime meridian of date with respect to planet J.

Common Blocks Required: CONST

Subroutines Required: M50EPM, M50LEQ, M50MDT

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	B	3,3	Calling Operand	Transformation matrix
I	J	1	Calling Operand	Planet number
I	RAD	1	CONST(1)	Degrees per radian
I	TDS	1	CONST(43)	Seconds per day
I	UJT	1	Calling Operand	Modified Julian Date
I	WP	12	CONST(29)	Spin rate of planets (rad/sec)

Theory:

The transformation from Earth mean of 1950 to true equator and prime meridian of date with respect to the desired planet is composed of two separate transformations. The first of these, Earth mean of 1950 to Earth mean of date, is calculated and explained in M50MDT. The second, Earth mean of date to true equator and prime meridian of date, with respect to the desired planet, is explained here.

The tranformation, T, involves three rotations about three axes.

$$\begin{pmatrix} T \end{pmatrix} = \begin{pmatrix} CP & SP & 0 \\ -SP & CP & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & CD & SD \\ 0 & -SD & CD \end{pmatrix} \begin{pmatrix} CR & SR & 0 \\ -SR & CR & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

where

CR and SR represent the cosine and sine of RA + 90^{o} respectively

CD and SD represent the cosine and sine of 90^{o} - DEC respectively

CP and SP represent the cosine and sine of PHE respectively

and

RA and DEC are the right ascension and declination of the planet's spin axis with respect to Earth mean equator and equinox of date.

PHE is the angle to the planet's prime meridian from the ascending node of the planet's equator on the Earth's equator.

The values of RA, DEC and PHE are calculated through equations of the type

$$\alpha = \alpha_0 + \dot{\alpha}\ (t - t_0)$$

where α_0 is the value of α at t_0 and $\dot{\alpha}$ is the rate of change of α with respect to time.

The following constants are used for α_0, $\dot{\alpha}$ and t_0,

MARS

	α_0	t_0 (modified Julian date)	$\dot{\alpha}$
RA	$317.^{o}3$	16846.024	$6.^{o}751 \times 10^{-3}$/year
DEC.	$52.^{o}7$	16846.024	$3.^{o}46 \times 10^{-3}$/year
PHE	$344.^{o}41$	18322.	$.70882 \times 10^{-4}$rad/sec

JUPITER

	α_0	t_0(modified Julian date)	$\dot{\alpha}$
RA	4.6775435 rad	18673.	$.102917 \times 10^{-3}$ rad/year
DEC	1.126778 rad	18673.	$.29089 \times 10^{-5}$ rad/year
PHE*	$344.^{o}41$	18322.	$.175849 \times 10^{-3}$rad/sec

*Note: The surface features of Jupiter are relatively unknown, so that the location of its prime meridian is a matter of speculation. For this reason, its initial position is set equal to that of Mars.

Description:

M50JPM calculates the transformation matrix B from Earth mean of 1950 to true equator and prime meridian of date with respect to planet J. The date is given in terms of the modified Julian date UJT. At present, only the following planets may be used: Earth, Moon, Mars, and Jupiter (i.e., J = 3, 11, 4 and 5 respectively).

Subroutines M50EPM and M50LEQ calculate the matrix B for the planet Earth and the Moon respectively, thus the calculation of B for Mars and Jupiter represents the bulk of M50JPM. These calculations are done in three main steps. First, the right ascension and declination of the planet's spin axis with respect to the Earth mean equator and equinox of date are found in addition to the angle of prime meridian. These angles represent rotations which make up the transformation from Earth mean of date to true equator and prime meridian of date with respect to the planet. Second, the transformation from Earth mean of 1950 to Earth mean of date is calculated in M50MDT. Third, these matrices are multiplied to give the total desired transformation.

SUBROUTINE M50LEQ

Calling Sequence: CALL M50LEQ (UJT, SELNEQ)

Purpose: M50LEQ calculates the transformation matrix from the Earth mean equator and equinox of 1950 to the true lunar equator and prime meridian of date.

Common Blocks Required: None

Subroutines Required: NUTATE

Input / Output

I/O	SYMBOLIC NAME	DIMENSTION	COMMON BLOCK	DEFINITION
O	SELNEQ	3, 3	Calling Operand	Transformation matrix
I	UJT	1	Calling Operand	Modified Julian Date

Theory:

The transformation from Earth mean of 1950.0 to true lunar equator and prime meridian can be broken up into three transformations: (1) Earth mean equator and equinox of date to true lunar equator and ascending node, (2) Earth mean equator and equinox of 1950 to Earth mean equator and equinox of date, (3) True lunar equator and ascending node to true lunar equator and prime meridian.

The matrix for the first transformation is calculated in subroutine NUTATE, and the theory is described there.

The second matrix describes the precession of the Earth's mean equator since 1950.0. This precession is defined by the three small Euler angles

$$\alpha = 2304.''997\,(T) + 0.''302(T)^2 + 0.''019(T)^3$$

$$\beta = 2304.''997(T) + 1.''093(T)^2 + 0.''0192(T)^3$$

$$\gamma = 2004.''298(T) - 0.''426(T)^2 + 0.''0416(T)^3$$

The transformation A is defined by the three rotations

$$A = \begin{bmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The elements of $[A]$ may be expanded in powers of T to yield:

$$a_{11} = 1-.00029697\,(T)^2 - (0.13)\,10^{-6}\,(T)^3$$

$$a_{12} = -a_{21} = -.02234988(T) - (0.676)\,10^{-5}\,(T)^2 + (0.221)10^{-5}\,(T)^3$$

$$a_{13} = -a_{31} = -.00971711(T) + (0.207)10^{-5}\,(T)^2 + (0.96)10^{-6}(T)^3$$

$$a_{22} = 1-.00024976(T)^2 - (0.15)10^{-6}\,(T)^3$$

$$a_{23} = a_{32} = -.00010859\,(T)^2 - (0.3)\,10^{-7}(T)^3$$

$$a_{33} = 1-.00004721(T)^2 + (0.2)10^{-7}\,(T)^3$$

The third transformation involves the simple rotation about the Z-axis (perpendicular to the true equator of date and positive up) from the ascending node to the prime meridian. The angle between the ascending node and prime meridian is calculated in NUTATE, and its derivation is described there. If the angle is δ, the rotation matrix is

$$B = \begin{bmatrix} \cos\delta & \sin\delta & 0 \\ -\sin\delta & \cos\delta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Once these three matrices have been calculated, the total transformation matrix is

$$[T] = [B]\,[N]\,[A]$$

where $[N]$ is the matrix from NUTATE (for the transformation from Earth mean of date to true lunar equator and ascending node).

Description:

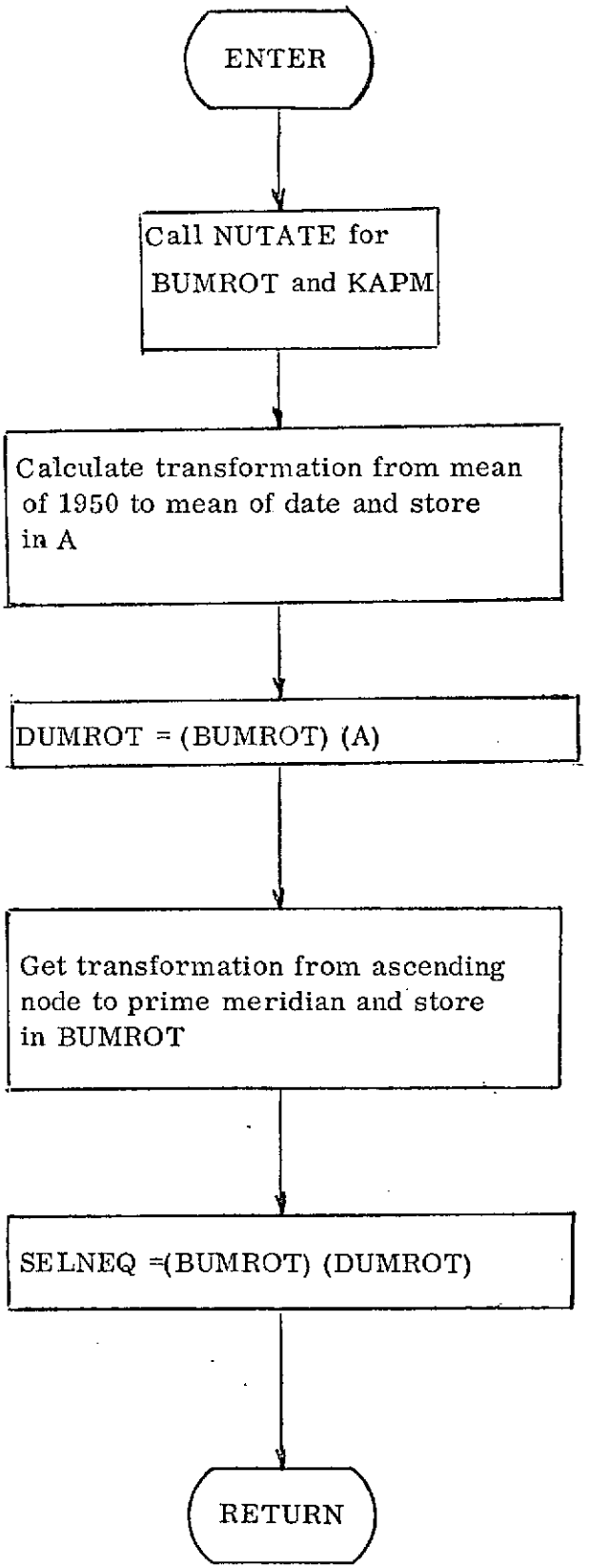
M50LEQ first calls NUTATE to get BUMROT, the transformation matrix from Earth mean equator and equinox of date to Moon true equator with the ascending node on the ecliptic. Also output from NUTATE is RAPM which is the angle from the lunar equator ascending node on the ecliptic to the prime meridian.

The next step in the transformation is obtaining the transformation matrix A from Earth mean of 1950.0 to Earth mean of date. Then A is multiplied by BUMROT to get DUMROT, which is the transformation matrix from Earth mean of 1950.0 to Moon true equator and ascending node.

The final transformation needed is from ascending node to prime meridian in the Moon's true equator of date. This involves a simple rotation about the Z-axis (perpendicular to the true equator and positive up) through the angle RAPM. This matrix is stored in BUMROT.

The final step is the multiplication of DUMROT by BUMROT which is stored in SELNEQ and is output as the total transformation matrix.

SUBROUTINE M50LEQ



SUBROUTINE M50MDT

Calling Sequence: CALL M50MDT (T, A)

Purpose: M50MDT calculates the transformation matrix from Earth mean equator and equinox of 1950.0 to Earth mean equator and equinox of date.

Common Blocks Required: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	A	3,3	CALLING OPERAND	Transformation Matrix
I	T	1	CALLING OPERAND	Modified Julian Date

Description:

M50MDT calculates the time in Julian centuries and then calculates the elements of the matrix as polynomials in time.

Theory:

The transformation from Earth mean equator and equinox of 1950.0 to Earth mean equator and equinox of date is accomplished by three rotations through three Euler angles that represent the precession of the Earth since 1950.0.

The three angles are as follows:

$$\alpha = 2304.''997(T1) + 0.''302\,(T1)^2 + 0.''019(T1)^3$$

$$\beta = 2304.''997(T1) + 1.''093(T1)^2 + 0.''0192(T1)^3$$

$$\gamma = 2004.''298(T1) - 0.''426(T1)^2 + 0.''0416(T1)^3$$

where T1 is the time in Julian centuries since the beginning of the Besselian year 1950.

The transformation matrix [A] is defined by

$$A = \begin{bmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The elements of A may be expanded into the following polynomials of T1:

$$a_{11} = 1-.00029697(T1)^2 - (0.13)\ 10^{-6}\ (T1)^3$$

$$a_{12} = -a_{21} = -.02234988(T1) - (0.676)10^{-5}\ (T1)^2 + (0.221)\ 10^{-5}\ (T1)^3$$

$$a_{13} = -a_{31} = -.00971711(T1) + (.207)10^{-5}\ (T1)^2 + (.96)\ 10^{-6}\ (T1)^3$$

$$a_{22} = 1- .00024976(T1)^2 - (0.15)\ 10^{-6}\ (T1)^3$$

$$a_{23} = a_{32} = -.00010859(T1)^2 - (0.3)\ 10^{-7}\ (T1)^3$$

$$a_{33} = 1-.00004721(T1)^2 + (0.2)\ 10^{-7}\ (T1)^3$$

SUBROUTINE NUTAIT

Calling Sequence: CALL NUTAIT (TIME, EN)

Purpose: NUTAIT calculates the matrix for the transformation from Earth mean equator and equinox of date to Earth true equator and equinox of date

Common Blocks Required: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	EN	3,3	Calling Operand	Transformation Matrix
I	TIME	1	Calling Operand	Days since 1950.0

Description:

NUTAIT calculates the mean longitude of the Sun and Moon (VL & CR) and the mean longitude of perigee of the Sun and Moon (G & GP) as functions of time. Then the nutations of the obliquity of the ecliptic and longitude of the Sun (DE & DT) are calculated as functions of VL, CR, G and GP. The mean obliquity of the ecliptic is calculated as a function of time, and then the elements of the matrix are calculated and stored in EN for output.

Theory:

The transformation from mean equator to true equator can be approximated by

$$EN = \begin{bmatrix} 1 & -\delta L \cos(\bar{\epsilon}) & -\delta L \sin(\bar{\epsilon}) \\ \delta L \cos\bar{\epsilon} & 1 & -\delta\epsilon \\ \delta L \sin\bar{\epsilon} & \delta\epsilon & 1 \end{bmatrix}$$

where the nutations δL and $\delta\epsilon$ are given by

$$\begin{aligned}\delta\epsilon \times 10^4 = \;& 25^\circ.5844 \cos \bar{\Omega} + 0^\circ.2456 \cos 2\bar{L}' \\ & -0^\circ.2511 \cos 2\bar{\Omega} + 0^\circ.0508 \cos (2\bar{L}' - \bar{\Omega}) \\ & +1^\circ.5336 \cos 2\bar{L} + 0^\circ.0369 \cos (3\bar{L}' - \bar{\Gamma}') \\ & +0^\circ.0666 \cos (3\bar{L} - \bar{\Gamma}) - 0^\circ.0139 \cos (\bar{L}' + \bar{\Gamma}') \\ & -0^\circ.0258 \cos (\bar{L} + \bar{\Gamma}) - 0^\circ.0086 \cos (\bar{L}' - \bar{\Gamma}' + \bar{\Omega}) \\ & -0^\circ.0183 \cos (2\bar{L} - \bar{\Omega}) + 0^\circ.0083 \cos (\bar{L}' \bar{\Gamma}' - \bar{\Omega}) \\ & -0^\circ.0067 \cos (2\bar{\Gamma}' - \bar{\Omega}) + 0^\circ.0061 \cos (3\bar{L}' + \bar{\Gamma}' - 2\bar{L}) \\ & + 0^\circ.0064 \cos (3\bar{L}' - \bar{\Gamma}' - \bar{\Omega})\end{aligned}$$

$$\begin{aligned}\delta L \times 10^{-4} = \;& -(47^\circ.8927 + 0.0482\,T) \sin \bar{\Omega} - 0^\circ.5658 \sin (2\bar{L}') \\ & + 0^\circ.5800 \sin (2\bar{\Omega}) \qquad - 0^\circ.0950 \sin (2\bar{L}' - \bar{\Omega}) \\ & - 3^\circ.5361 \sin (2\bar{L}) \qquad - 0^\circ.0725 \sin (3\bar{L}' - \bar{\Gamma}') \\ & - 0^\circ.1378 \sin (3\bar{L} - \bar{\Gamma}) \qquad + 0^\circ.0317 \sin (\bar{L}' + \bar{\Gamma}') \\ & + 0.0594 \sin (\bar{L} + \bar{\Gamma}) \qquad + 0^\circ.0161 \sin (\bar{L}' - \bar{\Gamma}' + \bar{\Omega}) \\ & + 0^\circ.0344 \sin (2\bar{L} - \bar{\Omega}) \qquad + 0^\circ.0158 \sin (\bar{L}' - \bar{\Gamma}' - \bar{\Omega}) \\ & + 0^\circ.0125 \sin (2\bar{L}' - \Omega) \qquad - 0^\circ.0144 \sin (3\bar{L}' + \bar{\Gamma}' - 2\bar{L}) \\ & + 0^\circ.3500 \sin (\bar{L} - \bar{\Gamma}) \qquad - 0^\circ.0122 \sin (3\bar{L}' - \bar{\Gamma}' - \bar{\Omega}) \\ & + 0^\circ.0125 \sin (2\bar{L} - 2\bar{\Gamma}') \qquad + 0^\circ.1875 \sin (\bar{L}' - \bar{\Gamma}') \\ & \qquad + 0.0078 \sin (2L' - 2\bar{\Gamma}') \\ & \qquad + 0^\circ.0414 \sin (\bar{L}' + \bar{\Gamma}' - 2\bar{L}) \\ & \qquad + 0^\circ.0167 \sin (2\bar{L}' - 2\bar{L}) \\ & \qquad - 0^\circ.0089 \sin (4\bar{L}' - 2\bar{L})\end{aligned}$$

and

$$\begin{aligned}\bar{L}' = \;& 64^\circ.37545167 + 13^\circ.1763965268\,d \\ & - .^\circ 001131575T - .^\circ 00113015\,T^2 + .^\circ 019 \times 10^{-4}\,T^3\end{aligned}$$

$$\begin{aligned}\bar{L} = \;& 280^\circ.08121009 + 0^\circ.9856473354\,d \\ & + .^\circ 000303T + .^\circ 000303T^2\end{aligned}$$

$$\begin{aligned}\bar{\Gamma}' = \;& 208^\circ.8439877 + 0^\circ.1114040803\,d \\ & - .^\circ 01334T - .^\circ 010343T^2 - .^\circ 12 \times 10^{-4}\,T^3\end{aligned}$$

$$\begin{aligned}\bar{\Omega} = \;& 12^\circ.1127902 - 0^\circ.0529539222\,d \\ & + .^\circ 0020795T + .^\circ 002081\,T^2 + .^\circ 02 \times 10^{-4}\,T^3\end{aligned}$$

$$\bar{\Gamma} = 282.^{\circ}08053028 + 0.^{\circ}470684 \times 10^{-4}\, d + .^{\circ}00045525T + .^{\circ}0004575\, T^2 + .^{\circ}03 \times 10^{-4}\, T^3$$

$$\bar{\epsilon} = 23.^{\circ}4457587 - 0.^{\circ}01309404T - .^{\circ}0088 \times 10^{-4}\, T^2 + .^{\circ}0050 \times 10^{-4}\, T^3$$

where the above symbols represent the following quantities,

Symbol	Quantity
L	Longitude of Sun, from the equinox of date
Γ	Longitude of perigee of the Sun
L'	Longitude of the Moon, measured in the ecliptic from the mean equinox of date to the ascending node of the lunar orbit, and then along the orbit.
Γ'	Longitude of the perigee of the Moon, measured as L'
ϵ	Obliquity of the ecliptic
Ω	Longitude of the ascending node of the Moon's orbit on the ecliptic
d	Time in days since 1950.0
T	Time in Julian centuries since 1950.0

Note: Barred quantities are mean values and δL represents the nutation in longitude of the Sun and $\delta\epsilon$ represents the nutation in obliquity.

SUBROUTINE NUTATE

Calling Sequence: CALL NUTATE (K, TW, TF, TN, TM, RA, MG)

Purpose: To compute the transformations from Earth's mean equator, equinox to Earth's and Moon's true equator, equinox and equator, node.

Common Blocks Required: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	K	1	ARGUMENT LIST	Output option key.
I	TW, TF	1	ARGUMENT LIST	Whole and fractional days since 1950.0, days
O	TN	3, 3	ARGUMENT LIST	Earth's transformation matrix
O	TM	3, 3	ARGUMENT LIST	Moon's transformation matrix
O	RA	1	ARGUMENT LIST	Right ascension of Moon's prime meridian, rad.
O	OMG	1	ARGUMENT LIST	Moon's rotation rate, rad/sec

Output Options

If K = 1, only TN is computed, and the lunar orientation variables TM, RA, OMG are ignored.

If K = 2, TM, RA, OMG are computed, and the Earth orientation matrix, TN, is ignored.

If K = 3, all list items are computed.

Equinox Coordinate Systems

Reference: Holdridge, D. B., "Space Trajectories Program for the IBM 7090 Computer." JPL Technical Report No. 32-223, Pasadena, California, March 2, 1962.

The cartesian coordinate systems defined using the vernal equinox and ecliptic or equatorial planes, and the transformations relating those systems are described below. Numerical expressions for the angles defining relative orientations are taken from the cited reference, though different computational forms are used. For completeness, the orientations of body-fixed systems are also treated here.

1. Notation

The symbols used for the angles follow the general rules:

a. Quantities not otherwise noted are true values of date.

b. Quantities superscribed by a bar are mean values.

c. Quantities subscripted "50" are 1950.0 values.

d. The prefix "δ" denotes a nutation or libration in the quantity prefixed.

In general, quantities defining Earth-Moon relationships use the symbol primed of the equivalent quantity from the Earth-Sun relationships. Unless otherwise stated, longitude is measured in the Earth's ecliptic, from the vernal equinox, and right ascension is measured in the equator, from the vernal equinox.

For each of the coordinate systems, the z-axis is taken normal to the reference plane, positive in the northern hemisphere, and the x-axis is taken along the reference direction in the reference plane.

The reference direction is the vernal equinox for the Earth-equinox systems, and the descending node of the Moon's equator on the ecliptic for the Moon-equinox systems. For the body-fixed systems, the reference direction is the prime meridian of the body.

A list of symbols is given in Table 1. The symbols used by the reference are also shown for ease of reference.

2. Equations

The computational equations are based on the equations given by the cited reference. Numerical values are taken from that report, but the form of many of the equations has been changed.

2.1 Earth's Mean Equator, Equinox of Date

The precession of the Earth's equator is defined by the small angles

$$\begin{aligned} \zeta_o &= 2304\overset{''}{.}997T + 0\overset{''}{.}302T^2 + 0\overset{''}{.}0179T^3 \\ z &= 2304\overset{''}{.}997T + 1\overset{''}{.}093T^2 + 0\overset{''}{.}0192T^3 \\ \theta &= 2004\overset{''}{.}298T - 0\overset{''}{.}426T^2 - 0\overset{''}{.}0416T^3 \end{aligned} \tag{1}$$

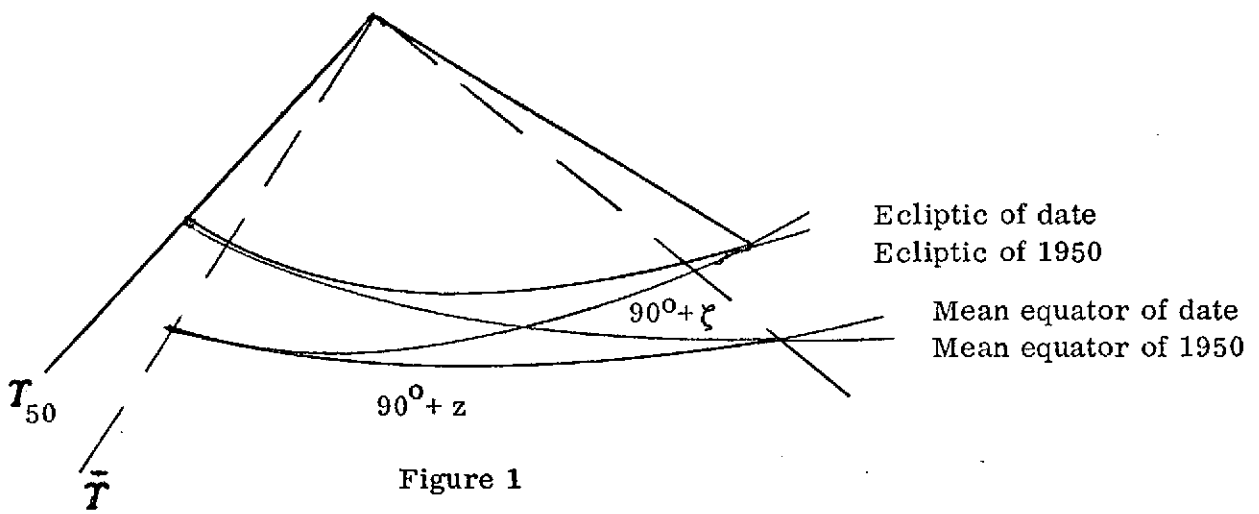


Figure 1

The transformation

$$A = \left\{ a_{ij} \right\}$$

$$= \begin{bmatrix} \cos z & -\sin z & 0 \\ \sin z & \cos z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\zeta_o & -\sin\zeta_o & 0 \\ \sin\zeta_o & \cos\zeta_o & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

may be expanded in powers of T with the result

$$\begin{aligned} a_{11} &= - - 0.00029697T^2 - 0.00000013T^3 \\ a_{12} &= -a_{21} = - 0.02234988T - 0.00000676T^2 + 0.00000221T^3 \\ a_{13} &= -a_{31} = - 0.0097171lT + 0.00000207T^2 + 0.00000096T^3 \end{aligned} \tag{3}$$

$$a_{22} = 1 - 0.00024976T^2 - 0.00000015T^3$$

$$a_{23} = a_{32} = -0.00010859T^2 - 0.00000003T^3$$

$$a_{33} = 1 - 0.00004721T^2 + 0.00000002T^3$$

given by the reference.

2.2 Earth's True Equator, Equinox of Date

The transformation from mean equator to true equator is approximated by

$$N = \begin{bmatrix} 1 & -\delta L \cos \bar{\epsilon} & -\delta L \sin \bar{\epsilon} \\ \delta L \cos \bar{\epsilon} & 1 & -\delta \epsilon \\ \delta L \sin \bar{\epsilon} & \delta \epsilon & 1 \end{bmatrix} \quad (4)$$

where the nutations δL, $\delta \epsilon$ are given by

$$\begin{aligned} \delta\epsilon \times 10^4 = {} & 25^{\circ}.5844 \cos \bar{\Omega} + 0^{\circ}.2456 \cos 2\bar{L}' \\ & -0^{\circ}.2511 \cos 2\bar{\Omega} + 0^{\circ}.0508 \cos (2\bar{L}' - \bar{\Omega}) \\ & +1^{\circ}.5336 \cos 2\bar{L} + 0^{\circ}.0369 \cos (3\bar{L}' - \bar{\Gamma}') \\ & +0^{\circ}.0666 \cos (3\bar{L} - \bar{\Gamma}) - 0^{\circ}.0139 \cos (\bar{L}' + \bar{\Gamma}') \\ & -0^{\circ}.0258 \cos (\bar{L} + \bar{\Gamma}) - 0^{\circ}.0086 \cos (\bar{L}' - \bar{\Gamma}' + \bar{\Omega}) \\ & -0^{\circ}.0183 \cos (2\bar{L} - \bar{\Omega}) + 0^{\circ}.0083 \cos (\bar{L}' - \bar{\Gamma}' - \bar{\Omega}) \\ & -0^{\circ}.0067 \cos (2\bar{\Gamma}' - \bar{\Omega}) + 0^{\circ}.0061 \cos (3\bar{L}' + \bar{\Gamma}' - 2\bar{L}) \\ & + 0^{\circ}.0064 \cos (3L' - \bar{\Gamma}' - \bar{\Omega}) \end{aligned}$$

$$\delta L \times 10^4 = -(47^{\circ}.8927 + 0.0482T) \sin \bar{\Omega} - 0^{\circ}.5658 \sin 2\bar{L}' \quad (5)$$

$$\begin{aligned} & +0^{\circ}.5800 \sin 2\bar{\Omega} && - 0^{\circ}.0950 \sin (2\bar{L}' - \bar{\Omega}) \\ & -3^{\circ}.5361 \sin 2\bar{L} && - 0^{\circ}.0725 \sin (3\bar{L}' - \bar{\Gamma}') \\ & -0^{\circ}.1378 \sin (3\bar{L} - \bar{\Gamma}) && + 0^{\circ}.0317 \sin (\bar{L}' + \bar{\Gamma}') \\ & +0^{\circ}.0594 \sin (\bar{L} + \bar{\Gamma}) && + 0^{\circ}.0161 \sin (\bar{L}' - \bar{\Gamma}' + \bar{\Omega}) \\ & +0^{\circ}.0344 \sin (2\bar{L} - \bar{\Omega}) && + 0^{\circ}.0158 \sin (\bar{L}' - \bar{\Gamma}' - \bar{\Omega}) \end{aligned}$$

$+0^{\circ}.0125 \sin (2\bar{\Gamma}' - \bar{\Omega})$
$+0^{\circ}.3500 \sin (\bar{L} - \bar{\Gamma})$
$+0^{\circ}.0125 \sin (2\bar{L} - 2\bar{\Gamma}')$
$-0^{\circ}.0144 \sin (3\bar{L}' + \bar{\Gamma}' - 2\bar{L})$
$-0^{\circ}.0122 \sin (3\bar{L}' - \bar{\Gamma}' - \bar{\Omega})$
$+0^{\circ}.1875 \sin (\bar{L}' - \bar{\Gamma}')$
$+0^{\circ}.0078 \sin (2\bar{L}' - 2\bar{\Gamma}')$
$+0^{\circ}.0414 \sin (\bar{L}' + \bar{\Gamma}' - 2\bar{L})$
$+0^{\circ}.0167 \sin (2\bar{L}' - 2\bar{L})$
$-0^{\circ}.0089 \sin (4\bar{L}' - 2\bar{L})$

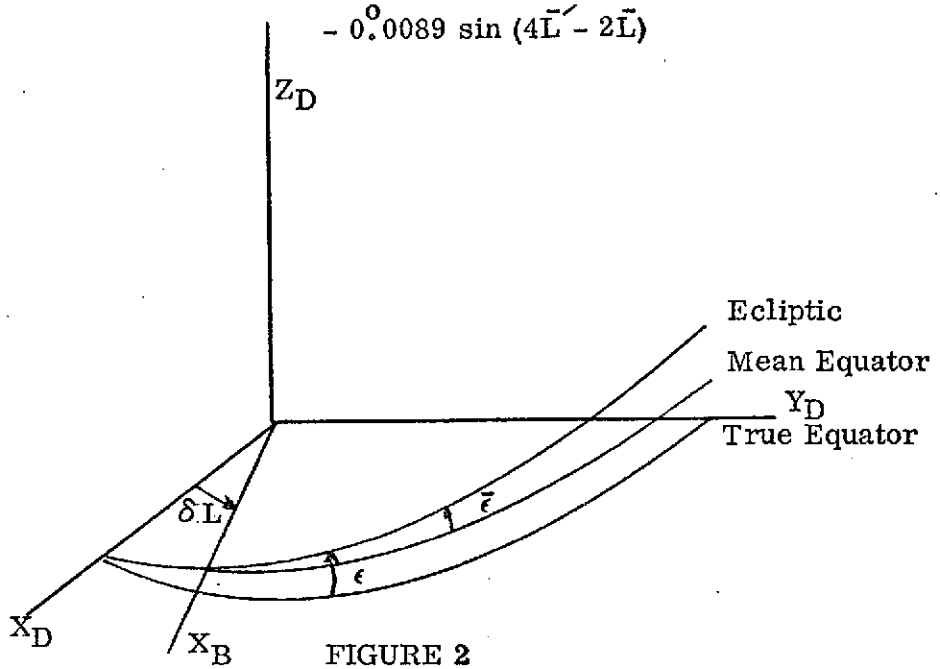


FIGURE 2

$$\bar{L}' = 64^{\circ}.37545167 + 13^{\circ}.1763965268d$$
$$-^{\circ}.001131575T - ^{\circ}.00113015T^2 + ^{\circ}.019\times10^{-4}\,T^3$$

$$\bar{L} = 280^{\circ}.08121009 + 0^{\circ}.9856473354d$$
$$+^{\circ}.000303T + ^{\circ}.000303T^2$$

$$\bar{\Gamma}' = 208^{\circ}.8439877 + 0^{\circ}.1114040803d \qquad (6)$$
$$-^{\circ}.010334T - ^{\circ}.010343T^2 - ^{\circ}.12\times10^{-4}T^3$$

$$\bar{\Omega} = 12^{\circ}.1127902 - 0^{\circ}.0529539222d$$
$$+^{\circ}.0020795T + ^{\circ}.002081T^2 + ^{\circ}.02\times10^{-4}T^3$$

$$\bar{\Gamma} = 282^{\circ}.08053028 + 0^{\circ}.470684\times10^{-4}d$$
$$+^{\circ}.00045525T + ^{\circ}.0004575T^2 + ^{\circ}.03\times10^{-4}T^3$$

$$\bar{\epsilon} = 23^{\circ}.4457587 - 0^{\circ}.01309404T - {}^{\circ}.0088\times10^{-4}T^2 + {}^{\circ}.0050\times10^{-4}T^3$$

Equations (5) are computationally inefficient, requiring the evaluation of 39 sines and cosines. The subroutine NUTATE computes the transformation by computing the sines and cosines of the angles $\bar{\epsilon}$, $\bar{\Omega}$, $\bar{\Gamma}-\bar{\Omega}$, $\bar{L}-\Omega$, $\bar{\Gamma}'-\Omega$, $\bar{L}'-\Omega$, with the remaining sines and cosines computed by trigonometric identities.

2.3 Earth-Fixed

The transformation from true equator, equinox of date to Earth-fixed coordinates is

$$T = \begin{bmatrix} \cos\xi & \sin\xi & 0 \\ -\sin\xi & \cos\xi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where

$$\bar{\xi} = 100^{\circ}.07554260 + 0^{\circ}.9856473460[d] + (2^{\circ}.9015)10^{-13}[d]^2 + \omega t$$

$$\omega = 0.00417807417/(1 + (5.21)10^{-13}[d]) \text{ deg/sec} \tag{8}$$

$$\delta\xi = \delta L \cos\bar{\epsilon}$$

$$\xi = \bar{\xi} + \delta\xi$$

The nutation in right ascension, $\delta\xi$ is the element n_{21} of the nutation matrix, equation (4).

2.4 Moon's Equator, Node of Date

The transformation from Earth's equator, equinox to Moon's equator, node may be written as shown on the next page:

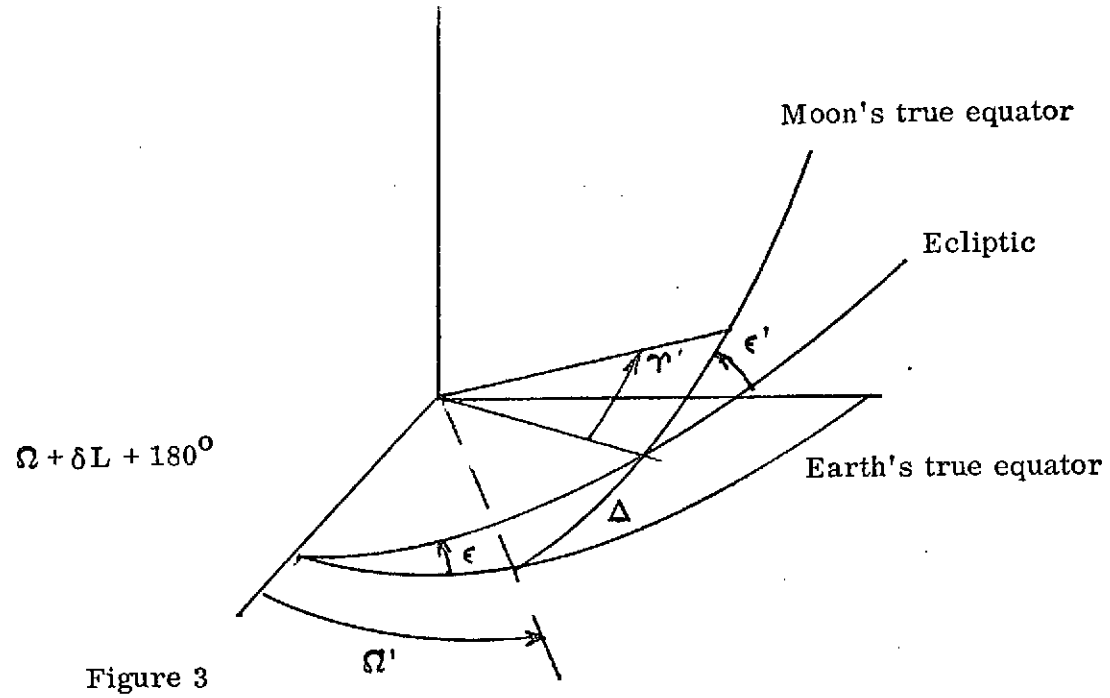


Figure 3

$$M = \begin{bmatrix} \cos\Delta & \sin\Delta & 0 \\ -\sin\Delta & \cos\Delta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos i & \sin i \\ 0 & -\sin i & \cos i \end{bmatrix} \begin{bmatrix} \cos\Omega & \sin\Omega & 0 \\ -\sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{9}$$

as in the reference, or

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\epsilon' & \sin\epsilon' \\ 0 & -\sin\epsilon' & \cos\epsilon' \end{bmatrix} \begin{bmatrix} -\cos(\Omega+\delta L) - \sin(\Omega+\delta L) & 0 \\ \sin(\Omega+\delta L) - \cos(\Omega+\delta L) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\epsilon & \sin\epsilon \\ 0 & -\sin\epsilon & \cos\epsilon \end{bmatrix} \tag{10}$$

where

$$\epsilon' = \bar{\epsilon} + \delta\epsilon'$$

$$\epsilon = \bar{\epsilon} + \delta\epsilon$$

$$\Omega = \bar{\Omega} + \delta\Omega$$

Retaining linear terms in $\delta\epsilon$, δL,

$$M = \bar{M}\, N^T$$

$$\bar{M} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\epsilon' & \sin\epsilon' \\ 0 & -\sin\epsilon' & \cos\epsilon' \end{bmatrix} \begin{bmatrix} -\cos\Omega & -\sin\Omega & 0 \\ \sin\Omega & -\cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\bar{\epsilon} & \sin\bar{\epsilon} \\ 0 & -\sin\bar{\epsilon} & \cos\bar{\epsilon} \end{bmatrix}$$

where N is the nutation matrix (4). Clearly, $\bar{M}$ is the transformation from Earth's mean equator, equinox to Moon's true equator, node.

The Earth's mean obliquity, $\bar{\epsilon}$, is given by (6). The mean inclination of the Moon's equator with the ecliptic, $\bar{\epsilon}'$, is

$$\bar{\epsilon}' = 1^\circ.535$$

and the librations in inclination and longitude are

$$\delta\epsilon' = -0.0297222 \cos(\bar{L}' - \bar{\Gamma}') + 0.0102777 \cos(\bar{L}' + \bar{\Gamma}' - 2\bar{\Omega})$$
$$-0.00305555 \cos(2\bar{L}' - 2\bar{\Omega})$$

$$\delta\Omega = \csc\bar{\epsilon}' \left\{-0.0302777 \sin(\bar{L}' - \bar{\Gamma}') + 0.0102777 \sin(\bar{L}' + \Gamma' - 2\bar{\Omega})\right.$$
$$\left.-0.0030555 \sin(2\bar{L}' - 2\bar{\Omega})\right\}$$

The transformation $\bar{M}$ is computed by subroutine NUTATE.

2.5 Moon-Fixed

The transformation from Moon's equator, node of date to equator, prime-meridian is

$$L = \begin{bmatrix} \cos\Upsilon' & \sin\Upsilon' & 0 \\ -\sin\Upsilon' & \cos\Upsilon' & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where

$$\Upsilon' = L' - \Omega$$

$$\delta L' = -0.^{\circ}003333 \sin (\bar{L}' - \bar{\Gamma}') + 0.^{\circ}0163888 \sin (\bar{L} - \bar{\Gamma})$$
$$+ 0.^{\circ}005 \sin (2\bar{\Gamma}' - 2\bar{\Omega})$$

The angular velocity of the Moon is the vector sum of the angular velocities

1. $\dot{L}' - \dot{\Omega}$ about the Moon's polar axis, and
2. $\dot{\Omega}$ about the normal to the ecliptic.

Since the inclination of the two axes is ϵ', the angular velocity is

$$\omega' = \begin{bmatrix} 0 \\ -\dot{\Omega} \sin \epsilon' \\ (\dot{L}' - \dot{\Omega}) + \dot{\Omega} \cos \epsilon' \end{bmatrix} \tag{14}$$

in equator, node components.

Subroutine NUTATE computes the polar component

$$\omega' = (\dot{L}' - \dot{\Omega} + \dot{\Omega} \cos \epsilon') \tag{15}$$

from

$$\omega' = \dot{L}' - \dot{\Omega} (1 - \cos \bar{\epsilon}' + \delta\epsilon' \sin \bar{\epsilon}')$$

$$1 - \cos \bar{\epsilon}' = .000358852$$

$$\sin \bar{\epsilon}' = .026787599$$

$$\dot{\bar{L}}' = 0.266170762 \times 10^{-5} - 0.12499171 \times 10^{-13} T \text{ (rad/sec)} \tag{16}$$

$$\dot{\bar{\Omega}} = -0.106969844 \times 10^{-7} + 0.23015329 \times 10^{-13} T$$

$$\dot{\delta} L' = -0.1535272946 \times 10^{-9} \cos (\bar{L}' - \bar{\Gamma}') + 0.569494067 \times 10^{-10} \cos (\bar{L} - \bar{\Gamma})$$
$$+ 0.579473484 \times 10^{-11} \cos (2\bar{\Gamma}' - 2\bar{\Omega})$$

$$\delta\Omega = -0.520642191\times10^{-7}\cos(\bar{L}'-\bar{\Gamma}) + 0.181177445\times10^{-7}\cos(\bar{L}'+\bar{\Gamma}-2\bar{\Omega}) - 0.106405786\times10^{-7}\cos(2\bar{L}'-2\bar{\Omega})$$

Retaining nine significant figures,

$$\begin{aligned}\omega' = {} & 0.266171146\times10^{-5} - .13499\times10^{-9}\cos(\bar{L}'-\bar{\Gamma}') \\ & + .5695\times10^{-10}\cos(\bar{L}-\bar{\Gamma}) - .645\times10^{-11}\cos(\bar{L}'+\bar{\Gamma}'-2\bar{\Omega}) \\ & + .380\times10^{-11}\cos(2\bar{L}'-2\bar{\Omega}) + .579\times10^{-11}\cos(2\bar{\Gamma}'-2\bar{\Omega}) \\ & + \delta\epsilon'\left[.139.10^{-8}\cos(\bar{L}'-\bar{\Gamma}') - .485\times10^{-9}\cos(\bar{L}'+\bar{\Gamma}'-2\bar{\Omega})\right. \\ & \left. + .285\times10^{-9}\cos(2\bar{L}'-2\bar{\Omega})\right]\end{aligned} \qquad (17$$

LIST OF SYMBOLS

Symbol	Reference	Quantity
L	L	Longitude of the Sun, from the equinox of date.
Γ	Γ	Longitude of perigee of the Sun.
g	g'	True anomaly of the Sun
L'	☾	Longitude of the Moon, measured in the ecliptic from the mean equinox of date to the ascending node of the lunar orbit, and then along the orbit.
Γ'	Γ'	Longitude of perigee of the Moon, measured as L'.
g'	g	True anomaly of the Moon.
ϵ	ϵ	Obliquity of the ecliptic.
ϵ'	I	Inclination of the Moon's equator with the ecliptic.
ω	ω	Angular velocity of the central body about its north polar axis.
ξ	ξ	Right ascension of the prime meridian (hour angle of the vernal equinox).
Ω	Ω	Longitude of the ascending node of the Moon's orbit on the ecliptic.
ω_p	ω	Argument of perigee of the Moon.
i	i	Inclination of the Moon's equator with the Earth's equator.
Ω'	Ω'	Right ascension of the ascending node of the Moon's equator on the Earth's equator.
Δ	Δ	Arc in the Moon's equator from the Earth's equator to the ecliptic.

Symbol	Reference	Quantity
ξ'	$\Delta - \Delta$	Right ascension of the Moon's prime meridian, from the ascending node of the Moon's equator on the ecliptic.
δL	$\delta\Psi$	Nutation in longitude of the Sun
$\delta\xi$	$\delta\xi$	Nutation in right ascension of the prime meridian.
$\delta\Omega$	σ	Libration in longitude of the ascending node.
$\delta L'$	t	Libration in longitude of the Moon.
$\delta\epsilon'$	δI	Libration in inclination.
$\delta\epsilon$	$\delta\epsilon$	Nutation in obliquity.
$90^o + z$	$90^o + z$	Right ascension of the mean equator of 1950, from the mean equinox of date.
$90^o - \zeta_o$	$90^o - \zeta_o$	Right ascension of the mean equator of date, from the mean equinox of 1950.
θ	θ	Inclination of the mean equator of date with the mean equator of 1950.

SUBROUTINE OBLATE

Calling Sequence: CALL OBLATE

Purpose: This subroutine calculates the approximate effect of the Earth's J_2 gravitational term on a trajectory. This subroutine is used with the Multiconic propagator.

Common Blocks Required: CONST, DUM, INPUT, STATE.

Subroutines Required: ORBIT, TRMN.

References: Paul Penzo, Computing Earth Oblateness Effects on Lunar and Interplanetary Trajectories, AIAA paper No. 70-97, January, 1970.

Inputs / Outputs

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
O	AM	1	DUM (9)	Mean anomaly at end of step
O	ELM	12	STATE(14)	Orbital elements and sines and cosines of elements of state
I	GME	1	CONST(7)	Earth's gravitational constant
O	PM	1	DUM(7)	Reciprocal of spacecraft's mean motion
I	T	1	STATE(10)	Time since state epoch
I	TF	1	INPUT(4)	Final time
I/O	X	6	STATE(1)	Spacecraft's position and velocity vectors.

Theory:

The approximate effect of J_2 on the orbital elements is determined in this subroutine. The time derivatives of the angular elements can be expressed by:

$$\frac{di}{dt} = \frac{r \cos\Theta}{h} A_n$$

$$\frac{d\Omega}{dt} = \frac{r \sin\Theta}{h \sin i} A_n \qquad (1)$$

$$\frac{d\omega}{dt} = -\cos i \frac{d\Omega}{dt} - \frac{1}{eh} \left[P \cos f A_r - (P+r) \sin f A_t \right]$$

where,

r = radial distance
f = true anomaly
ω = argument of perigee
i = inclination
Ω = longitude of the ascending node
$\Theta = \omega + f$
h = angular momentum
e = eccentricity
P = semi-latus rectum

and where A_r, A_t, A_n are the components of the disturbing acceleration caused by Earth oblateness, see Figure 1.

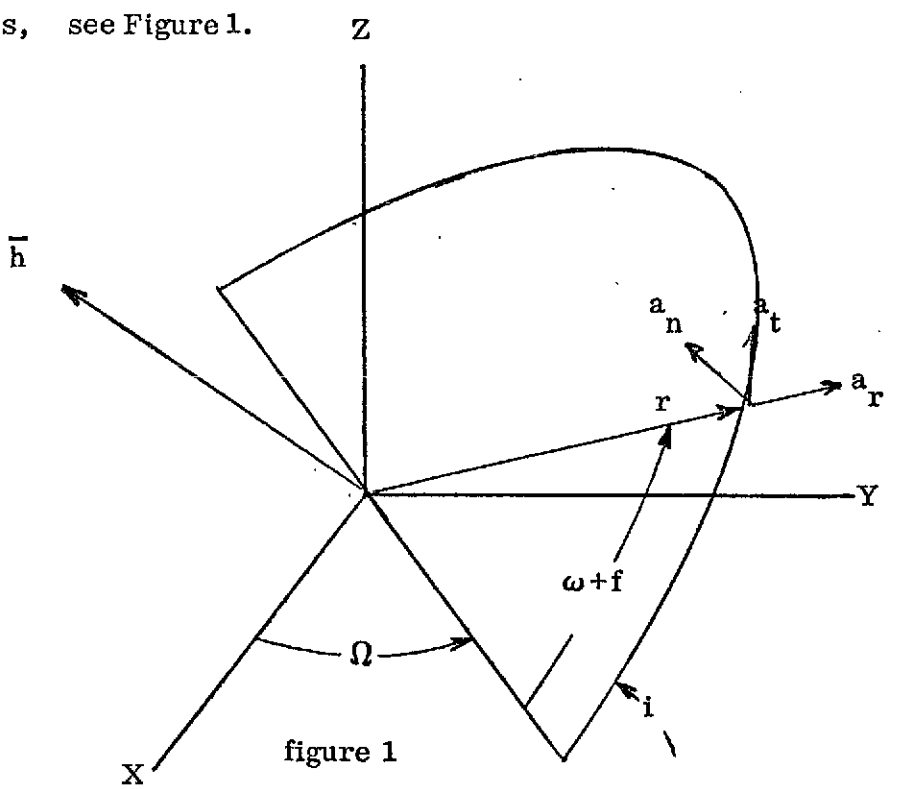


figure 1

It is more convenient to exchange the true anomaly for time as the independent variable. To do this, the expression for angular momentum may be used, as

$$h = r^2 \frac{df}{dt}$$

then,

$$\frac{di}{dt} = \frac{di}{df} \frac{df}{dt} = \frac{h}{r^2} \frac{di}{df} \tag{2}$$

Similar equations are used for the other elements. Thus equation (1) becomes,

$$\frac{di}{df} = \frac{r^3 \cos \theta}{h^2} A_n$$

$$\frac{d\Omega}{df} = \frac{r^3 \sin\theta}{h^2 \sin i} A_n \tag{3}$$

$$\frac{d\omega}{df} = -\cos i \frac{d\Omega}{df} - \frac{r^2}{eh^2} \left[P \cos f \; A_r - (P + r) \sin f \; A_t \right]$$

The acceleration due to the second harmonic term is given by,

$$\begin{Bmatrix} A_r \\ A_t \\ A_n \end{Bmatrix} = \frac{-3 \mu J_2 r_{eq}^2}{2 r^4} \begin{pmatrix} -3 \sin^2 \theta & \sin^2 i \\ \sin^2 \theta & \sin^2 i \\ \sin \theta & \sin^2 i \end{pmatrix} \tag{4}$$

where μ is the Earth's gravitational constant

r_{eq} is the Earth's equatorial radius

$J_2 = .0010823$

The substitution of equation (4) into equation (3) provides the final form of the derivatives of the orbital elements. The only variable remaining besides f and θ is r. However, r can be eliminated by the use of

$$r = \frac{P}{1 + e \cos f} \tag{5}$$

Now, equations (3), (4) and (5) can be combined and analytically integrated to provide the effects of J_2 on the orientation elements. Since the changes in the elements are small through the integration interval, the orbital elements can be held constant during the integration for a first approximation to their variations. Thus, the variation in the elements becomes,

$$\Delta i = K \sin i \cos i \left[\frac{1}{2} \sin^2\theta - \frac{e}{3}\cos\omega\cos^3\theta + \frac{e}{3}\sin\omega\sin^3\theta\right]_{\theta_0}^{\theta_1}$$

$$\Delta \Omega = K \cos i \left[\frac{1}{2}(\theta - \sin\theta\cos\theta) + \frac{e}{3}(\cos\omega\sin^3\theta + \sin\omega\cos^3\theta) + e \sin\omega \sin^3\theta\right]_{\theta_0}^{\theta_1} \tag{6}$$

$$\Delta \omega = - \cos i \Delta \Omega + \Delta \omega_{r\theta}$$

where

$$K = \frac{-3\mu^2 J_2 r_{eq}}{h^4}$$

$$\Delta \omega_{r_\theta} = \frac{-K}{2e}\left[e\left(1 - \frac{3}{2}\sin^2 i\right) f + \left(1 + \frac{3}{2}e^2\right)\sin f + e \sin f \cos f + \frac{e^2}{3}\cos^2 f \sin f\right]_{f_0}^{f_1}$$

$$+ \frac{K}{2e}\sin^2 i \sin\omega\cos\omega\left[4\cos f + 3e\cos^2 f - \frac{2}{3}(7 - e^2)\cos^3 f - 6e\cos^4 f - 2e^2\cos^5 f\right]_{f_0}^{f_1}$$

$$+ \frac{K}{2e}\sin^2 i \left(A \sin f + B \sin f \cos f + C \cos^2 f \sin f + 3D\cos^3 f \sin f + eD\cos^4 f \sin f\right)_{f_0}^{f_1}$$

where

$$A = \frac{1}{3}(7 + 2e^2)\cos^2\omega + \frac{3}{2}(1 + 2e^2)\sin^2\omega$$

$$B = \frac{3}{2}e$$

$$C = \frac{1}{3}(e^2 - 7)\cos^2\omega + \frac{1}{3}(7 + 2e^2)\sin^2\omega$$

$$D = -e(\cos^2\omega - \sin^2\omega)$$

The in-plane elements, e and p, are determined if the energy, c, and the angular momentum, h, are known. The variation in the energy is determined since the force field of an oblate Earth is derivable from a potential function. Considering only the second harmonic term, the energy per unit mass is,

$$C = \frac{V^2}{2} - \frac{\mu}{r} - \frac{\mu J_2 \, r_{eq}^2}{2 r^2}(1 - 3\cos^2\phi) \tag{7}$$

where ϕ is the colatitude. The Keplerian portion of the energy remains constant. Thus, the variation in energy is only the third term of equation (7), or

$$\Delta C = \left[\frac{-\mu J_2 \, r_{eq}^2}{2 r^3}(1 - \cos^2\phi) \right]_{r_0}^{r_1} \tag{8}$$

The angular momentum does vary and its derivative is given by,

$$\frac{d\bar{h}}{dt} = \bar{r} \times \dot{\bar{r}}$$

However, since the potential has no longitude dependence, the z - component of the derivative in equation (9) is zero. As seen from Figure 1, the z - component of the angular momentum is given by,

$$h_z = h \cos i$$

then

$$\frac{dh_z}{dt} = \frac{dh}{dt} \cos i - h \sin i \frac{di}{dt} = 0 \qquad (10)$$

Thus, the variation in the magnitude of the angular momentum over the same interval as Δi is,

$$h = h \tan i \Delta i$$

The elements at the end of a step can now be obtained from,

$$i_1 = i + \Delta i$$

$$\Omega_1 = \Omega + \Delta\Omega$$

$$\omega_1 = \omega + \Delta\omega$$

$$P_1 = \frac{h_1^2}{\mu}$$

$$e_1 = \sqrt{1 + \frac{2P_1 C_1}{\mu}}$$

where

the subscript 1 denotes the values at the end of a step, and

$C_1 = C_k + \Delta C$

C_k is the Keplerian energy

$h_1 = h + \Delta h$

Description:

The variations in the orbital elements are determined in a straightforward manner from the above equations. The initial state is brought into the routine as position and velocity vectors in x of STATE common.

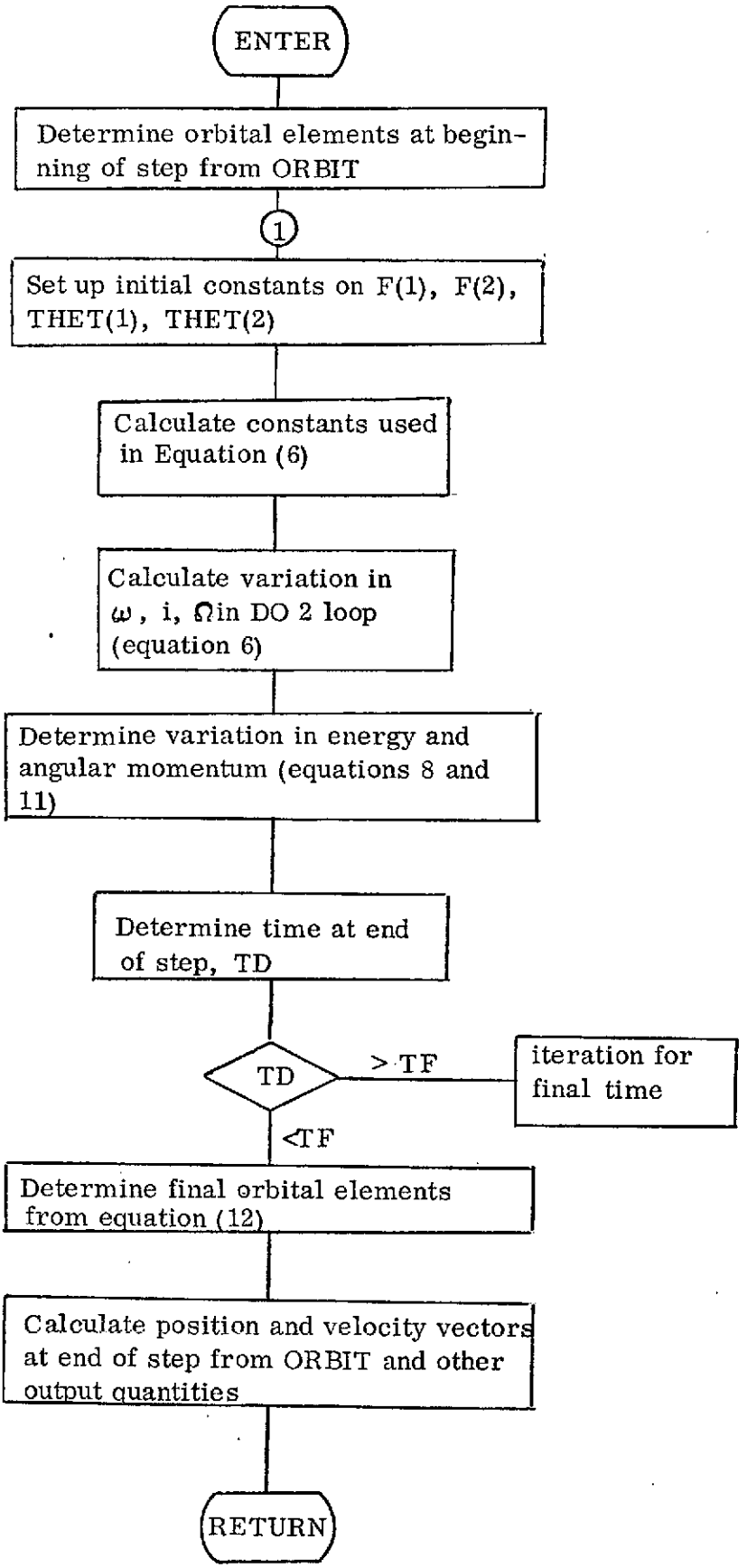
The orbital elements are determined using subroutine ORBIT. Next, the true anomaly f, and $\omega + f$ are determined at the beginning and at the end of the step and stored in the arrays F and THET, respectively. The variation in ω, i and r is determined in the DO 2 loop, where the index refers to the limits of integration.

The variation in the energy and angular momentum is calculated from equations (8) and (11). Finally, the orbital elements at the end of a step are determined from equation (12).

If the time at the end of the step is greater than the final time, logic flow transfers to statement 5 where an iteration scheme is employed to determine the radius corresponding to the final time.

The position and velocity vectors, mean, anomaly, and reciprocal of the mean motion are determined at the end of the step before the subroutine terminates.

SUBROUTINE OBLATE



SUBROUTINE OBLE

Calling Sequence: CALL OBLE

Purpose: This subroutine calculates the oblateness accelerations of the Earth.

Common Block Required: CONST, CNTRL, GRAVTY, STATE, PERT

Subroutines Required: M5OMDT, NUTATE, ROTATE

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	GM	12	CONST	Gravitational potential
I	JC	1	CNTRL	Central Planet
I	POS	3	GRAVTY	Position vector
I	UJT	1	STATE(32)	Current julian date
I/O	XDUM	3	PERT	Acceleration Vector

Usage: JC must = 3
Position Vector must be in GRAVTY (Mean Equator of 1950).
Acceleration Vector must be in PERT (Mean Equator of 1950).

Description:

This subroutine first transforms the position vector of the S/C to mean equator of date. It then calculates the acceleration acting on the S/C due to the aspherical Earth with harmonics A_2, A_3 and A_4. The equations are as follows:

$$\ddot{x} = \frac{3x}{2r^5} A_2 \, GM(3) \, (1-5U_z^2) + \frac{5xz}{2r^7} A_3 \, GM(3) \, (7U_z^2-3)$$

$$+ \frac{5x}{8r^7} A_4 \, GM(3) \, (3-42U_z^2 + 63U_z^4)$$

$$\ddot{y} = \frac{y}{x} \ddot{x}$$

$$\ddot{z} = \frac{-3z}{2r^5} A_2 \, GM(3) \, (3-5U_z^2) + \frac{3}{2r^5} A_3 \, GM(3) \, (1-10U_z^2 + \frac{35}{3} U_z^4)$$

$$+ \frac{5z}{8r^7} A_4 \, GM(3) \, (15-70U_z^2 + 63U_z^4)$$

where $U_z = \frac{z}{r}$

These accelerations are then transformed back into Earth mean equinox and equator of 1950, and then added to the total acceleration vector XDUM.

SUBROUTINE OBLTY

Calling Sequence: CALL OBLTY (EP, B)

Purpose: OBLTY calculates the transformation matrix that rotates from the Earth's ecliptic to the Earth's equator.

Common Blocks Required: CONST, INPUT

Subroutines Required: None

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
O	B	3,3	Calling Operand	Transformation Matrix
I	EP	1	Calling Operand	Julian centuries since 1900
I	RAD	1	CONST (1)	Degrees per radian

Theory:

The transformation from the Earth's ecliptic to the Earth's equator involves a simple rotation about the vernal equinox through the angle θ ,

where

$$\theta = 1.03 \times 10^{-8} t^{3} - 1.23 \times 10^{-7} t^{2} - 3.562 \times 10^{-3} t + 23.452294 \text{ degrees}$$

The time variable, t, represents the number of Julian centuries since 1900 (days since 1900.0 times 10^{-4}).

The matrix is given by:

$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

Description:

OBLTY calculates the transformation matrix for the rotation from the Earth's ecliptic to the Earth's equator. This matrix is loaded into B. The input EP represents the number of days since 1900.0 times 10^{-4}.

SUBROUTINE OBSET

Calling Sequence: CALL OBSET (MODEL)

Purpose: OBSET initializes the spherical harmonics coefficients for a given field model.

Common Blocks Required: FIELDM

Subroutines Required: None

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
O	MMOD	1	FIELDM(309)	Highest tesseral order
I	MODEL	1	Calling Operand	Model number
O	NMOD	1	FIELDM(308)	Highest zonal order
O	TSRL	16, 17	FIELDM (17)	Tesseral array
O	ZONL	16	FIELDM (1)	Zonal vector

Theory:

The relationships between zonals and tesserals and the sine and cosine coefficients of a spherical harmonic potential field are as follows:

$ZONL(i) = -C_{io}$ $\quad i = 1, 2, \ldots\ldots\ldots$

$TSRL(i, j) = C_{ij}$ $\quad o \; j \; i, \; i = 1, 2, \ldots\ldots\ldots$

$TSRL(j, i + 1) = S_{ij}$ $\quad o \; j \; i = 1, 2, \ldots\ldots$

Description:

OBSET initializes the spherical harmonic coefficients according to the field model selected. The values NMOD and MMOD are also set to indicate the size of the selected model. NMOD indicates the highest-ordered zonal and MMOD indicates the highest tesseral order. The following chart indicates the possible values of MODEL and the corresponding fields:

MODEL	FIELD DESCRIPTION	NMOD	MMOD
1	Houşton L1 model of lunar field	3	3
3	JPL 15-8 model of the lunar field	15	8
4	Mars field--post Mariner 1971	4	3
5	Clears coefficient arrays for loading of input field	0	0

See Figure 1 for the values of the C and S terms for the first three models above.

HOUSTON L1 MODEL OF LUNAR FIELD, NMOD = 3, MMOD = 3

```
C( 2, J)= -0.2071D-03  0.0          0.2072D-04
C( 3, J)=  0.2100D-04  0.3400D-04   0.0          0.2583D-05
S( 2, J)=  0.0         0.0          0.0
S( 3, J)=  0.0         0.0          0.0          0.0
```

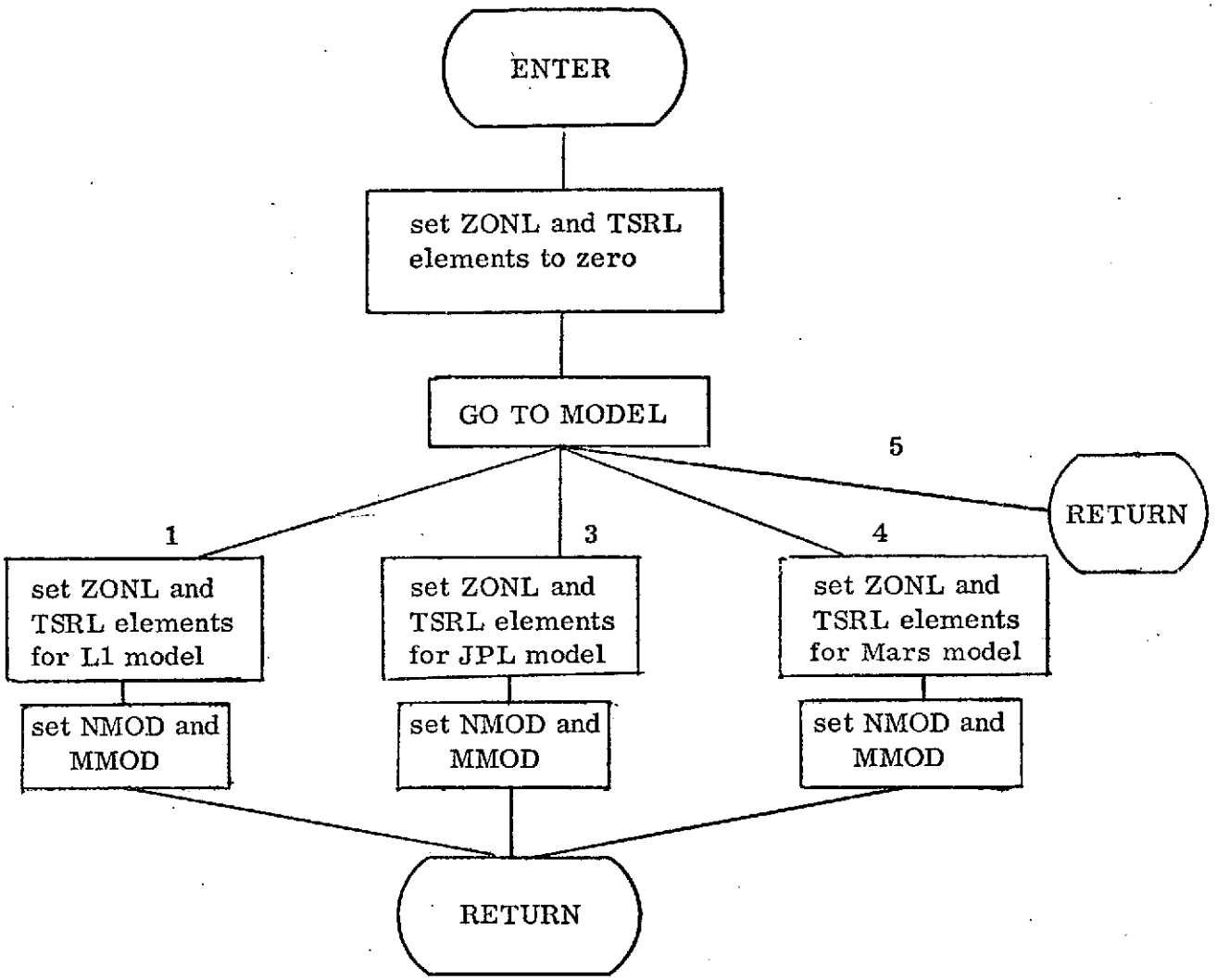
JPL 15-8 MODEL OF LUNAR FIELD, NMOD = 15, MMOD = 8

```
C( 2, J)= -0.1996D-03  0.8171D-05  0.2359D-04
C( 3, J)= -0.5878D-05  0.3001D-04  0.4698D-05  0.4847D-05
C( 4, J)=  0.1195D-04 -0.2226D-05 -0.2418D-05  0.2306D-06 -0.4547D-06
C( 5, J)= -0.4544D-05 -0.2455D-05  0.8880D-06 -0.7774D-06  0.1027D-06
          -0.3044D-08
C( 6, J)=  0.1088D-05 -0.6373D-05 -0.6917D-06 -0.2041D-06  0.4620D-07
          -0.7564D-09 -0.7277D-08
C( 7, J)=  0.1779D-04  0.1324D-05 -0.1293D-06 -0.1305D-06 -0.3523D-08
          0.6061D-08 -0.2837D-09  0.2454D-10
C( 8, J)= -0.5967D-05 -0.8040D-05  0.2650D-06 -0.6771D-07  0.1925D-07
          0.4844D-09  0.1252D-08 -0.4523D-10 -0.3178D-11
C( 9, J)= -0.3206D-05  0.0         0.0         0.0         0.0
          0.0         0.0         0.0         0.0
C(10, J)=  0.1367D-05  0.0         0.0         0.0         0.0
          0.0         0.0         0.0         0.0
C(11, J)= -0.7311D-05  0.0         0.0         0.0         0.0
          0.0         0.0         0.0         0.0
C(12, J)=  0.1251D-04  0.0         0.0         0.0         0.0
          0.0         0.0         0.0         0.0
C(13, J)= -0.3315D-04  0.0         0.0         0.0         0.0
          0.0         0.0         0.0         0.0
C(14, J)=  0.1044D-04  0.0         0.0         0.0         0.0
          0.0         0.0         0.0         0.0
C(15, J)= -0.2977D-04  0.0         0.0         0.0         0.0
          0.0         0.0         0.0         0.0
S( 2, J)=  0.0        -0.7213D-05  0.4538D-05
S( 3, J)=  0.0         0.1421D-05  0.5748D-06 -0.2919D-05
S( 4, J)=  0.0         0.3299D-05 -0.2389D-05 -0.6222D-06  0.4248D-06
S( 5, J)=  0.0        -0.6925D-05 -0.3178D-06  0.1159D-05  0.6296D-07
          -0.3012D-07
S( 6, J)=  0.0         0.5342D-05 -0.1157D-05 -0.3484D-06 -0.9870D-07
          -0.1654D-07  0.2373D-08
S( 7, J)=  0.0        -0.1579D-05  0.1152D-06  0.2440D-06  0.1295D-07
          0.5582D-08  0.1227D-08 -0.8131D-10
S( 8, J)=  0.0         0.6208D-05 -0.1767D-06 -0.1437D-06 -0.1757D-07
          0.2391D-08 -0.2390D-09 -0.5813D-10  0.7798D-11
```

```
MARS FIELD-POST MARINER 1971, NMOD = 4, MMOD = 3

C( 2, J)= -0.1960D-02  0.0          -0.5400D-04
C( 3, J)= -0.3050D-04  0.2770D-05    0.0          0.0
C( 4, J)= -0.3140D-04  0.0           0.0          0.0
S( 2, J)=  0.0         0.0           0.2900D-04
S( 3, J)=  0.0         0.2500D-04    0.0          0.0
```

SUBROUTINE OBSET



SUBROUTINE ORBIT

Calling Sequence: CALL ORBIT (K, J, X, XD, GM, ELM)

Purpose: ORBIT calculates orbital elements from position and velocity, or position and velocity from orbital elements.

Common Blocks: Required: CONST

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	ELM	6 or 12	Calling Operand	Orbital elements (see description)
I	GM	1	Calling Operand	Gravitational constant of planet
I	J	1	Calling Operand	Direction flag (see description)
I	K	1	Calling Operand	Condition flag (see description)
I	PI2	1	CONST(3)	$2*\pi$
I/O	X	3	Calling Operand	Position vector
I/O	XD	3	Calling Operand	Velocity vector

Theory:

Consider first the calculation of the orbital elements P, e, f, ω, i, Ω from the position and velocity vectors $\bar{X}$ and $\bar{V}$. The angular momentum vector is given by

$$\bar{H} = \bar{X} \times \bar{V} = \left(h \sin(i) \sin(\Omega),\ -h \sin(i) \cos(\Omega),\ h \cos(i)\right)$$

so that $|\bar{H}| = |\bar{X} \times \bar{V}| = h$

and $P = h^2 / GM$

The equation of motion of a body rotating about a central body is given by

$R = P / (1+e \cos f)$ where R is the distance to the body.

Thus $R = |\bar{X}|$ and $e \cos(f) = (P-R)/R$,

which implies $- e \sin(f)\, \dot{f} = - (P/R^2)\, \dot{R}$

$$\text{or } e \sin(f) = \frac{P}{h} \left(\frac{X_1 V_1 + X_2 V_2 + X_3 V_3}{R} \right) \quad \text{since } h = R^2 f.$$

From e cos (f) and e sin (f), both e and f can be calculated, as long as e is not equal to zero, which is explained later on.

The calculation of i and Ω is straightforward given the components of $\vec{H}$, as long as the inclination is not 0^o or 180^o, in which case Ω is set to zero.

In order to find ω, let $u = f + \omega$. Then the geometry of the orbit can be used to yield the following equations:

$$R \cos(u) = X_1 \cos(\Omega) + X_2 \sin(\Omega)$$
$$R \sin(u) = \left(-X_1 \sin(\Omega) + X_2 \cos(\Omega)\right) \cos(i) + X_3 \sin(i)$$

Now the calculation of u is straightforward, and

$\omega = u - f$, unless e equals zero,

in which case ω is set to zero and $f = u$.

The second part of the subroutine is the reverse case, i.e., given orbit elements find $\bar{X}$ and $\bar{V}$. The first step in this process is to calculate the sines and cosines of ω, i, and Ω (if K = 5 this is input, also if K = 6 e, ω, and f must be calculated as well as the above sines and cosines). After this, a transformation is set up which represents rotations through ω, i, and Ω to transform orbit coordinates to whatever coordinate system you may be in. The rotations and the resulting matrix are as shown on the following page.

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos\Omega & -\sin\Omega & 0 \\ \sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos i & -\sin 1 \\ 0 & \sin i & \cos i \end{pmatrix} \begin{pmatrix} \cos\omega & -\sin\omega & 0 \\ \sin\omega & \cos\omega & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X' \\ Y' \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} \cos(\omega)\cos(\Omega) - \cos(i)\sin(\Omega)\sin(\omega) & -\sin(\omega)\cos(\Omega) - \cos(i)\sin(\Omega)\cos(\omega) \\ \cos(\omega)\sin(\Omega) + \cos(i)\cos(\Omega)\sin(\omega) & -\sin(\omega)\sin(\Omega) + \cos(i)\cos(\Omega)\cos(\omega) \\ \sin(i)\sin(\omega) & \sin(i)\cos(\omega) \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix}$$

Here x' and y' refer to a coordinate system in the orbit plane with the x-axis pointing toward pericenter. Thus finding position and velocity vectors is reduced to finding the vectors in the orbit plane and then transforming them.

For position, R can be found by

$$R = \frac{P}{1 + e\cos(f)}$$

and then

$$x' = R\cos(f)$$

$$y' = R\sin(f)$$

For the velocity, the magnitude can be found by the energy relationship

$$E = -\frac{\mu}{2a} = \frac{V^2}{2} - \frac{\mu}{R}$$

$$\text{or } V = \sqrt{\frac{2\mu}{R} - \frac{(1-e^2)}{P}\mu}$$

thus $V_1' = -V\sin(f-\gamma)$ and $V_2' = V\cos(f-\gamma)$

where γ is the flight path angle and is given by

$$\gamma = \arctan\left(e\sin(f) \,/\, \left(1 + e\cos(f)\right)\right)$$

Once V_1' and V_2' are calculated, the velocity vector may be found by transforming the primed system as was done before.

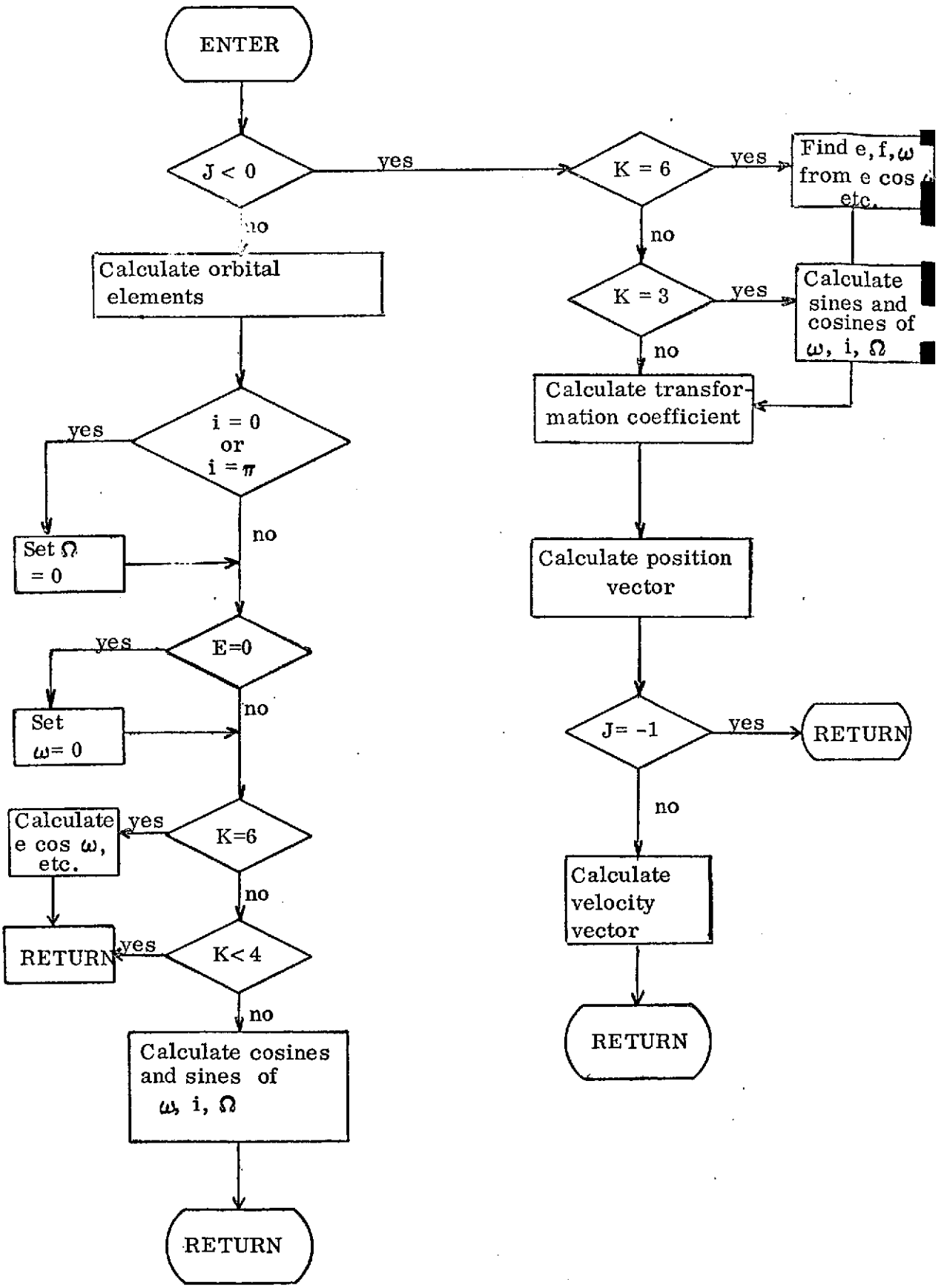
Description:

ORBIT provides the interchange between position and velocity coordinates and orbital elements. If the direction flag, J, is greater than or equal to zero, orbital elements will be calculated from position and velocity. Note that both position and velocity vectors are required for this transformation. Three different sets of orbital elements may be calculated, depending on the condition flag, K. If K is equal to three, only the standard orbital elements will be output, i.e., the semi-latus rectum (P), eccentricity (e), true anomaly(f), angle of pericenter (ω), inclination (i), and angle of ascending node (Ω) will be output in ELM(1) —— ELM(6). If K equals five, the following values will be output in ELM(7) —— ELM(12) in addition to the previous values: cos (ω), sin (ω), cos (i), sin (i), cos (Ω), and sin (Ω). If K equals six, the following values will be output in ELM(1) —— ELM(6): P, e cos (ω), e sin (ω), ω + f, i, Ω. No other values will be output.

If J is negative on input, then position and/or velocity will be output as follows: J equal to minus one implies that only the position vector is desired. Other negative values indicate that both the position vector and the velocity vector are desired.

The value of K in this case indicates the type of orbital elements that were input. K equal to one implies that only the standard orbital elements were input. K equal to five implies that the sines and cosines of ω, i, and Ω were input in addition to the standard elements. K equal to six implies that e cos (ω), e sin (ω), and ω+ f were input in place of e, f, and ω.

SUBROUTINE ORBIT



SUBROUTINE ORIENT

Calling Sequence: CALL ORIENT (A, U, S, CD, SC, K)

Purpose: To find the angle through which one vector must be rotated about a second vector in order that the first vector should form a given angle with a third vector.

Common Blocks Required: None

Subroutines Required: CROSS

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	A	3	ARGUMENT LIST	Rotated vector.
I	U	3	ARGUMENT LIST	Rotation vector.
I	S	3	ARGUMENT LIST	Fixed reference vector.
I	CD	1	ARGUMENT LIST	Cosine of desired angle.
O	SC	4	ARGUMENT LIST	Sines and cosines of the solutions
O	K	1	ARGUMENT LIST	Indicator for solution type

Description:

A principal use of this subroutine is to locate the times of day (right ascension of the Earth-fixed orbital plane's ascending node) when the touch-down point for Earth returns lies in the return orbit plane. This is accomplished by asking the subroutine for sines and cosines of the incremental right ascension required for the dot product of the normal to the return orbit plane and the touch-down radius vector to be zero. This subroutine can also solve the slightly more general problem of achieving a desired non-right-angle between two vectors by rotating one of them about a given axis.

Let $T_u(\theta)$ be the 3x3 orthogonal matrix which, when applied to any non-zero vector not parallel to u, rotates that vector about u through an angle, θ, measured in the plane normal to u. Thus, $\bar{a} = T_u(\theta)\,a$ is a vector which sweeps out a cone about u as θ increases from 0 degrees to 360 degrees. The problem solved by this subroutine is then: Find θ (or $\sin\theta$ and $\cos\theta$) such that

$$\bar{a} \cdot s = |\bar{a}|\,|s| \cos\varphi.$$

Now
$$\bar{a} \cdot s = \bar{a}^T s = \bar{a}^T\, T_u(-\theta)\, s$$

$$= a^T \left(\cos\theta\, I + (1-\cos\theta)\,\frac{uu^T}{u^T u} - \sin\theta\,\frac{ux}{|u|}\right) s$$

$$= \cos\theta\; a^T s + (1-\cos\theta)\,\frac{a^T u\, u^T s}{u^T u} - \frac{a^T uxs}{|u|}\,\sin\theta.$$

The equation to be solved then becomes

$$|a|\,|s|\cos\varphi = a\cos\theta + \beta\sin\theta + \gamma,$$

where
$$\gamma = \frac{(a^T u)\,(u^T s)}{u^T u}$$

$$\beta = \frac{(sxu)^T a}{|u|}$$

and

$$\alpha = a^T s - \gamma$$

or

$$\bar{\alpha}\cos\theta + \bar{\beta}\sin\theta + \bar{\gamma} = 0,$$

where
$$\bar{\gamma} = \frac{\gamma}{|a|\;\;|s|} - \cos\varphi,$$

$$\bar{\beta} = \frac{\beta}{|a|\;|s|},$$

$$\bar{\alpha} = \frac{\alpha}{|a|\;\;|s|}.$$

The solutions of this equation for $\sin\theta$ amd $\cos\theta$ are given by

$$\begin{bmatrix} \sin\theta \\ \cos\theta \end{bmatrix} = \frac{1}{\bar{a}^2 + \bar{\beta}^2} \begin{bmatrix} \bar{\beta} & \bar{\alpha} \\ \bar{\alpha} & -\beta \end{bmatrix} \begin{bmatrix} -\bar{\gamma} \\ \pm\sqrt{\bar{\alpha}^2 + \bar{\beta}^2 - \bar{\gamma}^2} \end{bmatrix}$$

It can be seen that two solutions exist - one for each sign on $\sqrt{\bar{\alpha}^2 + \bar{\beta}^2 - \bar{\gamma}^2}$.

When $\bar{\alpha}^2 + \bar{\beta}^2 = \bar{\gamma}^2$, however, only one solution exists. This case will occur when, for the use mentioned, the latitude of touch-down is exactly equal to the inclination of the return orbit. If the latitude is greater than the inclination $\bar{\alpha}^2 + \bar{\beta}^2 < \bar{\gamma}^2$ and a real solution is impossible. The subroutine then sets the indicator, K, to a positive value and provides the solutions for θ when the angle between a and s is a maximum or minimum.

SUBROUTINE OUTPUT

Calling Sequence: CALL OUTPUT (T, X)

Purpose: This routine outputs the spacecraft's state with respect to the launch and/or target planets in the desired coordinate system.

Common blocks required: CNTRL, CONST, INPUT, INTVAR, PLNET, STATE

Subroutines required: M5OLEQ, M5OMDT, M5OMDT, OBLTY, ORBIT, ROTATE, SHORB2, TRMN.

Input / Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DESCRIPTION
I	DJO	1	INPUT (46)	Modified julian date of state epoch
I	GM	12	CONST (5)	Gravitational constants
I	JC	1	CNTRL (7)	Central planet number
I	JL	1	INPUT (1015)	Launch planet number
I	JT	1	INPUT (1031)	Target planet number
I	KOUTPT	2	INPUT (1039)	Launch and target planet output coordinate systems
I	KOUT9	1	INPUT (1058)	Auxiliary output flag
I	KP	12	INPUT (1001)	Planets in system
I	KTERM	1	INPUT (1059)	Type of auxiliary output
I	METH	1	INPUT (1013)	Trajectory propagation method
I	PCON	1	INPUT (05)	Position units conversion factor
I	RAD	1	CONST (1)	Radians to degrees conversion factor
I	T	1	calling argument	Time since DJO
I	VCON	1	INPUT (6)	Velocity units conversion factor
I	X	6	Calling argument	Position and velocity vectors of state to be output

Description:

This subroutine outputs the state position and velocity vectors and orbital elements with respect to the launch and target planets in a desired coordinate system. The state and time are input via the argument list. The state is translated to the launch and target planets and rotated to the desired coordinate system before output. The output coordinate system is determined by the KOUTPT flag as follows:

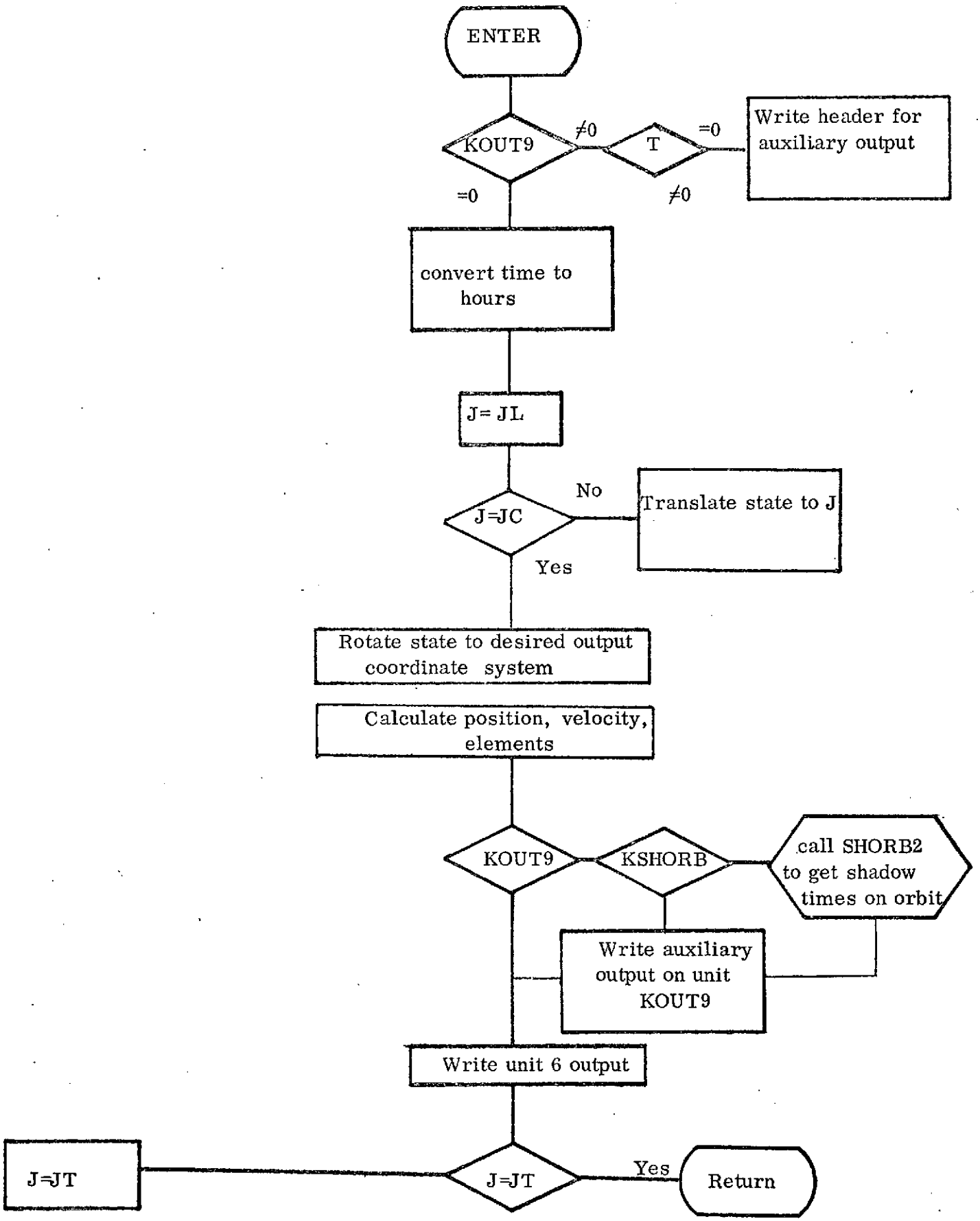
KOUTPT	= 1	Mean equinox and ecliptic of date
	= 2	True equator and prime meridian
	= 3	Mean equator and equinox of 1950
	= 4	True equator and equinox of date
	= 5	no output

The subroutine also has provisions for an auxiliary abbreviated output written on the unit number designated by KOUT9. If this flag is non-zero, output is presented on unit KOUT9 in a form specified by KTERM as follows:

KTERM	= 1	orbital elements
	= 2	position and velocity vectors
	= 3	1 and 2

The coordinate system of the output is also specified by KOUTPT. This subroutine also has a provision to determine the time in the shadow cone during one orbit assuming constant elements. The output is presented in the auxiliary abbreviated output. Thus, the KOUT9 flag must be non-zero. Also, the KSHORB flag must be 1 before subroutine SHORB2 is called to perform the shadow calculations.

SUBROUTINE OUTPUT



SUBROUTINE OUT1

Calling Sequence: CALL OUT1

Purpose: OUT1 determines the print interval and controls the interpolation logic for printing.

Common Blocks Required: INPUT, INTVAR, SAVE, STATE

Subroutines Required: INTERP, OUTPUT

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DJO	1	INPUT(46)	Modified Julian date of launch epoch
I	DTOUT	10	INPUT(460)	Printing intervals
I	T	1	STATE(10)	Seconds since state epoch
I	TOUT	10	INPUT(450)	Switching times of printing intervals
I	TOUTL	1	SAVE(40)	Time of last print
I	X	6	STATE(1)	Current state
I	UJT	1	STATE(32)	Current modified Julian date

Description:

This subroutine finds the print times and uses subroutine INTERP and OUTPUT to output the state at the print times. OUT1 is called from TIMEC at the beginning of each numerical integration step. The only print times determined on each call to OUT1 are print times which lie in the last numerical integration step.

The print interval is determined from the print table consisting of TOUT and DTOUT. The print time, TDUM, is determined from TDUM = TOUTL + DTOUT (IDUM) (1)

where TOUTL is the last print time and DTOUT(IDUM) is the current print interval determined from the print table.

If this time is greater than the current time, no printing is done. If not, the state is determined at the print time using the interpolation logic from subroutine INTERP and the state is then output in subroutine OUTPUT. If the print time is within 10 seconds of the current time, the state is not interpolated for and OUTPUT is called directly.

The next print time is then determined from equation (1) and the whole process is repeated until the print time is greater than the current time. The subroutine terminates when this criterion is satisfied.

SUBROUTINE PLANET

Calling Sequence: CALL PLANET

Purpose: Subroutine PLANET supervises the calculation of the planet's position and velocity vectors.

Common Blocks Required: CETBL2, CETBL3, CNTRL, INPUT, INTVAR, STATE

Subroutines Required: LUNA, SOL, READE

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	EJO	1	STATE(26)	Ephemeris date of launch epoch
I	EJT	1	STATE(28)	Current ephemeris date
I	JC	1	CNTRL(7)	Central planet number
I	JMN	1	INPUT(1017)	Ephemeris type flag
I	KP	12	INPUT(1001)	Planets in system
I	KREAD	1	CNTRL(8)	Flag to determine if tape (or disk) read necessary
I	TAB3	1	CETBL3(1)	Time of last tape (or disk) read

Description:

This subroutine controls the calculation of the planet's position and velocity vectors. The JMN flag determines the type of ephemeris used as follows:

JMN =
1. Mean elements
2. Mean elements for Sun and mean elements plus first-order corrections for Moon
3. Ephemeris tape read
4. Mean elements for Sun, osculating elements for Moon
5. Ephemeris disk using Goddard's direct read feature

If JMN is not equal to 3 or 5, subroutines SOL and LUNA are called to determine the planets' positions if the respective planets are in the system. If JMN is equal to 3 or 5, subroutine READE is used to obtain the position and velocity of the planets. Before this routine is called, the IREQ array is set to the KP array and CENT set to the central planet number. These are arrays used in READE. The KREAD flag is also set. Each read of the ephemeris disk on tape brings eight days of data into arrays in READE. The KREAD flag is set to zero (no read) if the time of the ephemeris call is within the eight day range of data already started in READE, otherwise, the KREAD flag is set to one for a tape or disk read.

SUBROUTINE POST

Calling Sequence: CALL POST

Purpose: POST computes certain parameters for output during midcourse guidance analysis.

Common Blocks Required: CONST, INPUT, INTVAR, MCCOM, PLNET, STATE

Subroutines Required: BELL, BURND, DOT, MVTRN, ORBIT, PLANET, RETRO, SUNMIN, TRMN, VISIB, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	WO	1	INPUT(38)	Initial spacecraft weight (kg)
I	DJO	1	INPUT(46)	Julian date of anchor vector epoch (days)
I	ASPMC	1	INPUT(441)	Specific impulse of the midcourse motor (sec)
I	WRETRO	1	INPUT(443)	Weight of retro fuel (kg)
I	AFUEL	1	INPUT(471)	Attitude fuel/angle factor (kg/rad)
I	WDROP	1	INPUT(473)	Retro drop-weight (kg)
I	JT	1	INPUT(1031)	Target body number
I	MCOUT	1	INPUT(1050)	Extra output key
I	KOUT9	1	INPUT(1058)	Logical unit for scope output.
I	MCUNIT	1	INPUT(1061)	Logical unit for writing out information.
I	IPROB	1	INPUT(1070)	Output probability scale (pct)
I	NORMIN	1	INPUT(1080)	Retro optimization key
I	XMC	6	MCCOM(6)	Pre-ignition midcourse state (km, km/sec)
I	DV	3	MCCOM(12)	Midcourse velocity impulse (km/sec)
I	DVMG	1	MCCOM(15)	Midcourse impulse magnitude(km/sec)

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DEC	1	MCCOM(16)	Declination of DV (deg)
I	RA	1	MCCOM(17)	Right ascension of DV (deg)
I	TMCS	1	MCCOM(18)	Midcourse ignition time (sec past anchor epoch)
I	XSUN	3	MCCOM(21)	Spacecraft -to-Sun unit vector Mean of 1950
I	VRET	1	MCCOM(25)	Retro delta-v (km/sec)
I	EXFUEL	1	MCCOM(26)	Expected second-midcourse fuel (kg)
I	BT	3	MCCOM(27)	Spin-axis unit vector at retro
I	DVT	1	MCCOM(30)	Trim velocity (km/sec)
I	FUELT	1	MCCOM(31)	Trim fuel (kg)
I	XR	6	MCCOM(32)	Pericynthion state (km, km/sec, Lunar Equator)
I	DJDIF	1	MCCOM(39)	Anchor epoch-launch epoch difference (days)
I	SIGOUT	6	MCCOM(40)	Expected end constraint errors from BELL
I	FFIRE	1	MCCOM(46)	Firing true anomaly for retro (rad)
I	WTF	1	MCCOM(47)	Spacecraft weight after midcourse (kg)
I	EXVZ	1	MCCOM(49)	Expected second midcourse velocity (m/sec)
I	IT	1	MCCOM(157)	Midcourse iteration counter from MDCORS
I	KMC	1	MCCOM(159)	Midcourse execution counter from PROTO
I	KT	1	MCCOM(161)	Flight time counter from PROTO
I	ICB	1	MCCOM(165)	Midcourse central body number
O	KEL	10	MCCOM(172)	Elevation indicator array for midcourse.
O	KELR	10	MCCOM(182)	Elevation indicator array for retro
I	X	6	STATE(1)	State at target closest approach (km, km/sec)
I	T	1	STATE(10)	Time at target closest approach (sec past anchor epoch)
I	ATT	3	STATE(11)	Spin-axis attitude before midcourse (unit vector)

Description:

POST is a subroutine in which auxiliary output calculations are performed for midcourse guidance subroutines PROTO and FIXATG. These calculations (in order of appearance) are:

1. Midcourse fuel (kg) *
2. Spin-axis-Sun angle at midcourse (deg)
3. Retro velocity impulse (km/sec)
4. Attitude fuel to midcourse attitude (kg)
5. Minimum spin-axis-Sun angle to midcourse (deg)
6. Array of midcourse visibility for trackers (KEL = $^{EL}/_{10}$ +1)
7. Burn time (sec)
8. Trim fuel (kg) *
9. Time of retro ignition (hours past launch) *
10. Right ascension of the spin axis at retro (deg) *
11. Declination of the spin axis at retro (deg) *
12. Total correction fuel (kg)*
13. Spin-axis-Sun angle at retro (deg)*
14. Trim velocity (m/sec)*
15. Minimum spin-axis-Sun angle to retro attitude (deg)*
16. Array of retro visibility for trackers KELR = $^{EL}/_{10}$ + 1 EL≥ 0
    = 0 EL<0

(* computation depends on NORMIN— if ≤0, retro attitude, trim fuel and firing time are as determined in TARGET, otherwise retro attitude and firing time are optimized in RETRO to minimize trim fuel)

The computed quantities are written on logical unit MCUNIT for later retrieval by PROTO or MCVERF. The logic of this subroutine is straightforward and merits no flowchart.

SUBROUTINE PRINT

Calling Sequence: CALL PRINT

Purpose: The purpose of this routine is to print the contents of various common blocks.

Common Blocks Required: CNTRL, CONST, INPUT, PERT, PLNET, STATE, SAVE

Subroutines Required: ORBIT

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	ELM	6	STATE(14)	Spacecraft's osculating orbital elements
I	GM	12	CONST(5)	Gravitational constants
I	JC	1	CNTRL(7)	Central planet number
I	METH	1	INPUT(1013)	Trajectory propagation indicator.
I	X	6	STATE(1)	Spacecraft's position and velocity vectors

Description:

This subroutine prints the contents of various common blocks on output unit 6. A series of flags are used to determine which common blocks are to be printed. If the input location in the table below is set to one, the designated common block will be printed.

INPUT LOCATION	COMMON BLOCK
1020	STATE
1021	CNTRL
1022	PLNET
1023	INPUT
1024	NOT USED
1025	PERT
1026	SAVE

The orbital elements of the spacecraft are determined from subroutine ORBIT if STATE common is to be output and the trajectory propagation method used is Cowell's.

SUBROUTINE PROTO

Calling Sequence: CALL PROTO

Purpose: PROTO controls the gross midcourse analysis logic and printout

Common Blocks Required: ANKOR, CNTRL, CONST, INPUT, MCCOM, STATE

Subroutines Required: FIXATG, MDCORS, POST

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DJL	1	INPUT(37)	Julian date of launch (days)
I	DJO	1	INPUT(46)	Julian date of anchor epoch (days)
I	RAI	1	INPUT(47)	Right ascension initially (deg)
I	DECI	1	INPUT(48)	Declination initially (deg)
I	HRO	1	INPUT(53)	Hours
I	XMINO	1	INPUT(54)	Minutes GMT of anchor epoch
I	SECO	1	INPUT(55)	Seconds
I	DTFIN	1	INPUT(422)	Desired time of flight (seconds)
I	DELTMC	1	INPUT(434)	Midcourse execution time step (sec)
I	SIGAT	1	INPUT(435)	Expected midcourse pointing error (rad)
I	SIGDV	1	INPUT(436)	Midcourse velocity proportional error (frac)
I	TMC2IN	1	INPUT(440)	Time of possible second midcourse (sec)
I	TMC	1	INPUT(478)	Initial midcourse execution time (sec)
I	JL	1	INPUT(1015)	Body center of anchor vector

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	MCOUT	1	INPUT(1050)	Extra midcourse output key
I	JMC	1	INPUT(1051)	Number of midcourse executions desired
I	MCUNIT	1	INPUT(1061)	Logical unit for midcourse output
I	IPROB	1	INPUT(1070)	Output-scaling probability (%)
I	IBURN	1	INPUT(1071)	Midcourse burn model key
I	KTF	1	INPUT(1077)	Scan selector (-fixed attitude, + flight time)
O	XMC	6	MCCOM(6)	Pre-ignition midcourse state (km, km/sec)
O	DV	3	MCCOM(12)	Midcourse velocity impulse(km/sec)
O	DVMG	1	MCCOM(15)	Midcourse impulse magnitude (m/sec)
O	DEC	1	MCCOM(16)	Declination of DV (deg)
O	RA	1	MCCOM(17)	Right ascension of DV (deg)
O	TMCS	1	MCCOM(18)	Midcourse execution time (sec past a.e.)
O	DVB4	1	MCCOM(24)	Expended midcourse velocity (km/sec)
O	EXFUEL	1	MCCOM(26)	Expected second-midcourse fuel (kg)
O	DVT	1	MCCOM(30)	Trim velocity (km/sec)
O	FUELT	1	MCCOM(31)	Trim fuel (kg)
O	DJDIF	1	MCCOM(39)	Anchor-launch epoch difference (sec)
O	SIGOUT	6	MCCOM(40)	Expected end constraint errors
O	EXV2	1	MCCOM(49)	Expected second-midcourse velocity (m/sec)
O	PSID	10	MCCOM(80)	Desired constraint vector
I/O	PSI	10	MCCOM(100)	Constraint error vector
O	KBURN	1	MCCOM(154)	Midcourse burn-type key
O	KENTRY	1	MCCOM(156)	MDCORS re-entry key to avoid initialization
O	IT	1	MCCOM(157)	MDCORS trial counter

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	KMC	1	MCCOM(159)	Midcourse execution time counter
O	KT	1	MCCOM(161)	Flight time counter
O	KGLAWI	1	MCCOM(164)	Guidance law indicator
O	KEL	10	MCCOM(172)	Tracking visibility indicators for midcourse
O	KELR	10	MCCOM(182)	Tracking visibility indicators for retro

Description:

PROTO operates in one of 3 modes, depending upon the setting of KTF.

KTF = 0 (one-dimensional scan of midcourse execution times)

In this mode, sample midcourse correction maneuvers are calculated at JMC execution times which begin at TMC and proceed at steps of DELTMC. Each maneuver, found by MDCORS and elaborated by POST, corresponds to a particular guidance law and the same set of desired end conditions. Characteristics of the maneuvers and arrival conditions are written on MCUNIT in POST, then read again for printout in PROTO at the conclusion of the scanning process.

KTF > 0 (Two-dimensional scan of midcourse times and flight times)

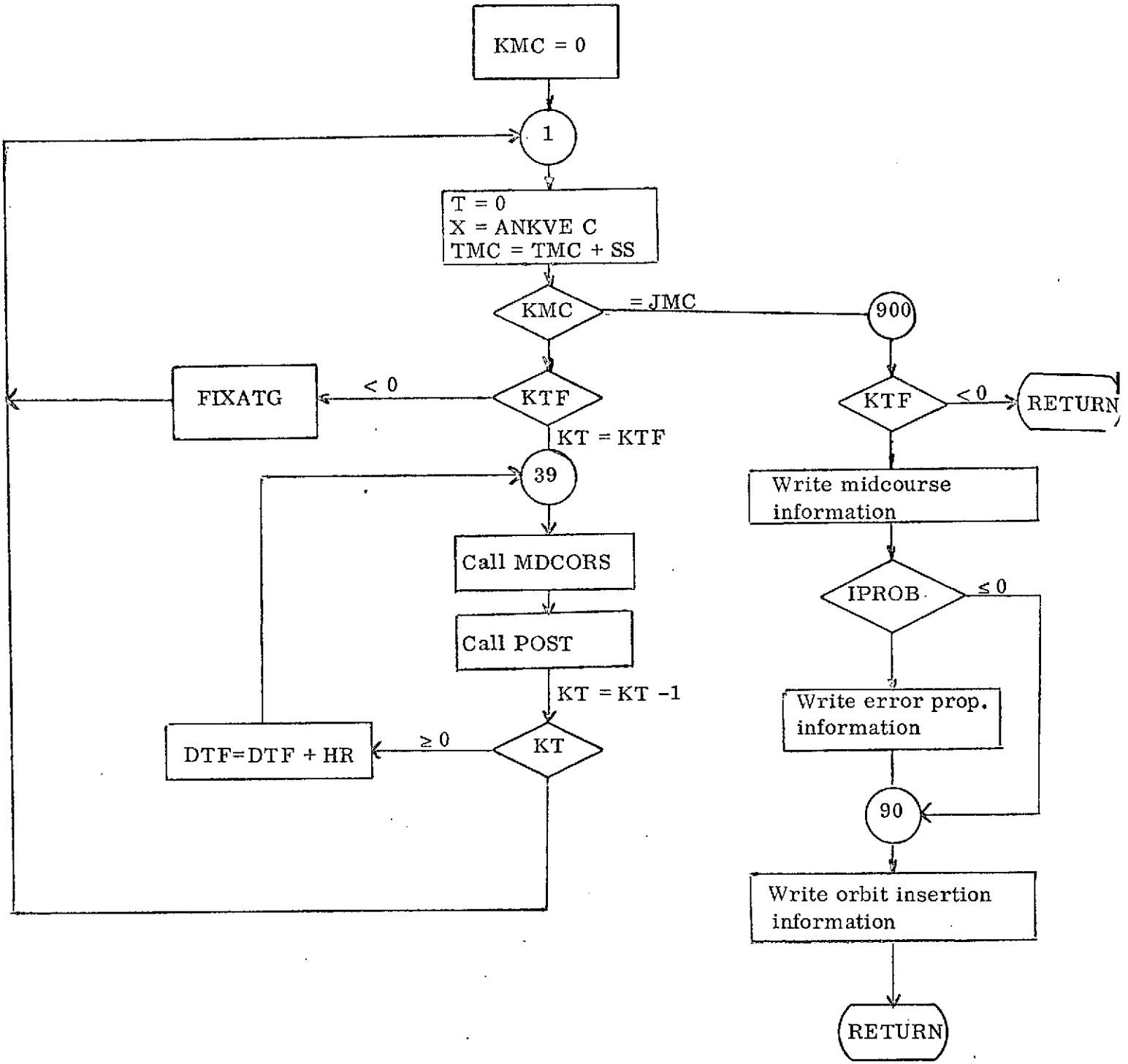
This mode injects a scan of flight times (using the FTA guidance law) into the execution-time scan. It enables generation of all those midcourse solutions which arrive at a specific closest approach distance and inclination, thereby encompassing all the available guidance law options. At each midcourse execution time, flight times are scanned in KTF one-hour steps beginning at DTFIN. At each point of the scan, MDCORS and POST compute the maneuver and arrival characteristics and write them on MCUNIT for later printout by PROTO. By scanning flight times within the execution-time loop, gradients for targeting may be salvaged for use at several flight times and some trajectory computation may be avoided. Gradients are re-generated every fourth hour of the flight time scan.

KTF < 0 (Two-dimensional scan of midcourse times and delta v)

This is the fixed-attitude-guidance mode. At each midcourse execution time, FIXATG

is called. In FIXATG, it is assumed that the spin-axis direction or thrust direction is fixed at RAI and DECI. The impulsive velocity magnitude is varied systematically in FIXATG from zero in -KTF steps of size DINK (location INPUT(479) in km/sec). The post-ignition trajectory is propagated to target closest approach wnere arrival characteristics are computed and printed out. MDCORS is not called in this mode, neither is POST and no summary printout is performed in PROTO.

SUBROUTINE PROTO



SUBROUTINE PUTELS

Calling Sequence: CALL PUTELS (ICOM, MSGERR, $)

Purpose: This subroutine writes a file describing the state for retrieval by the GTDS program.

Common Blocks Required: ELMNT

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ICOM	1	Calling Operand	Initialization flag 1 Initialize output data file 0 add new element to set
I	MSGERR	1	Calling Operand	Error message return number
O	$	-	-	Statement number for error return

Description:

This subroutine is used to output the state for retrieval by the GTDS program. The ICOM flag is used to initialize the output file. When this flag is set to 1, the output file is initialized. No writes are made with this flag setting. When the ICOM flag is zero, writes are made on unit 27.

The data to be written on unit 27 is transferred into the subroutine via the SET array of ELMNT common. The set array is defined as follows:

Location	Definition
1	Date of state in year, month and day written as YYMMDD.
2	Time of state in hours, minutes and seconds written as HHMMSS.SSS.
3-5	Cartesian position vector.
6-8	Cartesian velocity vector
9-14	Keplerian orbital elements. The order is (SMA, ECC, INCLIN, LAN, PA, MA)
15-35	Upper triangle of the state covariance matrix

Location	Definition
36	Start time of fitted data (year, month, day)
37	Start time of fitted data (hour, minute, second)
38	End of fitted data (year, month, day)
39	End of fitted data (hour, minute, second)
40	Root mean square of fit.
41	Satellite identification number
42	Reference coordinate system of state 1 for mean of 1950
43	Central body indicator 1 Earth 2 Moon 3 Sun
44	Element set number

SUBROUTINE QUARTC

Calling Sequence: CALL QUARTC (C, X, N)

Purpose: To find the real roots of a quartic * equation,

$$P = c_1 + \sum_{i=1}^{N} c_{i+1} x^i = 0$$

Common Blocks Used: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	DEFINITION
I	C	5	Coefficients of the polynomial
O	X	4	Solution vector if real solutions exist (Upper-loaded if less than four exist)
I/O	N	1	Input: Order of the polynomial (3 or 4) Output: Real solutions found (0, 1, 2, 3, 4)

* QUARTC may also be used to find the real roots of a cubic equation. The mathematical description which follows applies only to quartic equations.

Description:

The Descartes technique is used to solve the equation,
$AX^4 + BX^3 + CX^2 + DX + E = 0$. First divide through by A ($A \neq 0$) to obtain

$$X^4 + B'X^3 + C'X^2 + D'X + E' = 0 \quad (1)$$

Substituting $X = y + h$, where $h = \frac{B'}{4}$, into (1) obtain

$$y^4 + Py^2 + Qy + R = 0 \quad (2)$$

where

$$P = 6h^2 + 3B'h + C' \tag{3}$$

$$Q = 4h^3 + 3B'h^2 + 2C'h + D' \tag{4}$$

$$R = h^4 + B'h^3 + C'h^2 + D'h + E'. \tag{5}$$

Now (2) may be factored to produce equation (6)

$$(y^2 + \sqrt{R'}\, y + \xi)\ (y^2 - \sqrt{R'} + \beta) = 0 \tag{6}$$

which, in turn, yields four quadratic solutions if R', ξ and β are defined.

$$\xi = \frac{1}{2}\left(P + R' - \frac{Q}{\sqrt{R'}}\right) \tag{7}$$

$$\beta = \frac{1}{2}\left(P + R' + \frac{Q}{\sqrt{R'}}\right) \tag{8}$$

R' is the maximum of the real roots of

$$Z^3 + aZ + b = 0 \tag{9}$$

minus $\frac{2}{3}$ P. The coefficients, a and b, in (9) are given by

$$a = \frac{1}{3}\left[3\,(P^2 - 4R) - 4P^2\right] \tag{10}$$

and

$$b = \frac{1}{27}\left[16\,P^3 - 18P\,(P^2 - 4R) - 27Q^2\right] \tag{11}$$

To find the roots of (9), first compute the quantity

$$\Delta = \frac{a^3}{27} + \frac{b^2}{4}\ . \tag{12}$$

If $\Delta > 0$, the only real root is

$$Z_1 = \sqrt[3]{-\frac{b}{2} + \sqrt{\Delta}} + \sqrt[3]{-\frac{b}{2} - \sqrt{\Delta}} \quad . \tag{13}$$

If $\Delta = 0$, the real roots are

$$Z_1 = 2\sqrt[3]{-\frac{b}{2}}, \quad Z_2 = \sqrt[3]{\frac{b}{2}} \quad \text{and} \quad Z_3 = \sqrt[3]{\frac{b}{2}} \quad , \tag{14}$$

If $\Delta < 0$ the roots are

$$Z_1 = 2\sqrt{-\frac{a}{3}}\cos(\phi/3) \quad ,$$

$$Z_2 = 2\sqrt{-\frac{a}{3}}\cos(\phi/3 + 120^{\circ}), \tag{15}$$

and

$$Z_3 = 2\sqrt{-\frac{a}{3}}\cos(\phi/3 + 240^{\circ})$$

where

$$\cos\phi = -\frac{b}{2\sqrt{-\frac{a^3}{27}}} \tag{16}$$

After computing R', equations (6), (7), and (8) yield y_i, $i = 1, 4$. Then $X_i = y_i + h$ are solutions to the quartic. Only real solutions are considered by this subroutine.

SUBROUTINE QUIKIE

Calling Sequence: CALL QUIKIE (A, SI, CI, Ø0, SUNØ, TF1, TF2)

Purpose: This subroutine calculates approximate shadow times for circular lunar orbits with constant inclinations. The times calculated are shadow-free time and time until next shadow (or time until no shadow, if orbit is presently in shadow)

Common Blocks Required: CONST

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	A	1	CALLING OPERAND	Semi-major axis
I	SI	1	CALLING OPERAND	Sine (inclination)
I	CI	1	CALLING OPERAND	Cosine (inclination)
I	Ø0	1	CALLING OPERAND	Initial longitude of node
I	SUNØ	1	CALLING OPERAND	Initial longitude of Sun
I	GM	12	CONST	Planet gravitational constants
I	PI, PI2	1,1	CONST	π, 2π
O	TF1	1	CALLING OPERAND	Time until next shadow or neg. time until no shadow
O	TF2	1	CALLING OPERAND	Shadow-free time

Description:

This subroutine calculates the times when a given orbit around the Moon will cross the Moon's shadow. The orbit is assumed to have zero eccentricity and constant inclination. Input quantities should be expressed in an ecliptic coordinate system; however, the Moon's equator is probably acceptable for most work.

The output quantity TF2 represents the time between shadows, (i.e., the maximum shadow free time.

Note: For low inclination orbits, part of the orbit is always in shadow. For this case, both TF1 and TF2 are output as 999.9.

Theory:

In an ecliptic coordinate system, let Ψ be the longitude of the Sun, Ω be the longitude of the orbit's ascending node, and γ the angle between them.

Then $\gamma = \Psi - \Omega$, and $\dot{\gamma} = \dot{\Psi} - \dot{\Omega}$, where $\dot{\Psi}$ is the angular velocity of the Sun about the Earth (.998 deg/day), and $\dot{\Omega}$ is the precession rate of the orbit around the planet.

$\left(\dot{\Omega} \approx 1.5\, n c_2 \cos(i)\, RM^2/A^2 + .75\, n'^2\, \mu/n \cos(i)\right)$

n = mean motion of spacecraft

c_2 = lunar oblateness coefficient

i = inclination

RM = Radius of the Moon

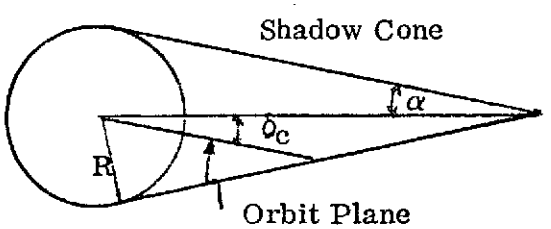
A = orbit radius

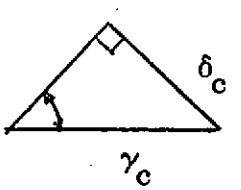
n' = mean motion of Moon

μ = grav. ratio

Since the orbit is circular, the radius A is constant. Thus, the angle from the shadow cone centerline to the edge of the cone at a distance A from the planet is

$\delta_c = \sin^{-1}(RM/A) - \alpha$, where α is the half shadow cone angle.





$$\gamma_c = \sin^{-1}\left(\frac{\sin \delta c}{\sin i}\right) \quad (\text{Note: } |\, \text{sm}\, \delta c| \leq |\, \text{smi}|\,)$$

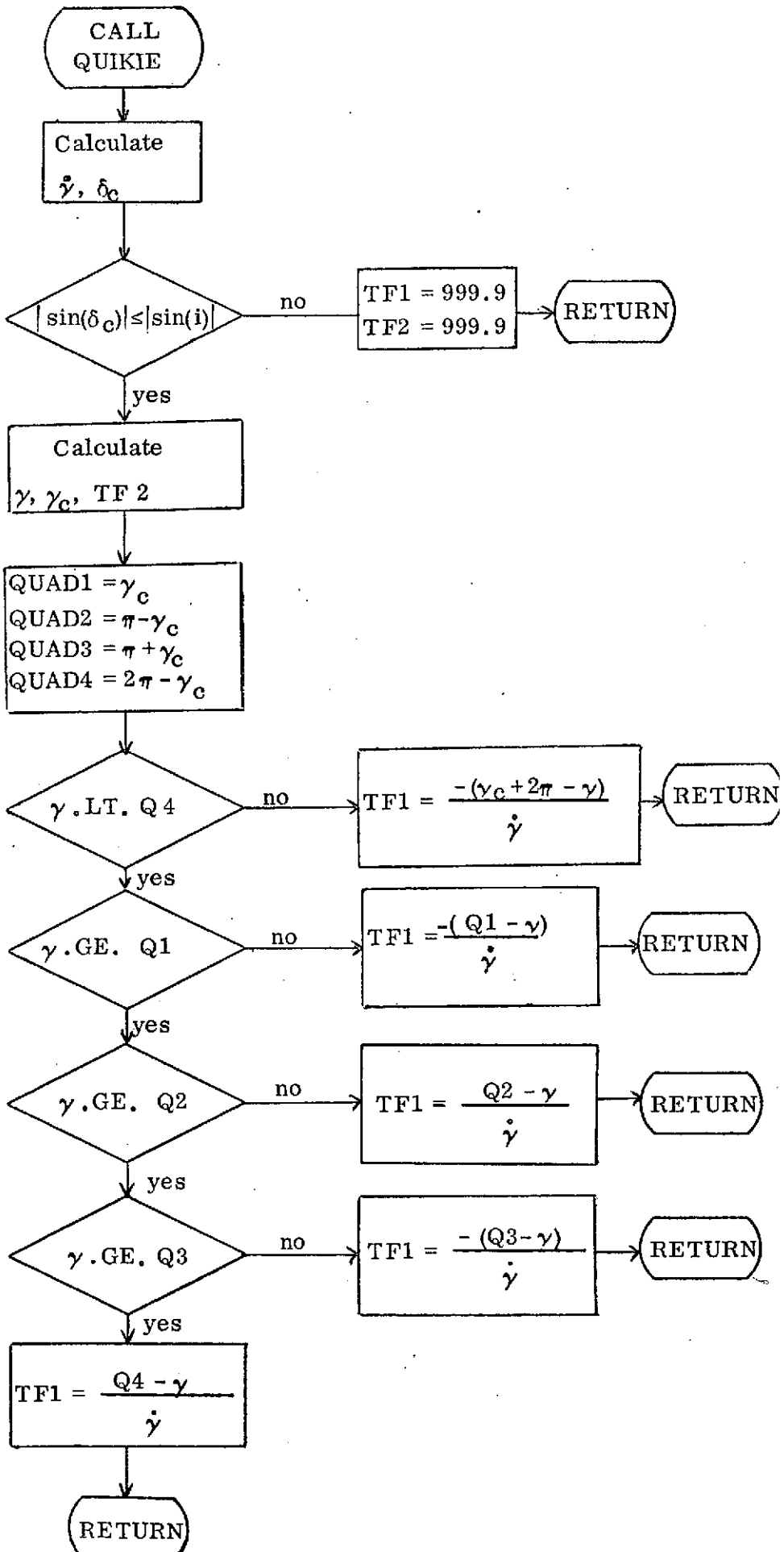
This implies that the orbit is a shadow whenever

$$-\gamma_c \leq \gamma \leq \gamma_c \quad \text{or} \quad \pi - \gamma_c \leq \gamma \leq \pi + \gamma_c \,.$$

Thus the problem is reduced to finding out what quadrant γ is in and what the angle is until the orbit status changes. For instance, if γ is in the third quadrant and the orbit is not in shadow, then $TF1 = \dfrac{2\pi - \gamma c}{\dot{\gamma}}$.

Similarly, if γ is in the second quadrant and the orbit is in shadow, then $TF1 = -(\pi + \gamma_c - \gamma)/\dot{\gamma}$. TF2 is independent on γ and is equal to $\dfrac{(\pi - 2\gamma_c)}{\dot{\gamma}}$.

SUBROUTINE QUIKIE



SUBROUTINE RANDM1

Calling Sequence: CALL RANDM1 (IY, YFL)

Purpose: This subroutine determines a random number uniformly distributed between 0 and 1.

Common Blocks Required: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	IY	1	CALLING ARGUMENT	Random number generator
O	YFL	1	CALLING ARGUMENT	Random number

Description:

The random number is determined from

YFL = IY (1027) (.465661 D-9).

The random number generator is updated on each call as,

IY = IY (1027)

This random number generator is designed for the word length of an IBM 360 series computer.

SUBROUTINE READE

Calling Sequence: CALL READE (JED, TSEC, TERR)

Purpose: This routine takes the data from the ephemeris tape and interpolates for the planet's position and/or velocity at the desired time. It also transfers to the central planet.

Common Blocks Required: CETBL2, CETBL3, CETBL9, PLNET

Subroutines Required: GETTAP

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ICENT	1	CETBL2(2)	Central Planet number
I	IERR	1	CALLING OPERAND	Error flag
I	IREQ	13	CETBL2(3)	Planet's ephemeris flag
I	JED	1	CALLING OPERAND	Reference julian ephemeris
O	NUT	4	PLNET(85)	Nutation output
I	NUTAT	204	CETBL3(830)	Nutation input data from ephemeris tape
O	TABOUT	6, 12	PLNET(1)	Position and velocity vectors of planets with respect to the central planet
I	TAB3	829	CETBL3(1)	Input planet position and velocity vectors from ephemeris tape
I	TSEC	1	CALLING OPERAND	Seconds of ephemeris time past JED

Description:

This subroutine determines the positions and velocities of the planets with respect to the central planet from the JPL ephemeris tape. The times of the positions and velocities to be obtained are brought through the argument list as JED and TSEC. The output positions and velocities are put in TABOUT of PLNET common.

The positions and velocities of the planets are set up in the TAB3 array through a call to GETTAP. This routine searches for the proper block of data from the tape (or disk) and reads into TAB3 the eight-day block of data that encompasses the desired time. READE next interpolates for the positions and velocities at the desired time from the eight-day block of data in TAB3. After the interpolation is complete, READE translates the positions and velocities to the desired central planet.

FUNCTION RETDV

Calling Sequence: DV = RETDV(DVMCM, WTØ, WTF, WTAR)

Purpose: RETDV computes the retro velocity as a function of midcourse velocity.

Common Blocks Required: CØNST, INPUT

Subroutines Called: None

Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	DVMCM	1	Argument List	Midcourse velocity magnitude (km/sec)
I	WTØ	1	Argument List	Spacecraft weight before midcourse (kg)
Ø	WTF	1	Argument List	Spacecraft weight after midcourse (kg)
Ø	WTAR	1	Argument List	Spacecraft weight after retro (kg)
Ø	RETDV	1	Function	Retro velocity (km/sec)
I	G	1	CØNST (45)	Earth's surface gravity (km/sec^2)
I	ASPMC	1	INPUT (441)	Specific impulse of midcourse motor (sec)
I	ASPR	1	INPUT (442)	Specific impulse of retro motor (sec)
I	WRETRØ	1	INPUT (443)	Weight of retro fuel (kg)

Method:

The rocket equation is applied to simulate both the midcourse and retro motor burns. Attitude fuel expenditure is ignored.

$$WTF = WT\emptyset \cdot e^{-[DVMCM/(G \cdot ASPMC)]}$$

$$WTAR = WTF - WRETR\emptyset$$

$$RETDV = -(G \cdot ASPR)\ \ell n\left(\frac{WTAR}{WTF}\right)$$

SUBROUTINE RETRO

Calling Sequence: CALL RETRO (ELM, DVR, DVT, FFIRE, RAO, DECO, X)

Purpose: RETRO finds the retro attitude which results in the minimum trim velocity.

Common Blocks Required: INPUT

Subroutines Required: MINV

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ELM	12	ARGUMENT LIST	Orbital elements of the approach hyperbola
I	DVR	1	ARGUMENT LIST	Retro velocity impulse (km/sec)
O	DVT	1	ARGUMENT LIST	Trim velocity required (km/sec)
O	FFIRE	1	ARGUMENT LIST	Firing true anomaly on the approach hyperbola (rad)
I	RAO	1	ARGUMENT LIST	Initial estimate of retro right ascension (rad)
I	DECO	1	ARGUMENT LIST	Initial estimate of retro declination (rad)
O	X	2	ARGUMENT LIST	Right ascension(rad) & declination of the solution retro orientation.
I	KROUT	1	INPUT(1072)	Extra output key (print if KROUT = 1)

Method:

A steepest descent hunting procedure is used to find the retro orientation resulting in the minimum trim velocity. The control vector, X, is defined by

$$X = \begin{bmatrix} \text{right ascension of the spin-axis} \\ \text{declination of the spin-axis} \end{bmatrix}$$

(The velocity impulse imparted by the retro burn is anti-parallel to spin-axis). The trim velocity, v_t, is defined (through formulae in MINV and TRIM) as a function of X and the firing true anomaly, θ, on the approach hyperbola.

$$v_t = f(X, \theta)$$

RETRO obtains $v_t^* = f(X, \theta^*)$ by calling MINV. θ^* is the value of θ which minimizes v_t for a given X and is, generally, also an implicit function of X. v_t^* is the θ-minimized trim velocity. The objective of the logic in RETRO is to find X^*, the control (specifying spin-axis in orientation) for which v_t is minimized.

$$\min v_t = f(X^*, \theta^*)$$

We will now denote v_t^* by the typographically-simpler v(X). Approximations to the partial derivatives,

$$P(X) = \frac{\partial v}{\partial X}$$

are obtained by the secant method evaluated at X. The first order change, δv, in v(X) due to a change, δX, in X is

$$\delta v = P \delta X$$

which we cannot solve directly for δX given δv. We choose, instead, to minimize a function , S^2, of δX subject to the constraint $P \delta X = \delta v$. Let $S^2 = \delta X^t G \delta X$, where G is a positive-definite weighting matrix. We adjoin the constraint to S^2 and differentiate with respect to δX to solve for δX as a function of the undetermined multiplier, λ.

$$\frac{\partial}{\partial \delta X} \left[\delta X^t G \delta X - 2\lambda (P\delta X - \delta v) \right] = 2\delta X^t G - 2\lambda P = 0$$

$$\delta X = G^{-t} P^t \lambda$$

$$P\delta X = P G^{-t} P^t \lambda = \delta v$$

$$\lambda = \frac{\delta v}{P G^{-t} P^t}$$

$$\delta X = \frac{G^{-t} P^t \delta v}{P G^{-t} P^t}$$

This is the steepest descent formulation for δX. RETRO's weighting matrix is defined by

$$G^{-t} = \begin{bmatrix} x_1^2 & 0 \\ 0 & x_2^2 \end{bmatrix}.$$

The hunting procedure used in RETRO to find X^* is to ask for an improvement, δv, (negative) at the k-th iteration, where X is X_r. This predicts a control change δX which leads to X_{r+1}.

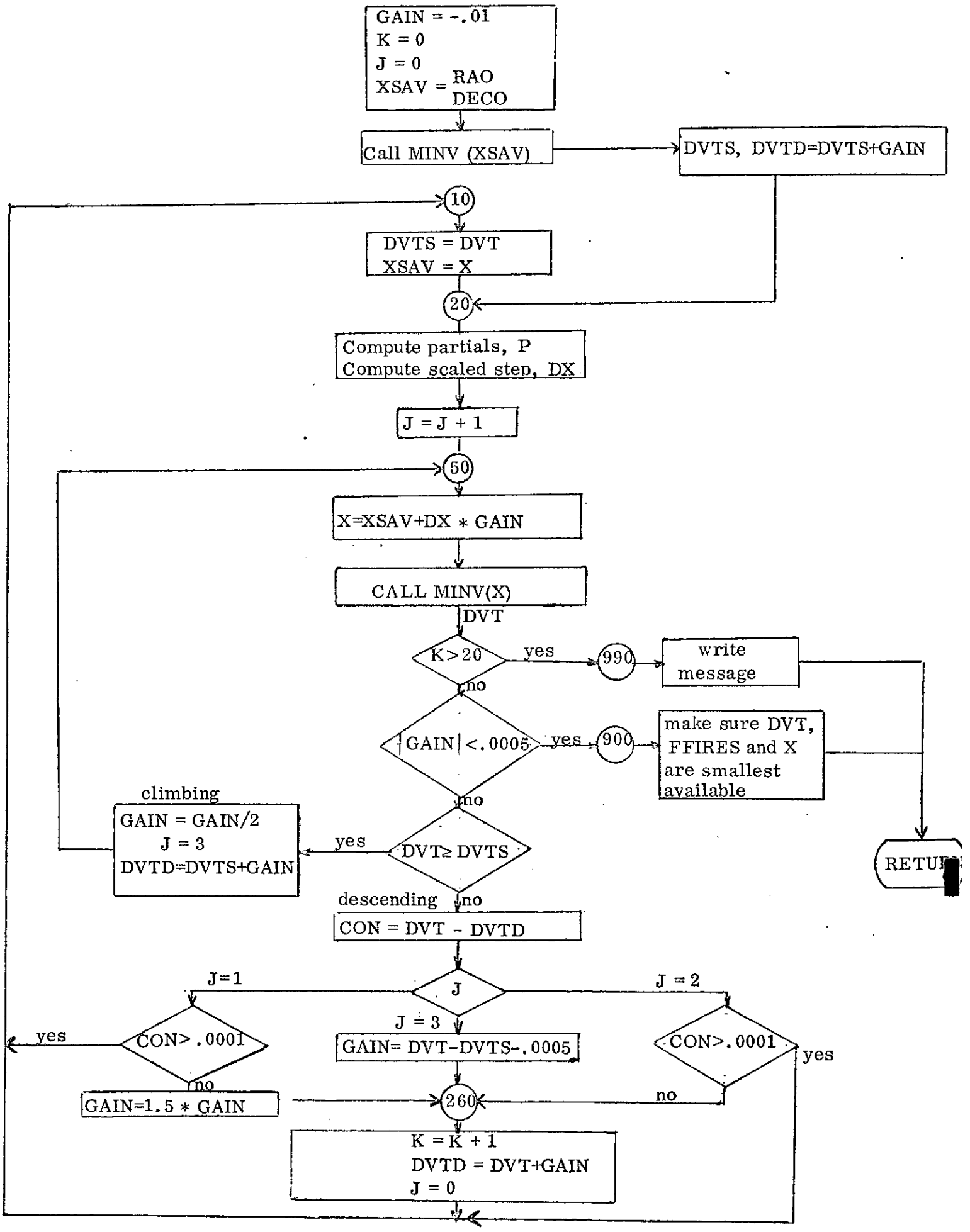
$$X_{r+1} = X_r + \delta X$$

If $$\Delta v = v(X_{r+1}) - v(X_r)$$

is positive, δv is halved and δX, X_{r+1} and $v(X_{r+1})$ are re-calculated (without re-calculating P) until Δv is negative. If the magnitude of δv is less than the specified tolerance, $X^* = \min(X_r, X_{r+1})$ and the iteration is complete.

If $\Delta v < \delta v$ on the first trial at any X_r, δv is set to $1.5\delta v$ for the next step from X_{r+1}. If Δv is negative on the first trial but $\Delta v > \delta v$, another step is computed using δv but starting at X_{r+1}. If $\Delta v_2 = v(X_{r+2}) - v(X_r)$ is less than δv, the iteration proceeds from X_{r+2}. If $\delta v < \Delta v_2 < 0$, then δv is set to Δv_2 - tolerance. This process is considered convergent when the magnitude of δv is less than the pre-set tolerance.

SUBROUTINE RETRO



SUBROUTINE RKSEVN

Calling Sequence: CALL RKSEVN (N, HO, XO)

Purpose: This subroutine integrates a set of simultaneous differential equations using a seventh-order ten-cycle Runge-Kutta scheme.

Common Blocks Required: CNTRL, INPUT, INTVAR

Subrountines Called: EQNS

Input / Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	ERRC	1	INPUT(1)	Error control
I/Ø	H	1	INTVAR(14)	Initial compute interval and next compute interval
Ø	HO	1	Calling Operand	Actual compute interval used
Ø	KHALT	1	CNTRL(6)	Error stop flag
I	N	1	Calling Operand	Number of differential equations
I/Ø	RATES	6	INTVAR(8)	First derivative of dependent variables.
Ø	X	1	INTVAR(1)	Independent variable
I	XO	1	Calling Operand	Initial independent variable
I/Ø	Y	6	INTVAR(2)	Dependent variables

Theory:

The integration scheme shown below was developed by D. Sarafyan, Reference 1. In his scheme,a set of simultaneous differential equations are numerically integrated using a seventh-order ten-cycle Runge-Kutta method. In the following equations f (x, y) refers to the value of the derivatives of the function at the independent variable X and dependent variable Y. This value is brought into the subroutine via RATES in INTVAR common. It is calculated in Subroutine EQNS.

The value of the independent variables, Y, at the end of the step is obtained from:

$$Y(X_0 + h) = Y_0 + \frac{1}{840}\left(41(K_0 + K_9) + 216(K_4 + K_8) + 27(K_5 + K_7) + 272\,K_6\right) \qquad (1)$$

where

$$K_0 = hf\,(X_0,\ Y_0)$$

$$K_1 = hf\,(X_0 + \tfrac{1}{3}h,\ Y_0 + \tfrac{1}{3}K_0)$$

$$K_2 = hf\left(X_0 + \tfrac{1}{2}h,\ Y_0 + \tfrac{1}{8}\left[K_0 + 3K_1\right]\right)$$

$$K_3 = hf\left(X_0 + h,\ Y_0 + \tfrac{1}{2}\left[K_0 - 3K_1 + 4K_2\right]\right)$$

$$K_4 = hf\left(X_0 + \tfrac{1}{6}h,\ Y_0 + \frac{1}{648}\left[83K_0 + 32K_2 - 7K_3\right]\right)$$

$$K_5 = hf\left(X_0 + \tfrac{1}{3}h,\ Y_0 + \frac{1}{54}\left[-3K_0 - 4K_2 + K_3 + 24K_4\right]\right) \qquad (2)$$

$$K_6 = hf\left(X_0 + \tfrac{1}{2}h,\ Y_0 + \frac{1}{5088}\left[-290K_0 - 524K_2 + 145K_3 + 1908K_4 + 1305K_5\right]\right)$$

$$K_7 = hf\left(X_0 + \tfrac{2}{3}h,\ Y_0 + \frac{1}{1431}\left[292K_0 + 108K_2 + 13K_3 - 318K_4 + 753K_5 + 106K_6\right]\right)$$

$$K_8 = hf\left(X_0 + \frac{5}{6}h,\ Y_0 + \frac{1}{68688}\left[14042K_0 + 11012K_2 - 4477K_3 + 5724K_4 - 6903K_5 + 6360K_6 + 31482K_7\right]\right)$$

$$K_9 = hf\left(X_0 + h,\ Y_0 + \frac{1}{4346}\left[-2049K_0 - 1836K_2 + 839K_3 + 5724K_4 - 4692K_5 + 12084K_6 - 9540K_7 + 3816K_8\right]\right)$$

Description:

The value of f in the above equations is determined by repeated calls to EQNS with the indicated independent variable set in X of INTVAR common and the dependent variables set in Y of INTVAR common. EQNS calculates the derivatives of the dependent variables and puts the answer in RATES of INTVAR common. The values of f are used to calculate K_0 through K_9. These K's are used to determine variables at the end of the computing step from the first equation. If a fixed computing interval is used (ERRC=0), the subroutine terminates.

When the automatic compute interval option is used (ERRC $\neq$ 0) the subroutine calculates the fourth-order solution at the end of the step from

$$Y_4\ (X_0\ th) = Y_0 + \frac{1}{6}(K_0 + 4K_2 + K_3) \tag{3}$$

where

K_0, K_2, K_3 are the same as defined above.

Next the fourth-order solution is compared to the seventh-order solution. The largest relative difference between the two is used to determine the compute interval from

$$H = H\ (\ ERRC/ERRELS\ )^{1/5} \tag{4}$$

where ERRELS is the largest relative difference. H is the compute interval to be used in the next step; however, if the relative error is more than 4 ERRC, then the error in the current solution is considered too large. In this case, the whole process is repeated again with the compute interval just determined. A limit of 10 repeats is allowed. If the limit is reached, the error halt flag is set and the

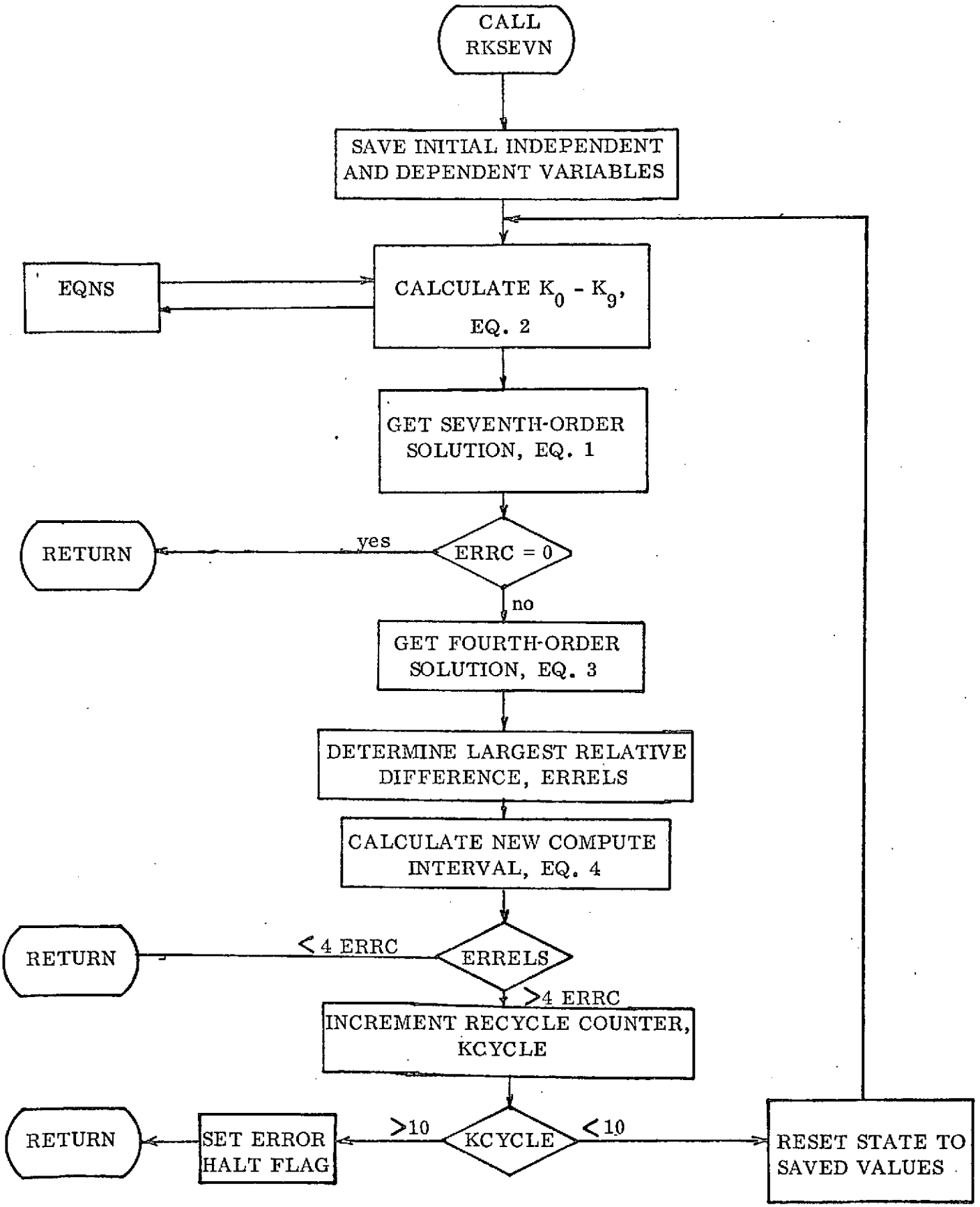
subroutine terminates. The independent variable is not incremented inside this routine. It should be done after the call to RKSEVN as shown below.

$$T = T + HO \tag{5}$$

where T is the independent variable.

Reference

1. Sarofyan, D.; "Seventh-Order Ten-Stage Runge-Kutta Formulas," Technical Report No. 38, Louisiana State University, Department of Mathematics, January 1970.


CALL RKSEVN
SAVE INITIAL INDEPENDENT AND DEPENDENT VARIABLES
EQNS
CALCULATE K_0 - K_9, EQ. 2
GET SEVENTH-ORDER SOLUTION, EQ. 1
RETURN
yes
ERRC = 0
no
GET FOURTH-ORDER SOLUTION, EQ. 3
DETERMINE LARGEST RELATIVE DIFFERENCE, ERRELS
CALCULATE NEW COMPUTE INTERVAL, EQ. 4
RETURN
< 4 ERRC
ERRELS
> 4 ERRC
INCREMENT RECYCLE COUNTER, KCYCLE
RETURN
SET ERROR HALT FLAG
> 10
KCYCLE
< 10
RESET STATE TO SAVED VALUES

SUBROUTINE ROTAIT

Calling Sequence: CALL ROTAIT (X, Y, S, C, U, V)

Purpose: To rotate two orthonormal vectors in their plane. The subroutine computes U=CX+SY, V=-SX+CY. If C=cosA, and S=sinA, U and V are obtained by rotating X and Y through the angle A in the sense X into Y.

Common Blocks used: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	DEFINITION
I	X	3	Orthonormal input vectors
	Y	3	Orthonormal input vectors
I	S	1	Sine of rotation angle
I	C	1	Cosine of rotation angle
O	U	3	Rotated vectors output
O	V	3	Rotated vectors output

ROTAIT is coded in such a way that U and V may share the same storage as X and Y.

SUBROUTINE ROTATE

Calling Sequence: CALL ROTATE (M, A, B, C)
CALL MVTRN (A, B, C, M, N)

Purpose: To form the matrix product $C = AB$ or $C = A^T B$ where A is a 3x3 matrix and B and C are 3x1 matrices (3xN in MVTRN).

Common Blocks Required: None

Subroutines Called: None

Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	A	9	Call List	Matrix multiplier
I	B	3,N	Call List	Matrix multiplier
Ø	C	3,N	Call List	Product matrix
I	M	1	Call List	Indicator: $C = AB$ if $M = 1$; $C = A^T B$ otherwise
I	N	1	Call List	Number of columns of B and C

SUBROUTINE SADOUT

Calling Sequence: CALL SADOUT

Purpose: SADOUT outputs the times of umbral, penumbral, and occultation times.

Common Blocks Required: INPUT, SHAD, STATE

Subroutines Required: CALEND

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DJO	1	INPUT(46)	Modified julian date of state epoch
I	DJ1	1	INPUT(37)	Modified julian date of liftoff
I	JL	1	INPUT(1015)	Launch planet number
I	JT	1	INPUT(1031)	Target planet number
I	KPLOT	1	INPUT(1093)	Plotting flag
I	T	1	STATE(10)	Current time
I	TSX	10	SHAD(19)	Table of umbral, penumbral, and occultation entrance and exit times

Description:

The shadow times to be output are stored in the TSX array. A test is made on the smallest time in the TSX array. If this time is greater than the current time, no output is presented. The TSX array is next ordered so that it is monotonically increasing. The KK array is included to keep track of the times in the ordered TSX array. Subroutine CALEND is used to determine the calendar date corresponding to the times in the TSX array. The dates are output on unit 6 if the time is less than T. If the KPLOT flag is set to 20, information is written on unit 20 for later use in a plotting program.

SUBROUTINE SENSO

Calling Sequence: CALL SENSØ

Purpose: SENSØ computes the gradient of end constraints with respect to midcourse velocity. It can also be used to provide end state and constraint errors as a function of midcourse velocity.

Common Blocks Required: INPUT, MCCØM

Subroutines Called: MCBURN, FØWARD, TARGET

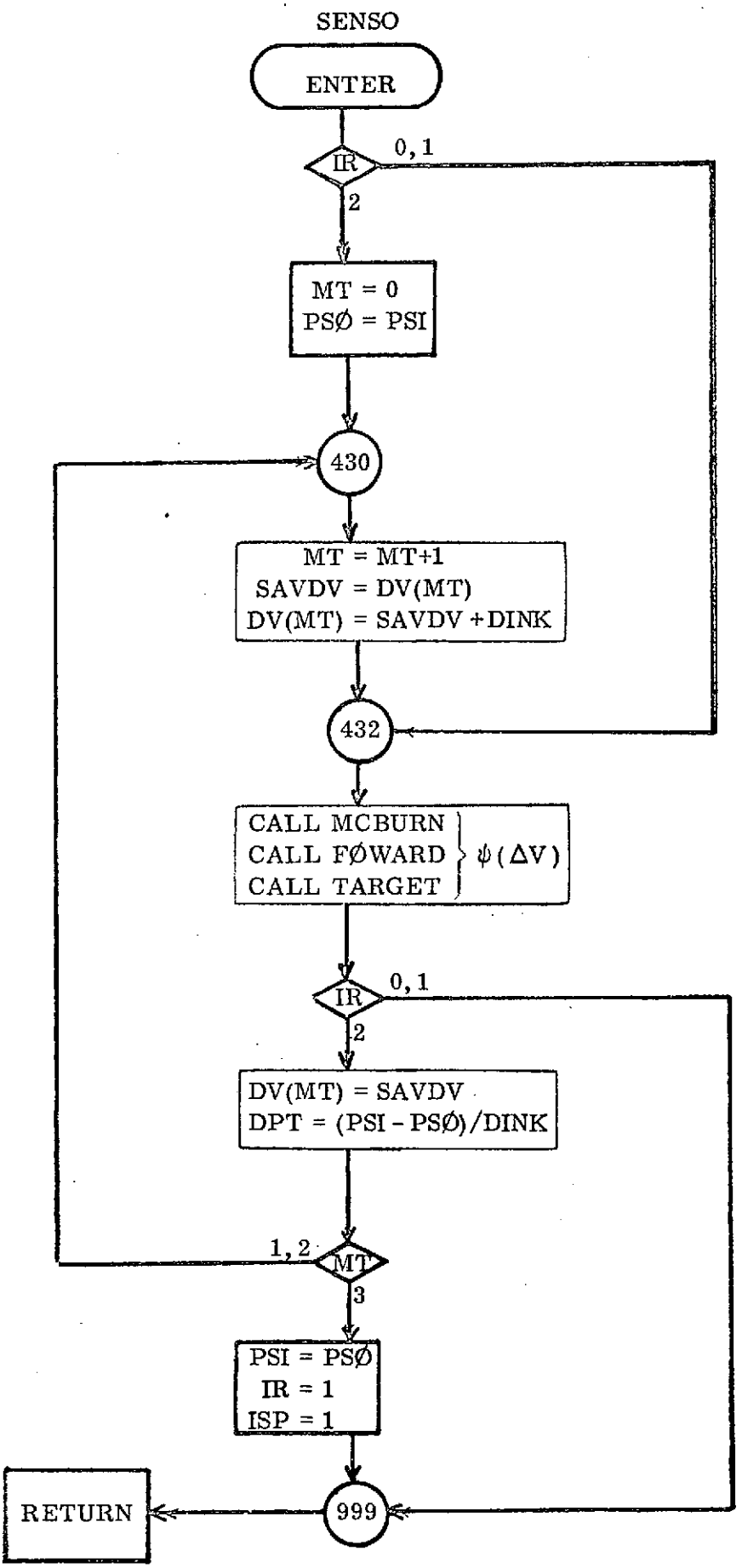
Input/Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DESCRIPTION
I	DINK	1	INPUT (479)	Midcourse velocity impulse for partials (km/sec)
Ø	KRASH	1	INPUT (1032)	Trajectory stop key for FØWARD
I	DV	3	MCCØM (12)	Midcourse correction impulse (km/sec)
Ø	DPT	3,10	MCCØM (50)	Gradient ($\partial\psi/\partial\Delta V$)
I/Ø	PSI	10	MCCØM (100)	End constraint error vector (see TARGET)
I/Ø	IR	1	MCCØM (158)	Gradient-or-not logic key (≥ 2 for gradient)
Ø	ISP	1	MCCØM (166)	Gradient-was-generated key (set 1)

Method:

The secant method of computing approximate partial derivatives is used in generating the gradient of constraint errors with respect to control variations. The control vector, DV, is the impulsive midcourse correction velocity vector. This vector is fed to MCBURN, which outputs the post-burn state. This state is fed to FØWARD, which outputs the state at target closest approach. TARGET takes this end state and generates the constraint error vector, PSI. The transpose of the gradient is stored in DPT.

$$DPT = \left(\frac{\partial PSI}{\partial DV}\right)^{T}$$


SENSO
ENTER
IR
0, 1
2
MT = 0
PSØ = PSI
430
MT = MT+1
SAVDV = DV(MT)
DV(MT) = SAVDV +DINK
432
CALL MCBURN
CALL FØWARD
CALL TARGET
ψ(ΔV)
IR
0, 1
2
DV(MT) = SAVDV
DPT = (PSI - PSØ)/DINK
1, 2
MT
3
PSI = PSØ
IR = 1
ISP = 1
RETURN
999

SUBROUTINE SETUP2

Calling Sequence:	CALL SETUP2
Purpose:	This subroutine initializes flags and constants before any of the program options are initiated.
Common Blocks Required:	ANKOR, CETBL2, CETBL3, CNTRL, CONST, ELMNT, INPUT, INTVAR, MCCOM, MOON, OBSIT, PERT, PLNET, SAVE, STATE
Subroutines Required:	CLOSE, DATE, DVMAG, FIND, M50EPM, M50LEQ, M50MDT, MVTRN, NUTAIT, NUTATE, OBLTY, OBSET, ORBIT, PLANET, TRMN

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	ATT	3	STATE(11)	Attitude unit vector in mean of 1950.
I	AU	1	CONST(4)	Astronomical unit
O	COV	3,3	INPUT(56)	State estimation covariance matrix.
I	DAYL	1	INPUT(21)	Launch day
I	DAYO	1	INPUT(51)	State epoch day
I	DEC	1	INPUT(48)	Initial declination of attitude
O	DJL	1	INPUT(37)	Modified julian launch date
O	DJO	1	INPUT(46)	Modified julian date of state epoch.
O	DX	3	STATE(4)	Velocity
O	EJO	1	STATE(26)	Ephemeris date of state epoch
O	ELM	6	STATE(14)	Orbital elements of initial state
I/O	ETC	1	INPUT (39)	Ephemeris time correction
I	GM	12	CONST(5)	Gravitational constants of the planets
I	HRL	1	INPUT(23)	Hour of launch epoch
I	HRO	1	INPUT(53)	Hour of state epoch
I	IDATT	1	INPUT(1087)	Element set number of attitude desired from element set.
I	IDSAT	1	INPUT(1089)	Satellite identification number
I	IFIND	1	INPUT(1076)	Element set number of state desired from GTDS file
O	JC	1	CNTRL(07)	Central planet number
I	JL	1	INPUT(1015)	Launch planet number
I	JMN	1	INPUT(1017)	Planetary ephemeris flag
I	JT	1	INPUT(1031)	Target planet number

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	KINPT	1	INPUT(1019)	Input state coordinate system flag
I	KOBL	1	INPUT(1029)	Lunar oblateness flag
I	KP	12	INPUT(1001)	Planets in system to be integrated
I	MODLEN	1	INPUT(1035)	Lunar field model number
I	NATUNT	1	INPUT(1088)	I/O unit number of attitude file
I	OBSLAT	10	INPUT(480)	Observation sites latitudes
I	OBSLON	10	INPUT(410)	Observation sites longitudes
I	PCON	1	INPUT(5)	Position unit conversion factor
I	PI2	1	CONST(3)	Twice pi.
I	RA	1	INPUT(47)	Initial right ascension of attitude
I	RAD	1	CONST(1)	Radian to degrees conversion factor.
I	RE	12	CONST(17)	Mean radius of the planets
I	RINIT	1	INPUT(444)	Desired orbit radius at target planet
I	SECL	1	INPUT(25)	Seconds of launch epoch
I	SECO	1	INPUT(55)	Seconds of state epoch
O	SOL	1	STATE(36)	Solar pressure constant
I	SOLARA	1	INPUT(433)	Spacecraft area
I	SPRESS	1	INPUT(197)	Solar pressure at 1 au
I/O	T	1	STATE(10)	Initial time since state epoch.
O	TCA	1	STATE(29)	Time of closest approach
I/O	TIG	6	INPUT(380)	Ignition and burnout times of the engines.
I	TMM	1	INPUT(196)	Epoch of lunar elements
O	UJT	1	STATE(32)	Current modified julian date
I	VCON	1	INPUT(6)	Velocity conversion factor

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	VINIT	1	INPUT(445)	Circular velocity of desired target planet's final orbit
I	WO	1	INPUT(38)	Initial weight
I	WP	12	CONST(29)	Rotation rates of the planets
O	WT	1	STATE(34)	Current weight
O	WTI	1	STATE(35)	Weight after engine burn
O	X	3	STATE(1)	Position
I	XMINL	1	INPUT(24)	Minutes of launch epoch
I	XMINO	1	INPUT(54)	Minutes of state epoch
I	XMONL	1	INPUT(20)	Month of launch epoch
I	XMONO	1	INPUT(50)	Month of state epoch
I	XMOON	6	INPUT(190)	Initial position of moon
I	YRL	1	INPUT(22)	Year of launch epoch
I	YRO	1	INPUT(52)	Month of launch epoch

Description:

The function of this subroutine is to initialize constants and flags and perform coordinate rotations when necessary. This subroutine does not have a complicated logic flow or theory. Thus, this description will consist of a chronological description of the functions performed.

1. Initialize Encke rectification factor.
2. Set up initial state, epoch, and covariance matrix from a call to FIND, if IFIND is non-zero.
3. Determine the modified julian date of the state epoch and the ephemeris time correction. If the ephemeris time correction is not input, it is calculated from ETC = DJO + (38.66 + .0025921 (DJO -40000)) ,
   where ETC is the ephemeris time correction in seconds, and
   DJO is the modified julian date of the state epoch.
4. If the liftoff epoch is not input, set it equal to the state epoch.
5. Set up the solar pressure constant as follows:
   $SOL = 10^{-8}$ (SOLARA) (SPRESS) AU^2
   where SOL is the solar pressure constant,
   SOLARA is the spacecraft area in cm^2,
   SPRESS is the solara pressure at IAU in dynes/cm^2, and
   AU is the astronomical unit.
6. Set the central planet equal to the launch planet.
7. If the state is initialized through subroutine FIND (IFIND = 0) skip to 10, otherwise calculate the position and velocity vectors if orbital elements are input.
8. Convert input units to program units (KM and KM/SEC).
9. Rotate from input coordinate system to Earth mean equator and equinox of 1950.
10. Initialize weights in STATE common for use in engine burns.
11. Set up ANKOR common and determine the circular velocity of the desired final orbit.
12. If IDATT is non-zero, determine the initial right ascension and declination of the attitude vector from a read from unit number NATUNT.

13. Initialize ATT, a unit vector in the direction of the attitude, from the initial right ascension and declination.

14. If the Moon is to be simulated by osculating elements, determine those elements from XMOON using subroutine ORBIT and set up MOON common.

15. If the tape or disk ephemeris is to be used, perform initial reads.

16. Set up the DST array for use in subroutine CLOSE to determine the central planet.

17. Set up the observation site common. The observation site common consists of the following information:

    a. Vector from the center of the Earth to the site in Earth-fixed coordinates (XOBS).

    b. The velocity of the site (DOBS).

    c. Rotation matrix from the Earth equator and Greenwich to a site local coordinate system (OBSROT).

SUBROUTINE SHADOW

Calling Sequence: CALL SHADOW

Purpose: This routine calculates the times of umbral and penumbral passage with respect to the launch and target planets. It also calculates the times that the spacecraft is occulted by the target planet.

Common Blocks Required: CNTRL, CONST, INPUT, INTVAR, PLNET, SHAD, STATE

Subroutines Required: DVMAG, INTERP, PLANET

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I/O	DSAD	3,5	SHAD(1)	Array of back distances to the shadow cones
I	ELM	6	STATE(14)	Spacecraft's osculating orbital elements
I	JC	1	CNTRL(7)	Central planet number
I	JL	1	INPUT(1015)	Launch planet number
I	KFIRST	1	CNTRL(12)	First pass flag
I	METH	1	INPUT(1013)	Trajectory propagation indicator
I	T	1	STATE(10)	Current time
I	TSAD	1	SHAD (16)	Times of back shadow distances
O	TSX	1	SHAD(19)	Array of shadow passage times
I	X	6	STATE(1)	Current position and velocity vectors

Theory:

The distance to the shadow cone, D, can be determined from trigometric relationships by examining Figure 1.

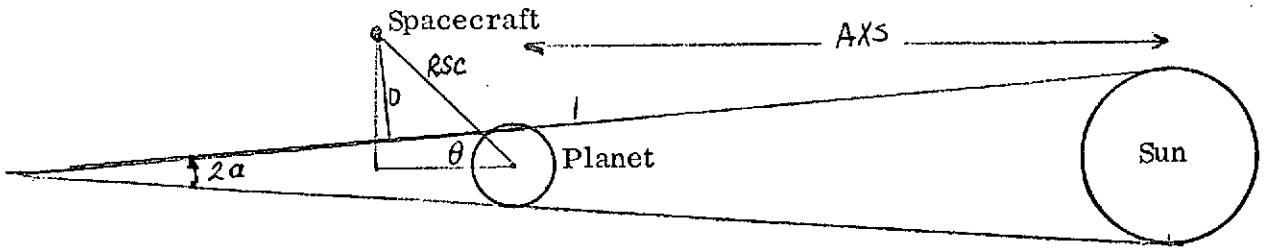


Figure 1

$$D = \left[RSC \sin\theta + (RSC \cos\theta - RE \sin\alpha) \tan\alpha - RE\right] / \cos\alpha \qquad (1)$$

where α is the half cone angle obtained from

$$\sin\alpha = (RE_{sun} - RE) / AXS$$

θ is the angle between the vector from the planet to the spacecraft and the Sun to the planet.

RE is the radius of the planet

The same type of cone is employed for occultation except that the Sun is replaced by the launch planet and the planet is the target planet.

The distance to the penumbral cone is obtained in a similar manner, (see Figure 2)

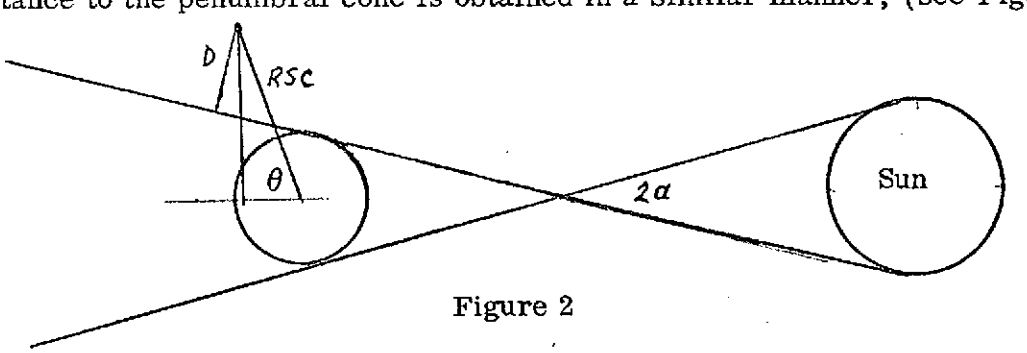


Figure 2

$$D = \left[RSC \sin\theta - (RSC \cos\theta - RE \tan\alpha) - RE\right] / \cos\alpha \qquad (2)$$

where in this case, the half cone angle, α, is obtained from

$$\alpha = (RE_{sun} + RE) / AXS.$$

Description:

The times of umbral and prenumbral passage are determined in this subroutine. There are a total of ten times which may be determined in this routine. These times consist of entrance and exit of the following cones:

KJ =
1. Launch planet umbra
2. Launch planet penumbra
3. Target planet umbra
4. Target planet penumbra
5. Occultation by target planet

There is a loop around the logic that determines the times. The internal flag, KJ, is the index of this loop and determines which shadow time is being calculated. (See above Table).

After the internal constants and flags are set up for loops and other purposes, the subroutine determines the distance from the shadow cone using equations (1) or (2). There are two criteria used to determine whether a shadow cone could have been passed on the last compute step.

1. Present position inside shadow cone while last position outside cone, or vice versa.
2. Spacecraft passed a minimum to the cone.

If neither of these criteria are satisfied, the spacecraft could not have flown through the shadow cone on the last step. At this point, flow is transferred to the location where the flags are set to loop on the next shadow time. If one of the criteria is satisfied, the spacecraft could have flown through the cone and the logic flow is transferred to the location where the time of shadow crossing is determined. The time of crossing is determined through a Newton-Raphson type iteration with subroutine INTERP. INTERP is used to determine the spacecraft state at the same time (T). This state is used in equations (1) or (2) to determine the distance to the shadow cone. T is adjusted by the Newton-Raphson scheme to drive the distance from the shadow cone to zero. The N-R

iteration scheme, as implemented, determines the time of shadow crossing from

$$T = T + DELT$$

$$DELT = \frac{dt}{dD} \; D \tag{3}$$

where D is the distance to the shadow cone and

$\frac{dt}{d\,D}$ is the inverse of the derivative of the distance from the shadow cone with respect to time.

The derivative of the distance to the shadow cone with respect to time is determined analytically from the spacecraft's velocity vector and position with respect to the shadow cone. The derivative is equal to the component of the spacecraft's velocity along the direction normal to the shadow cone. This is represented vectorially by

$$\frac{d\,D}{dt} = \bar{V} \cdot \hat{z} \tag{4}$$

where $\bar{V}$ is the spacecraft's velocity and

$\hat{z}$ is the unit vector normal to the shadow cone (see Figure below)

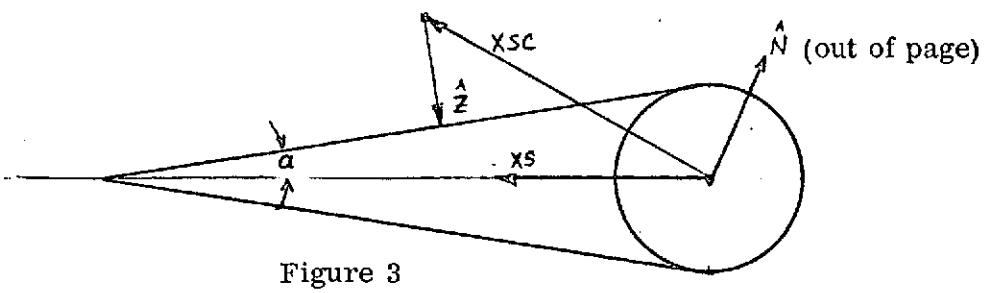


Figure 3

Define the X', Y', Z' coordinate system with X' along the XS vector, Z' along the $\bar{N}$ vector, and Y' defining a right-handed system, Figure 3. $\bar{N}$ is obtained from

$$\hat{N} = (\overline{XSC} \times \overline{XS}) \,/\, |\overline{XSC}|\;|\overline{XS}| \; . \tag{5}$$

Then, the vector normal to the shadow cone, $\hat{z}$, in the primed system is expressed

as $$\begin{Bmatrix} \hat{x} \\ \hat{z} \end{Bmatrix} = \begin{Bmatrix} -\sin\alpha \\ \cos\alpha \\ 0 \end{Bmatrix} \quad (6)$$

$\hat{z}$ in the original coordinate system is $$\begin{Bmatrix} \hat{X} \end{Bmatrix} = [B] \begin{Bmatrix} z \end{Bmatrix} \quad (7)$$

where $[B]$ is the rotation matrix from the primed system to the original system.

The first column of the B matrix is a unit vector along $\bar{XS}$. The third column is the vector $\hat{N}$ while the middle column is a vector obtained by the cross-product of $\hat{N}$ and a unit vector along $\bar{XS}$. Equations (4) to (6) define the derivative of the distance shadow cone with respect to time.

There is a limit to the time step allowed on each iteration. There are usually no convergence problems when one point is inside the shadow cone and the other outside. However, there is some convergence problem when a minimum to the shadow cone is passed. The problem is illustrated in Figure 4.

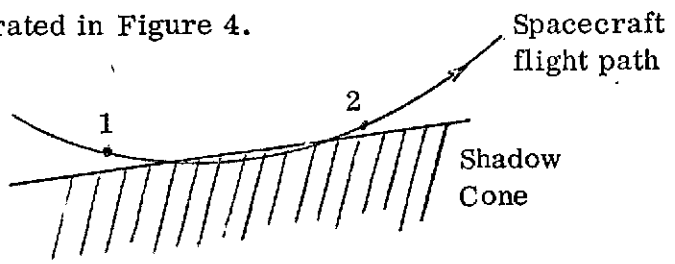
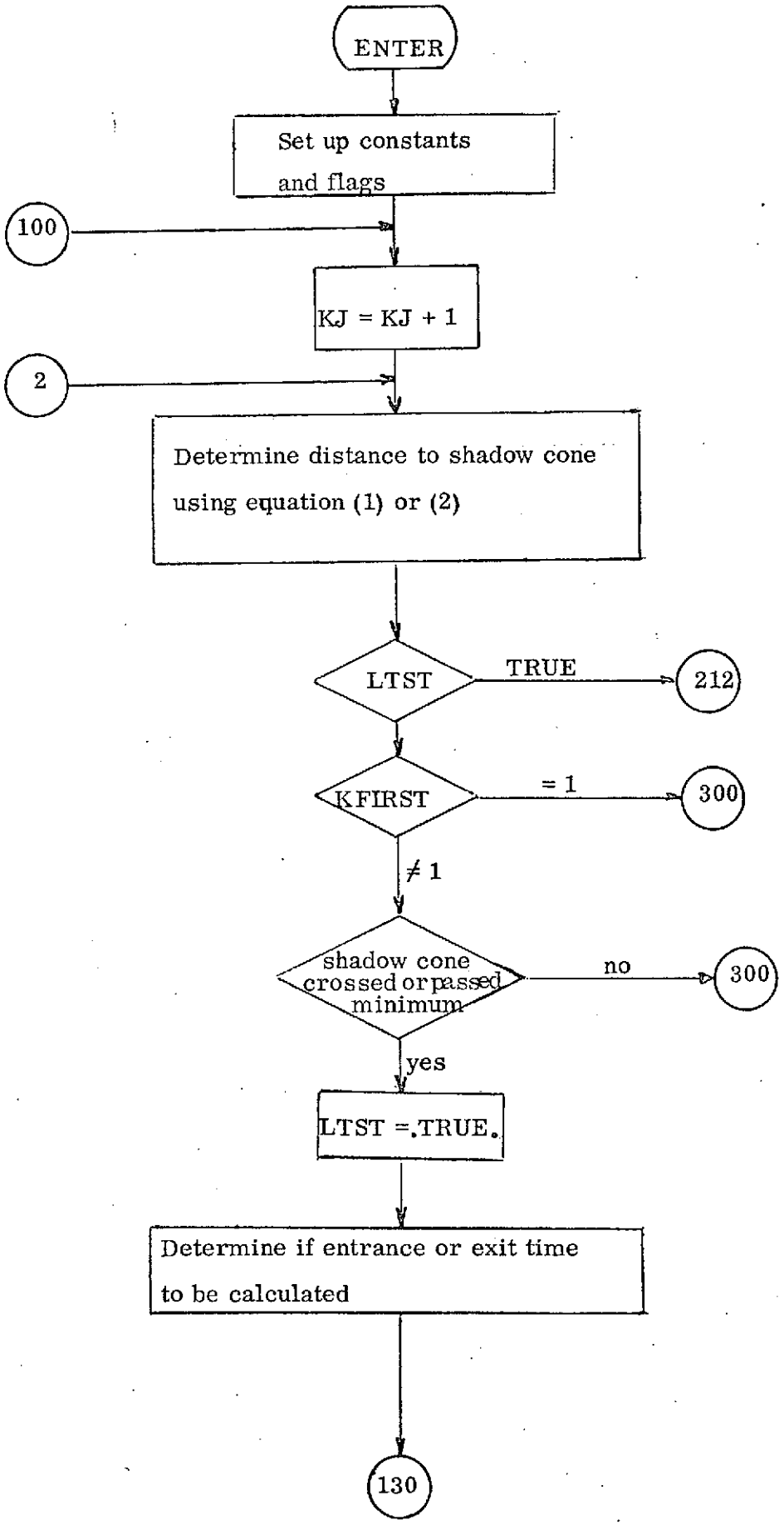


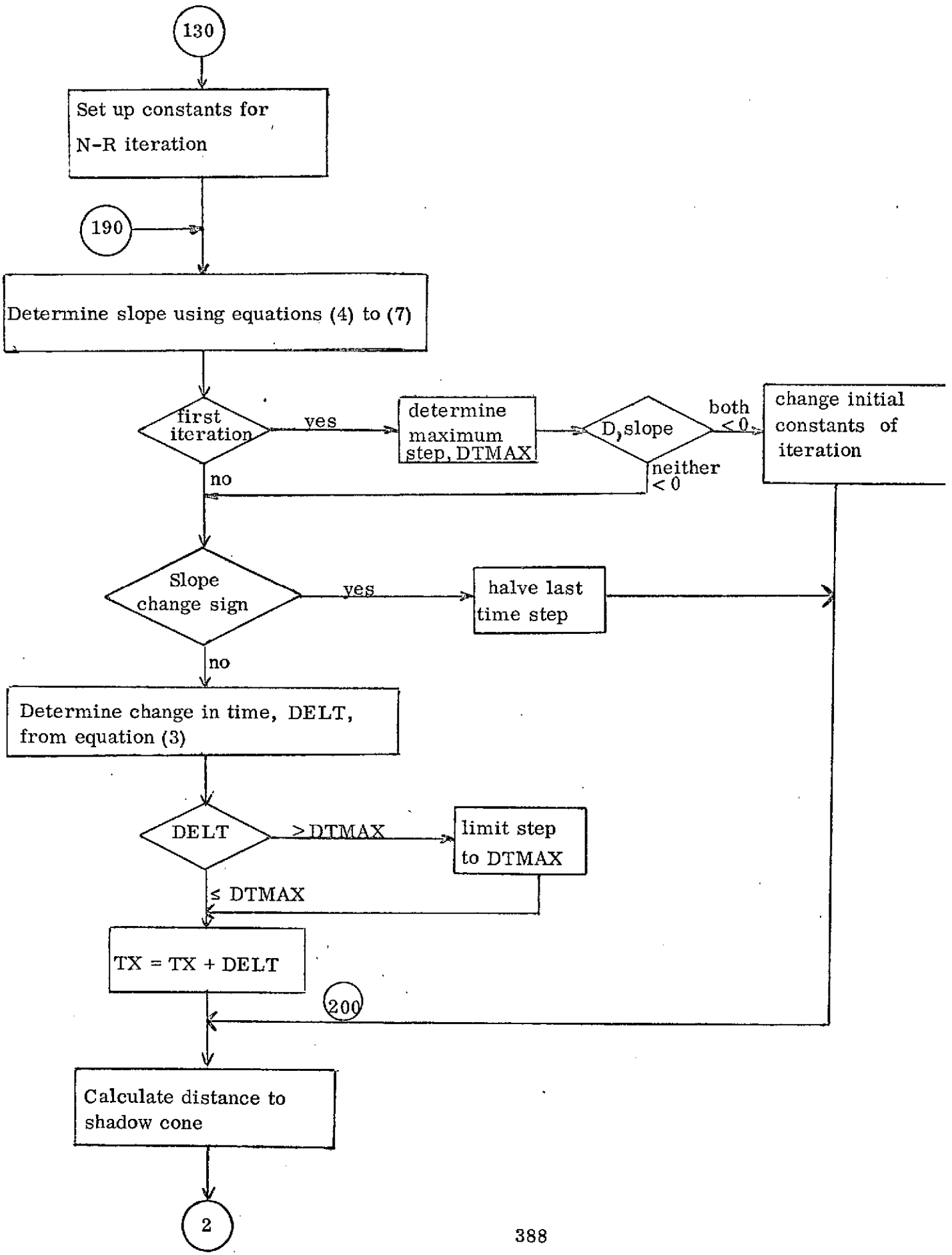
Figure 4

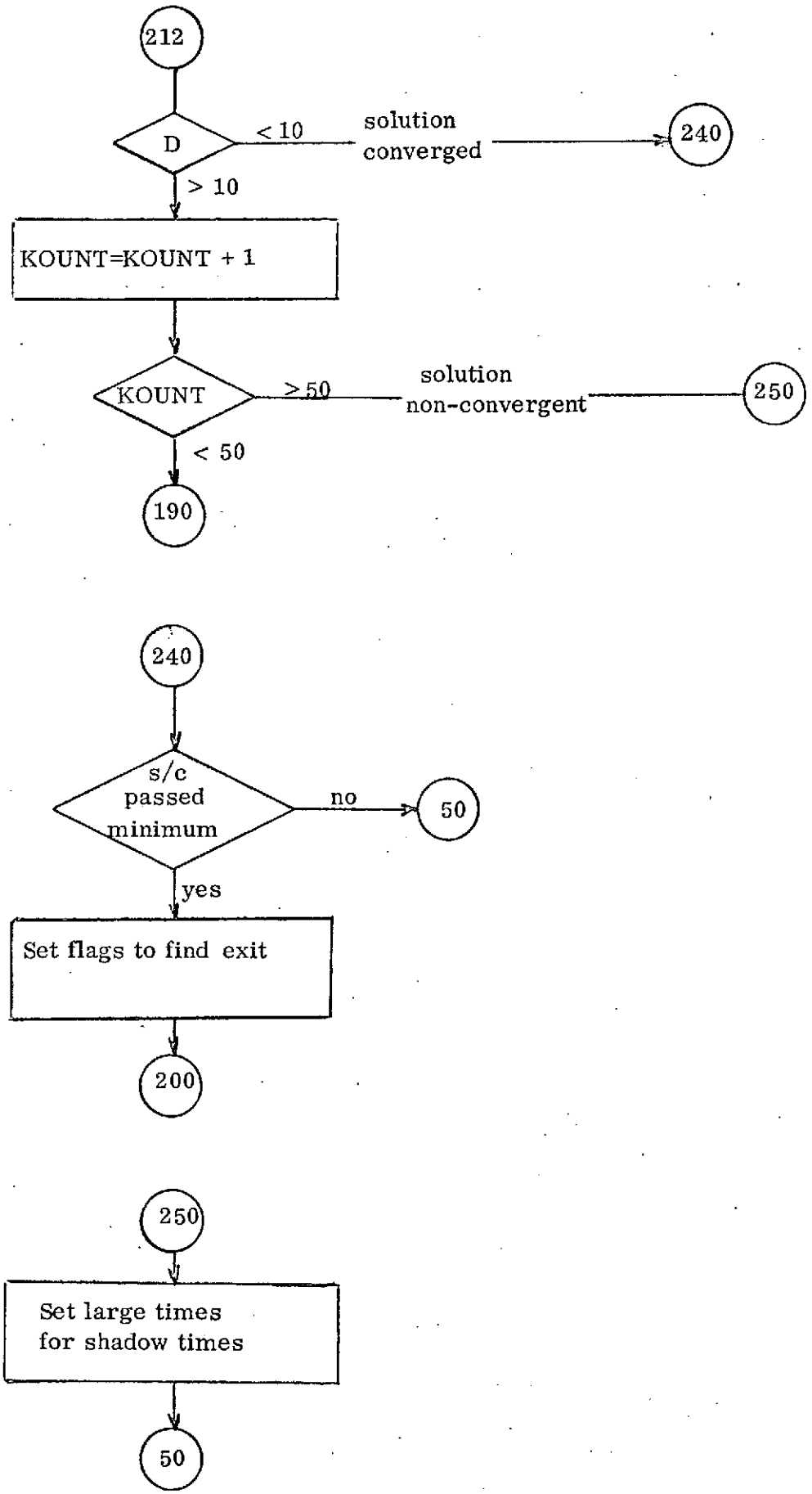
In this example, the spacecraft flight path just dips into the shadow cone. The problem arises when the derivative is calculated at point 1. The N-R technique would calculate point 2 as the entrance time. Since the minimum was skipped over, the iteration would converge on the shadow exit time instead of the entrance time it was expecting to determine. This problem was solved by testing on the sign of the derivative. Logic is included to reset the iteration to the last step and halve the calculated change in time, if the derivative changes sign. If the derivative changes sign more than five times, it is assumed that the spacecraft does not fly through the shadow cone. The shadow times determined are stored in the TSX array of SHAD common. The times are output from the array in subroutine SADOUT.

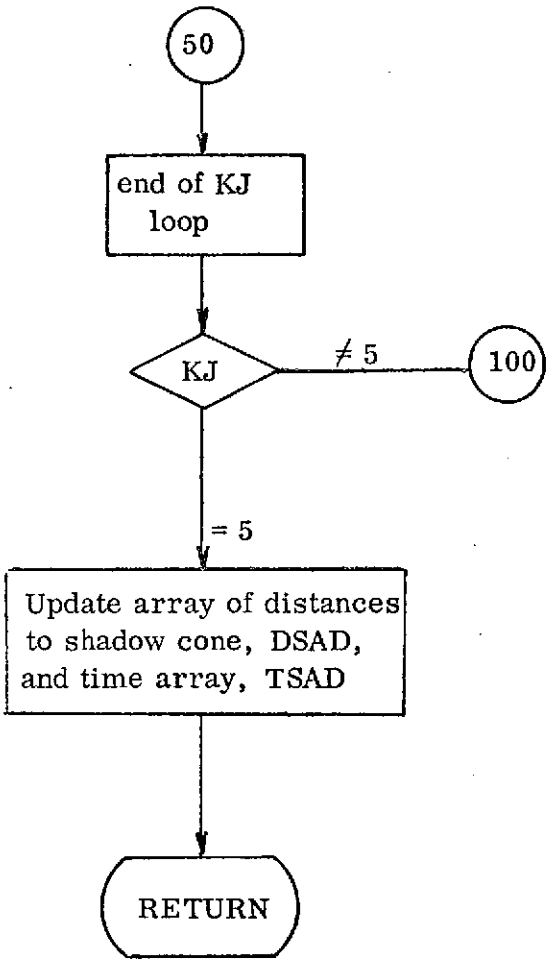
The DSAD array is updated on each of the passes in the KJ loop. On completion of this loop, the TSAD array is updated and the subroutine terminates.

SUBROUTINE SHADOW




130
Set up constants for N-R iteration
190
Determine slope using equations (4) to (7)
first iteration
yes
determine maximum step, DTMAX
D, slope
both < 0
neither < 0
change initial constants of iteration
no
Slope change sign
yes
halve last time step
no
Determine change in time, DELT, from equation (3)
DELT
> DTMAX
limit step to DTMAX
≤ DTMAX
TX = TX + DELT
200
Calculate distance to shadow cone
2


212
D
< 10
solution converged
240
> 10
KOUNT=KOUNT + 1
KOUNT
≥ 50
solution non-convergent
250
< 50
190
240
s/c passed minimum
no
50
yes
Set flags to find exit
200
250
Set large times for shadow times
50


50
end of KJ loop
KJ
≠ 5
100
= 5
Update array of distances to shadow cone, DSAD, and time array, TSAD
RETURN

SUBROUTINE SHORB2

Calling Sequence: CALL SHORB2 (ELM, SUN, RC, RSUN, JUMB, JPEN, UMBIN, UMBOUT, PENIN, PENOUT)

Purpose: This subroutine calculates the true anomaly of a given orbit that intersects the umbra and/or penumbra of a planet with respect to the Sun or another planet.

Common Blocks Required: None

Subroutine Required: ROTATE, QUARTC

Input / Output

I/Ø	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
I	ELM	6	Calling Operand	Orbital Elements
I	SUN	3	Calling Operand	Position coordinates of Sun
I	RC	1	Calling Operand	Radius of Planet
I	RSUN	1	Calling Operand	Radius of Sun
I/O	JUMB/JPEN	1	Calling Operand	Umbra and Penumbra Flag
O	UMBIN	1	Calling Operand	True anomaly when spacecraft enters umbra.
O	UMBOUT	1	Calling Operand	True anomaly when spacecraft leaves umbra.
O	PENIN	1	Calling Operand	True anomaly when spacecraft enters penumbra.
O	PENOUT	1	Calling Operand	True anomaly when spacecraft leaves penumbra.

Note: ELM and SUN must be expressed in the same coordinate system. All input/output angles are in radians.

Description: The true anomaly at which a spacecraft enters or leaves the UMBRA/PENUMBRA may be found by solving a quartic in the cosine of the true anomaly. For near - circular orbits (E .LT. .0015), a direct calculation is made for a close approximation of the intersections.

On input, the flags JUMB/JPEN must be set not equal to zero to calculate the UMBRAL/PENUMBRAL intersections respectively.

On output, the flags will have values 1 - 5 as follows:

(1) Impact or escape central planet (no shadows calculated)

(2) No shadow possible

(3) No real solutions on night side of terminator

(4) Solutions found

(5) No real solutions found

If JUMB/JPEN .NE. 4 on return, the output quantities UMBIN, UMBOUT/PENIN, PENOUT will be zero.

Note: The output quantities UMBIN, UMBOUT/PENIN, PENOUT will have values $-2\pi \leq \emptyset \leq 2\pi$ with UMBOUT $\geq$ UMBIN and PENOUT $\geq$ PENIN.

Theory: The distance of the spacecraft from the central planet is given by

$$r = \frac{P}{1+e\cos(\theta+\gamma)}$$

where P is the semi-latus rectum, e the eccentricity, γ the angle between pericenter and the projection of the shadow cone on the orbit plane, and θ is the in-orbit plane angle from the shadow cone centerline projection to the spacecraft.

The distance to the shadow is given by $\rho = R/\sin(\alpha+\beta)$ where R is the planet radius, α is half the shadow cone angle, and β is the angle between the line from the planet's center to the point on the shadow cone and the shadow cone centerline. (See Figure SHORB2.1)

THETA and BETA can be related through spherical trigonometry by

$\cos\beta = \cos\delta\ \cos\theta$, where δ is the angle between the shadow cone centerline and the orbit plane.

Since δ, γ, P and e are known constants, r and ρ can be equated, yielding the following equation

$$\frac{P}{1+e\cos(\theta+\gamma)} = \frac{R}{(\sin\alpha\cos\delta\cos\theta + \cos\alpha\sqrt{1-\cos^2\delta\ \cos^2\theta})}$$

which can be squared twice to yield the following quartic in cos (θ).

$$\text{COEF (5)} * x^4 + \text{COEF (4)} * x^3 + \text{COEF (3)} * x^2 + \text{COEF (2)} * x + \text{COEF (1)} = 0$$

where

$x = \cos(\theta)$

$\text{COEF (1)} = C^2 - D^2$

$\text{COEF (2)} = -2\ BC$

$\text{COEF (3)} = B^2 + 2AC + D^2(1+\cos^2\delta)$

$\text{COEF (4)} = -2\ AB$

$\text{COEF (5)} = A^2 - D^2\cos^2\delta$

and

$A = (P\sin\alpha\cos\delta - R\ e\cos\gamma)^2 + (R\ e\sin\gamma)^2 + (P\cos\alpha\cos\delta)^2$

$$B = 2R(P \sin\alpha \cos\delta - R e \cos\gamma)$$

$$C = R^2(1 - e^2 \sin^2\gamma) - P^2 \cos^2\alpha$$

$$D = 2PR e \sin\gamma \cos\alpha$$

This quartic is solved in subroutine QUARTC, which also outputs the number of roots found. If real roots exist, then the correct sign of θ must be found by returning to the original equation (since cosine is an even function). [cf. check quadrants of solutions]. Once the sign is known, cos θ is replaced by θ. Since the angle γ is the true anomaly of the shadow cone centerline projection in the orbit plane, the smallest absolute values of θ will be the desired intersections with the cone. Therefore the roots are ordered smallest first.

It is still possible that the intersections lie on the wrong side of the planet, so a check is made to make sure that θ is less than the angle from the centerline projection to the edge of the planet. [cf. check that solutions are on right side of umbral terminator].

The actual true anomalies of intersection are given by $\theta + \gamma$ and the output quantities are ordered so that the spacecraft enters the UMBRA/PENUMBRA before exiting (i.e. UMBOUT is greater than UMBIN).

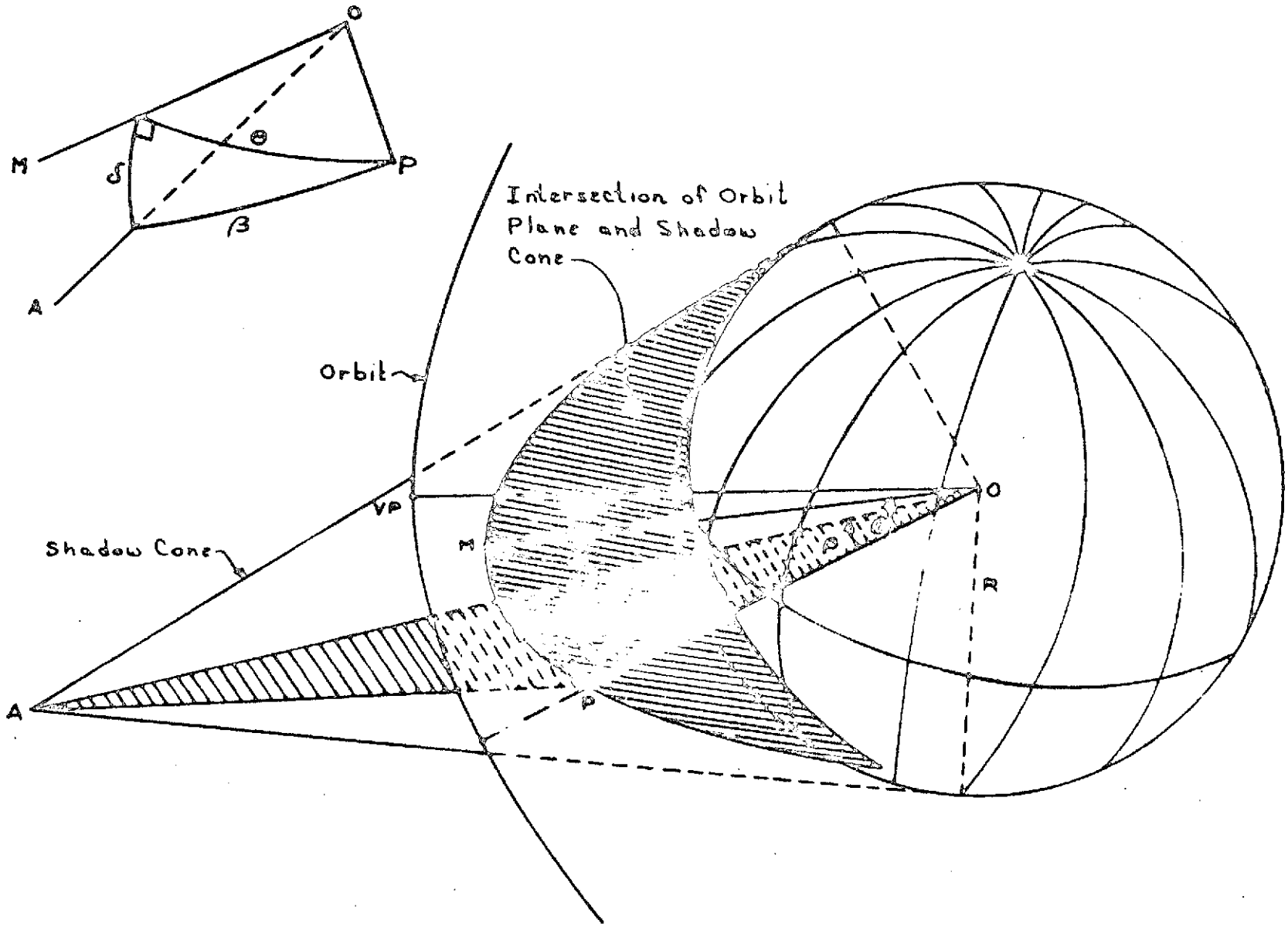
If the orbit is circular, then r is constant and $r = \frac{R}{\sin(\alpha + \beta_c)}$, where β_c = arc sin (R/r) - α which implies that θ_c = arc cos (cos β_c/cos δ). Here the quadrant checks are unnecessary since there are no extraneous roots.

The theory is exactly the same for PENUMBRAL cone intersections except for the angles α and γ, describing the cone. The half cone angle α is

arc sin ((radius of Sun + radius of planet)/ distance to Sun.) The angle γ_{PENUMBRA} equals γ_{UMBRA} minus π, since the cone is on the opposite side of the planet for PENUMBRA. The quadrant checks are slightly different in that the maximum values of θ are desired and θ must be greater than the angle from the Sun to the edge of the planet. The remainder of the calculations remain the same.

It is important to note that the assumption has been made that δ, γ, P and e are relatively constant throughout one period of revolution of the spacecraft.

Note: Items in square brackets refer to comment cards in subroutine.



GEOMETRICAL RELATIONSHIPS OF ORBIT AND SHADOW

SUBROUTINE SOL

Calling Sequence: CALL SOL

Purpose: This subroutine determines the position of the Sun with respect to the Earth.

Common Blocks Required: CNTRL, CONST, INPUT, INTVAR, PLNET, STATE

Subroutines Required: M50MDT, OBLTY, ROTATE

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	EJO	1	STATE(26)	Ephemeris time of state epoch
I	EJ1900	1	STATE(27)	Ephemeris time since 1900
I	JC	1	CNTRL(7)	Central planet number
I	T	1	INTVAR(1)	Current time
O	XP	6,12	PLNET(1)	Positions and velocities of the planets

Theory:

The position of the Sun with respect to the Earth is determined from the mean motion of the Sun as described in the Supplement to the Nautical Ephemeris. The mean anomaly of the Sun is obtained from

$$G = 358.47584^{o} + .985600267\ d - .1.12(10)^{-5}\ D^{2} - 7(10)^{-8}\ D^{3} \quad (1)$$

where

d is the number of days since 1900, and

D is the number of Julian centuries since 1900.

The argument of perifocus is obtained from

$$GAM = 281.22083 + 4.70684(10)^{-5}\ d + 3.39(10)^{-5}\ D^{2} + 7(10)^{-8}\ D^{3} \quad (2)$$

The eccentric anomaly can be determined from the power series expansion of Kepler's equation since the eccentricity, e, is small. This equation is

$$E = G + e \sin G + \frac{e^2}{2} \sin 2G + \frac{e^3}{8} (3 \sin^3 G - \sin G) \tag{3}$$

The X and Y components of the Sun are obtained from the ellipse as

$$\begin{aligned} X' &= a (\cos E - e) \\ Y' &= a \sqrt{1-e^2} \sin E \\ Z' &= 0 \end{aligned} \tag{4}$$

where

a is the semi-major axis of Earth's orbit about the Sun.

This X and Y position is rotated about the Z axis to account for the argument of perifocus. Thus

$$\begin{aligned} X &= \cos (GAM)\, X' - \sin (GAM)\, Y' \\ Y &= \sin (GAM)\, X' + \cos (GAM)\, Y' \\ Z &= 0 \end{aligned}$$

This vector represents the position of the Sun with respect to the Earth in the mean equinox and ecliptic of date. This vector is rotated to the mean equator and equinox of 1950 using subroutines M50MDT, OBLTY to establish the rotation matrices and ROTATE to perform the matrix multiplication.

SUBROUTINE SOLP

Calling Sequence: CALL SOLP

Purpose: This subroutine calculates the acceleration due to solar pressure.

Common Blocks Required: CONST, GRAVTY, INPUT, PLNET, PERT, STATE

Subroutines Required; CROSS, DVMAG, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ATT	3	STATE(11)	Unit vector along spacecraft attitude
I	KP	12	INPUT(1001)	Planets in the system
I	KSOLP	1	INPUT(1084)	Solar pressure flag
I/O	RCART	3	PERT(1)	Perturbing acceleration
I	REFLEK	1	INPUT(198)	Spacecraft reflectivity constant
I	SOL	1	STATE(36)	Solar pressure constant
I	W	1	STATE(35)	Current spacecraft mass
I	X	3	GRAVTY(1)	Current spacecraft position vector with respect to central planet

Theory:

The acceleration due to solar pressure can be calculated using two different models. In the first model, the solar pressure force acts along the radial direction from the Sun. This model assumes the spacecraft is a sphere or a flat plate perpendicular to the Sun's rays. Thus the acceleration can be obtained from,

$$\bar{a}_{sp} = \frac{SOL}{W R_{sun}} (1 - REFLEK) \hat{X}_{sun} \quad (1)$$

where

W	is the spacecraft mass
R_{sun}	is the distance from the Sun
REFLEK	is the reflectivity coefficient
X_{sun}	is the unit vector from the Sun to the spacecraft, and
SOL	is a constant defined by

$$SOL = Au^2 \ (SOLARA) \ (SPRESS) \ (10)^{-8}$$

where SOLARA is the spacecraft area

SPRESS is the solar pressure at 1 Au

In the second model, the spacecraft is assumed to be an object spinning about its centerline. This kind of a model will contribute two components to the solar pressure force. The first component is due to the absorbed light and is along the radial direction while the second component is due to reflected light and is normal to the centerline, see Figure 1.

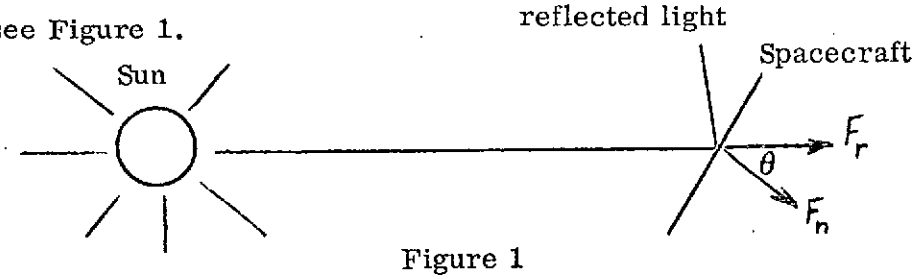


Figure 1

For this model, the radial component is determined from

$$(\bar{a}_{sp})_{rad} = \frac{SOL}{W R_{sun}} \cos\theta \, (1\text{-REFLK}) \, \hat{X}_{sun} \qquad (2)$$

where θ is defined in Figure 1.

The reflective contribution to the solar pressure is obtained from

$$(a_{sp})_{nor} = 2 \frac{SOL}{W R_{sun}} \text{REFLEK} \cos\theta \, \hat{F}_n \qquad (3)$$

where $\hat{F}_n$ is a unit vector normal to the plane containing the centerline and the Sun.

The total acceleration is the sum of the radial and normal components.

Description:

The subroutine determines if the spacecraft is in the umbral cone of any planet before the solar pressure acceleration is determined. The distance from the umbral cone is calculated from equation (1) of the SHADOW subroutine description. If the spacecraft is in an umbral cone,the solar pressure is assumed to be zero and the subroutine returns. If the spacecraft is not in shadows, the solar pressure is calculated from equation (1) or equations (2) and (3) according to the setting of the KSOLP flag. The solar pressure acceleration is added to the perturbing acceleration in RCART before the subroutine returns.

SUBROUTINE SPER

Calling Sequence: CALL SPER (X, Y)

Purpose: To convert the Cartesian coordinates X(1), X(2), and X(3) to spherical coordinates.

$$Y(1) = \sqrt{X(1)^2 + X(2)^2 + X(3)^2}$$

$$Y(2) = \tan^{-1} (X(3) / \sqrt{X(1)^2 + X(2)^2}), \quad -90^o < Y(2) < 90^o$$

$$Y(3) = \tan^{-1} (X(2) / X(1)), \quad -180^o < Y(3) < 180^o$$

Units of Y(2) and Y(3) are degrees.

Common Blocks used: None

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	DEFINITION
I	X	3	Input Cartesian Vector
O	Y	3	Output Spherical Coordinates

SUBROUTINE SPNM

Calling Sequence: CALL SPNM (NMAX, S, C, P)

Purpose: SPNM calculates the Legendre polynomials from $\bar{P}_o^o(S)$ to $P_{NMAX-1}^{NMAX+1}(S)$

Common Blocks Required: None

Subroutines Required: None

Reference: GULICK, L. J., (1970), "A comparison of Methods for Computing Gravitational Potential Derivatives", ESSA Technical Report, C & GS 40.

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	C	1	Calling Operand	$(1 - S^2)^{1/2}$
I	NMAX	1	Calling Operand	Highest degree desired + 1
O	P	17, 19	Calling Operand	Array of polynomials
I	S	1	Calling Operand	Argument of polynomials

Theory:

The associated Legendre polynomials are calculated from the following recursion relationship:

$$P_o^o(x) = 1 \qquad P_1^o(x) = x \qquad P_1^1(x) = (1-x^2)^{1/2}$$

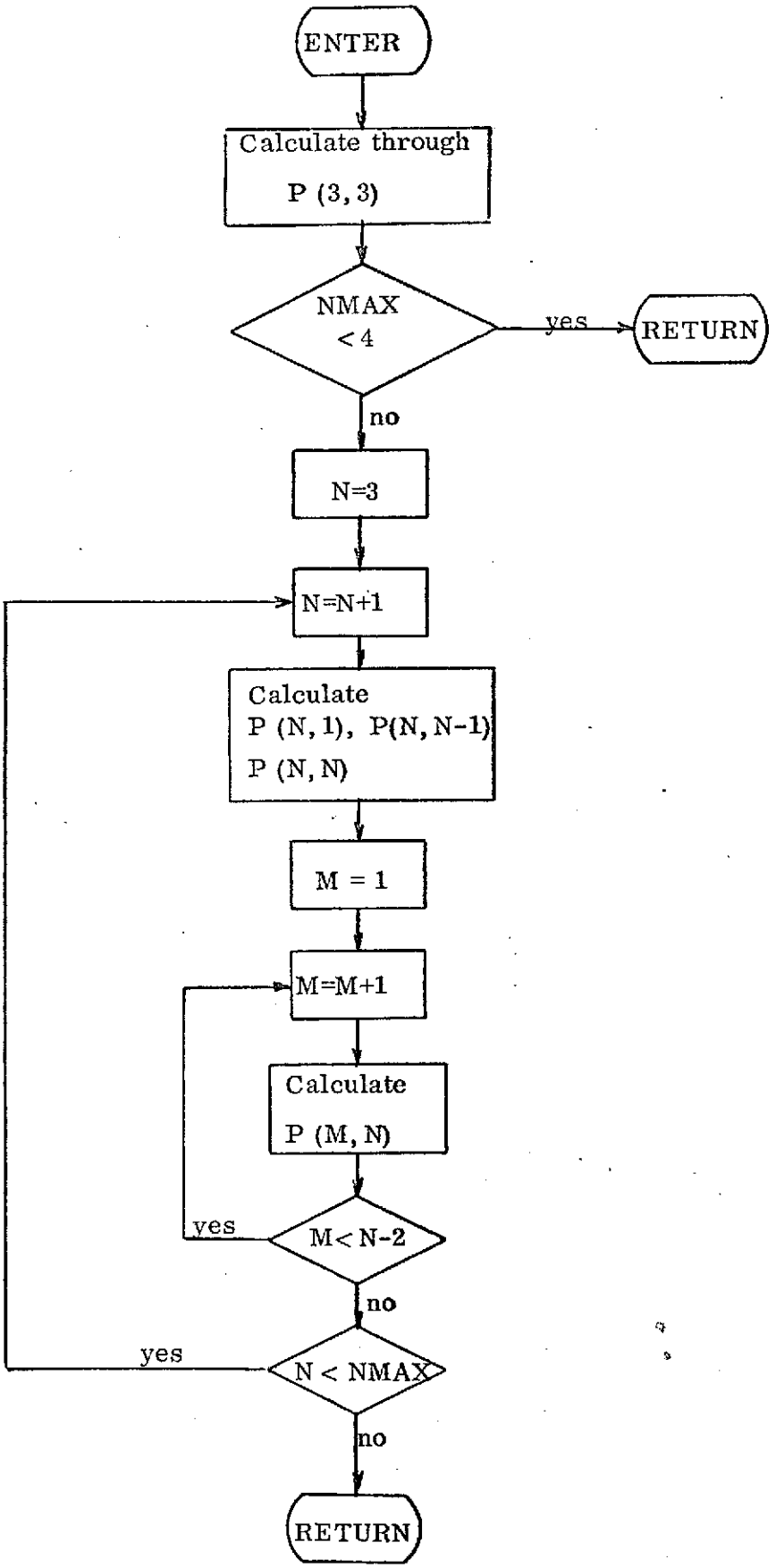
$$P_2^o(x) = \frac{1}{2}(3x^2-1) \qquad P_2^1(x) = 3x(1-x^2)^{1/2} \qquad P_2^2(x) = 3(1-x^2)$$

$$P_n^m(x) = (2m-1)(1-x^2)^{1/2} \qquad P_{n-1}^{m-1}(x) + P_{n-2}^m(x)$$

For a derivation of this relationship see the above reference.

Description:

SPNM calculates the Legendre polynomials of S for all degrees from zero to NMAX-1 and for all orders up to NMAX +1, and stores them in the array P. C must be equal to $+(1-S^2)^{1/2}$. NMAX must be less than or equal to 17. The indices are both one greater than the corresponding index used in the literature. Thus $P(3,1) = P_2^0$ in the usual notation for the associated Legendre polynomials.


ENTER
Calculate through
P (3, 3)
NMAX
< 4
yes
RETURN
no
N=3
N=N+1
Calculate
P (N, 1), P(N, N-1)
P (N, N)
M = 1
M=M+1
Calculate
P (M, N)
yes
M< N-2
no
yes
N < NMAX
no
RETURN

FUNCTION SUNMIN

Calling Sequence: FUNCTION SUNMIN (X, Y, Z)

Purpose: This function determines the minimum angle between vector Z and a plane formed by vectors X and Y. The angle must lie between X and Y.

Common Blocks Required: CONST

Subroutines Required: CROSS, ROTATE, VNORM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	PI2	1	CONST(3)	Twice pi
I	RAD	1	CONST(1)	Radian to degree conversion factor
I	X	3	CALLING OPERAND	Vector X as described in Purpose
I	Y	3	CALLING OPERAND	Vector Y as described in Purpose
I	Z	3	CALLING OPERAND	Vector Z as described in Purpose

Theory:

A rotation matrix is determined which transforms to a coordinate system with the X-axis along the X vector, the Y-axis normal to the plane formed by the vectors X and Y, and the Z-axis forming a right-handed system. This matrix is established by first calculating the unit vectors R and S defined by,

$$\hat{R} = \bar{X} \times \bar{Y} \; / \; |\bar{X}| \; |\bar{Y}|$$

$$\hat{S} = \bar{R} \times \bar{X} \; / \; |\bar{X}|$$

Then the first column of the matrix is the unit vector along vector X. The second column is composed of $\hat{S}$ while the last column is $\hat{R}$. The longitude of vectors Y and Z are

determined in the new coordinate system. Note, the longitude of vector X is zero. If the longitude of $\hat{Z}$ is between zero and the longitude of $\bar{Y}$, the minimum angle is equal to

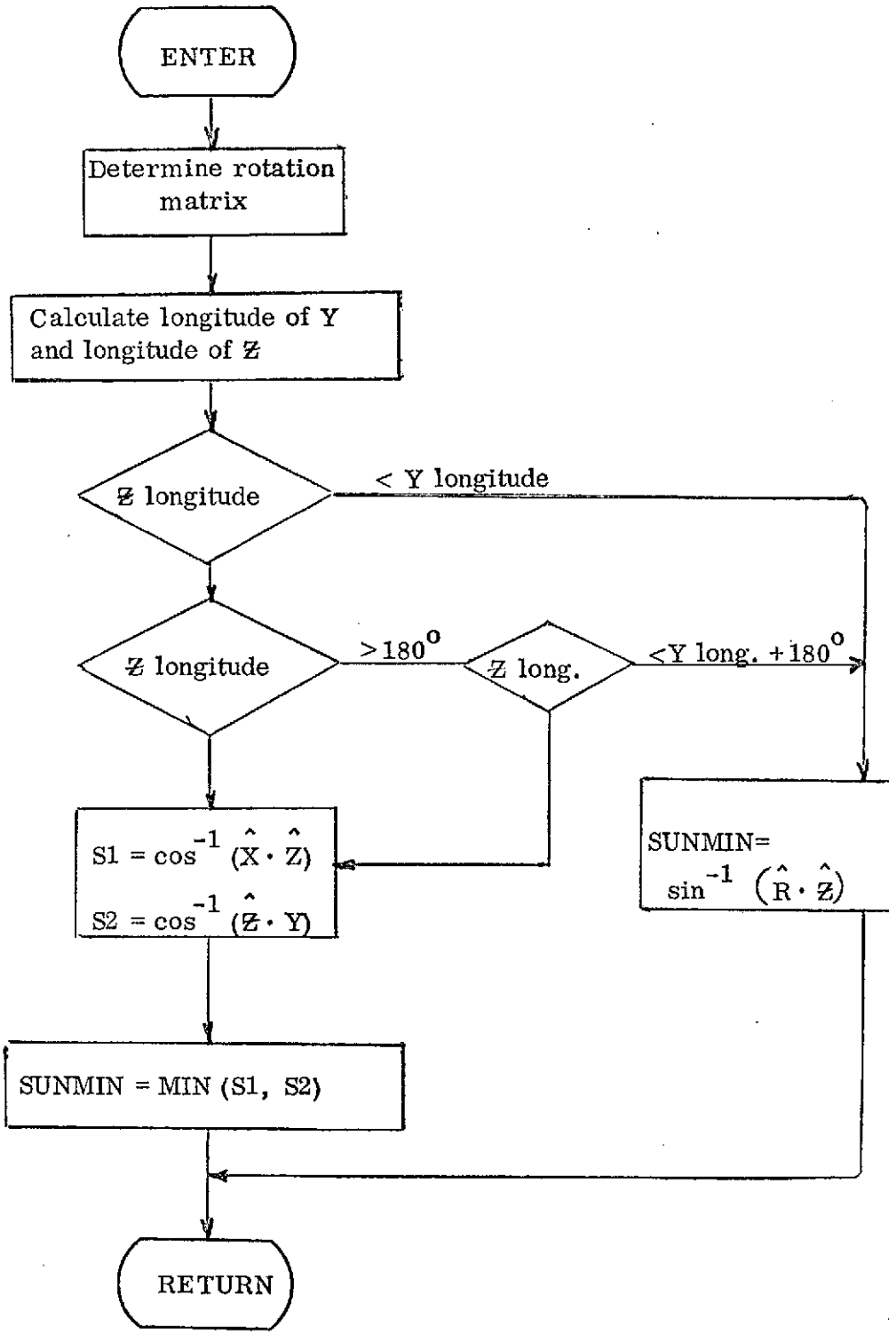
$$\text{SUNMIN} = \sin^{-1} (\hat{R} \cdot Z) / |\bar{Z}| \ .$$

If the longitude of $\bar{Z}$ is not between $\bar{X}$ and $\bar{Y}$, the minimum angle is equal to the minimum of

$$\text{SUNMIN}_1 = \cos^{-1} (\hat{Z} \cdot \hat{X})$$

$$\text{SUNMIN}_2 = \cos^{-1} (\hat{Z} \cdot \hat{Y}) \ .$$

SUBROUTINE SUNMIN



SUBROUTINE TABINT

Calling Sequence: CALL TABINT (X, K, J, N, R)

Purpose: This subroutine determines the spacecraft thrust and mass at the current time from input thrust and mass flow tables.

Common Block Required: INPUT

Subroutines Required: None

Input/Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	J	1	CALLING OPERAND	Location of the beginning of the dependent variable table in the input array
I	K	1	CALLING OPERAND	Location of the beginning of the independent variable table in the input array
I	N	1	CALLING OPERAND	Number of points in the table
O	R	1	CALLING OPERAND	Output current thrust or mass
I	X	1	CALLING OPERAND	Time from ignition of the desired engine.

Description:

The tables that are used to determine thrust or mass values are in two parts. The first part is the independent variable beginning in location K of the input array. This array consists of the times from engine ignition and must be monotonically increasing. The second part of the table is an array beginning in location J of the input array. This array contains values of the thrust or mass flow rate at the corresponding times of the first array. For example, the thrust in location J+2 occurs at the time since ignition input in location K+2. The value of J is used to determine if thrust or mass values are to be determined.

If J is greater than 350 and less than 379, then the mass is to be determined. Otherwise, a thrust table is being used.

If the thrust is to be determined, the thrust is obtained from

$$R = A(L) + \Big(A(L) - A(L-1)\Big)\Big(X - A(M)\Big) \Big/ \Big(A(M) - A(M-1)\Big) \qquad (1)$$

where the time X lies between

$$A(M) \text{ and } A(M-1)$$

and A(L) is the thrust at time A(M)

The mass is determined by a trapezoidal integration of the mass flow table up to the current time, X, from

$$M = \sum_{i=1}^{N} \frac{1}{2} \Big(A(J+i) - A(J+i-1)\Big) \Big(A(K+i) + A(K+i-1)\Big) \qquad (2)$$

where

N is the location of the last time in the independent variable array less than X.

The final mass is given by

$$R = M + \frac{1}{2} \Big(X - A(J+N)\Big)\Big(A(K+N) + R\Big) \qquad (3)$$

where

R is determined similar to equation (1) except the mass flow rate table is used.

SUBROUTINE TARGET

Calling Sequence: CALL TARGET

Purpose: TARGET computes the end constraint error vector for midcourse guidance calculations.

Common Blocks Required: CONST, CNTRL, INPUT, MCCOM, PLNET, STATE

Subroutines Called: M50LEQ, MVTRN, BVE, ORIENT, DVMAG, RETDV, ORBIT, RETRO, CROSS, VNORM, ROTAIT

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	JC	1	CNTRL(7)	Central body number of end state
I	RTD	1	CONST(1)	Radians-to-degrees conversion factor
I	UM	12	CONST(5)	Gravitational constant array (km^3/sec^2)
I	GKS	1	CONST(45)	Earth's surface gravity (km/sec^2)
I	WTO	1	INPUT(38)	Initial spacecraft weight (kg)
I	DJO	1	INPUT(46)	Julian date of anchor epoch (days)
I	ASPMC	1	INPUT(441)	Specific impulse of the trim motor (sec)
I	RCIRC	1	INPUT(444)	Final desired orbit radius (km)
I	WDROP	1	INPUT(473)	Post-retro drop weight (kg)
I	JTARG	1	INPUT(1031)	Target body number
I	IVTI	1	INPUT(1078)	Overburn strategy key
I	NORMIN	1	INPUT(1080)	Retro optimization key
I	DV	3	MCCOM(12)	Midcourse correction impulse (km/sec)
O	DVS	3	MCCOM(15)	Spherical components of DV

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	BVD	2	MCCOM(19)	Desired miss-vector if IBTR=1 (km)
I	DVB4	1	MCCOM(24)	Previously expended midcourse velocity (km/sec)
O	DVRET	1	MCCOM(25)	Retro velocity magnitude (km/sec)
O	TV	3	MCCOM(27)	Anti-retro unit vector (lunar equator ref)
O	DVT	1	MCCOM(30)	Trim velocity (km/sec)
O	FUELT	1	MCCOM(31)	Trim fuel (kg)
O	XS	6	MCCOM(32)	Cartesian end state (km, km/sec, lunar equator)
O	WTF	1	MCCOM(47)	Post-midcourse spacecraft weight(kg)
I	PSID	10	MCCOM(80)	Desired end constraint vector
O	PSI	10	MCCOM(100)	End constraint error vector
I	IBTR	1	MCCOM(167)	Miss vector type key (1=BT, BR, 2= RCA, INC)
I	XP	6, 12	PLNET(1)	Celestial body states at T (km, km/sec)
I	X	6	STATE(1)	End state (km, km/sec)
I	T	1	STATE(10)	End time corresponding to X (sec)

Description:

TARGET is a very important subroutine in the computation of midcourse guidance corrections. It computes the end constraint error vector, ψ, as a function of the desired end conditions and the end state. In the normal mode of operation, each guidance law is formulated to constrain radius of closest approach and inclination. Desired values of these quantities are combined with actual arrival energy and the direction of the approach asymptote to arrive at equivalent "desired" miss-vector components, $B \cdot T_d$ and $B \cdot R_d$. This computation is described in Appendix A of Reference 1. The first two constraint error vector components are then formulated:

$$PSI(1) = B \cdot T_d - B \cdot T_{actual}$$

$$PSI(2) = B \cdot R_d - B \cdot R_{actual}$$

where the actual miss-vector is computed in subroutine BVE. The third and fourth constraint error components are for time of flight and hyperbolic excess speed, respectively.

$$PSI(3) = TFS - T$$

$$PSI(4) = VINFD - VINF$$

TFS is the desired time at closest approach measured from anchor epoch and VINFD is the square root of the desired arrival energy (C_3). The fifth constraint error component is post-retro circular excess velocity, assuming the retro to burn at periapsis anti-parallel to the velocity there.

$$PSI(5) = VDAR - VAR$$

VDAR, velocity desired after retro, is supplied by subroutine MCSET, having been computed to be circular velocity at the desired arrival radius plus an optional input increment. VAR, velocity after retro, is computed as the scalar difference between the arrival periapsis speed and the retro velocity magnitude, DVRET. DVRET is, in turn, computed from the rocket equation and the midcourse correction velocity in function RETDV. The sixth component of PSI is not an error, but rather is the total correction fuel.

$$PSI(6) = WTO - WTF + FUELT$$

WTO-WTF is the midcourse correction fuel and FUELT is the trim correction fuel. FUELT is computed by the rocket equation from the trim velocity, DVT, about which more will be said later. The seventh and eighth components of PSI are the errors in achieving the desired periapsis radius, PRD, and the desired inclination, OINC.

$$PSI(7) = PRD - PR$$
$$PSI(8) = OINC - SINC$$

PR is the actual closest approach radius as computed from the end state in BVE. SINC is the post-retro inclination, which differs from the approach orbit's inclination only when the variable target inclination procedure is used.

Retro and Trim Calculations

The retro-strategy implemented in TARGET depends on:

1. arrival energy, C_3
2. radius of closest approach, r_p
3. inclination, i
4. retro-velocity impulse, δv
5. input option keys, NORMIN, IVTI and IBTR.

TARGET computes the "desired" B·T and B·R values as functions of these parameters as well as predicting:

a. firing true anomaly, θ, on the approach hyperbola.
b. direction, TV, of the spin-axis at retro-fire.
c. trim velocity and fuel for in-plane trim.

(Case 1: NORMIN = 2)

In this case, subroutine RETRO is used to optimize θ and TV and to provide the trim velocity. Overburn strategies are ignored. Desired miss vector computation depends on IBTR and input miss parameters.

(Case 2: NORMIN <2, IBTR = 1)

Overburn strategies are ignored and retro is anti-velocity at periapsis.

(Case 3: NORMIN <2, IBTR = 2, IVTI = 0)

Same as case 2 except for computation of desired miss vector.

(Case 4: NORMIN <2, IBTR = 2, IVTI = ±1)

Same as case 3 for underburns*, otherwise variable target inclination computations are invoked.

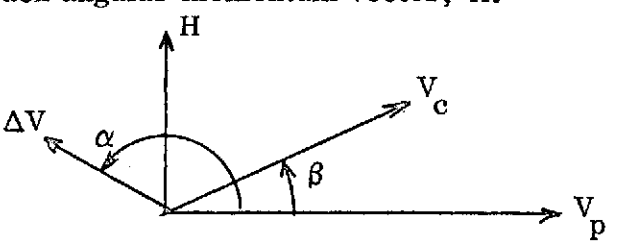
(Case 5: NORMIN <2, IBTR = 2, IVTI = ±2)

Same as case 3 for underburns*, otherwise variable approach distance computations are performed for the desired miss vector, retro direction, and trim velocity.

* An underburn is defined as the condition when the retro impulse would be sufficient for circularizing at the desired radius if the closest approach radius were indeed the desired circular orbit radius.

Variable Target Inclination

The VTI procedure is derived in Appendix B of Reference 1. It amounts to defining desired miss vector components, B·T and B·R, in such a way that an out-of-plane retro-burn at periapsis of the approach hyperbola can render a circular orbit of the desired radius and inclination. This procedure is applicable only for overburns. The sketch shows the retro velocity impulse, ΔV, in the plane of the approach periapsis velocity, V_p, and the approach angular momentum vector, H.



Geometry of the VTI Procedure

The direction of ΔV is given by

$$\hat{\Delta V} = \hat{V}_p \cos\alpha + \hat{H} \sin\alpha$$

and the new angular momentum vector's direction is

$$\hat{H} = \hat{H} \cos\beta - \hat{V}_p \sin\beta.$$

The angles, α and β, are defined in the derivation of the VTI procedure as functions of C_3, δv, and the desired final orbit radius. The desired miss parameters are computed in such a way that the approach orbit's track differs by β from the desired orbit's tract at r_p. If the closest approach and inclination constraint errors are exactly zero, the post-retro orbit will require no time.

Variable Approach Distance

The VAD procedure, like the VTI, is applicable only to overburns. The basic idea is to define the desired miss vector magnitude in such a way that if the retro is fired appropriately in-plane at the desired circular orbit radius, r_c, the post-retro orbit will be circular. It is assumed, first of all, that closest approach radius, r_p, of the arrival hyperbola can be varied without significantly changing the arrival energy, C_3, or the retro velocity impulse, δv. Assuming further that r_p is less than the desired circular orbit radius, r_c, we can write

$$V(r_c) = v_r \hat{R}_c + v_\theta \hat{\theta}$$

where v_r is the radial component of velocity and v_θ is the tangential component. The circular, post-retro velocity is:

$$V_c(r_c) = v_c \hat{\theta}$$

where $v_c = \sqrt{\frac{\mu}{r_c}}$. The required retro impulse, ΔV, is

$$\Delta V = V_c - V = v_r \hat{R} + (v_c - v_\theta) \hat{\theta}$$

and

$$\delta v^2 = v_r^2 + (v_c - v_\theta)^2 = v^2 - 2 v_c v_\theta + v_c^2$$

$$= c_3 + \frac{2\mu}{r_c} - 2 \frac{\mu}{r_c} \frac{h}{r_c} + \frac{\mu}{r_c}$$

$$= c_3 + \frac{3\mu}{r_c} - 2 \frac{\mu}{r_c} \frac{r_p}{r_c} \quad c_3 + \frac{2\mu}{r_p}$$

The only unknown in the above equation is r_p. We can solve for r_p as follows,

$$c_3 r_p^2 + 2\mu r_p = \left\{ \left[\delta v^2 - c_3 - \frac{3\mu}{r_c} \right] \frac{r_c}{2 v_c} \right\}^2 = b^2$$

$$r_p = \sqrt{\frac{\mu^2 + c_3 b^2}{c_3}} - \frac{\mu}{c_3}$$

The periapsis constraint error is re-defined for this case as:

PSI (7) = RP - PR.

The desired miss-vector magnitude is easily formulated from r_p.

$$BMAG = \left(\frac{\mu}{c_3} + r_p \right) \sin \left(\tan^{-1} \frac{v_p v_\infty r_p}{\mu} \right)$$

$$v_p = \sqrt{c_3 + \frac{2\mu}{r_p}} \quad , \quad v_\infty = \sqrt{c_3}$$

The true anomaly at which the retro motor is fired is found from

$$\cos \theta = \left(\frac{p}{r_c} - 1 \right) / e$$

where the sign of θ is determined by input: if IVTI is positive, θ is negative and vice versa. Defining unit vectors $\hat{P}$ and $\hat{Q}$ along periapsis position and velocity vectors, respectively,

$$\hat{R} = \hat{P} \cos \theta + \hat{Q} \sin \theta$$

$$\hat{\theta} = \hat{P} \sin \theta + \hat{Q} \cos \theta$$

We define ΔV in terms of an out-of-plane angle, α, and a flight path angle, γ.

$$\Delta V = \delta v \left[(\hat{\theta} \cos \alpha + \hat{H} \sin \alpha) \cos \gamma + \hat{R} \sin \gamma \right]$$

If the periapsis constraint error is zero, $\alpha = \pi$, and the maneuver will be in-plane. If δv is too large to circularize at r_c for the actual r_p and c_3, the maneuver will be out-of-plane. The direction out-of-plane is chosen to minimize the resultant inclination error. If δv is too small, γ will be chosen so that the radial component of post-retro velocity will be nulled, with the tangential component falling wherever it may as a result.

$$\sin \gamma = \frac{v_r}{\delta v}$$

$$\cos \alpha = \frac{v_c - v_\theta}{\delta v \cos \gamma} \quad , \quad \text{or} \quad \alpha = \pi \text{ if } |\cos \alpha| > 1$$

Trim Velocity

The trim maneuvers in TARGET do not attempt to remove inclination errors. They are treated simply as two-impulse Hohmann transfers from the post-retro orbit to the desired circular orbit. We require only the specification of periapsis radius and apo-apsis radius of the post-retro orbit to compute the required trim velocity. (See TRIM) These may be computed,given the post-retro energy, C_{3e}, and angular momentum, he. The general form of the post-retro velocity is

$$V_a = V + \Delta V$$

$$= (v_r \hat{R} + v_\theta \theta) + \delta_v \left[(\theta \cos \alpha + H \sin \alpha) \cos \gamma + \hat{R} \sin \gamma \right]$$

so that the energy (C_{3e}) is

$$C_{3e} = v_a^2 - \frac{2\mu}{r} = (v_r + \delta v \sin \gamma)^2 + (v_\theta + \delta v \cos \alpha \cos \gamma)^2$$

$$+ (\delta v \sin \alpha \cos \gamma)^2 - \frac{2\mu}{r}$$

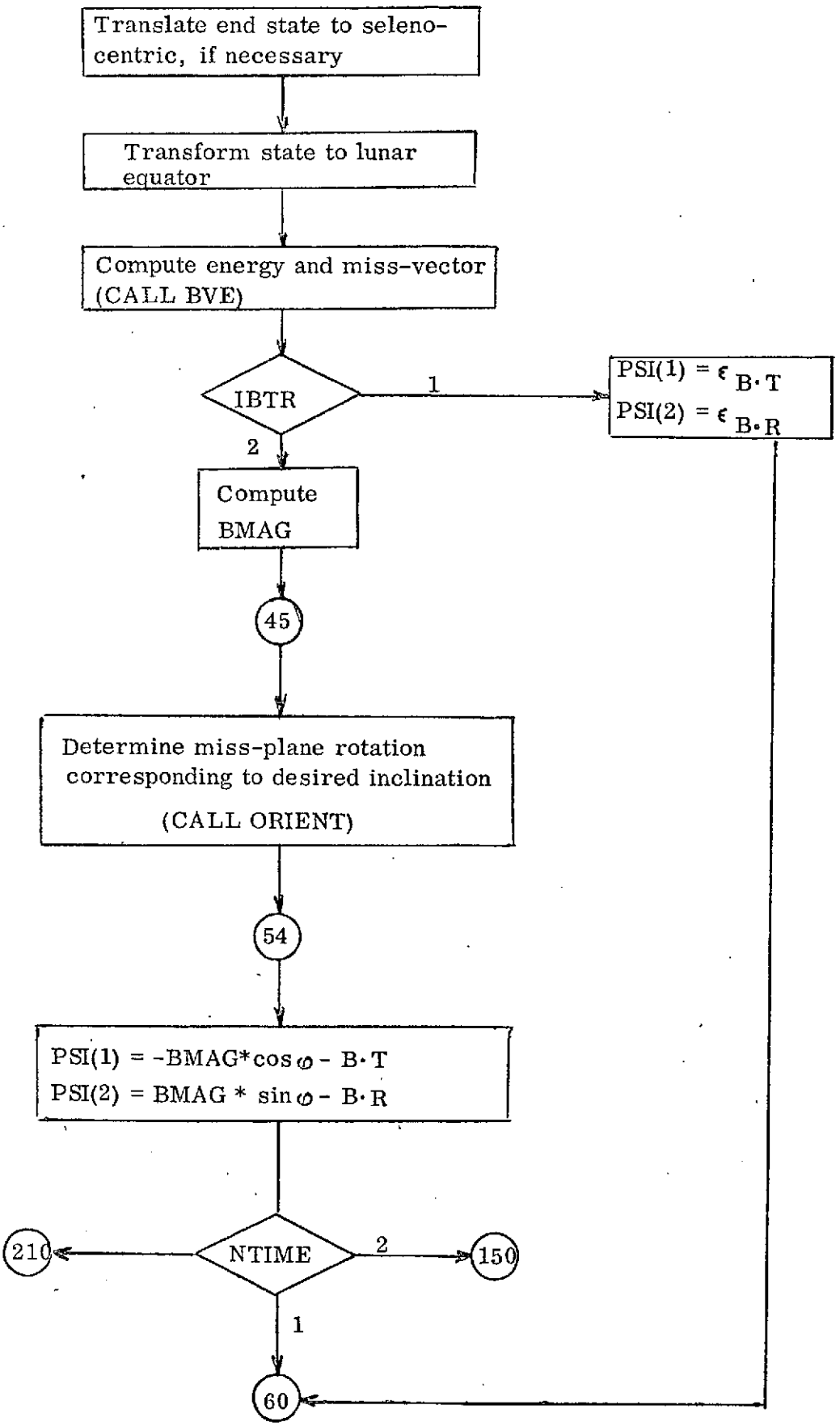
where r is the radius of retro-fire. The tangential post-retro velocity is the magnitude of $\hat{R} \times V_a$.

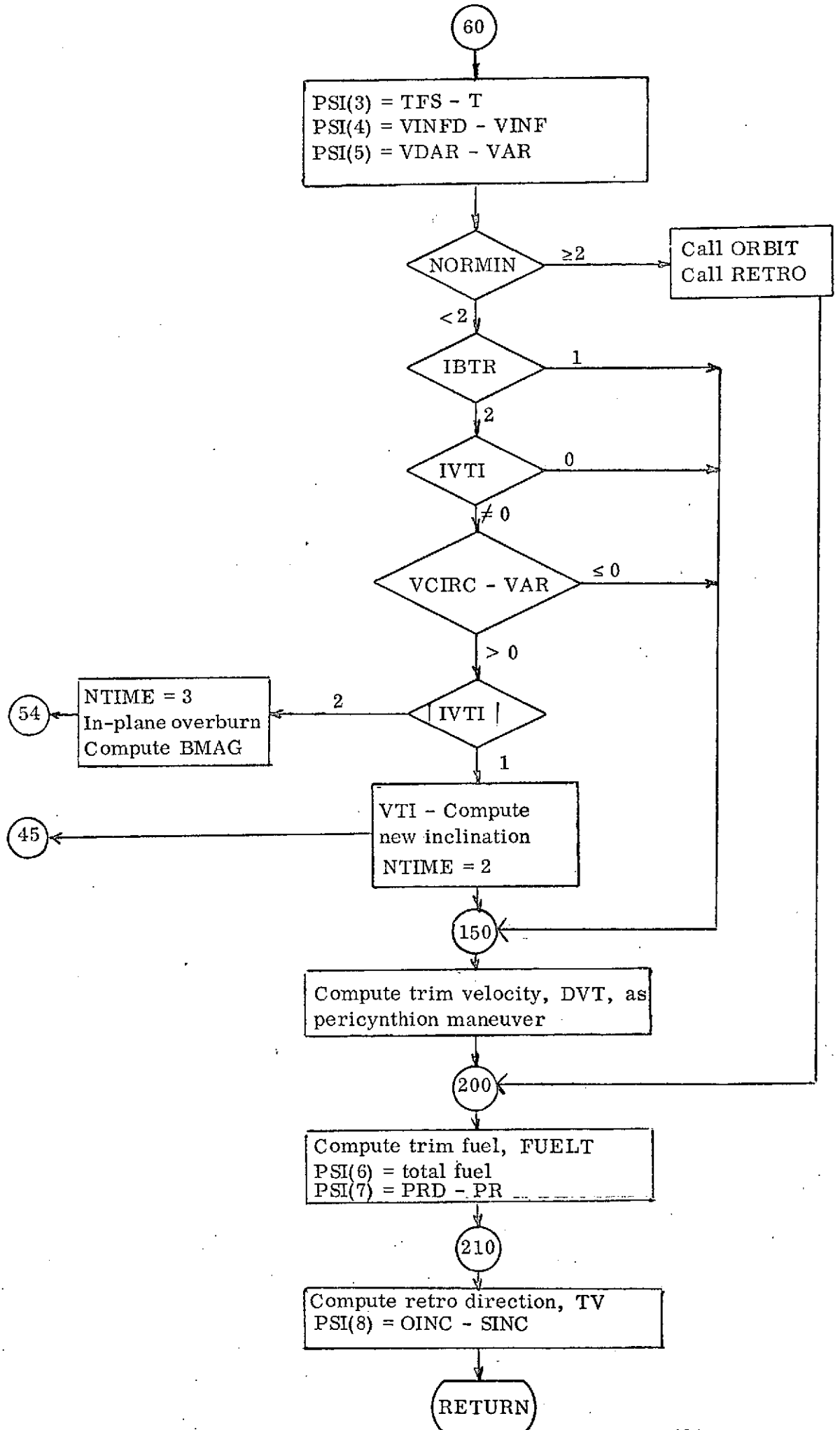
$$\hat{R} \times V_a = (v_\theta + \delta v \cos\alpha\cos\gamma)\,\hat{H} - (\delta v \sin\alpha\cos\gamma)\,\hat{\theta}$$

$$h_e = r\sqrt{(v_\theta + \delta v \cos\alpha\cos\gamma)^2 + (\delta v \sin\alpha\cos\gamma)^2}.$$

Reference 1 Bjorkman, W.S., Midcourse Guidance for Lunar and Planetary Orbiting Missions, AMA 71-16, March, 1971.

SUBROUTINE TARGET




60
PSI(3) = TFS - T
PSI(4) = VINFD - VINF
PSI(5) = VDAR - VAR
NORMIN
≥2
Call ORBIT
Call RETRO
<2
IBTR
1
2
IVTI
0
≠0
VCIRC - VAR
≤0
>0
NTIME = 3
In-plane overburn
Compute BMAG
54
2
IVTI
1
VTI - Compute
new inclination
NTIME = 2
45
150
Compute trim velocity, DVT, as
pericynthion maneuver
200
Compute trim fuel, FUELT
PSI(6) = total fuel
PSI(7) = PRD - PR
210
Compute retro direction, TV
PSI(8) = OINC - SINC
RETURN

SUBROUTINE TIMEC

Calling Sequence: CALL TIMEC

Purpose: This subroutine controls the time logic during numerical integration. Its primary functions are to determine the compute interval, discontinuity times, and stopping criteria.

Common Blocks Required: CONST, CNTRL, INPUT, INTER, INTVAR, PERT, SAVE, STATE

Subroutines Required: CLOSE, CRASH, DOPLER, INTEG, MOTORS, OUT1, ORBIT, OUTPUT, SADOUT, SHADOW, TRMN, UPDATE.

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DELT	10	INPUT(180)	Compute intervals
I	ERRC	1	INPUT(1)	Error control limit for automatic compute interval determination
I	DELTMN	1	INPUT(3)	Minimum compute interval
I	DELTO	1	INPUT(2)	Initial compute interval
I	DX	3	STATE(4)	Spacecraft velocity
I/O	ELM	6	STATE(14)	Spacecraft orbital elements
I	GM	12	CONST(5)	Gravitational constants
I	JC	1	CNTRL(7)	Central planet number
I	JT	1	INPUT(1031)	Target planet number
I/O	KCA	1	CNTRL(11)	Counter used in closest approach iteration

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	KCRASH	1	INPUT(1032)	Closest approach flag
I/O	KFIRST	1	CNTRL(12)	First pass flag
I	KFMOD	1	INPUT(1047)	Thrusting mode
O	KDIS	1	CNTRL(5)	Discontinuity flag
I	KDOP	1	INPUT(1045)	Doppler flag
I	KHALT	1	CNTRL(6)	Error return flag
I	KMETH	3	INPUT(36)	Input trajectory propagator method
I	KOUT	1	INPUT(1030)	Output frequency flag
I	KTHRST	1	CNTRL(2)	Thrusting flag
I	KSADOW	1	INPUT(1049)	Shadow flag
O	METH	1	INPUT(1013)	Trajectory propagator to be used on this step.
I	T	1	STATE(10)	Current time
I	TCA	1	STATE(29)	Time of closest approach
I	TCATST	1	INPUT(513)	Time to begin closest approach testing
I	TCOMP	10	INPUT(170)	Switching times of compute interval table
I	TF	1	INPUT(4)	Run stop time
I	TIG	6	INPUT(380)	Engine ignition and burnout times.
I	TMETH	3	INPUT(10)	Switching times of trajectory propagator method table
I	X	3	STATE(1)	Spacecraft position

Description:

This subroutine controls the flow of logic during trajectory propagation by numerical integration. When the subroutine is initiated, the initial spacecraft state is in STATE common. The state at the final time resides in STATE common when the subroutine terminates.

The primary function of this subroutine is to determine the compute interval, discontinuity times, and stopping criteria. The discontinuity times consist of engine ignition and burnout times, final time, and time of closest approach. The compute interval is adjusted so that the integrator stops at those times exactly. The state and its derivatives are saved at the beginning of each step. If a discontinuity time is passed during the step, the state is restored to the saved value and the compute interval adjusted to integrate to the discontinuity time.

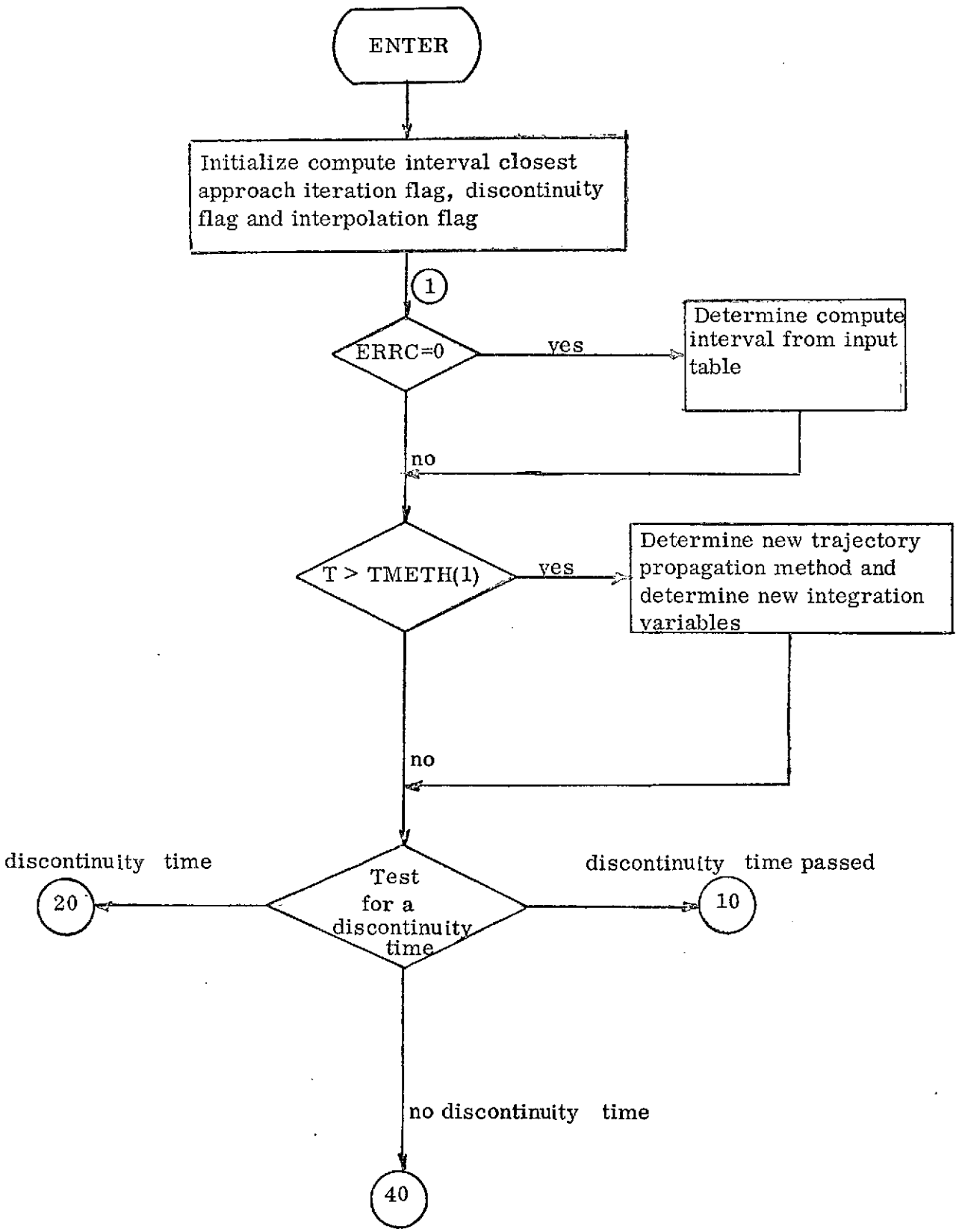
The compute interval is obtained from the input compute interval table, from the discontinuity logic to hit a discontinuity time, or, if the automatic compute interval option is used, the compute interval is determined by subroutine RKSEVN and passed through INTVAR common.

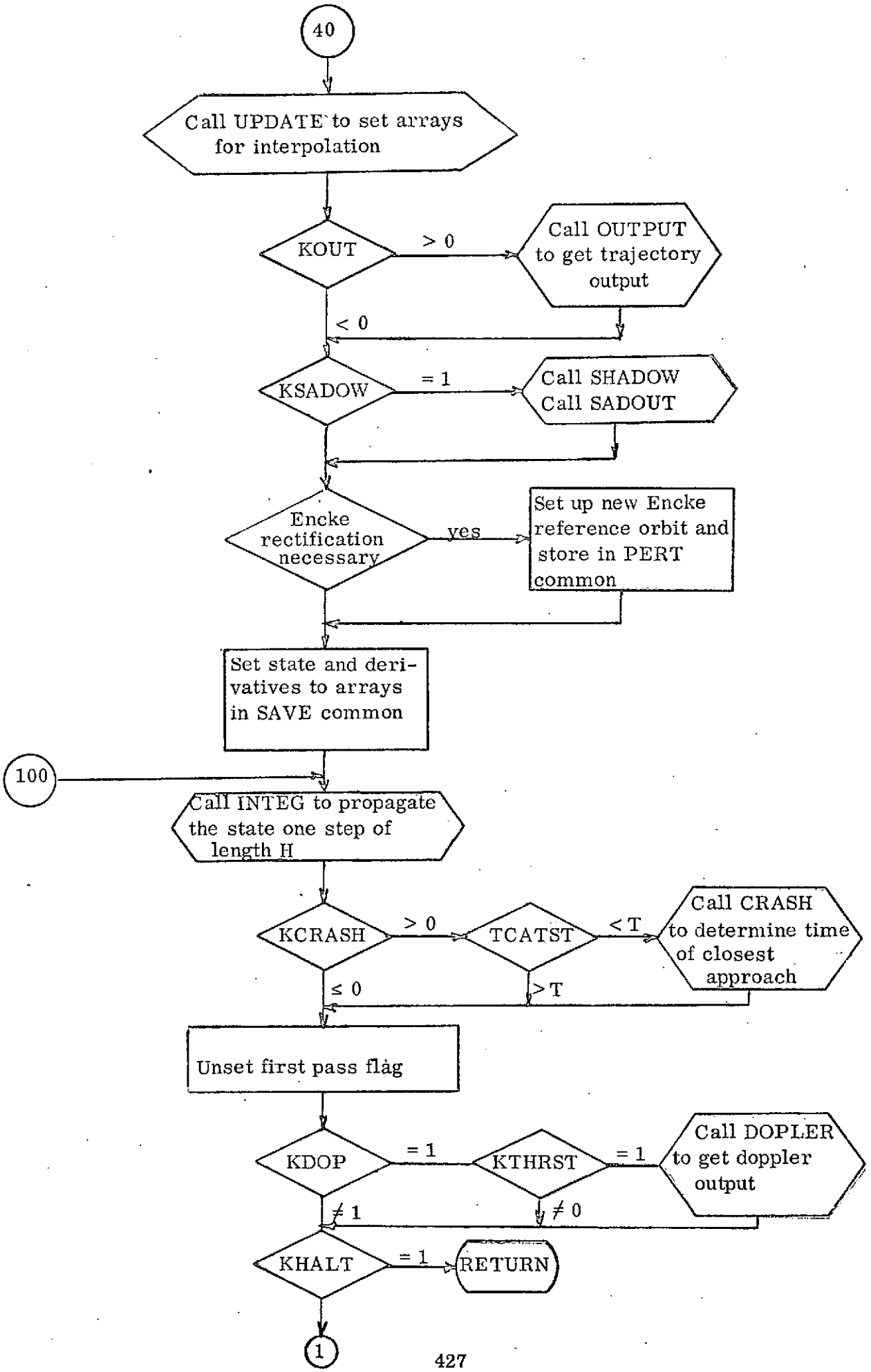
The time of closest approach to the target planet is determined by an iteration using subroutines CRASH and TIMEC. If the time of closest approach, determined by subroutine CRASH, is less than the current time, the state is restored to the last step and the state propagated to the time of closest approach. At this point, the time of closest approach is recalculated by subroutine CLOSE. If the last time of closest approach is within a specified tolerance of the new closest approach time, the iteration is converged and the time of closest approach determined. If the difference of the times is larger than a specified tolerance, the iteration has not converged and the process is repeated. A total of seven iterations are allowed with a conversion tolerance of ten seconds.

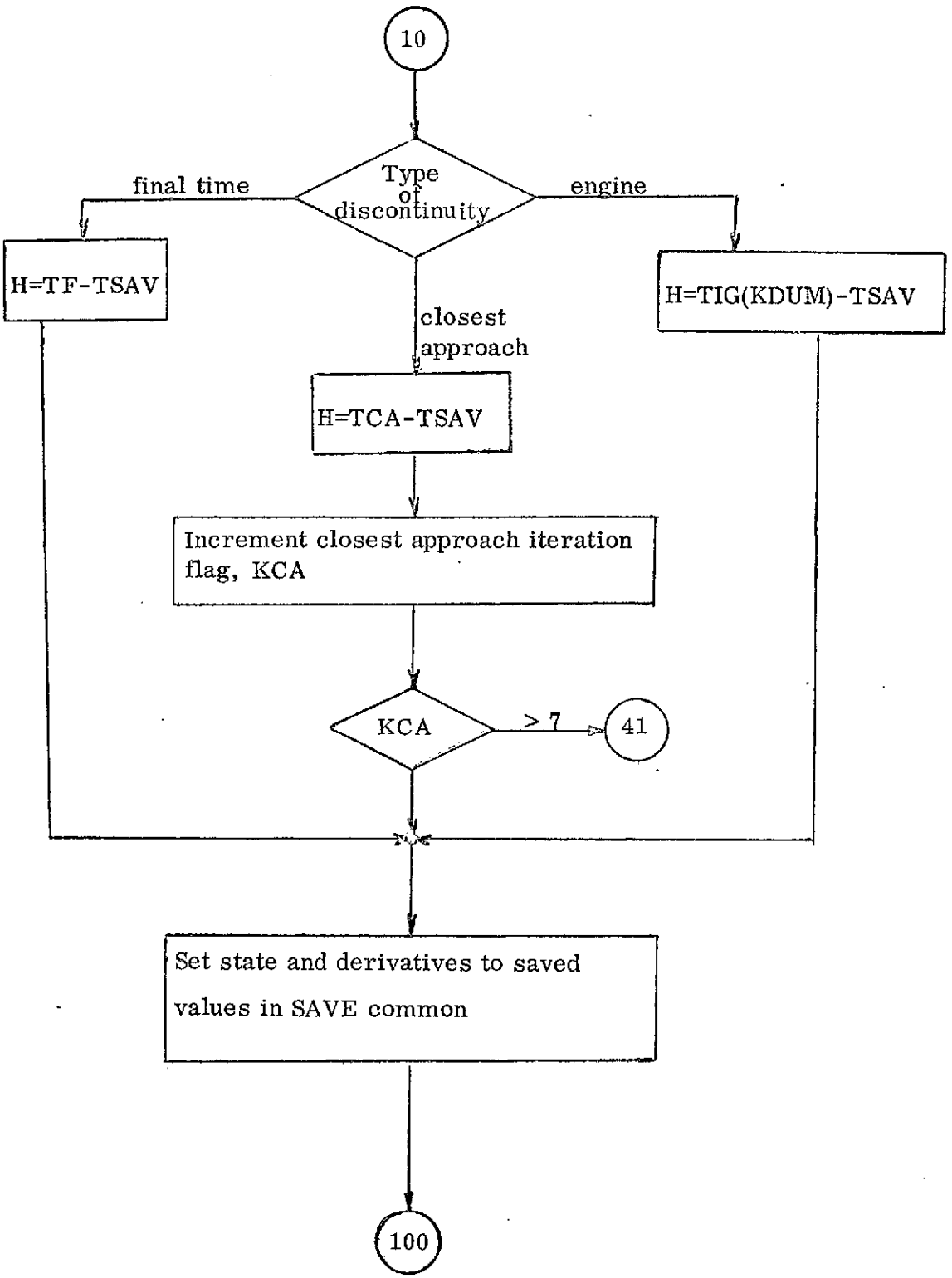
Besides its primary functions, TIMEC also performs many other functions. These consist of:

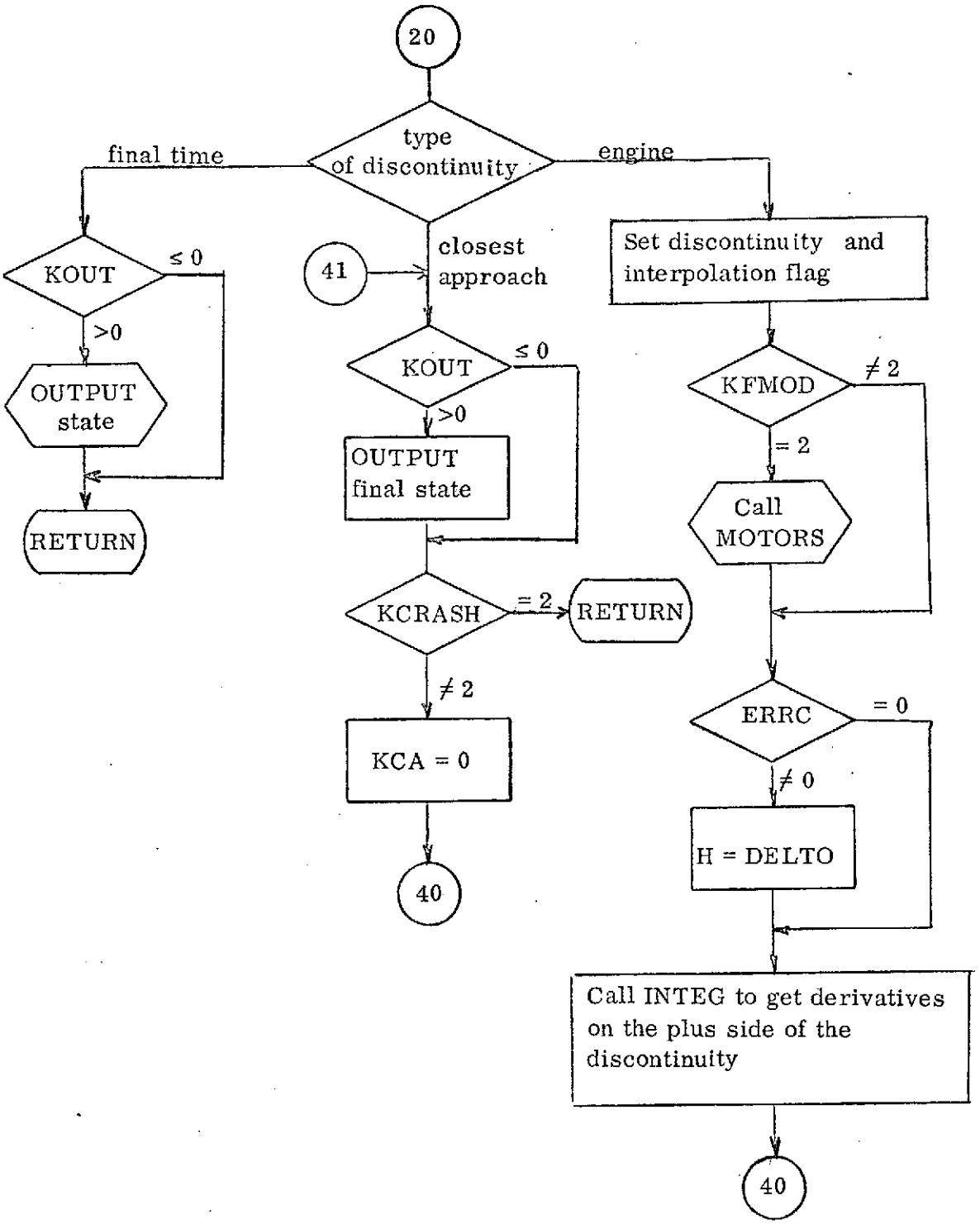
1. Calls to subroutine SHADOW to determine umbral, penumbral occultation times.

2. Determining the trajectory propagation method. If the method changes, logic is included to change the integration variables to the new set of integration variables.
3. Calls to UPDATE to set up arrays for interpolation.
4. Calls to OUT1 for trajectory output.
5. Logic to rectify the reference orbit when propagating the state using Encke.
6. Calls to subroutine CLOSE to determine the central planet.
7. Calls to subroutine DOPLER to get doppler output.
8. Calls to subroutine MOTORS to simulate an engine burn by impulsive velocity.


ENTER
Initialize compute interval closest approach iteration flag, discontinuity flag and interpolation flag
1
ERRC=0
yes
Determine compute interval from input table
no
T > TMETH(1)
yes
Determine new trajectory propagation method and determine new integration variables
no
discontinuity time
Test for a discontinuity time
discontinuity time passed
20
10
no discontinuity time
40


40
Call UPDATE to set arrays for interpolation
KOUT
> 0
Call OUTPUT to get trajectory output
< 0
KSADOW
= 1
Call SHADOW
Call SADOUT
Encke rectification necessary
yes
Set up new Encke reference orbit and store in PERT common
Set state and derivatives to arrays in SAVE common
100
Call INTEG to propagate the state one step of length H
KCRASH
> 0
TCATST
< T
Call CRASH to determine time of closest approach
≤ 0
> T
Unset first pass flag
KDOP
= 1
KTHRST
= 1
Call DOPLER to get doppler output
≠ 1
≠ 0
KHALT
= 1
RETURN
1


10
Type of discontinuity
final time
engine
H=TF-TSAV
H=TIG(KDUM)-TSAV
closest approach
H=TCA-TSAV
Increment closest approach iteration flag, KCA
KCA
> 7
41
Set state and derivatives to saved values in SAVE common
100


20
type of discontinuity
final time
closest approach
engine
KOUT
≤ 0
>0
OUTPUT state
RETURN
41
KOUT
≤ 0
>0
OUTPUT final state
KCRASH
= 2
RETURN
≠ 2
KCA = 0
40
Set discontinuity and interpolation flag
KFMOD
≠ 2
= 2
Call MOTORS
ERRC
= 0
≠ 0
H = DELTO
Call INTEG to get derivatives on the plus side of the discontinuity
40

SUBROUTINE TOBODY

Calling Sequence: CALL TOBODY (JC)

Purpose: This subroutine flies along a Keplerian conic an increment of time, and determines the cartesian state at the end of the step.

Common Blocks Required: CONST, DUM, STATE

Subroutines Required: ORBIT, TRMN

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	DT	1	DUM(10)	Input time step along conic
O	ELM	12	STATE(14)	Osculating orbital elements along with sines and cosines of i, ω, Ω
I	GM	12	CONST(5)	Gravitational constants
I	JC	1	CALLING OPERAND	Central planet number
I/O	X	6	STATE(1)	Position and velocity vectors of the state at the beginning of the step and, on output, these vectors at the end of the step

Description:

First, the orbital elements are determined from the initial position and velocity vectors using subroutine ORBIT. Next the mean motion is determined from

$$PV = GM(JC) / SEM^3$$

where

GM(JC) is the gravitational constant of the central planet, and

SEM is the semi-major axis

The initial mean anomaly, AMO, is determined from the initial true anomaly using

subroutine TRMN. Next, the mean anomaly at the end of the time step, DT, is determined from

$$AM = AMO + PV\ (DT)$$

The true anomaly at the end of the time step is determined from the mean anomaly using subroutine TRMN. Finally, the position and velocity at the end of the time step is determined using subroutine ORBIT using the final true anomaly in the orbital elements.

SUBROUTINE TRIM

Calling Sequence: CALL TRIM (ELM, DELV)

Purpose: TRIM calculates the trim velocity to circularize at a desired radius and correct inclination

Common Blocks Required: CONST, INPUT

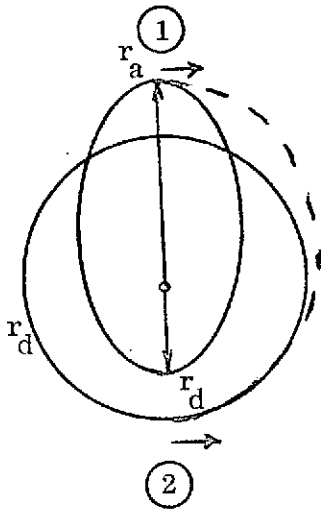
Subroutines Required: None

Input / Output

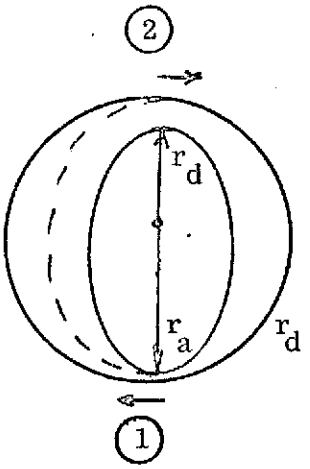
I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	ELM	6	Call List	Pre-trim orbital elements (use-ELM(1)=p ELM(2)-e, ELM(4)=w, ELM(5) = i
O	DELV	1	Call List	Trim velocity (km/sec)
I	RAD	1	CONST(1)	Radians-to-degrees conversion factor
I	GM	12	CONST(5)	Gravitational constant array(km^3/sec^2)
I	RD	1	INPUT(444)	Desired orbit radius (km)
I	VD	1	INPUT(445)	Circular velocity at RD (km/sec)
I	CID	1	INPUT(446)	Desired inclination (deg)
I	TRINC	1	INPUT(449)	Inclination tolerance (deg)
I	JT	1	INPUT(1031)	Central body number

Method:

Once the retro motor has fired, the spacecraft is in lunar orbit. Assuming this orbit to be elliptical with energy, c_{3e} and angular momentum h_e, we will proceed to define the trim sequence and the trim fuel cost. The in-plane trim is accomplished with a two-impulse Hohmann transfer to circular orbit at the desired radius, r_d.



Case 1: $r_a > r_d$



Case 2: $r_a < r_d$

The periapsis radius, r_p, and apoapsis radius, r_a, can be defined from c_{3e} and h_e.

$$p = \frac{h^2}{\mu}$$ semi-latus rectum

$$e = \sqrt{1 + \frac{c_{3e} p}{\mu}}$$ eccentricity

$$r_p = \frac{p}{1 + e}$$ periapsis radius

$$r_a = \frac{p}{1 - e}$$ apoapsis radius

$$a = \frac{r_p + r_a}{2} = - \frac{\mu}{c_{3e}}$$ semi-major axis

Case 1: $r_a > r_d$ The semi-major axis, a_i, and energy, c_{3i} of the intermediate orbit must be

$$a_i = \frac{r_a + r_d}{2}$$

and $$c_{3i} = - \frac{\mu}{a_i} = - \frac{2\mu}{r_a + r_d}$$

The velocity at r_a before the first impulse is

$$v_a^- = \sqrt{c_{3e} + \frac{2\mu}{r_a}} = \sqrt{\frac{2\mu r_p}{r_a(r_a + r_p)}}$$

and after the impulse,

$$v_a^+ = \sqrt{c_{3i} + \frac{2\mu}{r_a}} = \sqrt{\frac{2\mu r_d}{r_a(r_a + r_d)}}$$

so that the first maneuver requires an impulsive velocity

$$\delta v_1 = \sqrt{\frac{2\mu}{r_a}} \left| \sqrt{\frac{r_d}{r_a + r_d}} - \sqrt{\frac{r_p}{r_a + r_p}} \right| .$$

The velocity on the intermediate orbit at r_d is

$$v_d^- = \sqrt{c_{3i} + \frac{2\mu}{r_d}} = \sqrt{\frac{2\mu r_a}{r_d(r_a + r_d)}}$$

while the desired circular velocity is

$$v_d^+ = \sqrt{\frac{\mu}{r_d}}$$

so that the second impulse is

$$\delta v_2 = \sqrt{\frac{\mu}{r_d}} \left| \sqrt{\frac{2 r_a}{r_a + r_d}} - 1 \right|$$

and the total in-plane trim velocity is

$$\delta v = \delta v_1 + \delta v_2$$

Case 2: $r_a < r_d$. In this case, it is slightly cheaper to transfer first from r_p to r_d. By a derivation similar to that for Case 1, we obtain

$$\delta v_1 = \sqrt{\frac{2\mu}{r_p}} \left| \sqrt{\frac{r_d}{r_p + r_d}} - \sqrt{\frac{r_a}{r_a + r_p}} \right|$$

$$\delta v_2 = \sqrt{\frac{\mu}{r_d}} \left| \sqrt{\frac{2 r_p}{r_p + r_d}} - 1 \right|$$

It should be noted that the two cases can be computed with the same equations by interchanging roles between r_a and r_p.

Inclination Change

The inclination is changed by a third impulse executed at the node of the orbit on the equator. The impulse is applied on the intermediate orbit of the in-plane adjustment at the nodal crossing of larger radius. Although this strategy is not usually optimal, it is convenient: it leaves the node invariant, it always permits a solution, and it separates the in-plane and out-of-plane trim costs. Let the larger nodal radius vector (of either the pre-trim or intermediate orbit, actually) be R and its corresponding velocity be V^-. If the inclination is to be changed by δi, the rotated post-impulse velocity, V^+ will be

$$V^+ = \cos \delta i \; V^- + (1-\cos \delta i) \hat{R} \cdot V^- \hat{R} + \sin \delta i \, \hat{R} \times V^-$$

and

$$\delta V_3 = V^+ - V^-$$

$$= (\cos \delta i - 1) \; V^- + (1-\cos \delta i) \hat{R} \cdot V^- \hat{R} + \sin \delta i \; \hat{R} \times V^-$$

$$\delta v_3^2 = \left[(1-\cos \delta i)^2 + \sin^2 \delta i \right] \left[V^- \cdot V^- - (\hat{R} \cdot V^-)^2 \right]$$

$$= 2 (1 - \cos \delta i) \left| \hat{R} \times V^- \right|^2$$

$$= 4 \sin^2 \left(\frac{\delta i}{2} \right) \left| \hat{R} \times V^- \right|^2$$

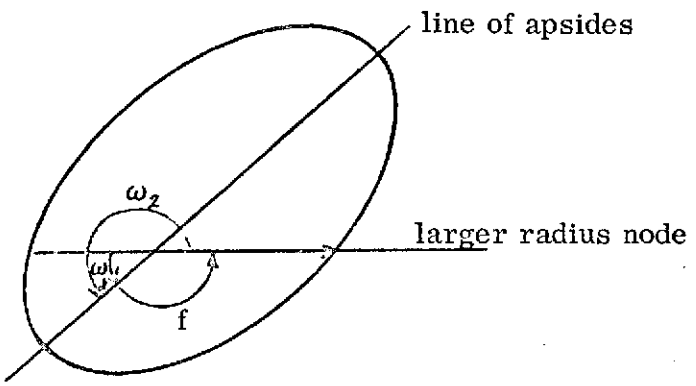
$$\delta v_3 = 2 \sin \left(\frac{\delta i}{2} \right) \left| \hat{R} \times V^- \right| = 2 \sin \left(\frac{|\delta i|}{2} \right) v_{\text{tangential}}$$

To roughly estimate the cost of correcting inclination, assume $\left|\hat{R} \times \bar{V}\right| = \sqrt{\frac{\mu}{r_d}}$

and $\sin \frac{\delta i}{2} = \frac{\delta i}{2}$.

$$\delta v \simeq \sqrt{\frac{\mu}{r_d}}\ \delta i = 22.94 \left(\frac{m}{sec} / \text{deg}\right) \delta i^o$$

Determination of the larger-radius node is somewhat tricky and deserves some explanation.



The argument of pericenter, ω, is either ω_1 or ω_2 as shown in the sketch, depending on which nodal sense is the ascending one. The true anomaly of the larger-radius node is computed according to the following scheme.

$$-\pi/2 \leq \omega \leq \pi/2 \quad : \quad f = \pi - \omega$$

$$\frac{\pi}{2} < \omega < \frac{3\pi}{2} \quad : \quad f = 2\pi - \omega$$

In either case, $\cos f = -\left|\cos \omega\right|$.

$$r = \frac{p}{1 + e \cos f}$$

and the tangential velocity (needed in the plane-change formula) is

$$v_{\text{tangential}} = \frac{h}{r} = \frac{\sqrt{\mu p}}{r} \quad .$$

If r is smaller than RD, the plane-change is executed at a node of the circularized orbit. An inclination tolerance, TRINC, offers a band within which no inclination correction is made. The correction is made only over to the tolerance band.

SUBROUTINE TRIM2

Calling Argument: CALL TRIM2 (ELMI, F, DV, LOPT)

Purpose: This subroutine determines the optimum two impulse 180^{o} transfer between two orbits

Common blocks required: CNTRL, CONST, INPUT, PIT

Subroutines required: MVTRN, ORBIT

Inputs/Outputs

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	CINF	1	INPUT(446)	Desired inclination of final orbit
O	DV	1	Calling Operand	Magnitude of the first and second trim maneuvers
O	DX	3, 2	PIT(3)	First and second trim maneuvers in same system as ELMI
I	ELMI	12	Calling Operand	Orbital elements of initial orbit
I	JC	1	CNTRL(7)	Central planet number
I	LOPT	1	Calling Operand	Flag used to calculate trim velocity components. Non-zero to calculate components
I	F	1	Calling Operand	True anomaly of the first maneuver on the initial orbit
I	RFINAL	1	INPUT(444)	Desired final orbit radius
I	VFINAL	1	INPUT(445)	Desired final orbit velocity
O	XS	6	PIT(12)	State before first trim in same coordinate system as ELMI

Reference:

F. T. Sun, "Analytic Solution for Optimal Two-Impulse 180^{o} Transfer Between Noncoplanar Orbits and the Optimal Orientation of the Transfer Plane, AIAA Journal, Vol 7, No. 10, 1969.

Theory:

This subroutine determines the optimal two-impulse 180° transfer between noncoplanar orbits using the method described in the reference. Since a 180° transfer is specified, the first impulse must be applied at the intersection of the initial and final orbit planes. Thus, the angle between the initial and final orbit planes and the position on the initial orbit where the maneuver is made can be obtained from the spherical trigonometric relationships, see Figure 1.

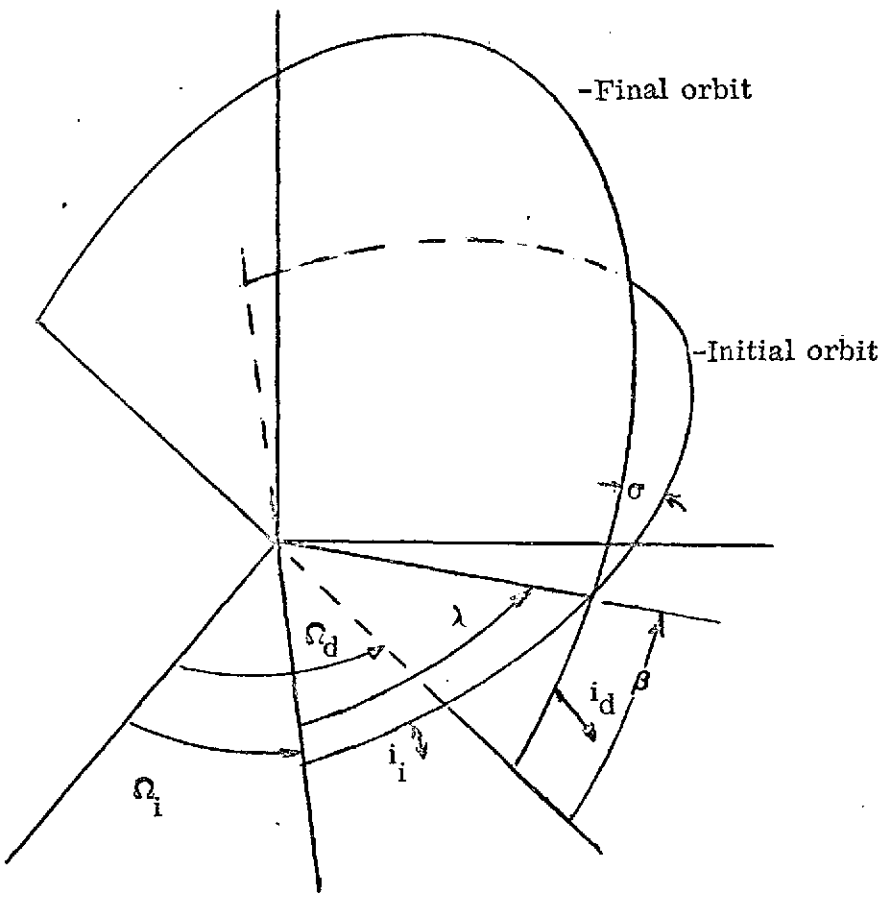


Figure 1

The angle from the reference plane to the common line of nodes in the initial orbit, λ, can be determined from the input initial true anomaly, f, and the argument of the ascending node of the initial orbit as,

$$\lambda = f + \omega \tag{1}$$

Then, the angle between the two orbit planes, σ , can be determined from,

$$\sin \beta = \sin i_i \ \sin i_d \ / \sin \lambda \qquad (2)$$

$$\tan \frac{\sigma}{2} = \sin \left(\frac{\lambda - \beta}{2}\right) \overline{\sin \left(\frac{\lambda + \beta}{2}\right)} \tan \left(\frac{i_d - i_i}{2}\right)$$

where i is the inclination

β is the angle from the reference plane to the common line of nodes in the final orbit plane.

and the subscripts i and d refer to the initial and desired orbits, respectively.

The radius and velocity components can be determined from standard orbital relationships at the initial true anomaly, f.

The orientation of the transfer plane with respect to the initial and final orbit planes is described in Figure 2.

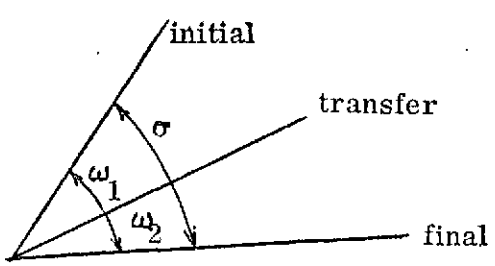


Figure 2

The angles ω_1 and ω_2 describe the orientation of the transfer plane with respect to the initial and final orbit planes, respectively.

If the inclination of the transfer plane with respect to the initial plane is specified, then the optimal velocity can now be determined using Sun's equation 10.

$$\Delta V = \sqrt{\frac{\mu}{r_1}} \left\{ V_{r1}^2 + \left[\left(V_{T1}^2 - 2 \sqrt{\frac{2n}{n+1}} \ V_{T1} \cos \omega_1 + \frac{2n}{n+1} \right)^{1/2} + \left(1 - 2 \sqrt{\frac{2}{n+1}} \cos \omega_2 + \frac{n}{n+1} \right)^{1/2} \right]^2 \right\}^{1/2} \qquad (3)$$

where μ is the gravitational constant, r_i is the radius at $f = \lambda - \omega_d$, V_{R1} and V_{T1} are the radial and transversal velocity components on the initial orbit relative to local circular velocity ($V/\sqrt{\mu/r}$), and n is r_d/r_i.

The equation above is somewhat simplified from Sun's equation since the final orbit is circular. Thus,

$$V_{R2} = 0$$
$$V_{T2} = 1. \tag{4}$$

If the velocity of each trim maneuver is desired, then

$$\frac{\Delta V_1}{\Delta V} = 1 + \frac{1}{\sqrt{n}} \left[\frac{1 - 2\sqrt{\frac{2}{n+1}} \cos\omega_2 + \frac{2}{n+1}}{V_{T1}^2 - 2\sqrt{\frac{2n}{n+1}} V_{T1} \cos\omega_1 + \frac{2n}{n+1}} \right]^{1/2}$$

and

$$\frac{\Delta V_2}{\Delta V} = 1 + \sqrt{n} \left[\frac{V_{T1}^2 - 2\sqrt{\frac{2n}{n+1}} V_{T1} \cos\omega_1 + \frac{2n}{n+1}}{1 - 2\sqrt{\frac{2}{n+1}} \cos\omega_2 + \frac{2}{n+1}} \right]^{1/2} \tag{5}$$

where ΔV_1 and ΔV_2 denote the magnitudes of the first and second trim maneuvers.

The direction these impulses are applied can be determined by noting the following relationships,

$$\Delta V_{R1} = V_{RT1} - V_{R1}$$
$$\Delta V_{N1} = -V_{T1} \sin\omega_1 \tag{6}$$
$$\Delta V_{T1} = V_{TT1} - V_{T1} \cos\omega_1$$

The second trim is determined in a similar manner as

$$\Delta V_{R2} = V_{R2} - V_{RT2}$$
$$\Delta V_{N2} = -V_{T2} \sin\omega_2 \tag{7}$$
$$\Delta V_{T2} = V_{T2} \cos\omega_2 - V_{TT2}$$

In the above equations the components of velocity are defined as follows,

$V_{R1,2}$	radial component of velocity of the initial or final orbit.
$V_{T1,2}$	Transversal component of velocity of the initial or final orbit.
$V_{TT1,2}$	Transversal component of velocity of the initial or final orbit written in the transfer plane.
$V_{RT1,2}$	Radial component of velocity of the transfer orbit at initial and final orbit crossings.

All components of velocity, except the radial components of the transfer orbit in equations 6 and 7, are fixed by specifying the initial and final orbits. The radial component of velocity is determined from the condition that the total trim velocity is to be minimized. The total trim velocity is

$$\Delta V = \Delta V_1 + \Delta V_2 = \sqrt{\Delta V_{T1}^2 + \Delta V_N^2 + (V_{RT1} - V_{R1})^2} + \sqrt{\Delta V_{T2}^2 + \Delta V_{N2}^2 (V_{R2} - V_{RT2})^2} \quad (8)$$

Also,

$$V_{RT1} = -V_{RT2}$$
$$V_{R2} = 0 \quad (9)$$

since a 180^o transfer is specified and the final orbit is to be circular. Now, the partial derivative of the trim velocity with respect to V_{RT1} can be written as,

$$\frac{\partial \Delta V}{\partial V_{RT1}} = \frac{V_{RT1} - V_{R1}}{\Delta V_1} + \frac{V_{RT1}}{\Delta V_2} = 0$$

or

$$V_{RT1} = \frac{\Delta V_2 \; V_{R1}}{\Delta V_1} \quad (10)$$

The components of the trim velocity obtained from equations 6, 7, and 10 describe the trim velocity with respect to the transfer plane. The trim velocity vector in the same coordinate system of the initial orbit is obtained through a three Euler angle rotation pictured in the figure below.

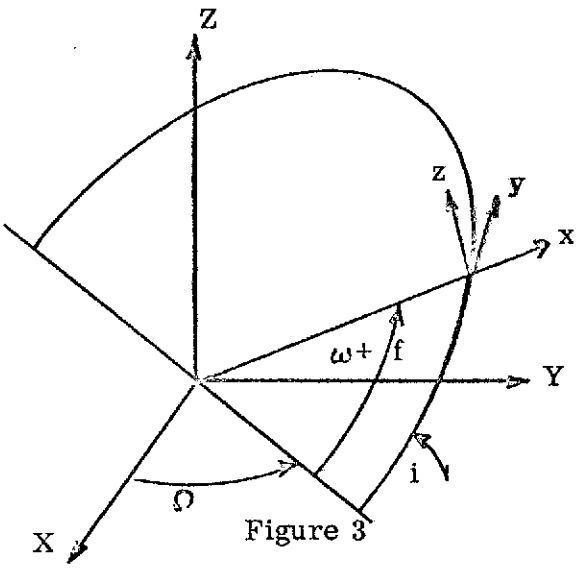


Figure 3

In the figure above, x corresponds to the radial direction, y to the transversal direction, and z to the normal direction.

The angular elements of the reference orbit define the Euler angles. Thus the transformation from the x, y, z system to the X, Y, Z system is

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix} = \begin{bmatrix} \cos\psi\cos\Omega - \cos i \sin\Omega\sin\psi & -\sin\psi\cos\Omega - \cos i \sin\Omega\cos\psi & \sin i \sin\Omega \\ \cos\psi\sin\Omega + \cos i \cos\Omega\sin\psi & -\sin\psi\sin\Omega + \cos i \cos\Omega\cos\psi & -\sin i \cos\Omega \\ \sin i \cos\psi & \sin i \sin\psi & \cos i \end{bmatrix} \begin{Bmatrix} x \\ y \\ z \end{Bmatrix} \tag{11}$$

where $\psi = f + \omega$

The above equation is used to transform the trim velocity components from an orbit plane coordinate system to the system of the reference orbit.

Optimum Inclination of the Transfer Plane

The condition for the optimal orientation of the transfer plane is expressed by

$$\frac{\sin\omega_2}{\sin\omega_1} = -n\left(\frac{1 - 2\rho_2\cos\omega_2 + \rho_2^2}{1 - 2\rho_1\cos\omega_1 + \rho_1^2}\right)^{1/2} \tag{12}$$

$$\text{where} \quad \rho_1 = \left(\frac{2\mu\, r_d/r_1}{r_d + r_1}\right)^{1/2} \frac{1}{V_{T1}}$$

$$\rho_2 = \left(\frac{2\mu\, r_1/r_d}{r_d + r_1}\right)^{1/2}$$

Equation 12 along with the condition

$$\omega_2 = \sigma + \omega_1 \tag{13}$$

yields a set of equations which can be solved for ω_1 or ω_2 to yield the optimum orientation of the transfer plane. The solutions to equation 12 resulted in a sixth order polynomial in sin ω. The equation was solved numerically in order to avoid the cumbersome task of solving a sixth order equation. A Newton-Raphson procedure was employed to determine the solution to equation 13. Sun, in the reference, states that the solution is unique. Thus, the task of finding multiple solutions with the Newton-Raphson method is not required.

Description:

The initial orbital elements and argument of the ascending node of the final orbit is brought into the subroutine via the argument list. The angular components of the initial and final orbits are used in equations 1 and 2 to determine the position on the initial orbit where the trim maneuver will take place. Next, the transversal and radial components of velocity and other quantities required for Sun's equations are determined. The Newton-Raphson technique described by equations 12 and 13 are used to determine the inclination of the transfer orbit with respect to the initial orbit.

The magnitude of the velocity is determined from Sun's equation 10. If only the magnitude of the velocity is required, the subroutine returns. This option is executed through the LOPT flag brought in through the argument list. If the flag is equal to zero, only the velocity magnitude is calculated. Otherwise, the components of the trim velocity in the same coordinate system as the initial orbital elements are determined.

Equations 6, 7 and 10 are used to determine the components of the trim velocity in the transfer plane. The first trim velocity is rotated from the transfer plane to the initial

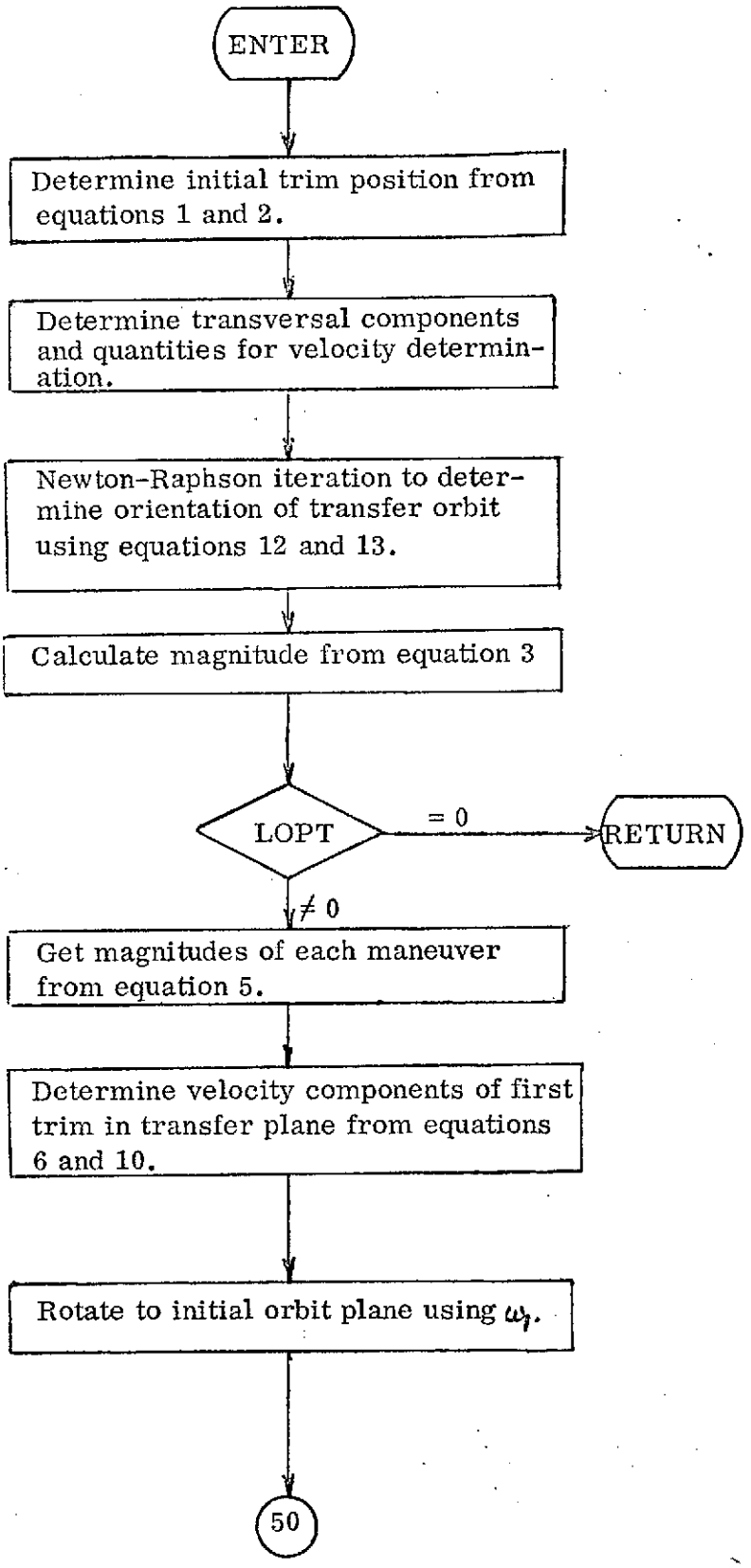
orbit plane by a rotation through an angle ω_1 about the radial velocity component. This rotation is expressed by,

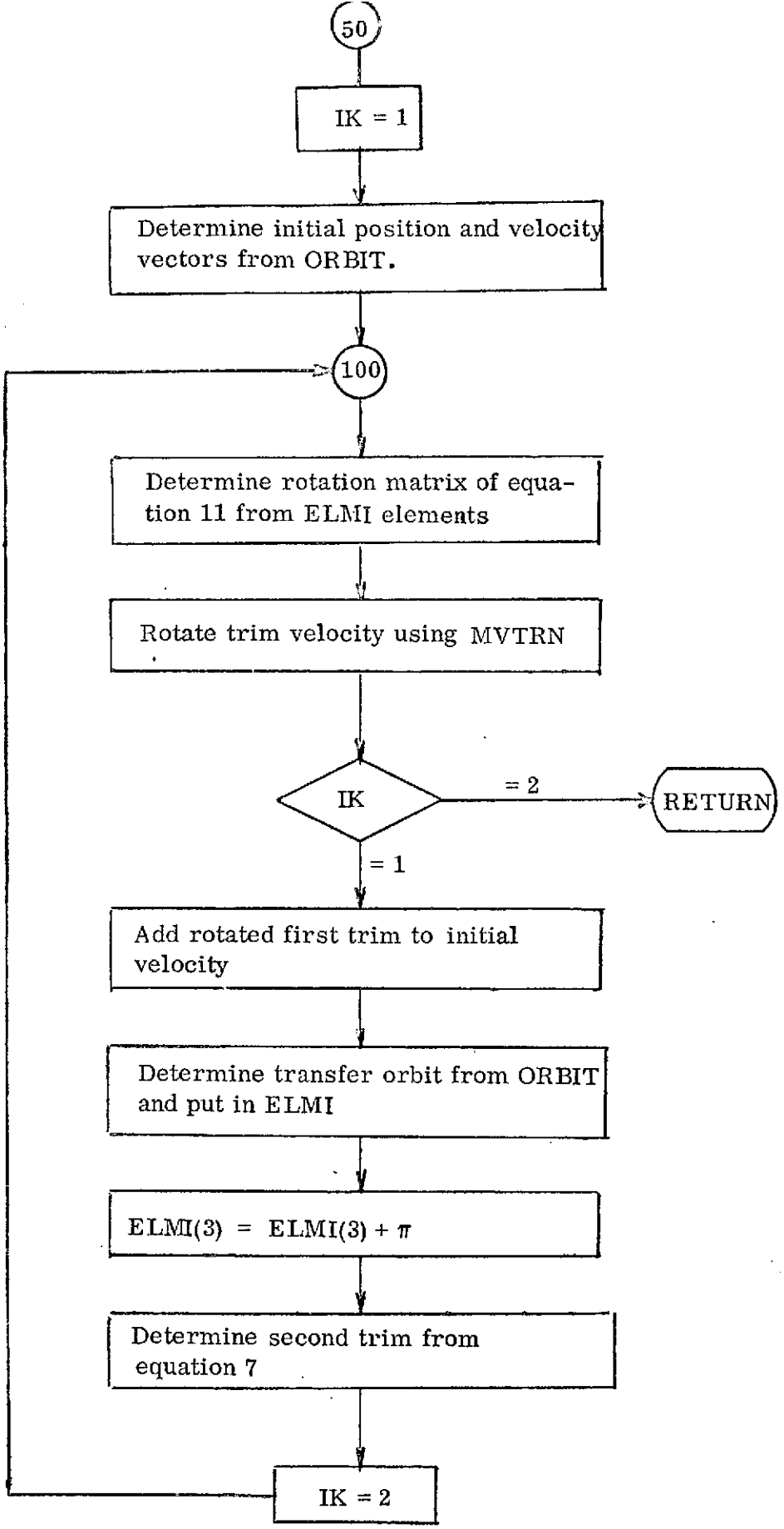
$$\begin{Bmatrix} DX \\ DY \\ DZ \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\omega_1 & \sin\omega_1 \\ 0 & -\sin\omega_1 & \cos\omega_1 \end{bmatrix} \begin{Bmatrix} \Delta V_R \\ \Delta V_T \\ \Delta V_N \end{Bmatrix} \tag{14}$$

Next, the rotation defined by equation 11 is employed to obtain the first trim velocity in the desired system. The angular quantities in this rotation are obtained from the initial orbit.

The trim velocity from equation 14 is added to the initial state to determine the transfer orbit initial conditions. These conditions are used in ORBIT to determine the elements of the transfer orbit. The normal, radial, and transversal components from equation 7 are used directly in equation 11 to determine the second trim velocity in the desired system. The elements used in equation 11 are from the transfer orbit for the second trim. The logic to rotate the trim components are accomplished in a loop. The IK flag is used to determine the current maneuver.

SUBROUTINE TRIM2




50
IK = 1
Determine initial position and velocity vectors from ORBIT.
100
Determine rotation matrix of equation 11 from ELMI elements
Rotate trim velocity using MVTRN
IK
= 2
RETURN
= 1
Add rotated first trim to initial velocity
Determine transfer orbit from ORBIT and put in ELMI
ELMI(3) = ELMI(3) + π
Determine second trim from equation 7
IK = 2

FUNCTION TRMN

Calling Sequence: Y = TRMN (J, Q, E)

Purpose: This function calculates the mean anomaly from the true anomaly and vice-versa.

Common Blocks Required: CONST

Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	E	1	Calling Operand	Eccentricity
I	J	1	Calling Operand	Flag to determine the computation =1 Calculate mean from true anomaly =-1 Calculate true from mean anomaly
I	PI	1	CONST(2)	pi, π
I	PI2	1	CONST(3)	twice pi
I	Q	1	Calling Operand	Input mean or true anomaly

Theory:

I. Calculation of the mean anomaly from the true anomaly. The equations to determine the mean anomaly are dependent on the eccentricity, e. If e is less than 1, the mean anomaly is obtained as follows,

$$E = \cos^{-1}\left[(\cos f + e) / (1+e \cos f)\right] \quad (1)$$

where E is the eccentric anomaly and

f is the input true anomaly.

Then the mean anomaly, M, is obtained from Kepler's equation, as

$$M = E - e \sin E \quad (2)$$

When hyperbolic motion is encountered $e > 1$, the mean anomaly is obtained by first calculating the auxiliary variable, F, from

$$\tanh\left(\frac{F}{2}\right) = \sqrt{\frac{e-1}{e+1}} \tan\left(\frac{f}{2}\right) \tag{3}$$

Then the mean anomaly is determined from

$$M = e \sinh F - F \tag{4}$$

II. Calculating the true anomaly given the mean anomaly. Kepler's equation, equation (2), is solved for the eccentric anomaly given the mean anomaly. This is a transcendental equation and an iterative procedure is required to solve for E.

If a first-order Taylor series is used to represent Kepler's equation, it becomes

$$M = \phi(E_0) + \phi'(E_0)\,\Delta E_0 \tag{5}$$

where $\phi(E_0) = E_0 - e \sin E_0$

$$\phi'(E_0) = 1 - e \cos E_0$$

If we assume that

$$\phi(E_0) = M_0,$$

then

$$M - M_0 = \Delta M = \phi'(E_0)\,\Delta E_0,$$

or

$$\Delta E_0 = \frac{M - M_0}{1 + e \cos E_0}. \tag{6}$$

A corrective term is added to the derivative of ϕ in order to help convergence. Thus the change in eccentric anomaly is given by

$$\Delta E_0 = \frac{M - M_0}{1 + e \cos E_0 + .01e \cos^3 E_0} \tag{7}$$

The iterative process uses equations (2) and (7) as follows:

1. The initial mean anomaly is input, M_0.
2. A guess is made for the corresponding eccentric anomaly, E_0.
3. Equation (2) is used to determine the mean anomaly corresponding to E_0.
4. The change in eccentric anomaly, ΔE_0, is determined from equation (7).
5. A new eccentric anomaly is determined from $E = E + \Delta E$.
6. Equation (2) is used to obtain a new mean anomaly M.
7. If $(M - M_0)$ is sufficiently small, the solution has converged. If not, steps 4 through 7 are repeated until it is small.

The true anomaly is calculated from the eccentric anomaly using

$$\sin f = \frac{a}{r} \sqrt{1-e^2} \quad \sin E \tag{8}$$

and

$$\cos f = \frac{a}{r} \quad \cos E - e$$

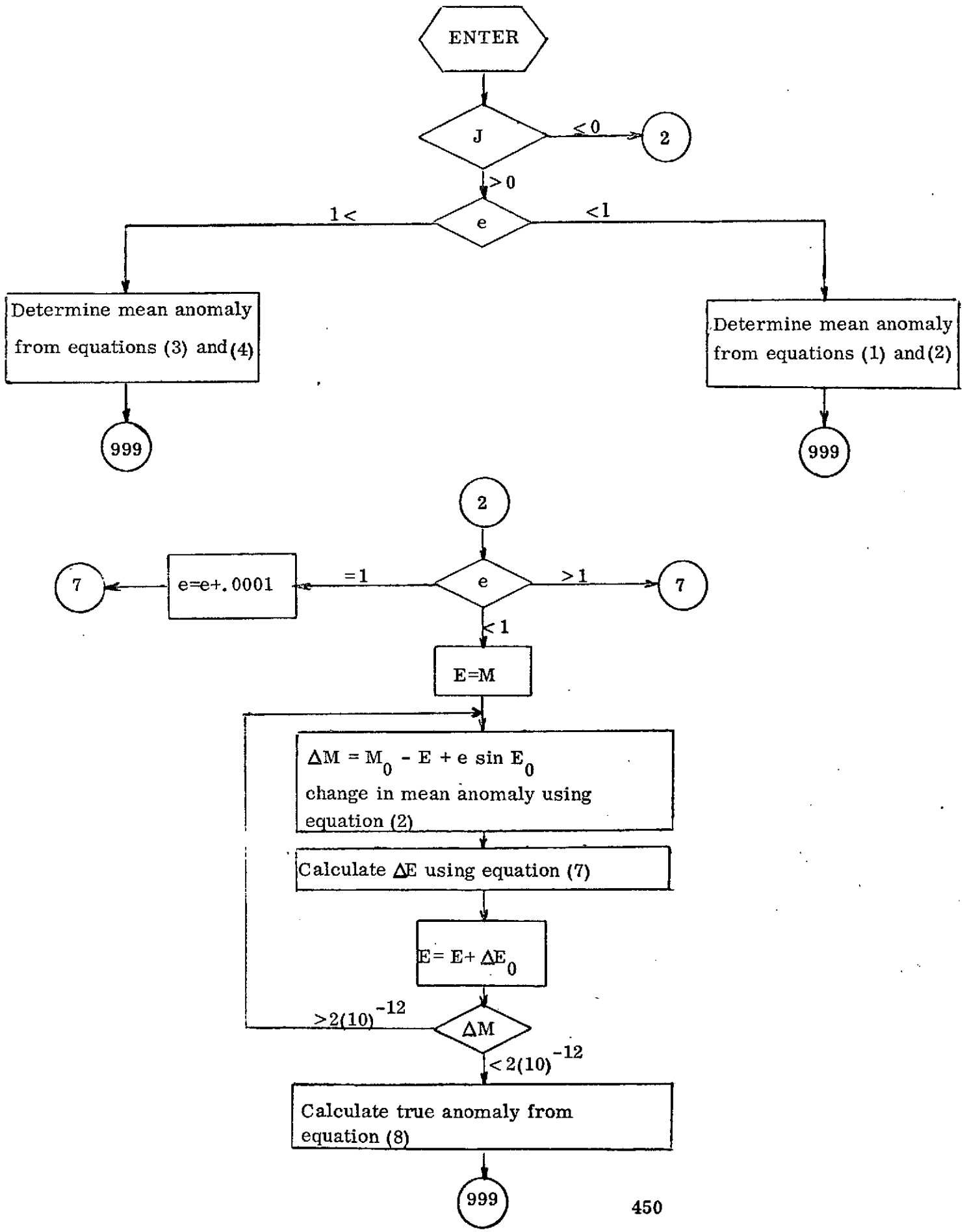
The solution of Kepler's equation for hyperbolic orbits is essentially the same except that equation (2) is replaced by equation (4) and equation (7) is replaced by

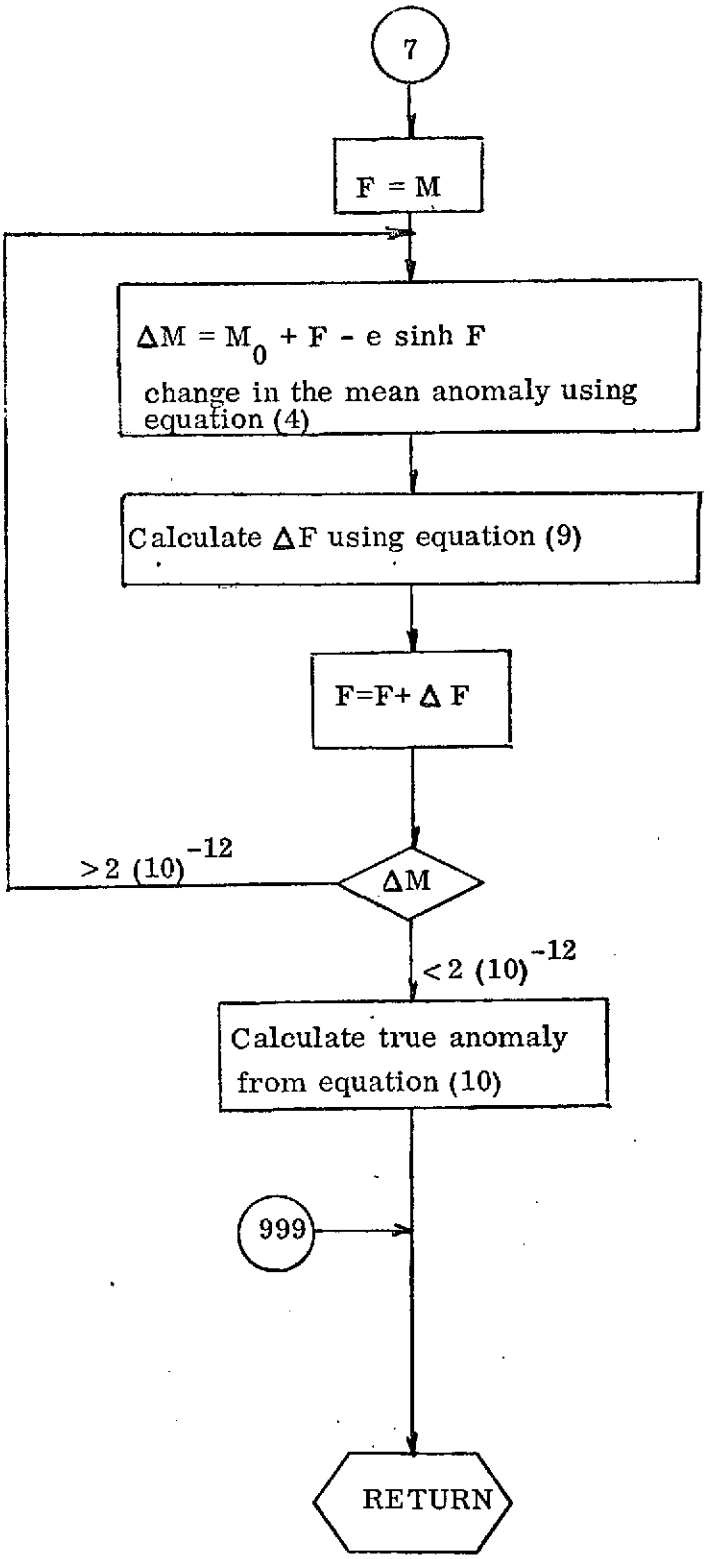
$$\Delta F = - \frac{M - M_0}{1 + e \cosh F_0} \tag{9}$$

The true anomaly is calculated from the auxiliary variable, F, using the inverse tangent

$$\tan f = \frac{\sqrt{e^2 - 1} \quad \sinh F}{\cos F - e} \tag{10}$$

FUNCTION TRMN




7
F = M
ΔM = M0 + F - e sinh F
change in the mean anomaly using equation (4)
Calculate ΔF using equation (9)
F=F+ Δ F
>2 (10)-12
ΔM
<2 (10)-12
Calculate true anomaly from equation (10)
999
RETURN

SUBROUTINE TUBE1

Available from NASA Goddard Space Flight Center.

SUBROUTINE TWELVE

Calling Sequence: CALL TWELVE (T)

Purpose: This subroutine integrates a set of simultaneous differential equations using a twelfth-order predictor-corrector type method.

Common Blocks Required: CONST, CNTRL, INPUT, INTVAR, PERT

Subroutines Required: DVMAG, EQNS, RKSEVN

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	H	1	INTVAR(14)	Compute interval
I	JC	1	CNTRL(7)	Central planet number
I	KDIS	1	CNTRL(5)	Discontinuity flag
I	METH	1	INPUT(13)	Trajectory propagation indicator
I	RATES	6	INTVAR(8)	Derivatives of the dependent variables
I/O	T	1	CALLING OPERAND	Initial time on input and time on end of step or return
I	TOL	1	INPUT(8)	Corrector convergence tolerance
I	X	1	INTVAR(1)	Independent variable
I/O	Y	6	INTVAR(2)	Dependent variables

Theory:

A set of simultaneous differential equations are numerically integrated over the time step, H. The scheme employed is a predictor-corrector scheme of summed ordinate form. Two sets of predictor and corrector equations are contained in this routine. When second derivatives of the quantities to be integrated are available, a Störmer predictor and a Cowell corrector are used. An Adams-Bashforth predictor and an Adams-Moulton corrector are employed when only the first derivatives are available. A predictor-corrector numerical integration scheme determines the state at the end of the step as follows:

1. The predictor equation is used to extrapolate for the state at the end of the step (T + H) using the back derivatives (or second derivatives) at times T, T-H, T-2H, etc.

2. The derivatives are determined at time T+H using the state determined in 1.

3. The corrector equation is used to determine the state at T + H using the back values of the derivatives at T, T-H, T-2H, etc. and the derivative at T + H from 2.

4. The state obtained from the corrector equation is compared to the predictor state. If the difference is less than TOL, the solution is converged.

5. If the solution has not converged, the derivative is determined using the corrector state at T+H and flow is transferred to Step 3. A total of 3 iterations are allowed.

The predictor equations are

$$Y^{p}_{K+1} = h^2 \left[\Pi S_k + \sum_{i=1}^{10} \alpha_i \ddot{Y}_{11-i} \right] \qquad \text{Störmer} \qquad (1)$$

$$Y^{p}_{K+1} = h \left[{}^{I}S_k + \sum_{i=1}^{10} \alpha^{*}_i \dot{Y}_{11-i} \right] \qquad \text{Adams-Bashforth} \qquad (1)$$

where α_i and α^{*}_i are constants described in Table I.

and ${}^{I}S_k$ and ${}^{II}S_k$ are the first and second sums defined by

$${}^{I}S_{k+1} = {}^{I}S_k + \ddot{Y}_{k+1}$$

$${}^{II}S_{k+1} = {}^{II}S_k + {}^{I}S_{k+1}$$

or if first derivatives only are available

$${}^{I}S_{k+1} = {}^{I}S_k + \dot{Y}_{k+1}$$

The Cowell and Adams-Moulton corrector equations used are,

$$Y^{c}_{K+1} = h^2 \left[{}^{II}S_k + \sum_{i=1}^{10} \beta_i \ddot{Y}_{11-i} \right] \qquad \text{Cowell}$$

$$Y^{c}_{K+1} = h \left[{}^{I}S_k + \sum_{i=1}^{10} \beta^{*}_i \dot{Y}_{11-i} \right] \qquad \text{Adams-Moulton} \qquad (2)$$

where β_i and β^{*}_i are constants described in Table I

The table of back derivatives is established using a seventh-order ten-cycle Runge-Kutta integration scheme when the discontinuity flag is set to 1. This flag is set on the first step, or at engine ignition or burnout times. The initial first and second sums are obtained from the corrector equation as follows,

$${}^{II}S_9 = \frac{Y_{10}}{h^2} - \sum_{i=1}^{10} \beta_i \ddot{Y}_{11-i} \qquad (3)$$

$$I_{S_9} = \frac{\dot{Y}_{10}}{h} - \sum_{i=1}^{10} \beta_i^* \ddot{Y}_{11-i}$$

$$I_{S_{10}} = I_{S_9} + \ddot{Y}_{10} \qquad (3)$$

$$\Pi_{S_{10}} + \Pi_{S_9} + I_{S_9} .$$

If the first derivative is used, the sum is obtained from

$$I_{S_9} + \frac{Y_{10}}{h} - \sum_{i=1}^{10} \beta_i^* \dot{Y}_{11-i}$$

$$I_{S_{10}} = I_{S_9} + \dot{Y}_{10}$$

Only the first sum is needed.

Description:

Logic to numerically integrate either first or second derivative equations is contained in the routine. The METH flag is tested to determine which set of equations to use. A series of flags are internally set according to the value of METH. These flags are used to determine the size of DO LOOPS and values of subscripts. The purpose of the flags are

FLAG	1st Deriv.	2nd Deriv.	DESCRIPTION
NK	6	3	Number of quantities integrated
NO	0	3	Used to pick out the first derivatives when integrating second derivative equations.
IN	2	1	Subscript to identify the integration coefficients.
IS	1	2	Subscript to identify the first or second sum.

KDIS is tested to determine if it is necessary to proceed into the start-up logic. This logic is used to set up the back values of the derivatives in the XDD array. The values are obtained by integrating the set of equations with the seventh-order Runge-Kutta scheme. The compute interval is halved and the integration accomplished with RKSEVN. The derivatives at the end of every second step are loaded into the XDD array. After the start-up integration, the first and second sums are initialized using equations (3).

Next, the predicted value is obtained from equation (1). Subroutine EQNS is used to get the derivatives of the function using the predicted state.

The corrector equation is employed to obtain the corrected value and the test for convergence is made. The integration is complete if convergence is obtained. If not, the derivation is obtained with the corrected state and the corrector equation employed again. Three iterations are allowed.

The derivative array, XDD, and the sums, S, are updated after convergence is achieved.

Note: This subroutine must be used with a constant, fixed step compute interval as there are no provisions for modifying the back value table.

SUBROUTINE TWELVE

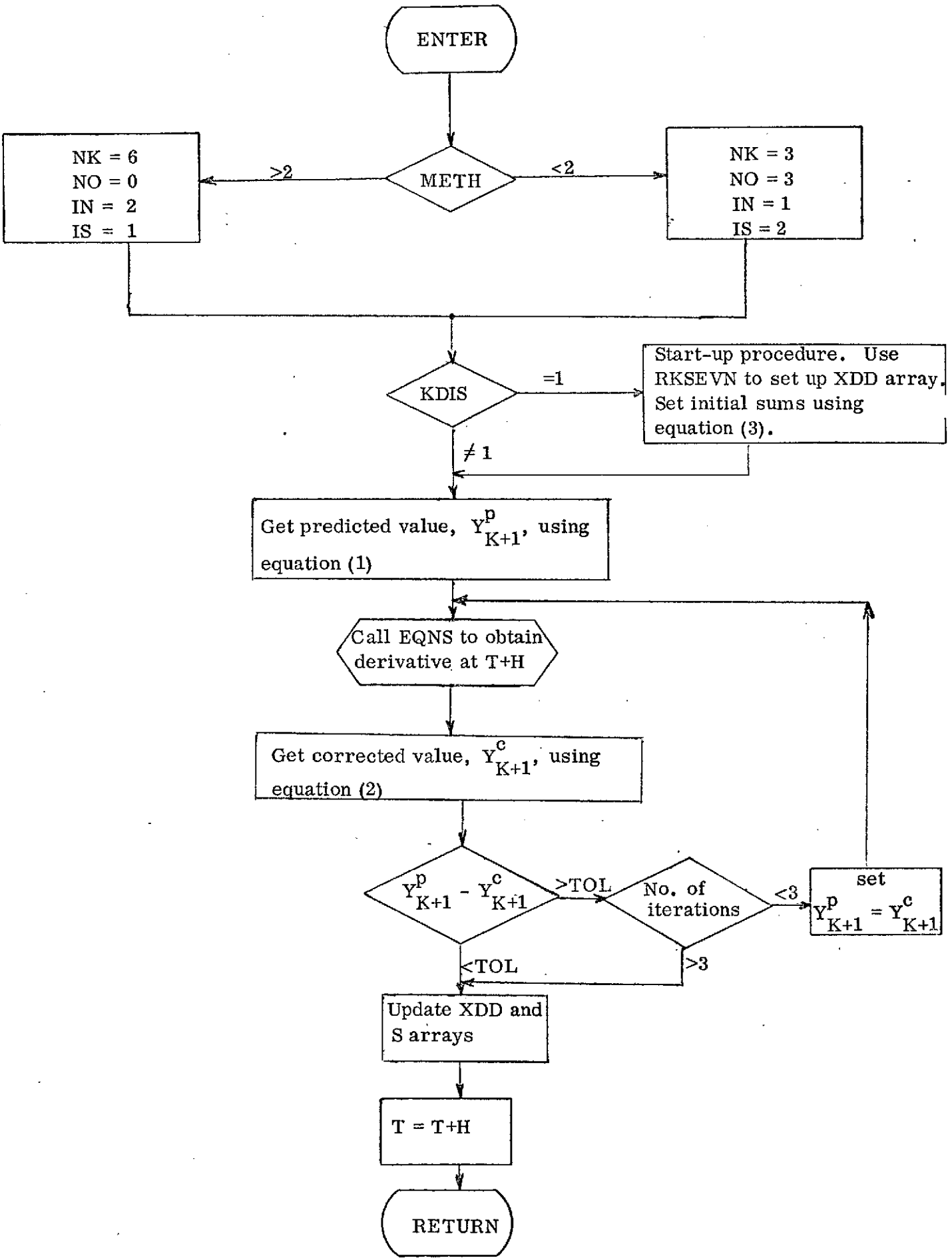


TABLE I

INTERPOLATION COEFFICIENTS

Störmer Predictor	Adams-Bashforth Predictor	Cowell Corrector	Adams-Moulton Corrector
α	α^*	β	β^*
.709333000140291806	3.4519884004562823	.0592405641233766233	.280189596443936721
-2.94977592767957351	-13.8168372652617444	.116927358906525573	.650092436016915182
7.56536563552188551	35.4336533489658489	-.283950542127625460	-1.20830542528459194
-12.9585332742103575	-60.8353967251883918	.456997940716690716	1.81090177569344235
15.3441157607824274	72.1837392401194484	-.518014808301266634	-1.99558147196168029
-12.6715074479918229	-59.7076813146344396	.415493601691513357	1.57596093624739458
7.19372036335578001	33.9395391414141414	-.230988982082732082	-.867866061407728073
-2.68438193943402276	-12.6774970438512105	.0848526685505852171	.316787568141734808
.594237726972101971	2.80868181442400192	-.0185565538820747153	-.0689652038740580406
-.0592405641233766233	-.280189596443936721	.00183208573833573833	.00678584998463470685

SUBROUTINE TWOPIT

Calling Sequence: CALL TWOPIT(ELM, DUT)

Purpose: This subroutine is used in conjunction with TRIM 2 to determine the minimum two-impulse, plane-change post-injection trim maneuver.

Common Blocks Required: CONST

Subroutines Required: TRIM2

Input/Output

I/O	SYMBOLIC NAME	PROGRAM DIMENSION	COMMON BLOCK	DEFINITION
O	DVT	1	Calling Argument	Total velocity requirement
I	ELM	6	Calling Argument	Pre-trim orbit
I	PI	1	CONST(2)	

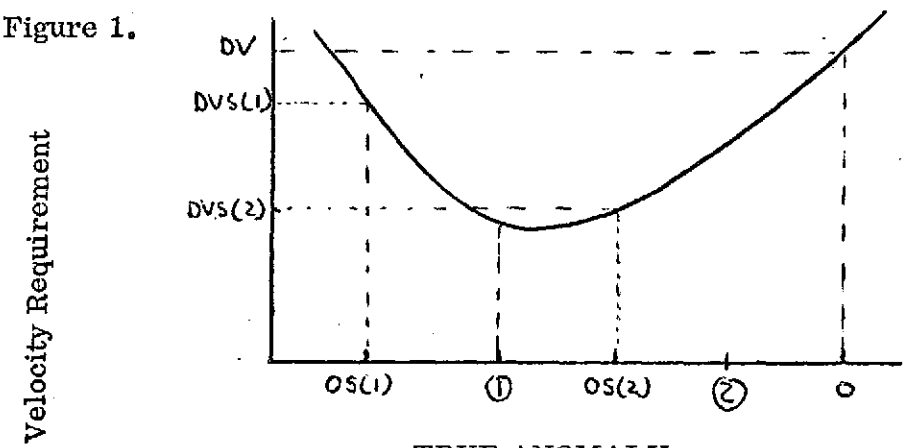
Description:

This subroutine is used in conjunction with TRIM2 to determine the optimum place on the initial orbit to initiate the two-impulse plane-change maneuver. Subroutine TRIM2 determines the optimum maneuver between two orbits. However, the position on the initial orbit must be specified. This subroutine uses a half-interval search to find the position on the initial orbit which results in the minimum trim requirements.

The search is accomplished by stepping the true anomaly around the initial orbit. On each step, subroutine TRIM 2 is called to determine the velocity requirement.

As TWOPIT steps along the true anomaly, the true anomalies on the two previous steps and their corresponding velocity requirements are saved in the OS and DVS arrays. The DVS array and the current velocity, DVT, is used to determine if a minimum velocity requirement has been passed. If a minimum is not passed the logic continues to step true anomaly. The subroutine goes into its half-interval search phase when the minimum criterion is satisfied. The LPASS flag is non-zero when in the half-interval phase.

It is known that the minimum occurs between the current true anomaly and the true anomaly stored in OS(1) when the half-interval search phase is begun, see Figure 1.



The true anomaly interval is halved and added onto the value in OS(1). Then the velocity requirement at this point is determined (① in the figure). If the velocity at ① is less than OS(2), the minimum must occur between OS(1) and OS(2). If this condition is true, the interval is halved again and the process repeated between the points OS(1) and ① . If the velocity at ① is greater than OS(2), the minimum lies between ① and ϕ. The velocity requirement is then determined at ② . If the velocity at ② is less than DVS(2), the minimum must occur between OS(2) and ϕ. Then,

$$OS(1) = OS(2)$$
$$OS(2) = ①$$
$$DVS(1) = DVS(2)$$
$$DVS(2) = DV\ (①)$$

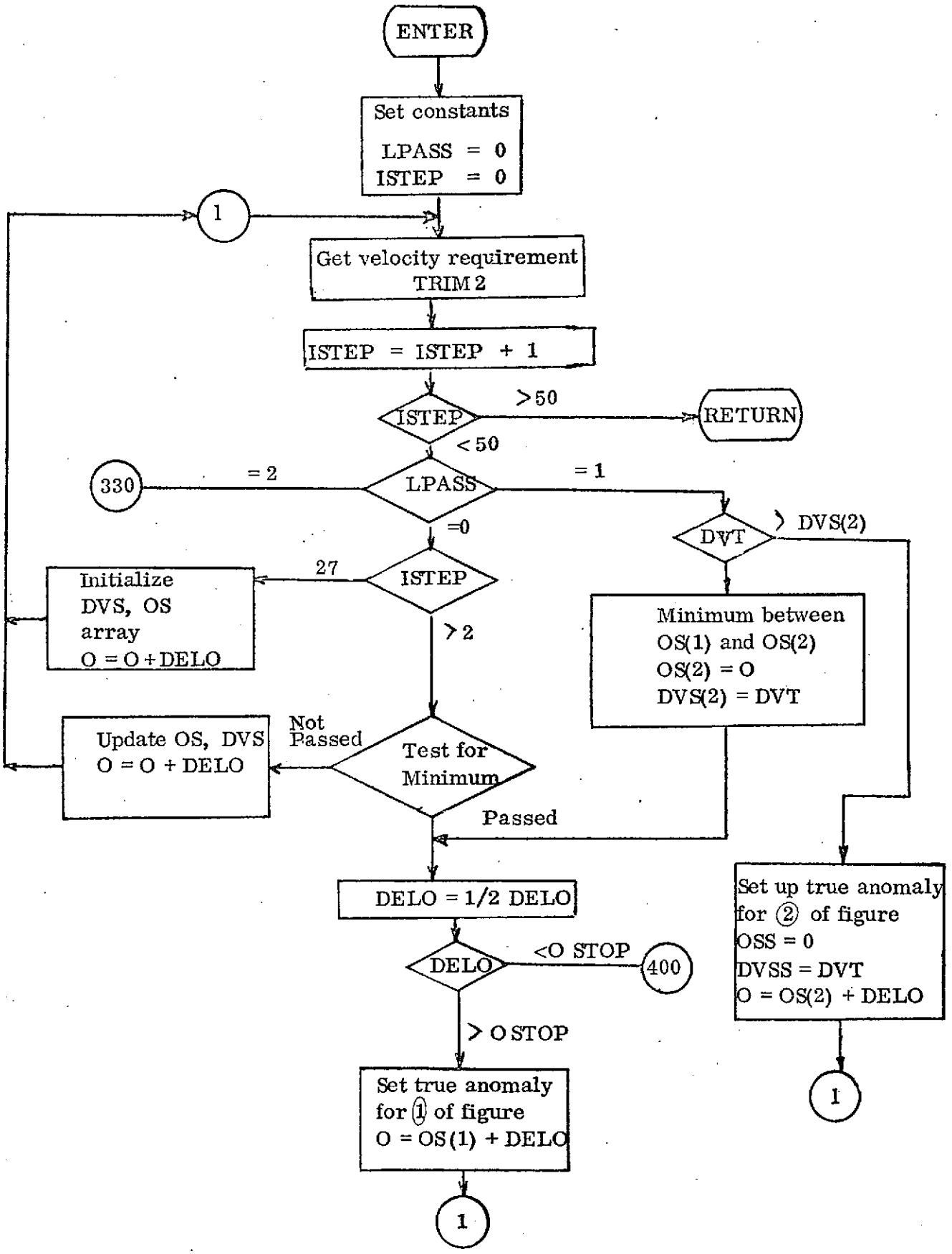
and flow transfers back to the beginning of the half-interval search. If the velocity at 2 is greater than DVS(2), the minimum occurs between ① and ② . For this case

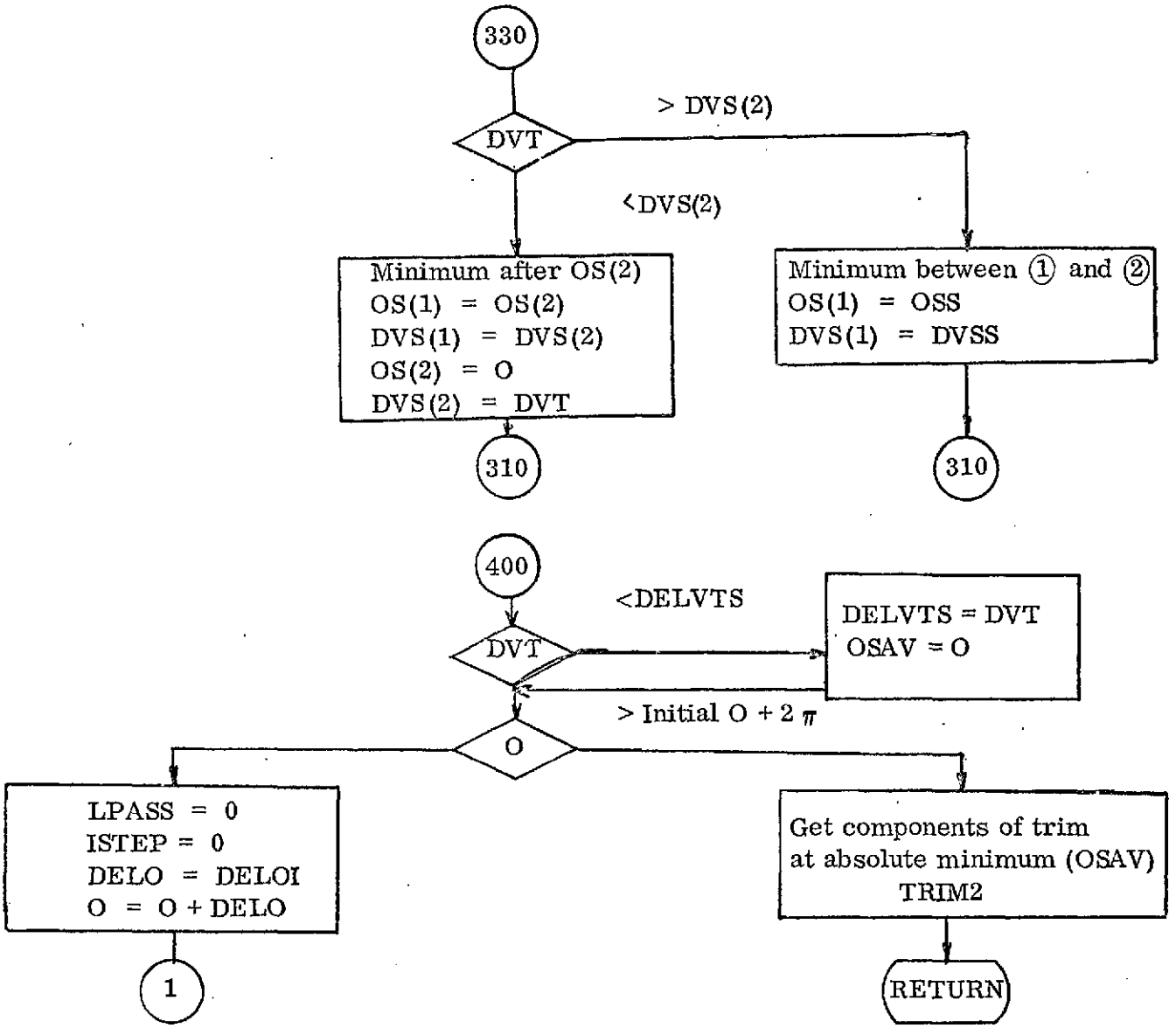
$$OS(1) = ①$$
$$OS(2) = OS(2)$$
$$DVS(1) = DV(①)$$
$$OS(2) = DVS(2)$$

The flow returns to the beginning of the search.

The search is terminated when the interval in true anomaly has been halved enough times to cause it to become less than a stopping value. Logic is included to search the entire orbit for minimums. This is required because there are many local minimums. The subroutine terminates when the final true anomaly is more than 2π greater than the initial value. Prior to termination, TRIM2 is called with the value of true anomaly corresponding to the absolute minimum and the input flag set to obtain components of the trim along with its magnitude.

SUBROUTINE TWOPIT




330
DVT
> DVS(2)
<DVS(2)
Minimum after OS(2)
OS(1) = OS(2)
DVS(1) = DVS(2)
OS(2) = O
DVS(2) = DVT
310
Minimum between ① and ②
OS(1) = OSS
DVS(1) = DVSS
310
400
DVT
<DELVTS
DELVTS = DVT
OSAV = O
> Initial O + 2 π
O
LPASS = 0
ISTEP = 0
DELO = DELOI
O = O + DELO
1
Get components of trim
at absolute minimum (OSAV)
TRIM2
RETURN

SUBROUTINE UPDATE

Calling Sequence: CALL UPDATE

Purpose: This subroutine sets up the array of back derivatives used for interpolation in subroutine INTERP.

Common Blocks Used: INPUT, INTER, INTVAR, STATE

Subroutines Used: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
O	ACL	6, 10	INTER(71)	Array of back derivatives of the integration variables
I	INT	1	INTER(131)	Counter used to indicate the current value in arrays
O	POS	6, 10	INTER(11)	Array of back values of the integration variables
I	RATES	6	INTVAR(8)	Current derivatives of the integration variables
I	T	1	STATE(10)	Current time
O	X	10	INTER(1)	Times of the back values
I	Y	6	INTVAR(2)	Current integration variables

Description:

This subroutine sets up arrays which contain the back values of the integration variables, derivatives of the integration variables and times at which the back values are stored in the arrays. The INT flag is used as an indicator to determine which back value is current. The back value arrays can hold up to ten back values. Only five of these back values are used. The longer table decreases the number of times components in the table must be moved. If INT is larger than 10, the last five back values of the table are set in the first five slots and INT reset to 5. If INT is zero, the back values arrays are restored to zero.

SUBROUTINE VIEW

Calling Sequence: CALL VIEW (X, Y, J, A)

Purpose: To calculate the lighting characteristics of a planet as seen by the spacecraft.

Common Blocks Required: CONST

Subroutines Required: None

Inputs / Outputs

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DESCRIPTION
O	A	1	CALLING ARGUMENT	Percent of planet lighted
I	J	1	CALLING ARGUMENT	Planet number
I	RE	12	CONST(17)	Equatorial radius of the planets
I	X	3	CALLING ARGUMENT	Vector from spacecraft to the planet.
I	Y	3	CALLING ARGUMENT	Vector from Sun to planet.

Theory:

Let θ be the s/c-planet-sun-angle

x, y, z system centered at the planet with

x toward the s/c, z normal to the s/c-planet-sun plane, and y in the plane

x', y', z' system same as x, y, z, except rotated about z an angle θ such that x' is toward the Sun.

Then the equation for the terminator circle formed on the planet by the Sun is described as

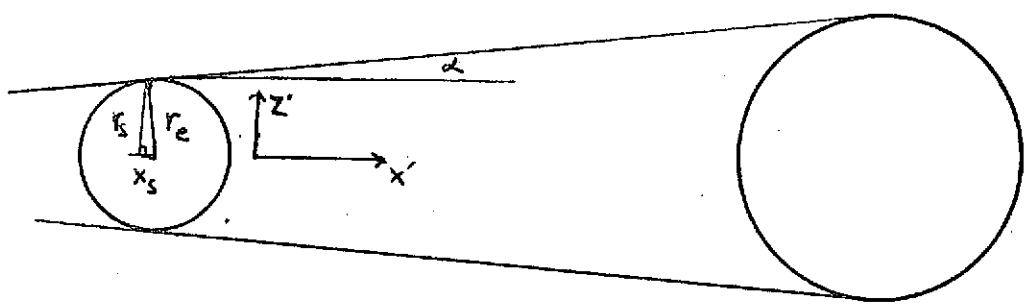
$$y'^2 + z'^2 = r_s^2 \qquad (1)$$

$$x' = -x_s$$

where

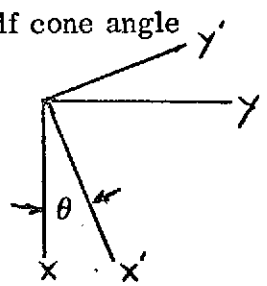
$$r_s = r_e \cos a$$

$$x_s = r_e \sin a$$



and r_e is the planet's radius

a is the half cone angle



The rotation from the x, y, z system to x', y', z' is obtained from

$$\begin{Bmatrix} x' \\ y' \\ z' \end{Bmatrix} \quad \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x \\ y \\ z \end{Bmatrix}$$

Equation (1) written in the s/c system becomes

$$x\cos\theta + y\sin\theta = -x_s$$

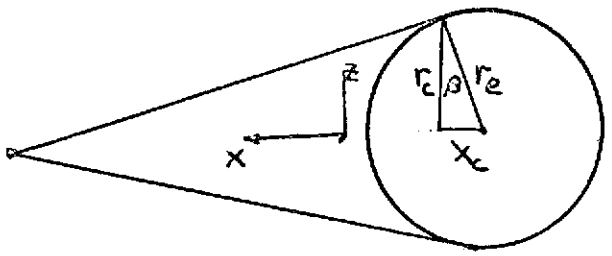
$$(-x\sin\theta + y\cos\theta)^2 + z^2 = r_s^2 \qquad (2)$$

The s/c also sees a circle on the planet described by

$$y^2 + z^2 = r_c^2$$

$$x = x_c \qquad (3)$$

where



$$x_c = r_e \sin\beta$$

$$r_c = r_e \cos\beta$$

$$\beta = \sin^{-1}\frac{r_e}{r}$$

The terminator ellipse will intersect the viewing circle where the plane $x = x_c$ intersects the ellipse, or

$$x_c \cos\theta + y \sin\theta = -x_s$$

and (4)

$$(-x_c \sin\theta + y \cos\theta)^2 + z^2 = r_s^2$$

thus the intersection points are

$$y_1 = \frac{-x_s - x_c \cos\theta}{\sin\theta}$$

$$z_{1,2} = \pm\sqrt{r_s^2 - y_c'^2} \qquad (5)$$

where

$$y_c' = -x_c \sin\theta + \left[\frac{-x_s - x_c \cos\theta}{\sin\theta}\right] \cos\theta$$

$$= \frac{-x_c \sin^2\theta - x_s \cos\theta - x_c \cos^2\theta}{\sin\theta} = \frac{-x_c - x_s \cos}{\sin\theta}$$

The sun ellipse can be rewritten as

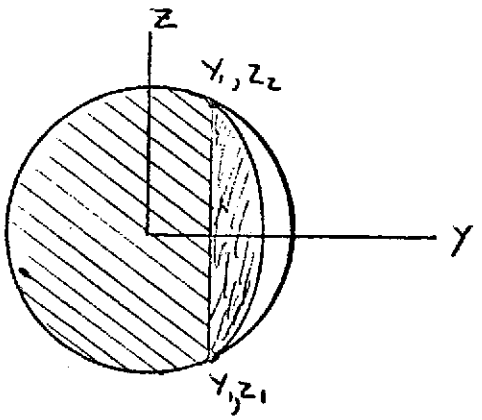
$$x = \frac{-x_s + y \sin\theta}{\cos\theta}$$

$$\left[\left(\frac{-x_s + y \sin\theta}{\cos\theta}\right) \sin\theta + y \cos\theta\right]^2 + z^2 = r_s^2$$

$$\left[\frac{-x_s \sin\theta + y \sin^2\theta + y \cos^2\theta}{\cos\theta}\right]^2 + z^2 = r_s^2 \qquad (6)$$

$$\left(\frac{-x_s \sin\theta + y}{\cos\theta} \right)^2 + z^2 = r_s^2$$

The shaded area of the disk in the figure below is obtained from



$$A = \int_{z_1}^{z_2} (y - y_1)\, dz$$

$$= \int \left\{ \cos\theta \sqrt{r_s^2 - z^2} + x_s \sin\theta - y_1 \right\} dz$$

$$= \left[\frac{\cos\theta}{z} \left[z\sqrt{r_s^2 - z^2} + r_s^2 \sin^{-1} \frac{z}{r_s} \right] + z(x_s \sin\theta - y_1) \right]_{z_1}^{z_2} \qquad (7)$$

The area of the rest of the circle (the crossed-hatched area) is determined from

$$\frac{\pi r^2}{z} + \left[y\sqrt{r_c^2 - y_1^2} + r_c^2 \sin^{-1} \frac{y_1}{r_c} \right] \qquad (8)$$

Description:

The areas determined in equations (7) and (8) are determined in a straightforward manner. The initial vectors are input via the argument list while the planet's radius is in CONST common. The area determined by the sum of equations (7) and (8) is divided by the total area to determine the percent lighted. If the spacecraft-planet-Sun angle is greater than 90 degrees, the percent calculated is the darkened area. Thus, the percent is subtracted from 100 to determine the lighted area.

SUBROUTINE VISIB

Calling Sequence: CALL VISIB (TIME, XIN, JC, KEL)

Purpose: VISIB computes tracking station visibility.

Common Blocks Required: CONST, INPUT, OBSIT, PLNET

Subroutines Required: DOT, DVMAG, MVTRN, M50EPM

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	TIME	1	Call List	Time from anchor epoch (sec)
I	XIN	6	Call List	Spacecraft's position vector(EE50, km)
I	JC	1	Call List	Central body of XIN
O	KEL	10	Call List	Visibility or elevation array $\frac{\text{deg}}{10}$
I	RTD	1	CONST(1)	Radians-to-degrees conversion factor
I	RAD	12	CONST(17)	Planetary or lunar radii (km)
I	DJO	1	INPUT(46)	Julian date of anchor epoch (days)
I	OBSL	10	INPUT(410)	Tracker longitude array(deg)
I	XOBS	10,3	OBSIT(21)	Tracker radius vectors (EPM, km)
I	XP	6,12	PLNET(1)	Celestial body states at TIME relative to JC (km, km/sec)

Method:

VISIB is called both for visibility at midcourse time and for visibility at retro ignition. In either case, the spacecraft's position, X, is transformed into Earth's equator and prime meridian coordinates where the trackers' positions, R_t, reside. The slant range vector, $S = X - R_t$, is used to compute the elevation angle, El.

$$El = 90^{o} - \cos^{-1}\left(\frac{S \cdot R_t}{s\, r_t}\right)^{o}$$

The KEL element corresponding to El is the integer part of

$$\frac{El}{10} + 1,$$

if El is positive and zero otherwise. Thus, if the elevation is negative at the I-th tracker, KEL (I) = 0. If elevation is between 0^{o} and 10^{o}, KEL (I) = 1, if it is between 10^{o} and 20^{o}, KEL (I) = 2, etc.

If the central body is not the Earth, the spacecraft's position is first translated to be Earth-centered before computing the slant range vector. In addition, occultation by the central body is checked. If the central body indeed occults the spacecraft from the I-th tracker, KEL (I) = 0, even if the elevation is positive. The criterion for occultation is as follows. Let R be the spacecraft's position relative to JC and r_m be the physical radius of JC. Then if

$$R \cdot S \geq s \sqrt{r^2 - r_m^2}$$

the spacecraft is occulted. Otherwise it is not. If the spacecraft is occulted while the state is Earth-centered (way out past the Moon), it would not be detected because the test would not be made.

FUNCTION VNORM

Calling Sequence: CALL VNORM (X, Y)

Purpose: This function determines the magnitude of a vector and a unit vector in the same direction as the input vector.

Common Blocks Required: None
Subroutines Required: None

Input / Output

I/O	SYMBOLIC NAME	DIMENSION	COMMON BLOCK	DEFINITION
I	X	3	CALLING OPERAND	Input vector
O	Y	3	CALLING OPERAND	Unit vector along X
O	VNORM	1	FUNCTION NAME	Absolute value of X

Description:

The absolute value of the input vector, X, is determined from

$$R = \sqrt{X_1^2 + X_2^2 + X_3^2}$$

where X_i, i=1, 3, are the components of $\bar{X}$. The components of the unit vector $\bar{Y}$ are determined from

$$Y_i = X_i / R \qquad i=1,3$$