CREW PROCEDURES ORBITAL GUIDANCE AND NAVIGATION PROGRAM

NAVIGATION SECTION

1.0 PROGRAM DESCRIPTION

The CPB Guidance and Navigation Program, henceforth referred to as AAP, is a modification set to Program BETELGEUSE. The program library currently exists on tapes 183 or 184 in the Building 35 Tape Library.

AAP provides environment and estimated states for two vehicles, rendezvous sensors and an inertial platform. These vehicles may be in orbit about either the earth or moon.

The navigation filter is a generalization of the Apollo-LM square-root filter to two-vehicle estimation plus estimated sensor biases. A detailed exposition of the theory of this filter may be found in references 1 and 2, and of the covariance advancement method in reference 3. The filter accepts measurements of relative angle, range and/or range-rate, and updates the state of either or both vehicles in an optimal fashion.

The program is structured into a main overlay, and 1st and 2nd primary overlays. The main overlay lists input/output devices, zeros working core and sets values for sensor error models. The 1st primary overlay contains input/output routines, state and covariance integrators and the navigation package. The 2nd primary overlay contains subroutines necessary to compute rendezvous maneuvers as currently defined.

By the setting of appropriate flags, AAP can be caused to run in either single-run or monte-carlo modes. Data as specified by the user is collected at intervals on each cycle of a monte-carlo run, and stored on a local mass-storage file for later transference to a permanent data tape. This data may then be processed statistically by a separate program.

The following sections will deal with those portions of AAP which constitute the navigation function, and its controlling subroutines. It will be assumed that the user is otherwise familiar with standard BETELGEUSE functions and the operation of the guidance overlay.

2.0 PROGRAM CONTROL

The sequence of cards portrayed in figure 2.1 constitutes the controlling set to read the program library, update corrections, load, execute and transfer run data from local to permanent storage. The following notes corresponding to cards identified with the same number are provided for clarity:

1. Creates local mass-storage file for monte-carlo data

2. Requests program library tape

3. Copies program library to local file

4. Copies compile file to local file

5. Update changed subroutines from program library

6. Compile updated routines

7. Rewind update file

8. Copy changed subroutines into compile file

9. Request additional field length for loading

10. Load updated program

11. Set up overlay linkages

12. Reduce to execution field length

13. Execute

14. Create a dummy file

15. Rewind the program data storage file

16. Request the permanent data storage tape

17. Turn data storage tape past previously stored data

18. Copy new data onto tape

Should abnormal termination of the program occur, placement of an EXIT. card following UNLOAD(DTAPE) will cause control to be transferred to the EXIT. card, and execution of all cards following. The card sequence from REWIND(FAKE) to UNLOAD(DTAPE) should be duplicated and placed behind the EXIT. card. This will prevent loss of data in the event of abnormal termination. In order for a subroutine to be modified, it must already exist in the tape program library. Attempts to add a previously non-existent subroutine will cause error termination. To ameliorate the effect of this restriction, the first primary overlay contains five dummy subroutines, OPEN1 to OPEN5. These may be modified as required to provide currently undefined program operations.

3.0 REQUIRED INPUT DATA

In order to produce desired program operation, it is necessary to specify sensor and IU error models, navigation initializing and control information and platform alignment times. If maneuvers are to be performed, targeting data and instructions as to which will be stored for later processing are also required. Vehicle state vectors and their covariance are specified in the usual way for all BETELGEUSE programs requiring such information.

3.1 HARD-WIRED CONSTANTS

Rendezvous sensors, alignments and maneuver applications are all modelled by AAP as pure Gaussian processes. Values for the means and standard deviations of these processes are set in program MAIN of the main overlay by FORTRAN replacement statements. Figure 3.1 presents the relevant portions of program MAIN with the subject variables underlined. A correction set of similar replacement statements must be contained in the update portion of the operating deck if sensors being modelled have a different error model than that shown in figure 3.1:

VARS ACCELEROMETER SCALE FACTOR ERROR (DIMENSIONLESS)

VARA DELTA-V CUTOFF UNCERTAINTY (FT/SEC)

RVAR ACTUAL VALUE OF RANGE MEASUREMENT ERROR AS A FRACTION OF TOTAL RANGE (DIMENSIONLESS)

RVARMIN ACTUAL VALUE OF MINIMUM RANGE ERROR (FT)

VVAR ACTUAL VALUE OF RANGE-RATE MEASUREMENT ERROR AS A FRACTION OF TOTAL RANGE-RATE (DIMENSIONLESS)

VVARMIN ACTUAL VALUE OF MINIMUM RANGE-RATE ERROR (FT/SEC)

VARAZ ACTUAL VALUE OF AZIMUTH MEASUREMENT ERROR (RADIANS)

VAREL ACTUAL VALUE OF ELEVATION MEASUREMENT ERROR (RADIANS)

BR, BV, BAZ, BEL ACTUAL VALUES OF RANGE, RANGE-RATE, AZIMUTH AND ELEVATION MEASUREMENT BIASES. THESE ARE SET IN THE FIRST VISIT TO THE SENSOR NOISE SUBROUTINE ON THE BASIS OF THE VALUES OF BRO, BVO, BAZO AND BELO

BRO ACTUAL VALUE OF RANGE BIAS ERROR (FT)

BVO ACTUAL VALUE OF RANGE-RATE BIAS ERROR (FT/SEC)

BAZO ACTUAL VALUE OF AZIMUTH BIAS ERROR (RADIANS)

BELO ACTUAL VALUE OF ELEVATION BIAS ERROR (RADIANS)

GDR ACTUAL VALUE OF GYRO DRIFT RATE ERROR (RADIANS/SEC)

ALIGNB ACTUAL VALUE OF INITIAL MIS-ALIGNMENT BIAS ERROR (RADIANS)

NFAMA INITIALIZER FOR MANEUVER APPLICATION ERRORS

NFAMB INITIALIZER FOR SENSOR BIAS ERRORS

```
 03/08/73   MSC VERSION 9.0   SCOPE 3.3   296-4
13.30.57.OARJH3S
13.30.57.OARJH,CM60000,P27,T100 CO,MI01.
13.30.57.REWIND(CAL)
13.30.57.REWIND(STAT)                                          1.
13.30.57.REQUEST(TAPE1)REEL183                                 2.
13.31.14.POLLOUT COMPLETED.  (FL  60000)
14.33.51.POLLIN COMPLETED.
14.34.04.  (33 ASSIGNED)
14.34.04.REWIND(TAPE1)
14.34.04.COPYBF(TAPE1,KEN1)                                    3.
14.34.16.COPYBF(TAPE1,KEN2)                                    4.
14.34.25.UNLOAD(TAPE1)
14.34.26.REWIND(KEN1,KEN2)
14.34.26.UPDATE(P=KEN1,W)                                      5.
14.34.27.READING INPUT
14.35.16.UPDATE COMPLETE
14.35.16.FTN(I=COMPILE)                                        6.
14.36.30.     17.759 CP SECONDS COMPILATION TIME
14.36.31.REWIND(LGO)                                           7.
14.36.31.COPYL(KEN2,LGO,XXX)                                   8.
14.36.33. RK          UPDATED
14.36.34. INPUT       UPDATED
14.36.34. GNEXEC      UPDATED
14.36.34. DELTAV      UPDATED
14.36.34. STOPE1      UPDATED
14.36.35. OUTDAT      UPDATED
14.36.37. POPOUT      UPDATED
14.36.39. OPEN3       UPDATED
14.36.40. OPEN4       UPDATED
14.36.40. OPEN5       UPDATED
14.36.42. CLOD        UPDATED
14.36.42. ICERR       UPDATED
14.36.43. GIDSEL      UPDATED
14.36.49.COPYL DONE
14.36.49.REL,70000.                                            9.
14.36.49.LOAD(XXX)                                             10.
14.37.26.NOGO.                                                 11.
14.37.26.REL,60000.                                            12.
14.37.26.AAP.                                                  13.
14.38.44.POLLOUT COMPLETED.  (FL  60000)
14.52.30.POLLIN COMPLETED.
15.27.52.EXIT
15.27.52.REWIND(FAKE)                                          14.
15.27.52.REWIND(STAT)                                          15.
15.27.52.REQUEST(OTAPE)      ENABLE WRITE ON REEL              16.
15.27.52.285
15.28.17.  (32 ASSIGNED)
15.28.17.REWIND(OTAPE)
15.28.17.COPYBF(OTAPE,FAKE,2)                                  17.
15.28.30.COPYBF(STAT,OTAPE)                                    18.
15.29.13.RELEASE(FAKE)
15.29.13.UNLOAD(OTAPE)
15.29.14.MT 32 BLOCKS WRITTEN--000156
15.29.20.CP    1350.320 SEC.
15.29.20.PP     315.604 SEC.
15.29.20.IO     068.445 SEC.
```

Figure 2.1 AAP Control Cards

```
C       LOAD ERROR MODEL
        VARS=1.E-4
        VARA=1.E-1
C
          RVAR=0.
          RVARMIN = 33.
        VVAR=4.3E-3
        VVARMIN=.43
          VARAZ=2.E-3
          VAREL=2.E-3
        BR=0.
        BV=0.
        BAZ=0.
        BEL=0.
        BRO=0.
        BVO=0.
        BAZO=17.45E-3
        BELO=17.45E-3
C
        GDR=1.45E-7
        ALIGNB=3.E-4
        NFAMA= 46728
        NFAMB= 12944
        NFAMC= 31171
        NFAMV= 22222
C
          RVARB=0.
          RVARMNB = 33.
        VVARB=4.3E-3
        VVARMNB=.43
          VARAZB=2.E-3
          VARELB=2.E-3
C
```

Figure 3.1 Hard-wired constants

NFAMC INITIALIZER FOR PLATFORM MISALIGNMENT AND DRIFT ERRORS

NFAMV INITIALIZER FOR SENSOR RANDOM NOISE ERRORS

RVARB FILTER VALUE OF RANGE MEASUREMENT ERROR AS A FRACTION OF TOTAL RANGE (DIMENSIONLESS)

RVARMNB FILTER VALUE OF MINIMUM RANGE ERROR (FT)

VVARB FILTER VALUE OF RANGE-RATE MEASUREMENT ERROR AS A FRACTION OF TOTAL RANGE-RATE (DIMENSIONLESS)

VVARMNB FILTER VALUE OF MINIMUM RANGE-RATE ERROR (FT/SEC)

VARAZB FILTER VALUE OF AZIMUTH MEASUREMENT ERROR (RADIANS)

VARELB FILTER VALUE OF ELEVATION MEASUREMENT ERROR (RADIANS)

All values are to be given 1-sigma (standard deviation) including sensor biases. Bias processes are considered as having a zero mean; these are constructed at the beginning of program execution and become the mean value of any subsequent random process. To the extent that it models or ignores sensor biases, and compensates for random noise in the weighting process, the filter contains a model of every known random process affecting the value of a measurement. The difference between the actual value of a random process, and the filter value, is that in general the actual values can only be guessed at. Hence the actual and filter values are not in general the same, and it is customary to set the filter value larger than the largest expected value of the actual errors.

3.2 INPUT DATA CARDS

In addition to normal BETELGEUSE input cards, additional cards are defined which control the storage of data at selected maneuvers, the performance of navigation processes and the time of platform alignments. Also, some of the BETELGEUSE input features are utilized in the normal way to set flags and support the input of other required data. These will now be examined on a card-by-card basis as to placement and content:

CARD #1, INTEGER PARAMETER, 14I5

IDENT, C1-C5, Program will execute number of monte-carlo cycles equal to IDENT

NP, C6-C10, Number of P-array variables in data file

NINT, C11-C15, Starting value of NGUIDE (determines which maneuver of rendezvous sequence is first)

NFIRST, C16-C20, BETELGEUSE initial condition option

NTABLE, C21-C25, Number of tables in input data file

NV, C26-C30, Number of variables to be integrated (NV=37 for navigation runs)

NMORE, C31-C35, Number of additional integer parameters to be read in on succeeding cards

NT(1), C36-C40,

NCOL, C41-C45, Number of columns of W to be advanced (must be greater than or equal to the number of variables being estimated)

C46-C50, Not used

C51-C55, Not used

NPER, C56-C60, Number of navigation procedure cards in input file

IBLATE, C61-C65, Keplerian gravity flag. IBLATE=0 specifies point mass accelerations. IBLATE=1 causes computation of aspherical perturbations

IE, C66-C70, TPI angle search flag. IE=0 causes TPI to be done on time. Otherwise, angle.

CARD #2, CARD #3, HOLLERITH FIELD, 72H each
Two comment cards for the purpose of describing and labeling the run deck

CARD #4, DATA STORAGE CONTROL, 15I5
Flags for the storage of up to 8 sets state vector and delta-v information at selected rendezvous maneuvers. The number of the integer field in the sequence of 15 determines at which maneuvers data will be stored, by setting a right-adjusted 1 in that field. EXAMPLE: A '1' in column 25 of this card causes data to be stored at NGUIDE=5, that is, TPI.

CARD #5 et seq., BETELGEUSE FLOATING POINT (P-ARRAY), I5 ,E15.7
P(297)=Input value of NFAM2, initial state error initializer

NAVIGATION PROCESS CONTROL CARDS, CONTIGUOUS WITH END OF BETELGEUSE FLOATING POINT ENTRIES. THE NUMBER OF THESE CARDS MUST BE EQUAL TO NPER, AND CONVERSELY. 2I2,3F7.1,9F5.2,I2

The following explaination applies to Ith card (I=1,NPER), that is, the Ith navigation procedure to be executed during each run:

NE(I), C1-C2, Value of NE(I) dtermines when card will become active. If NE(I)=5, card will become active during guidance period when NGUIDE=5, i.e. preTPI.

NM(I), C3-C4, Value of NM(I) determines what type of measurement this card will cause to be taken:
NM(I)=1: Radar mark (R,RDOT, TWO ANGLES)
NM(I)=2: VHF mark (R ONLY)
NM(I)=3: OPTICS mark (TWO ANGLES ONLY)
NM(I)=4: SHUTTLE? (R, TWO ANGLES)

DTL(I), C5-C11, Number of minutes after previous maneuver that this card will become active. If there has been no previous maneuver, program will define previous maneuver as having occurred at time=0.

DTN(I), C12-C18, Number of minutes before next maneuver that this card will become inactive.

DTM(I), C19-C25, Number of minutes between measurements called for by this card.

USP(I), C26-C30, 1-sigma radius of S/C position uncertainty, in thousands of feet

USV(I), C31-C35, 1-sigma radius of S/C velocity uncertainty, in feet/second

UTP(I), C36-C40, 1-sigma radius of TGT position uncertainty, in thousands of feet

UTV(I), C41-C45, 1-sigma radius of TGT velocity uncertainty, in feet/second

SR(I), C46-C50, 1-sigma range measurement bias uncertainty, in thousands of feet

SRD(I), C51-C55, 1-sigma range-rate measurement bias uncertainty, in feet/second

SO(I), C56-C60, 1-sigma radius of angle measurement bias uncertainty about line-of-sight, milliradians

SC1(I) and SC2(I), C61-C70, Dummy estimated constants, not currently defined, corresponding to elements 17 and 18 of the estimated state vector. Unless defined, these should be set to 0.

NW(I), C71-C72, Reinitialization flag. NW(I)=0 causes a spherical reinitialization of the W-matrix to a diagonal form with values specified by C26-C70. NW(I)=1 inhibits the resetting of W and causes this procedure to become active with the existing value of W.

Figure 4.1 AAP Block Data

BETELGEUSE STATE 1320.0 A

	RAD VEC	LONGITUDE	LATITUDE	ALT RATE	HOR VEL	HEADING	X-BAR	Y-BAR	Z-BAR	U-BAR	V-BAR	W-BAR
NAV	215.9674.	36.134	-75.881	-.98	25557.69	173..48 B	127.	193.	-3.	-2.84	-4.49	.18
ACT	21513368.	36.138	-75.873	.92	25556.58	172.958	180.	286.	-13.	-2.73	-4.13	.40

CARTESIAN STATE 1320.0

	XS(BRF)	YS(BRF)	ZS(BRF)	XSD(BRF)	YSD(BRF)	ZSD(BRF)	XT(BRF)	YT(BRF)	ZT(BRF)	XTD(BRF)	YTD(BRF)	ZTD(BRF)
NAV	4242138.	-3088186.	-20859853.	12505.66	22276.39	-753.70	4242053.	-3088362.	-20859975.	12507.54	22280.53	-750.99
ACT	4240690.	-3096668.	-20862699.	12504.20	22275.49	-755.53	4240567.	-3096932.	-20862871.	12505.78	22279.45	-763.09

CARTESIAN STATE ERRORS IN MEASUREMENT FRAME

XSE(MF)	YSE(MF)	ZSE(MF)	XSDE(MF)	YSDE(MF)	ZSDE(MF)	XTE(MF)	YTE(MF)	ZTE(MF)	XTDE(MF)	YTDE(MF)	ZTDE(MF)
1041.	-8596.	2677.	-8.93	-7.27	-3.57	1047.	-8704.	2682.	-8.84	-7.52	-3.80

RELATIVE STATE ERRORS

BIAS ESTIMATION | POSITION AND VELOCITY

RANGE	R-RATE	AZIMUTH	ELEV	SC1	SC2	XRE(MF)	YRE(MF)	ZRE(MF)	XRDE(MF)	YRDE(MF)	ZRDE(MF)
1.0	0.00	10.347	6.307	0.000	.000	5.	-108.	5.	.09	-.35	-.23

D-R	S-DOT	SD-CDEL	DELTA-H	DELH-DOT
94.	-.326	-.234	52.	-.001

D

RELATIVE PARAMETERS

	RANGE	R-RATE	AZIMUTH	REL ELEV
NAV	23..	-5.31	6.944	33.793
ACT	339.	-4.96	4.483	32.152
MEAS.	361.	-4.43	1.574	1.395

MARKS C

MFLG	0	UR	17.5
RAD	17	URD	.3
VHF	0	UAZ	4.6
OPT	-9	UEL	4.8
		SR	.883
		SRD	.937
		SAZ	.401
		SEL	.397

F

COVARIANCE OF RELATIVE ERRORS (MF)

X	.144E+01	.452E+01	.566E-01	.228E-01	.591E-02	.386E-03
Y	F1.19554	.169E+02	.371E+01	.433E-01	.433E+00	.246E-01
Z	.02721	.15186	.145E+01	.454E-03	.629E-02	.225E-01
XD	.65958	.10667	.01398	.240E-01	.686E-04	.290E-05
YD	.15770	.92154	.15666	.10274	.278E-01	.574E-04
ZD	.01185	.06438	.68773	.00533	.09129	.226E-01

INERTIAL STANDARD DEVIATIONS E

S/C	5349.	8738.	4128.	9.36	11.71	2.81
TGT	5349.	8737.	4128.	9.36	11.71	2.81
BIAS	0.	.000	1.575	1.96	0.00	0.00

-.69 -4.62 1.70

43.5 RNG -.015839 ETADI -.689 DVNDR

-4.618 RNGRT .00467 BETADI -4.618 DVLOS

1.700 DVDOP

THE NAVIGATION BURN ESTIMATE IS DU= -2.412 DV= 4.344 DW= .109

THE ACTUAL COMPONENTS WERE DU= -2.484 DV= 4.294 DW= .187 G

BETELGEUSE STATE 1380.0

	RAD VEC	LONGITUDE	LATITUDE	ALT RATE	HOR VEL	HEADING	X-BAR	Y-BAR	Z-BAR	U-BAR	V-BAR	W-BAR
NAV	215.9453.	19.303	-75.796	-3.19	25561.83	-17..709	-31.	-56.	-12.	-.11	-.23	-.31
ACT	21513493.	19.404	-75.795	-1.69	25560.34	-17..810	15.	35.	22.	.07	-.06	.07

CARTESIAN STATE 1380.0

ALIGNMENT CONTROL CARD, CONTIGUOUS WITH THE END OF NAVIGATION PROCESS CONTROL CARDS. THIS CARD MUST BE PRESENT, EVEN IF BLANK. I2,10E7.1

Program will read up to 10 decimal fields on card, following integer field. Integer in first two columns determines how many F-fields will be read in subsequent columns. Subsequent columns contain the times, in seconds, of desired platform realignments during each run. If card is blank, an alignment is automatically performed at time=0.

INPUT STATE COVARIANCE MATRIX, IMMEDIATELY FOLLOWING ALIGNMENT CARD. THIS MATRIX MUST BE PRESENT WITH DIMENSION 24 x 24 IF P(297) IS OTHER THAN ZERO.

INPUT TABLES CALLED FOR BY NTABLE, IMMEDIATELY FOLLOWING INPUT COVARIANCE MATRIX. THESE TABLES MUST BE PRESENT IF NTABLE IS GREATER THAN ZERO.

4.0 OUTPUT

Program AAP originates printed and mass storage output. Printed output is originated by the 2nd primary overlay during maneuver computations, by the 1st primary overlay during maneuver applications, and at each platform alignment. Block data on all vehicle states, and the status of the navigation, is printed periodically as specified by the user. A sample of such output is shown in figure 4.1 and will be discussed below. Block data print interval is controlled by setting P(9) equal to the desired print interval, in seconds. Block data is automatically printed every time the guidance (2nd) overlay is called, or whenever the W-matrix is reinitialized.

Area A: Time, in seconds, of the block print.

Area B: Azimuth of ground track; east is 0°, south is 90°, etc.

Cartesian State: Earth centered inertial frame (BRF)

Cartesian State Errors in Measurement Frame (MF): Measurement frame is defined as follows-

$$\text{UNIT}(\underline{X}_{MF}) = \text{UNIT}(\underline{Y}_{MF} \times \underline{Z}_{MF})$$

$$\text{UNIT}(\underline{Y}_{MF}) = \text{UNIT}(\underline{R}_{TGT} - \underline{R}_{S/C})$$

$$\text{UNIT}(\underline{Z}_{MF}) = \text{UNIT}(\underline{R}_{S/C} \times \underline{Y}_{MF})$$

XSE, YSE, ZSE: SPACECRAFT POSITION ERROR
XSDE, YSDE, ZSDE: SPACECRAFT VELOCITY ERROR

XTE, YTE, ZTE: TARGET POSITION ERROR
XTDE, YTDE, ZTDE: TARGET VELOCITY ERROR

} STATE ERRORS IN MF

Relative State Errors: Bias estimation portion is the estimated bias minus actual bias for each sensor in ft, fps, mr.

XRE, YRE, ZRE: RELATIVE POSITION ERRORS IN MF (ft)
XRDE, YRDE, ZRDE: RELATIVE VELOCITY ERRORS IN MF (fps)

Relative Parameters: Navigated and actual values of range, range-rate, azimuth and elevation in the local vertical in-plane frame. Measured values of these quantities have noise and biases added and are given in the estimated measurement frame. Units are ft, fps, deg.

Area C: Navigation Status

MFLG: Filter status flag: MFLG=0 Marking is enabled on any procedure active during this period.
MFLG=1 Marking is suspended because of final maneuver computation or recycle
MFLG=2 Final comp has been done and the filter is waiting for ignition before resuming updates

RAD: Number of radar marks since last W-reinitialization

VHF: Number of VHF marks since last W-reinitialization

OPT: Number of angle marks since last W-reinitialization

UR: State uncertainty in next range measurement (ft)

URD: State uncertainty in next r-rate measurement (fps)

UAZ: State uncertainty in next azimuth measurement (mr)

UEL: State uncertainty in next elevation measurement (mr)

SR: Weighting given a-priori estimate of range on last mark (A value of 1.0 indicates 100% confidence)

SRD: Same as above for r-rate

SAZ: Same as above for azimuth

SEL: Same as above for elevation

Area D: Coelliptical relative errors at phase match. Space craft is advanced to phase match with target and curvilinear errors are computed

S-R: Down-range error

S-DOT: Horizontal speed error

SD-COEL: Horizontal speed error minus speed error if vehicle errors were coelliptical

DELTA-H: Vertical position error

DELH-DOT: Vertical speed error

Area E: List of square roots of the diagonal terms of the filter covariance matrix. 1-sigma estimated uncertainty in the value of each estimated quantity, expressed in measurement frame. Vehicle rows are in the order of X, Y, Z, XD, YD, ZD; bias row is in order of R, RDOT, AZ, EL, SC1,SC2.

Area F: Covariance of relative errors in the measurement frame. Diagonal elements are 1-sigma uncertainties in each of the frame directions in the order of X, Y, Z, XD, YD, ZD. Lower left portion (Area F1) is array of correlation coefficients.

Area G: Estimated and actual maneuver. The navigation burn estimate is that computed from the estimated vehicle states, The actual value is that maneuver actually applied in view of platform drift, scale factor error and cutoff uncertainty.

MASS DATA STORAGE

On each cycle of a monte-carlo set, AAP stores up to 350 items of data on a locally created file, in unformatted form, for later transference to a permanent data storage tape. Before writing the local file, the program writes the date, time and number of cycles (one word each) on the beginning of the file. At the end of each cycle, the DATA array of 350 words is dumped on the file. These words are allotted as follows:

DATA(1)=LOOP, Current value of the monte-carlo cycle index

DATA(2)-DATA(249), Up to 8 sets of state vector and maneuver information stored for selected maneuvers (see CARD #4). Each set consists of 31 elements:
1-6: S/C actual BRF vector
7-12: TGT actual BRF vector
13-18: S/C navigated BRF vector
19-24: TGT navigated BRF vector
25-27: Maneuver computed from actual states
28-30: Maneuver computed from estimated states
31: Time of ignition

DATA(251)-DATA(320), Up to ten sets of platform alignment information. Each set consists of 7 elements:
1-3: Platform angular drift rates (X,Y,Z)
4-6: Initial platform misalignment (X,Y,Z)
7: Time of alignment

DATA(321)-DATA(350), Arbitrary output specified by user.

Regardless of which maneuver is the first with a data storage flag set, the state vector/maneuver data is stacked, 31 elements at a time, beginning in DATA(2). All quantities are output in fundamental units of feet, seconds and radians. A dump of the DATA array in the indicated format is performed at the termination of each cycle.

5.0 PROGRAM MECHANIZATION

This section will discuss the implementation of the navigation function down to the level of FORTRAN code. As a preliminary, the assignment and definition of all variables associated with the navigation will be reviewed.

MASTER COMMON AND EQUIVALENCE LIST

BLANK COMMON: VAR(5600)		BETELGEUSE BLANK COMMON
LABEL COMMON: DELV		LISTS VARS, VARA, NFAMA, SD(3) FOR USE IN MANEUVER APPLICATION ROUTINE. VARS, VARA, AND NFAMA ARE DISCUSSED IN SECTION 3.1. SD(3) ARE THREE ACCELEROMETER SCALE FACTORS CREATED ON FIRST VISIT TO DELTAV (MANEUVER APPLICATION).
	GARB	LISTS ACTUAL STATISTICS OF SENSOR ERRORS FOR USE BY SENSOR NOISE ROUTINE (GARBAGE). SEE SECTION 3.1.
	ALIG	LISTS ACTUAL STATISTICS OF PLATFORM ALIGNMENT AND DRIFT ERRORS FOR USE BY PLAT ALIGN ROUTINE (ALIGN). SEE SECTION 3.1.
	BV	LISTS FILTER STATISTICS OF SENSOR ERRORS FOR USE BY GEOMETRY VECTOR SUBROUTINE (BVEC). SEE SECTION 3.1.

DIMENSIONED ARRAYS USED BY NAVIGATION

Y(100)	INTEGRATED VARIABLES
DYDX(100)	DERIVATIVES WRT TIME
NTEGER(100)	INTEGER PARAMETERS
P(5000)	BETELGEUSE WORKING ARRAY
SAVE(950)	NAVIGATION STORAGE ARRAY
BLK(700)	NAVIGATION SCRATCH PAD
DATA(350)	MONTE-CARLO MASS DATA STORAGE
COV(24,24)	INPUT STATE VECTOR COVARIANCE MATRIX
QQ(4)	MEASURED RELATIVE PARAMETERS
SIG(4)	CURRENT VALUE OF MEASURED RELATIVE RANDOM NOISE
C(10)	ARRAY OF CONTROL TIMES FOR GNEXEC
REFMAT(3,3)	ESTIMATED TRANSFORMATION MATRIX FROM BRF TO PLATFORM AXES
XNBN(3), YNBN(3), ZNBN(3)	ESTIMATED NAVIGATION BASE UNIT VECTORS
NE(10)	SEE SECTION 3.2
NM(10)	" " "
DTL(10)	" " "
DTN(10)	" " "
DTM(10)	" " "
USP(10)	" " "
USV(10)	" " "
UTP(10)	" " "
UTV(10)	" " "

```
OWING AREA IS BLANK COMMON FOR BETELGEUSE                                  POPOUT      6
      COMMON VAR                                                           POPOUT      7
                                                                           POPOUT      8
     *                                                                     POPOUT      9
     *NF,AI,METRN                                                          POPOUT     10
     *                                                                     POPOUT     11
     NSIO   VAR(5600), Y(100), OYDX(100), O(100), FIRSTY(100)              POPOUT     12
            NTEGER(100), O(100), P(5000)                                   POPOUT     13
     VALENCE (VAR(1),Y(1))                                                 POPOUT     14
             (VAR(101),OYDX(1))                                            POPOUT     15
             (VAR(201),O(1))                                               POPOUT     16
             (VAR(301),FIRSTY(1))                                          POPOUT     17
             (VAR(401),NTEGER(1))                                          POPOUT     18
             (VAR(501),O(1))                                               POPOUT     19
             (VAR(601),P(1))                                               POPOUT     20
     NSION SAVE(950), BLK(700), DATA(350), COV(24,24)                      POPOUT     21
     VALENCE (P(350),SAVE(1))                                              POPOUT     22
             (P(1300),BLK(1))                                              POPOUT     23
             (P(4074),DATA(1))                                             POPOUT     24
             (P(4424),COV(1,1))                                            POPOUT     25
     NSION QQ(4), SIG(4), C(10), REFMAT(3,3), XNBN(3), YNBN(3)             POPOUT     26
           ZNBN(3), NE(10), NM(10), OTL(10), OTN(10), OTM(10)              POPOUT     27
           USP(10), USV(10), UTP(10), UTV(10), SR(10), SRO(10)             POPOUT     28
           SO(10), SC1(10), SC2(10), NW(10), TLM(10), NS(3)                POPOUT     29
           ZTZ(4), SZ(4), TALIGN(10), XNBE(3), YNBE(3), ZNBE(3)            POPOUT     30
           X(18), WE(18,27)                                                NOSHIT     1
           XLVE(3),YLVE(3),ZLVE(3),XLVN(3),YLVN(3),ZLVN(3)                 POPOUT     31
     VALENCE (SAVE(1),QQ(1)),          (SAVE(5),SIG(1))                   POPOUT     32
             (SAVE(9),C(1)),           (SAVE(19),REFMAT(1,1))             POPOUT     33
             (SAVE(28),XNBN(1)),       (SAVE(31),YNBN(1))                 POPOUT     34
             (SAVE(34),ZNBN(1)),       (SAVE(37),NE(1))                   POPOUT     35
             (SAVE(47),NM(1)),         (SAVE(57),OTL(1))                  POPOUT     36
             (SAVE(67),OTN(1)),        (SAVE(77),OTM(1))                  POPOUT     37
             (SAVE(87),USP(1)),        (SAVE(97),USV(1))                  POPOUT     38
             (SAVE(107),UTP(1)),       (SAVE(117),UTV(1))                 POPOUT     39
             (SAVE(127),SR(1)),        (SAVE(137),SRO(1))                 POPOUT     40
             (SAVE(147),SO(1)),        (SAVE(157),SC1(1))                 POPOUT     41
             (SAVE(167),SC2(1)),       (SAVE(177),NW(1))                  POPOUT     42
             (SAVE(187),TLM(1)),       (SAVE(197),NS(1))                  POPOUT     43
             (SAVE(200),ZTZ(1)),       (SAVE(204),SZ(1))                  POPOUT     44
             (SAVE(208),TALIGN(1)),    (SAVE(218),NALIGN)                 POPOUT     45
             (SAVE(229),XNBE(1)),      (SAVE(232),YNBE(1))                POPOUT     46
             (SAVE(235),ZNBE(1)),      (SAVE(258),X(1))                   POPOUT     47
             (SAVE(276),WE(1,1))                                          POPOUT     48
             (NTEGER(30),ICOMP)                                           NOSHIT     2
     VALENCE (SAVE(762),XLVE(1))                                          NOSHIT     3
             (SAVE(765),YLVE(1))                                          NOSHIT     4
             (SAVE(768),ZLVE(1))                                          NOSHIT     5
             (SAVE(771),XLVN(1))                                          NOSHIT     6
             (SAVE(774),YLVN(1))                                          NOSHIT     7
             (SAVE(777),ZLVN(1))                                          POPOUT     49
```

FIGURE 5.1 STANDARD COMMON BLOCK

SR(10)	SEE SECTION 3.2
SRD(10)	" " "
SO(10)	" " "
SC1(10),SC2(10)	" " "
NW(10)	" " "
TLM(10)	STORED TIME OF THE LAST MARK ON EACH NAVIGATION PROCEDURE
NS(3)	NUMBER OF MARKS SINCE W-REINITIALIZATION ON RADAR, VHF AND OPTICS
ZTZ(4)	A PRIORI UNCERTAINTY IN VALUE OF SENSOR MEASUREMENT
SZ(4)	WEIGHTING ON A PRIORI ESTIMATE OF SENSOR MEASUREMENT
TALIGN(10)	STORED TIMES OF PLATFORM ALIGNMENTS SEE SECTION 3.2
XNBE(3),YNBE(3), ZNBE(3)	ACTUAL NAVIGATION BASE UNIT VECTORS
X(18)	LOCAL STORAGE FOR ESTIMATED CARTESIAN STATE
WE(18,18)	THE FAMOUS W-MATRIX
XLVE(3),YLVE(3) ZLVE(3)	ACTUAL LOCAL VERTICAL UNIT VECTORS
XLVN(3),YLVN(3) ZLVN(3)	ESTIMATED LOCAL VERTICAL UNIT VECTORS
DRIFT(3)	INTEGRATED PLATFORM MISALIGNMENT ANGLES
RATE(3)	PLATFORM GYRO DRIFT RATES

Subroutines which utilize the BLK array for intermediate local computations define and equivalence local variables as required. These will be individually discussed in the section on subroutine structure. Figure 5.1 presents the standard navigation common block . In addition to the equivalences shown, the following are scattered throughout:

NTEGER(29),		NGUIDE	CURRENT VALUE OF NGUIDE
	30	ICOMP	NAVIGATION STATUS FLAG (SEE MFLG, SECTION 4.0, AREA C)
	31	ISTEP	GNEXEC INTEGRATION STEP-SIZE MANAGEMENT FLAG
	32	NOVER	OVERLAY RETURN MANAGEMENT FLAG
	33	LIGN	CURRENT NUMBER OF PLATFORM ALIGNMENTS ALREADY PERFORMED
	34		NOT DEFINED
	35	NGATE	BRAKING GATE COUNTER
	36	NBRFL	SET WHEN IN BRAKING PHASE
	42	NLINE	PRINTED OUTPUT LINE COUNTER
C(1)		TW	TIME TAG ON W-MATRIX
8		STEP	SAVED VALUE OF INPUT INTEGRATION STEP SIZE
9		T2	SYNCH STEP SIZE FOR FINAL PASS AFTER MANEUVER APPLICATION
10		TGN	TIME FROM NEXT IGNITION

5.1 FUNCTIONAL DESCRIPTION

In spite of the overlay structure, the actual operation of AAP is practically indestinguishable from that of the traditional BETELGEUSE program. The guidance overlay (2nd) acts like a subroutine which when called, computes a maneuver specified by the current value of NGUIDE. Other than the filter computation routines, the most significant addition to the program is the guidance and navigation executive, which controls the taking of navigation marks and the calling of the guidance overlay. The following is a list of subroutines peculiar to or modified by the presence of the navigation function:

MAIN	REFERENCES MONTE-CARLO MASS STORAGE FILE (TAPE77=STAT) SETS VALUES FOR ERROR MODELS BY REPLACEMENT
RK	HAS OVERLAY RETURN FLAG (NOVER), CALL TO GNEXEC, CALL TO SETY, CALL TO POPOUT, ENTRY POINT FOR COVARIANCE ADVANCEMENT (ENTRY RKW), INTEGRATES PLATFORM DRIFTS
INPUT	WRITES ON MASS DATA STORAGE TAPE (77), READS DATA STORAGE CONTROL FLAGS, READS NAVIGATION PROCESS CONTROL CARDS, READS ALIGNMENT CONTROL CARD, INITIALIZES GUIDANCE, NAVIGATION AND ALIGNMENT PARAMETERS, CALLS ALIGN AND SETY
GNEXEC	CONTROLS TAKING OF NAVIGATION MARKS, COMPUTATION AND APPLICATION OF MANEUVERS
DELTAV	APPLIES AN ESTIMATED AND ACTUAL MANEUVER TO THE ESTIMATED AND ACTUAL STATES
STORE1	STORES 31 ELEMENTS OF STATE VECTOR AND MANEUVER INFORMATION AT SELECTED MANEUVERS
OUTDAT	DUMPS THE MASS STORAGE MONTE-CARLO DATA AT THE END OF EACH CYCLE
POPOUT	COMPUTES ACTUAL, ESTIMATED AND MEASURED RELATIVE PARAMETERS, SETS UP VECTORS AND FILTER STATUS DATA, PRINTS BLOCK DATA AS REQUIRED
CART1, CART2	SUBROUTINE CONVERTS BETELGEUSE VECTOR TO CARTESIAN (CART1), OR A CARTESIAN VECTOR TO BETELGEUSE (CART2)
SETY	LOADS CARTESIAN FORM OF ESTIMATED BETELGEUSE STATE INTO Y ARRAY. CURRENTLY ONLY ONE INSTRUCTION IS ACTIVE- ALL CALLS TO SETY MAY BE REPLACED WITH THE STATEMENT CALL CART1(Y(38),Y(2))
REL	COMPUTES RELATIVE PARAMETERS (R, RDOT, AZ, EL) GIVEN CARTESIAN VECTOR AND UNIT VECTORS OF MEASUREMENT FRAME

SETUP	SETS UP AN OUTPUT FORM OF THE BETELGEUSE VECTOR
GARBAGE	CREATES INITIAL SENSOR BIASES AND ADDS RANDOM NOISE TO RELATIVE PARAMETERS
ALIGN	CREATES INITIAL PLATFORM DRIFT RATES, DEFINES A REFMAT AND SETS MISALIGNMENT BIASES FOR EACH ALIGNMENT
P20	LOCAL SUPERVISORY ROUTINE FOR TAKING A NAVIGATION MARK
ADVW	COVARIANCE INTEGRATING SUBROUTINE
BVEC	CALCULATES AND CONSTRUCTS REQUIRED GEOMETRY VECTORS FOR FILTER
FILTER	CALCULATES WEIGHTING VECTOR, UPDATES STATE AND COVARIANCE
OPEN1-OPEN5	DUMMY SUBROUTINES FOR USER DEFINED FUNCTIONS

The navigation function operates on a cartesian vector which is created periodically from the estmated BETELGEUSE state vector. This cartesian vector, although not itself integrated, is stored in the Y array. Because it is required for covariance advancement, the previous value of this vector , called X, is stored in the SAVE array from the previous visit to ADVW. The allocation of the Y array to state and other variables is as follows:

Y(1)	TIME
Y(2)-Y(13)	ESTIMATED BETELGEUSE STATE
Y(14)-Y(19)	ESTIMATED VALUES OF SENSOR BIASES
Y(20)-Y(31)	ACTUAL BETELGEUSE STATE
Y(32)-Y(37)	ACTUAL VALUE OF SENSOR BIASES
Y(38)-Y(49)	ESTIMATED CARTESIAN STATE
Y(50)-Y(55)	ESTIMATED VALUE OF SENSOR BIASES (SAME AS Y(14)-Y(19))
Y(98)-Y(100)	TOTAL INTEGRATED PLATFORM DRIFT ANGLES. THESE ARE EQUIVALENCED TO DRIFT(3) IN SUBROUTINE ALIGN. ALSO, THEIR DERIVATIVES, RATE(3), ARE EQUIVALENCED TO DYDX(98)-DYDX(100) IN ALIGN.

5.2 COMPUTATIONAL ORGANIZATION

Figure 5.2.1 presents the interfaces of the major navigation functional subroutines. The remaining part of this section will present the FORTRAN code used to implement this function, with flow diagrams and explainatory notes where appropriate. The following comments refer to Figure 5.2.1:

MAIN — Main program of main overlay.

DUMMY1 — Called by MAIN to bring in first primary overlay AAP(1,0)

RK — Integrating subroutine for (1,0). Also called at ENTRY RKW by ADVW for integration of W-matrix.

INPUT — Controls input of data to program. Called by RK at the beginning of each monte-carlo cycle

ALIGN — Simulates performance of platform alignment. Called by INPUT at beginning of program execution, and by GNEXEC at times defined by alignment control card.

POPOUT — Handles computation of actual, estimated and measured relative quantities for output and navigation. Computes and organizes for output other quantities of interest. Prints block data at intervals defined by P(9), and whenever called through ENTRY POPW by GNEXEC. Computation section of POPOUT will be executed when POPOUT is called on each pass through RK, even if print is inhibited.

OVERLAY 2 — Called in from GNEXEC at termination of the last marking procedure in a premaneuver period. Because return from AAP(2,0) is to DUMMY1 and RK, special provision is made in the program to return to the next instruction in GNEXEC following the call to AAP(2,0). This is accomplished by the setting of the NOVER flag.

GNEXEC — Executive routine. Controls taking of navigation marks, performance of platform alignments and computation of maneuvers. Called once each pass through RK.

DELTAV — Applies maneuvers computed by guidance overlay to actual and estimated states. Called from GNEXEC. Also calls ADVW to advance covariance to time of ignition.

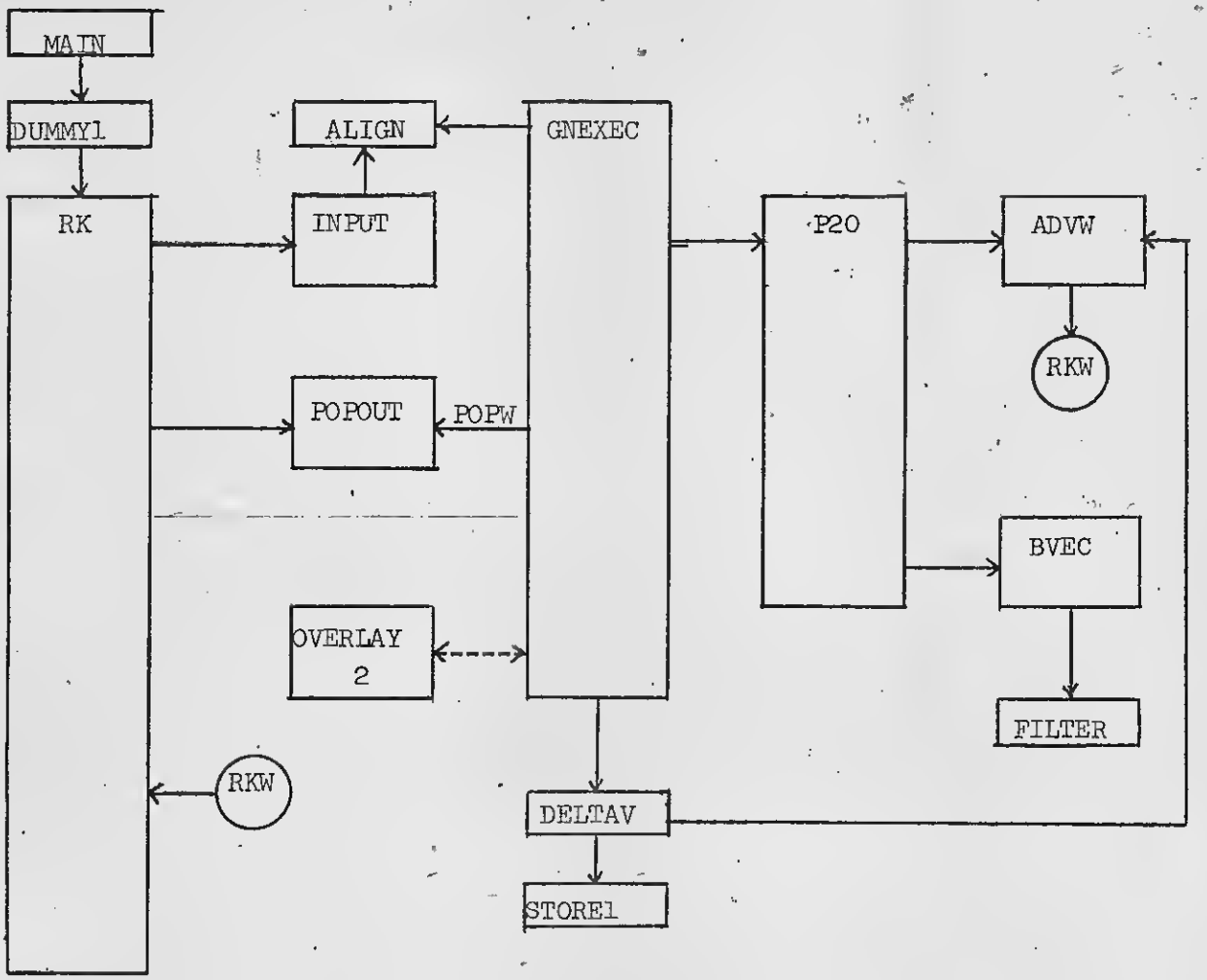
STORE1 — Loads state vector and delta-v data into DATA array for dump to mass storage at end of cycle. Called from GNEXEC.

P20 — Controls advancement of W, taking mark, updating state

ADVW . ADVANCES W-MATRIX TO CURRENT TIME WHENEVER CALLED. ENTERS RK AT RKW FOR COLUMEN ADVANCEMENT OF MATRIX.

BVEC DETERMINES WHAT SORT OF MARK IS DESIRED, CALCULATES APPROPRIATE GEOMETRY VECTORS, CALLS FILTER TO UPDATE STATE AND COVARIANCE

FILTER CALCULATES WEIGHTING VECTOR, UPDATES STATE AND COVARIANCE.



NOTE: ARROWS GO FROM CALLING ROUTINE TO CALLED ROUTINE.

FIGURE 5.2.1 NAVIGATION INTERFACES

MAIN OVERLAY

```
                                                  COC 6600 FTN V3.0-P308 OPT=1   08/29/72  11.32.27.

            OVERLAY(AAP,0,0)                                                      MAIN        2
      PROGRAM MAIN(INPUT,OUTPUT,STAT,TAPE5=INPUT,TAPE6=OUTPUT                     MAIN        3
     *,                                TAPE77=STAT)                             MAIN        4
C     GENERAL SUBROUTINES FOR SOLVING ORDINARY DIFFERENTIAL EQUATIONS            MAIN        5
C     BY MEANS OF FOUR POINT RUNGE KUTTA NUMERICAL INTEGRATION                   MAIN        6
C                                                                                MAIN        7
            COMMON VAR                                                           MAIN        8
C                                                                                MAIN        9
C     FOLLOWING IS LABELED COMMON FOR ERROR MODELS AND NAVIGATION                MAIN       10
C                                                                                MAIN       11
      COMMON/DELV/VARS,VARA,NFAMA,SO(3)                                          MAIN       12
C                                                                                MAIN       13
      COMMON/GARB/RVAR,RVARMIN,VVAR,VVARMIN,VARAZ,VAREL,NFAMV                    MAIN       14
     *,           BR,BV,BAZ,BEL,NFAMB                                          MAIN       15
     *,           BRO,BVO,BAZD,BELO                                            MAIN       16
C                                                                                MAIN       17
      COMMON/ALIG/GOR,ALIGNB,NFAMC                                               MAIN       18
C                                                                                MAIN       19
      COMMON/BV/RVARB,RVARMNB,VVARB,VVARMNB,VARAZB,VARELB                        MAIN       20
      DIMENSION SAVE(950), BLK(700)                                              MAIN       21
      EQUIVALENCE (P(350),SAVE(1))                                               MAIN       22
     *,           (P(1300),BLK(1))                                             MAIN       23
C                                                                                MAIN       24
            DIMENSION Y(100), DYDX(100), C(100), FIRSTY(100),                    MAIN       25
     1 P(5000),NTEGER(100),VAR(5600),D(100)                                      MAIN       26
            EQUIVALENCE (VAR(1),Y(1)), (VAR(101),DYDX(1)),                       MAIN       27
     1 (VAR(201),C(1)), (VAR(301),FIRSTY(1)), (VAR(401),NTEGER(1)),              MAIN       28
     2 (VAR(501),D(1)), (VAR(601),P(1))                                          MAIN       29
            EQUIVALENCE   (P(2301),TWOPI), (P(2302),CRAD),                       MAIN       30
     1 (P(2303),CNM)                                                             MAIN       31
```

```
C       ZERO CORE AT INITIAL LOADING                                    MAIN      33
              DO 20 J=1,5600                                            MAIN      34
20            VAR(J) = 0.0                                              MAIN
        SET IN FUNDAMENTAL CONSTANTS                                    MAIN      36
              TWOPI=6.2831853072                                        MAIN      37
              CRAD=57.2957795131                                        MAIN      38
              CNM=6076.10333                                            MAIN      39
C       LOAD ERROR MODEL                                          A     MAIN      40
        VARS=1.E-4                                                      MAIN      41
        VARA=1.E-1                                                      MAIN      42
C                                                                       MAIN      44
          RVAR=0.                                                       SENSOR     1
          RVARMIN = 33.                                                 SENSOR     2
        VVAR=4.3E-3                                                     MAIN      47
        VVARMIN=.43                                                     MAIN      48
          VARAZ=2.E-3                                                   SENSOR     3
          VAREL=2.E-3                                                   SENSOR     4
        BR=0.                                                           MAIN      52
        BV=0.                                                           MAIN      53
        BAZ=0.                                                          MAIN      54
        BEL=0.                                                          MAIN      55
        BRO=0.                                                          MAIN      57
        BVO=0.                                                          MAIN      58
        BAZO=17.45E-3                                                   MAIN      59
PROGRAM MAIN                    CDC 6600 FTN V3.0-P308 OPT=1  08/29/72  11.32.27.
        BELO=17.45E-3                                                   MAIN      60
C                                                                       MAIN      61
        GDR=1.45E-7                                                     MAIN      62
        ALIGNB=3.E-4                                                    MAIN      63
        NFAMA= 46728                                                    RANNO      1
        NFAMB= 12944                                                    RANNO      2
        NFAMC= 31171                                                    RANNO      3
        NFAMV= 22222                                                    RANNO      4
C                                                                       MAIN      65
          RVARB=0.                                                      SENSOR     5
          RVARMNB = 33.                                                 SENSOR     6
        VVARB=4.3E-3                                                    MAIN      68
```

```
      VARAZB=2.E-3                                                  SENSOR    7
      VARELB=2.E-3                                                  SENSOR    8
                                                                    MAIN
      SET DERIVATIVE OF INDEPENDENT VARIABLE WR/T ITSELF EQUAL TO ONE  MAIN    73
30          DYDX(1) = 1.0                                         MAIN      74
C     SET DERIV OF BIAS TERMS EQUAL ZERO                          MAIN      75
            DO 35 I=1,6                                           MAIN      76
            DYDX(I+13)=0.                                         MAIN      77
            DYDX(I+31)=0.                                         MAIN      78
35          CONTINUE                                              MAIN      79
C     SET JOB COUNTER                                             MAIN      80
            LOOP=0                                                MAIN      81
C     TRANSFER CONTROL TO NAVIGATION OVERLAY                      MAIN      82
            CALL DUMMY1                                           MAIN      83
            END                                                   MAIN      84
```

```
                SUBROUTINE DUMMY1                                       MAIN      85
C       ROUTINE TO LOAD NAVIGATION OVERLAY                              MAIN      86
                CALL OVERLAY(3HAAP,1,0,6HRECALL)                        MAIN      87
                RETURN                                                  MAIN      88
                END                                                     MAIN      89
```

```
                SUBROUTINE DUMMY2                                       MAIN      90
C     ROUTINE TO LOAD GUIDANCE OVERLAY                                  MAIN      91
                CALL OVERLAY(3HAAP,2,0,6HRECALL)                        MAIN      92
                RETURN                                                  MAIN      93
                END                                                     MAIN      94
```

1st Primary RK, DYDXS

```
      OVERLAY(1,0)                                                    MAIN      95
      PROGRAM NAVLAY                                                  MAIN      96
C     TRANSFER CONTROL TO INTEGRATION SUBROUTINE                      MAIN      97
      CALL RK                                                         MAIN      98
      END                                                             MAIN      99
```

PROGRAM MAIN

AREA A — THE PURPOSE AND HANDLING OF THESE INSTRUCTIONS IS DISCUSSED IN SECTION 3.1

AREA B — TIME DERIVATIVES OF BIAS PORTION OF INTEGRATED STATE VECTORS IS SET EQUAL TO ZERO. ALL ESTIMATED SENSOR BIASES ARE ASSUMED CONSTANT.

ROUTINE RK

AREA A — IF THIS PASS THROUGH RK IS A RETURN FROM THE GUIDANCE OVERLAY, NOVER WILL BE SET TO 1,2,3 OR 4, DEPENDING ON THE LOCATION IN GNEXEC WHICH CALLED THE OVERLAY. IN THIS CASE, IT IS DESIRED TO GO DIRECTLY BACK TO GNEXEC.

AREA B — CALL TO SETY LOADS A CARTESIAN FORM OF THE BETELGEUSE ESTIMATED STATE INTO Y(38) - Y(55) FOR USE BY OTHER SUBROUTINES. THIS MUST BE ACCOMPLISHED EACH INTEGRATION STEP BEFORE VISITING OUTPUT OR NAVIGATION EXECUTIVE.

AREA C — CALL TO POPOUT ACCOMPLISHES COMPUTATION OF RELATIVE STATE QUANTITIES FOR USE BY NAVIGATION, AND PERIODIC PRINTING OF BLOCK DATA DESCRIBED IN SECTION 4.0

AREA D — GUIDANCE AND NAVIGATION IS VISITED EACH INTEGRATION CYCLE TO PROVIDE FOR PERFORMANCE OF NAVIGATION, COMPUTATION OF MANEUVERS AND MANEUVER APPLICATION

AREA E — RK WILL BE CALLED PERIODICALLY AT ENTRY RKW FROM ADVW FOR ADVANCEMENT OF COVARIANCE. IN THIS CASE, IT IS DESIRED TO GO TO THE RETURN STATEMENT AT THE CONCLUSION OF THE STATE INTEGRATION INSTRUCTIONS. BY SETTING NFLGW=1 UPON ENTRY AT RKW, PROGRAM WILL RETURN TO CALLING ROUTINE ADVW INSTEAD OF PROCEEDING TO NEXT INTEGRATION STEP. FLAG IS RESET UPON NEXT NORMAL PASS THROUGH RK.

AREA F — LOGICAL OPERATOR TRANSFERS CONTROL TO RETURN STATEMENT IF NFLGW IS SET. OTHERWISE PLATFORM DRIFTS ARE INTEGRATED ONE STEP BEFORE GOING ON TO NEXT PASS THROUGH RK.

```
                SUBROUTINE RK                                                   RK        2
C       GENERAL SUBROUTINES FOR SOLVING ORDINARY DIFFERENTIAL EQUATIONS         RK        3
C       BY MEANS OF FOUR POINT RUNGE KUTTA NUMERICAL INTEGRATION                RK        4
C                                                                               RK        5
C       RK - INTEGRATING SUBPROGRAM                                             RK        6
                COMMON VAR                                                      RK        7
                DIMENSION Y(100), DYDX(100), Q(100), FIRSTY(100),               RK        8
     1 P(5000),NTEGER(100),VAR(5600),NT1(14),NT2(14),NT(14),D(100)              RK        9
                EQUIVALENCE (VAR(1),Y(1)), (VAR(101),DYDX(1)),                  RK       10
     1 (VAR(201),D(1)), (VAR(301),FIRSTY(1)), (VAR(401),NTEGER(1)),             RK       11
     2 (VAR(501),D(1)), (VAR(601),P(1)), (NTEGER(6), N)                         RK       12
     3,           (NTEGER(32),NOVER)                                            RK       13
C       CHECK IF THIS IS A RETURN FROM AN OVERLAY CALL                  A       RK       14
        IF(NOVER.GT.0) GO TO 25                                         A       RK       15
C       LOAD INPUT DATA INTO COMPUTER                                           RK       16
   10           CALL INPUT                                                      RK       17
C       CALCULATE THE INITIAL VALUES OF THE DERIVATIVES                         RK       18
   20           CONTINUE                                                        RK       19
        CALL SETY(Y)                                                    B       RK       20
        CALL POPOUT                                                     C       RK       21
   25   CONTINUE                                                                RK       22
        CALL GNEXEC                                                     D       RK       23
                IF(Y(1).GT.P(2)) GO TO 10                                       RK       24
        NFLGW = 0                                                       E       RK       25
        GO TO 29                                                                RK       26
        ENTRY RKW                                                               RK       27
        NFLGW = 1                                                       E       RK       28
   29   CONTINUE                                                                RK       29
                CALL DYDXS                                                      RK       30
C       WRITE INITIAL VALUES OF DERIVATIVES                                     RK       31
   30           CONTINUE                                                        RK       32
C       CALCULATE THE DELTA Y(J) AT Y(1) = 0                                    RK       33
   40           DO 50 J = 1,N                                                   RK       34
   50           D(J) = DYDX(J)*P(1)                                             RK       35
C       CALCULATE THE Y(J) AT T = 0                                             RK       36
   60           DO 90 J = 1,N                                                   RK       37
   70           R = .5*(D(J) - Q(J))                                            RK       38
   80           Y(J) = Y(J) + R                                                 RK       39
   90           Q(J) = Q(J) + 3.0*R - .5*D(J)                                   RK       40
C       CALCULATE THE DELTA Y(J) AT Y(1) = HALF STEP                            RK       41
  100           CALL DYDXS                                                      RK       42
  110           DO 120 J = 1,N                                                  RK       43
```

```
C       CALCULATE THE Y(J) AT Y(1) = HALF STEP                          RK      45
  130           DO 160 J = 1,N                                          RK      46
  140           R = .292893219*(D(J) - Q(J))                            RK      47
  150           Y(J) = Y(J) + R                                         RK      48
  160           Q(J) = Q(J) + 3.0*R - .292893219*D(J)                   RK      49
C       CALCULATE THE DELTA Y(J) AT Y(1) = HALF STEP (AGAIN)            RK      50
  170           CALL DYDXS                                              RK      51
  180           DO 190 J = 1,N                                          RK      52
  190           D(J) = DYDX(J)*P(1)                                     RK      53
C       CALCULATE THE Y(J) AT Y(1) = HALF STEP (AGAIN)                  RK      54
  200           DO 230 J = 1,N                                          RK      55
  210           R = 1.70710678*(D(J) - Q(J))                            RK      56
```

UBROUTINE RK CDC 6600 FTN V3.0-F308 DPT=1 08/29/72 11.32.27.

```
  220           Y(J) = Y(J) + R                                         RK      57
  230           Q(J) = Q(J) + 3.0*R - 1.70710678*D(J)                   RK      58
C       CALCULATE THE DELTA Y(J) AT Y(1) = FULL STEP                    RK      59
  240           CALL DYDXS                                              RK      60
  250           DO 260 J = 1,N                                          RK      61
  260           D(J) = DYDX(J)*P(1)                                     RK      62
C       CALCULATE THE Y(J) AT Y(1) = FULL STEP                          RK      63
  270           DO 300 J = 1,N                                          RK      64
  280           R = .1666666667*(D(J) - 2.0*Q(J))                       RK      65
  290           Y(J) = Y(J) + R                                         RK      66
  300           Q(J) = D(J) + 3.0*R - .5*D(J)                           RK      67
        IF(NFLGW.EQ.1) GO TO 330                                        F       RK      68
C       INTEGRATE PLATFORM DRIFTS                                               RK      69
              DO 305 I=98,100                                                   RK      70
  305         Y(I)=Y(I) + P(1)*DYDX(I)                                  F       RK      71
C       PROCEED TO THE NEXT INTEGRATION STEP                            RK      72
  310           NGO = 1                                                 RK      73
  320           GO TO (20,330),NGO                                      RK      74
  330           RETURN                                                  RK      75
                END                                                     RK      76
```

INPUT

ROUTINE INPUT

AREA A — FIRST TWO INSTRUCTIONS CHECK NW(I) ARRAY (SEE SEC 3.2) TO RESET W-MATRIX REINITIALIZATION FLAGS. IF ANY FLAG OF THIS ARRAY IS 0 ON INPUT NAVIGATION CONTROL CARD, IT IS SET TO -1 AT TIME W IS REINITIALIZED. LAST FOUR INSTRUCTIONS IN THIS ARE WRITE OUT ACCUMULATED MONTE-CARLO DATA TO LOCAL MASS STORAGE FILE AND THEN ZERO ARRAY FOR NEXT CYCLE.

AREA B — AT BEGINNING OF PROGRAM EXECUTION, SYSTEM ROUTINES ARE READ TO DETERMINE DATE AND TIME OF THIS RUN. THIS INFORMATION PLUS THE NUMBER OF MONTE-CARLO CYCLES TO BE EXECUTED ARE THE READ ONTO BEGINNING OF MASS STORAGE FILE.

AREA C — FOLLOWING INPUT OF BETELGEUSE HOLLERITH COMMENT CARDS, A CARD SPECIFYING THE MANEUVERS AT WHICH DATA WILL BE STORED IS READ IN AND PRINTED OUT. NO MORE THAN 8 ELEMENTS OF NST(I) MAY BE NON-ZERO; THE DATA ARRAY HAS ROOM FOR ONLY 8 MANEUVERS WORTH OF DATA.

AREA D — AFTER INPUT OF BETELGEUSE FLOATING POINT ENTRIES, A SERIES OF UP TO 10 NAVIGATION CONTROL CARDS IS READ IN. NUMBER OF THESE CARDS IS EQUAL TO NPER. AFTER INPUT, TIME, LENGTH, AND ANGLE QUANTITIES ARE RESCALED TO FUNDAMENTAL UNITS OF SECONDS, FEET AND RADIANS.

AREA E — FOLLOWING NAVIGATION CONTROL CARDS, AN ALIGNMENT CONTROL CARD, AS SPECIFIED IN SEC 3.2 IS READ AND ALIGNMENT TIMES PRINTED OUT.

AREA F — DATA ARRAY STACK INDEX IS RESET. THIS INDEX IS INCREMENTED BY 1 EACH TIME A 31 ELEMENT ARRAY OF VECTOR AND MANEUVER DATA IS STORED IN THE DATA ARRAY. SEE MASS DATA STORAGE, PAGE 9.

AREA G — RESET MANEUVER TYPE COMPUTATION FLAG TO STARTING VALUE.
RESET ALIGNMENT COUNTER FLAG (INCREMENTED BY 1 EACH TIME AN ALIGNMENT IS PERFORMED.
SET MANEUVER COMPUTATION FLAG TO CALL FOR AN INITIAL VISIT TO THE GUIDANCE OVERLAY AT FIRST VISIT TO GNEXEC.

AREA H — ZERO THE W-MATRIX TO BE SURE ITS NICE AND CLEAN FOR NEXT CYCLE.
LOAD ESTIMATED CARTESIAN STATE INTO Y(38)-Y(55).
PERFORM INITIAL ALIGNMENT TO DEFINE PLATFORM AXES.

```
                SUBROUTINE INPUT                                                INPUT     2
C       GENERAL SUBROUTINES FOR SOLVING ORDINARY DIFFERENTIAL EQUATIONS         INPUT     3
C       BY MEANS OF FOUR POINT RUNGE KUTTA NUMERICAL INTEGRATION                INPUT     4
C                                                                               INPUT     5
C       INPUT - SUBPROGRAM FOR READING IN DATA                                  INPUT     6
                COMMON VAR                                                      INPUT     7
                DIMENSION Y(100), DYDX(100), Q(100), FIRSTY(100),               INPUT     8
      1 P(5000),NTEGER(100),VAR(5600),NT1(14),NT2(14),NT(14),D(100)             INPUT     9
                EQUIVALENCE (VAR(1),Y(1)), (VAR(101),DYDX(1)),                  INPUT    10
      1 (VAR(201),Q(1)), (VAR(301),FIRSTY(1)), (VAR(401),NTEGER(1)),            INPUT    11
      2 (VAR(501),D(1)), (VAR(601),P(1)), (NTEGER(6),N),                        INPUT    12
      3 (NTEGER(1),IDENT), (NTEGER(2),NP), (NTEGER(3),NINT),                    INPUT    13
      4 (NTEGER(4),NFIRST), (NTEGER(5),NTABLE), (NTEGER(7),NMORE),              INPUT    14
      5 (NTEGER(8),NT(1)), (NTEGER(23),NTCR), (NTEGER(24),NTSKIP),              INPUT    15
      6 (NTEGER(41),LPRINT), (NTEGER(42),NLINE), (NTEGER(43),NSKIP),            INPUT    16
      7 (NTEGER(44),NPAGE), (NTEGER(45),NT1(1)), (NTEGER(60),NT2(1))            INPUT    17
        DIMENSION NST(15), TIG(15)                                              INPUT    18
        EQUIVALENCE (P(294),NFAM2)                                              INPUT    19
     *,             (P(295),LOOP)                                               INPUT    20
     *,             (P(334),NSTAT)                                              INPUT    21
     *,             (P(335),NST(1))                                             INPUT    22
     *,             (P(2107),TTPI)                                              INPUT    23
     *,             (P(2141),TIG(1))                                            INPUT    24
        EQUIVALENCE (NTEGER(12),NPER)                                           INPUT    25
     *,             (NTEGER(29),NGUIDE)                                         INPUT    26
     *,             (NTEGER(30),ICOMP)                                          INPUT    27
     *,             (NTEGER(33),LIGN)                                           INPUT    28
     *,             (NTEGER(35),NGATE)                                          INPUT    29
     *,             (NTEGER(36),NBRFL)                                          INPUT    30
        DIMENSION SAVE(950), BLK(700), DATA(350), COV(24,24)                    INPUT    31
        EQUIVALENCE (P(350),SAVE(1))                                            INPUT    32
     *,             (P(1300),BLK(1))                                            INPUT    33
     *,             (P(4074),DATA(1))                                           INPUT    34
     *,             (P(4424),COV(1,1))                                          INPUT    35
        DIMENSION DQ(4), SIG(4), C(10), REFMAT(3,3), XNBN(3), YNBN(3)           INPUT    36
     *,           ZNBN(3), NE(10), NM(10), DTL(10), DTN(10), DTM(10)           INPUT    37
     *,           USP(10), USV(10), UTP(10), UTV(10), SR(10), SRD(10)          INPUT    38
     *,           SO(10), SC1(10), SC2(10), NW(10), TLM(10), NS(3)             INPUT    39
     *,           ZTZ(4), SZ(4), TALIGN(10), XNBE(3), YNBE(3), ZNBE(3)         INPUT    40
     *,           X(18), WE(18,27)                                             INPUT    41
        EQUIVALENCE (SAVE(1),QQ(1)),          (SAVE(5),SIG(1))                  INPUT    42
     *,             (SAVE(9),C(1)),           (SAVE(19),REFMAT(1,1))            INPUT    43
```

```
     *,              (SAVE(34),ZNBN(1)),     (SAVE(37),NE(1))                INPUT     45
     *,              (SAVE(47),NM(1)),       (SAVE(57),DTL(1))               INPUT     46
     *,              (SAVE(67),DTN(1)),      (SAVE(77),DTM(1))               INPUT     47
     *,              (SAVE(87),USP(1)),      (SAVE(97),USV(1))               INPUT     48
     *,              (SAVE(107),UTP(1)),     (SAVE(117),UTV(1))              INPUT     49
     *,              (SAVE(127),SR(1)),      (SAVE(137),SRD(1))              INPUT     50
     *,              (SAVE(147),SO(1)),      (SAVE(157),SC1(1))              INPUT     51
     *,              (SAVE(167),SC2(1)),     (SAVE(177),NW(1))               INPUT     52
     *,              (SAVE(187),TLM(1)),     (SAVE(197),NS(1))               INPUT     53
     *,              (SAVE(200),ZTZ(1)),     (SAVE(204),SZ(1))               INPUT     54
     *,              (SAVE(208),TALIGN(1)),  (SAVE(218),NALIGN)              INPUT     55
     *,              (SAVE(229),XNBE(1)),    (SAVE(232),YNBE(1))             INPUT     56
```

UBPOUTINE INPUT CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
     *,              (SAVE(235),ZNBE(1)),    (SAVE(258),X(1))                INPUT     57
     *,              (SAVE(276),WE(1,1))                                     INPUT     58
C       ZERO THE WORKING ARRAYS                                              INPUT     59
        DO 5 I=1,700                                                         INPUT     60
    5   BLK(I)=0.                                                            INPUT     61
C       SET PAGE NO OF FIRST PAGE                                            INPUT     62
   10           NPAGE = 1                                                    INPUT     63
            LOOP = LOOP + 1                                                  INPUT     64
            IF(LOOP.EQ.1) GO TO 20                                           INPUT     65
        DO 11 I=1,10                                                    A    INPUT     66
   11   IF(NW(I).LT.0) NW(I) = 0                                             INPUT     67
        WRITE(77) (DATA(I),I=1,350)                                          INPUT     68
        CALL OUTOAT                                                          INPUT     69
        DO 15 I=1,350                                                        INPUT     70
   15   DATA(I)=0.                                                      A    INPUT     71
            IF(LOOP.GT.IDENT) GO TO 20                                       INPUT     72
            WRITE(6,80)                                                      INPUT     73
            GO TO 380                                                        INPUT     74
C       READ CONTROL INTEGERS INTO PROBLEM                                   INPUT     75
   20           LOOP=1                                                       INPUT     76
                READ(5,30) (NTEGER(J),J=1,14)                                INPUT     77
C       MAKE END-OF-FILE CHECK                                               INPUT     78
              IF(EOF(5)) 21, 22                                              INPUT     79
   21           CALL EXIT                                                    INPUT     80
   22           CONTINUE                                                     INPUT     81
```

```
C       WRITE DATE, TIME AND SIZE OF UPCOMING STATISTICAL SET                B
        CALL DATE(IDATE)                                                     B
        CALL TIME(ITIME)                                                     B
        WRITE (77) IDATE,ITIME,IDENT                                         B
   30         FORMAT(14I5)
   40         IF(NMORE)60,60,50
   50         NMO = 14 + NMORE
              READ(5,30) (NTEGER(J),J=15,NMO)
C       WRITE HEADING AT TOP OF PAGE
   60         CONTINUE
              NLINE=30
C       READ AND WRITE TWO CARDS OF RUN INFORMATION
              READ(5,80)
   80         FORMAT(72H
     1                          /72H
     2                                    )
              WRITE(6,80)
          WRITE(6,1000)
 1000     FORMAT(1H0,14X,29H** DATA INPUT FOR THIS RUN **//18X,
     1    9HPARAMETER,6X,4HDATA/)
C       READ CONTROL FLAGS FOR DATA STORAGE                                  C
        READ 475, (NST(I),I=1,15)                                            C
        PRINT 480,(NST(I),I=1,15)                                            C
  475   FORMAT(15I5)                                                         C
  480   FORMAT(//1X,51HBLOCK DATA WILL BE STORED AT THE FOLLOWING NGUIDES-   C
     *,15I5)                                                                 C
C       CHECK FOR INDIVIDUAL FLOATING POINT DATA ENTRY
   100         IF(NP)250,250,110
   110         DO 140 J = 1,NP
               READ(5,130) I,P(I)
```

UBROUTINE INPUT CDC 6600 FTN V3.0-P30$_8$ OPT=1 08/29/72 11.32.27.

```
   130         FORMAT(I5,E15.7)
               WRITE(6,1001) I,P(I)
  1001     FORMAT(17X,I5,3X,E15.8)
   140         CONTINUE
               IF(NTEGER(13).GT.0) WRITE(6,1003)
  1003     FORMAT(//20X,19H**********/
     1    20X,19H**********(20X,2H** 15X,2H**/
```

```
     2   20X,19H**  BE IT KNOWN  **/                                    INPUT     119
     3   20X,19H** THAT THIS RUN **/                                    INPUT     120
     4   20X,19H**  WAS MADE IN  **/                                    INPUT     12?
     5   20X,19H**   AN OBLATE   **/                                    INPUT     12
     6   20X,19H** ENVIRONMENT  **/                                     INPUT     123
     7   20X,2H**,15X,2H**/20X,19H**********************/               INPUT     124
     8   20X,19H**********************)                                 INPUT     125
C                                                                       INPUT     126
C    CHECK IF NAVIGATION DATA IS IN INPUT FILE                      D   INPUT     127
     IF(NPER.EQ.0) GO TO 155                                            INPUT     128
C    READ AND SCALE NAVIGATION DATA                                     INPUT     129
     DO 145 I=1,NPER                                                    INPUT     130
     READ(5,1004) NE(I),NM(I),OTL(I),OTN(I),OTM(I),USP(I),USV(I)        INPUT     131
    1,UTP(I),UTV(I),SR(I),SRD(I),SO(I),SC1(I),SC2(I)                    INPUT     132
    2,NW(I)                                                             INPUT     133
     PRINT 460,NE(I),NM(I),OTL(I),OTN(I),OTM(I),USP(I),USV(I)           INPUT     134
    1,UTP(I),UTV(I),SR(I),SRD(I),SO(I),SC1(I),SC2(I),NW(I)              INPUT     135
 460 FORMAT(/1X,134H   NE   NM        OTL        OTN        OTM        USP INPUT     136
    1       USV        UTP        UTV        SR         SRD        SO      INPUT     137
    2 SC1      SC2 NW       ,/1X,2I5,3F10.1,9F10.2,I3)                   INPUT     138
C    SCALE DATA                                                         INPUT     139
     OTL(I)=OTL(I)*60.                                                  INPUT     140
     OTN(I)=OTN(I)*60.                                                  INPUT     141
     OTM(I)=OTM(I)*60.                                                  INPUT     142
     USP(I)=USP(I)*1000.                                                INPUT     143
     UTP(I)=UTP(I)*1000.                                                INPUT     144
     SR(I)=SR(I)*1000.                                                  INPUT     145
     SO(I)=SO(I)/1000.                                                  INPUT     146
1004 FORMAT(2I2,3F7.1,9F5.2,I2)                                         INPUT     147
 145 CONTINUE                                                       D   INPUT     148
     READ 1005,NALIGN, (TALIGN(I),I=1,NALIGN)                       E   INPUT     149
1005 FORMAT(I2,10E7.1)                                                  INPUT     150
     IF(NALIGN.LT.1) GO TO 150                                          INPUT     151
     PRINT 465, NALIGN                                                  INPUT     152
 465 FORMAT(/1X,I2,44H ALIGNMENTS ARE SCHEDULED FOR THIS RUN AT-      ) INPUT     153
     DO 146 I=1,NALIGN                                                  INPUT     154
 146 PRINT 470, TALIGN(I)                                               INPUT     155
 470 FORMAT(/44X,F10.1)                                                 INPUT     156
 150 CONTINUE                                                           INPUT     157
 155 CONTINUE                                                       E   INPUT     158
C                                                                       INPUT     159
          ICOM = 0                                                      INPUT     160
          NFAM2=P(297)                                                  INPUT     161
C    CHECK IF COVARIANCE MATRIX IS TO BE READ IN                        INPUT     162
          IF(NFAM2) 240,250,240                                         INPUT     163
  240   CALL INPUTO                                                     INPUT     164
C    CHECK IF TABLE ENTRIES ARE TO BE MADE                              INPUT     165
  250     IF(NTABLE)380,380,260                                         INPUT     166
```

```
260           NT1(1) = NTCR                                          INPUT  167
              DO 370 M = 1,NTABLE                                    INPUT  168
280           IF(NT(M))290,370,310                                   INPUT  169
290           NT1(M+1) = NT1(M)                                      INPUT  170
300           GO TO 370                                              INPUT  171
310           NT1(M+1) = NT1(M) + NT(M)                              INPUT  172
320           NT2(M) = NT1(M) -1 + NT(M)                             INPUT  173
330           NT11 = NT1(M)                                          INPUT  174
340           NT12 = NT2(M)                                          INPUT  175
              READ(5,360) (P(J),J=NT11,NT12)                         INPUT  176
360           FORMAT(7E10.7)                                         INPUT  177
370           CONTINUE                                               INPUT  178
C       CALL INPUT WRITEOUT AND IC CALC ROUTINE                      INPUT  179
380           CALL INAID                                             INPUT  180
C       SET MANEUVER COMPUTATION FLAG                                INPUT  181
        P(10)=-(P(9) + 1.)                                           INPUT  182
        NSTAT=0                                                  F   INPUT  183
        DATA(1)=LOOP                                                 INPUT  184
        DATA(321) = NFAM2                                            INPUT  185
        PRINT 450, P(2)                                              INPUT  186
450     FORMAT(/1X,5HP(2) ,E15.6)                                    INPUT  187
              NGUIDE=NTEGER(3)                                   G   INPUT  188
              LIGN=0                                                 INPUT  189
              ICOMP=1                                            G   INPUT  190
C       SET TPI TIME INTO ERASABLE                                   INPUT  191
              TIG(5)=TTPI                                            INPUT  192
C     SET BRAKING GATE INDEX TO FIRST GATE                           INPUT  193
              NGATE=1                                                INPUT  194
C       SET BRAKING FLAG TO ZERO                                     INPUT  195
              NBRFL=0                                                INPUT  196
C       ZERO THE Q AND SET IN IC                                     INPUT  197
390           DO 420 J = 1,N                                         INPUT  198
400           Q(J) = 0.0                                             INPUT  199
410           Y(J) = FIRSTY(J)                                       INPUT  200
420           CONTINUE                                               INPUT  201
C     ZERO THE WE MATRIX                                         H   INPUT  202
        DO 421 I=1,18                                                INPUT  203
        DO 421 J=1,27                                                INPUT  204
        WE(I,J) = 0.0                                                INPUT  205
421     CONTINUE                                                     INPUT  206
C       SET UP VECTORS AND PERFORM INITIAL ALIGNMENT                 INPUT  207
        CALL SETY(Y)                                                 INPUT  208
        CALL ALIGN                                               H   INPUT  209
C       CONSTRUCT ENVIRONMENT VECTOR                                 INPUT  210
425           CALL ICERR                                             INPUT  211
430           RETURN                                                 INPUT  212
              END
```

GNEXEC

ROUTINE GNEXEC

AREA A — GUIDANCE OVERLAY RETURN CHECK. IF NOVER FLAG IS OTHER THAN ZERO, GNEXEC HAS BEEN CALLED ON RETURN FROM THE GUIDANCE OVERLAY. VALUE OF NOVER (1,2,3,4) INDICATES PLACE IN GNEXEC WHICH CALLED GUIDANCE OVERLAY. EXECUTION OF THE GO TO STATEMENT RETURNS CONTROL TO STATEMENT FOLLOWING ONE THAT CALLED OVERLAY.

AREA B — LIGN FLAG IS INCREMENTED EACH TIME A PLATFORM ALIGNMENT IS PERFORMED. CHECK IS MADE TO DETERMINE IF LIGN IS EQUAL TO NUMBER OF ALIGNMENTS SCHEDULED ON INPUT. CHECK IS THEN MADE TO SEE IF THE NEXT ALIGN IS LESS THAN ONE INTEGRATION STEP AWAY.

AREA C — IF NO NAVIGATION PROCEDURE CARDS WERE READ IN ON INPUT, CONTROL IS TRANSFERRED OUT OF THE NAVIGATION CONTROL PORTION.

AREA D — BEGIN DO LOOP WHICH CYCLES THROUGH THE NAVIGATION CONTROL CARDS READ IN ON INPUT. IF CARD IS NOT ACTIVE, LOOP TURNS TO NEXT CARD.

AREA E — IF CARD IS ACTIVE FOR THIS NGUIDE (NE(I)≠0), CHECK IS MADE TO SEE IF A SENSOR IS ON. IF SENSOR IS OFF, CONTROL IS TRANSFERRED TO W-MATRIX INITIALIZATION SECTION TO SEE IF THIS CARD IS PRESENT ONLY TO RESET W.

AREA F — IF FIRST RENDEZVOUS MANEUVER IS NC1, TIME SINCE LAST MANEUVER IS DEFINED AS PROGRAM ELAPSED TIME. OTHERWISE, PROGRAM ELAPSED TIME MINUS PREVIOUS TIG.

AREA G — TIME TO NEXT MANEUVER IS TIG FOR CURRENT MANEUVER MINUS PROGRAM ELAPSED TIME.

AREA H — SEE IF TIME SINCE LAST MANEUVER IS GREATER THAN MINIMUM TIME TO BEGIN THIS PROCEDURE.

AREA I — SEE IF TIME TO NEXT MANEUVER IS STILL GREATER THAN MINIMUM TIME TO TERMINATE BEFORE MANEUVER.

AREA J — IF IT IS TIME TO TERMINATE THIS PROCEDURE, MANEUVER COMPUTATION FLAG IS SET = 1 TO INDICATE A MARKING PROCEDURE HAS TERMINATED AND THE GUIDANCE CONTROL SECTION CAN CALL THE GUIDANCE OVERLAY TO COMPUTE A BURN. TIME SINCE LAST MARK ON THIS PROCEDURE IS SET TO ZERO FOR NEXT CYCLE ON MONTE-CARLO SET.

AREA K — CHECK TO SEE IF THIS PROCEDURE CALLS FOR W TO BE RESET. OTHERWISE GO DIRECTLY TO MARKING SECTION AREAS M-P.

AREA L — REINITIALIZE W:
SET SENSOR MARK COUNTERS TO ZERO.
SET THE CARD INITIALIZATION FLAG TO -1 INDICATING FOR FUTURE PASSES THAT REINITIALIZATION HAS BEEN DONE. THIS FLAG WILL BE RESET IN INPUT ON BEGINNING NEXT MONTE-CARLO CYCLE.
DEFINE TIME TAG ON W AS CURRENT ELAPSED PROGRAM TIME.
SET TIME OF LAST MARK ON THIS PROCEDURE EQUAL 0.
STORE THE CURRENT CARTESIAN ESTIMATED STATE FOR THE W-MATRIX ADVANCEMENT ROUTINE. ZERO OUT ELEMENTS OF W.
CALL FOR BLOCK PRINT AT THE REINITIALIZATION.

AREA M — IF SENSORS WERE OFF AND THIS CARD IS ONLY TO RESET W, TURN TO NEXT CARD.

AREA N — DEFINE THE TIME INTERVAL SINCE LAST MARK ON THIS PROCEDURE AS PROGRAM ELAPSED TIME MINUS TIME OF LAST MARK.

AREA O — SINCE MARKING IS ABOUT TO TAKE PLACE, MANEUVER COMPUTATION FLAG IS RESET TO INDICATE THAT ESTIMATED STATE IS GOING TO BE UPDATED AND A NEW MANEUVER COMPUTATION WILL BE NECESSARY.

AREA P — TAKING A MARK:
CHECK IF TIME SINCE LAST MARK IS GREATER THAN MINIMUM ALLOWED TIME BETWEEN MARKS ON THIS PROCEDURE. IF NOT, TURN TO NEXT CARD.
CALL P20 WITH CURRENT TIME, CURRENT ESTIMATED CARTESIAN STATE, AND SENSOR TYPE FLAG.
UPDATE THE MARK COUNTER. THE NM(I)=4 OPTION IS A LATE ADDITION COMBINING A RANGE AND OPTICS MARK. FOR THIS OPTION, BOTH THE RANGE AND OPTICS MARK COUNTERS ARE INCREMENTED.
SINCE THE CARTESIAN ESTIMATED STATE IS NOW UPDATED, IT IS NECESSARY TO RECONSTRUCT THE BETELGEUSE ESTIMATED STATE FOR INTEGRATION BY RK.
TRANSFER REVISED ESTIMATES OF SENSOR BIASES TO BETELGEUSE ESTIMATED VECTOR.
DEFINE TIME OF LAST MARK ON THIS PROCEDURE AS CURRENT PROGRAM TIME.

AREA Q — IF A MARKING PROCEEDURE HAS JUST TERMINATED, THE ICOMP FLAG IS EQUAL 1. IN THIS CASE, AREA Q COMPUTES A MANEUVER BASED ON THE VALUE OF NGUIDE, ONCE FOR THE ACTUAL STATES AND ONCE FOR THE ESTIMATED. IF ICOMP≠1, CONTROL IS TRANSFERRED TO THE MANEUVER APPLICATION AREA (R) TO SEE IF IT IS TIME FOR A MANEUVER.
COMPUTE A MANEUVER (RECYCLE OR FINAL)
LOAD ACTUAL BETELGEUSE STATE INTO GUIDANCE OVERLAY LOCATIONS

AREA Q

SET OVERLAY RETURN FLAG(NOVER) TO 1.
CALL GUIDANCE OVERLAY
CALL STORE1 TO STORE ACTUAL DELTAV.(CALLS TO THE STORE RESULT IN STORAGE OF INFORMATION ONLY IF NST(NGUIDE)=1 (SEE PAGE 5, CARD #4)).
CALL FOR BLOCK PRINT AT MANEUVER COMPUTATION.
SET OVERLAY RETURN FLAG, NOVER=2.
LOAD ESTIMATED STATES.
CALL GUIDANCE OVERLAY.
CALL STORE2 TO STORE ESTIMATED DELTAV.
RESET OVERLAY RETURN FLAG.
SET ICOMP=2 TO ADVERTISE THAT A MANEUVER HAS BEEN COMPUTED AND IS AVAILABLE.

AREA R

COMPUTE TIME TO GO UNTIL IGNITION.
SEE IF TGN IS LESS THAN 1 SECOND. IF IT IS, GO TO AREA R1 TO CALCULATE A MANEUVER IF THIS IS NOT ALREADY DONE.
IF TGN IS GREATER THAN 1 SECOND, SEE IF IT IS LESS THAN ONE INTEGRATION STEP. IF NOT, EXIT ROUTINE.
IF TGN IS LESS THAN 1 INTEGRATION STEP, SAVE THE DIFFERENCE BETWEEN TGN AND STEP FOR USE ON THE NEXT (SYNCH) PASS AFTER THE MANEUVER APPLICATION.
ALSO SAVE THE NOMINAL STEP SIZE.
DEFINE NEXT INTEGRATION STEP SIZE AS EQUAL TGN.
SET STEP SYNCH FLAG (ISTEP=1).
EXIT ROUTINE.

AREA RO

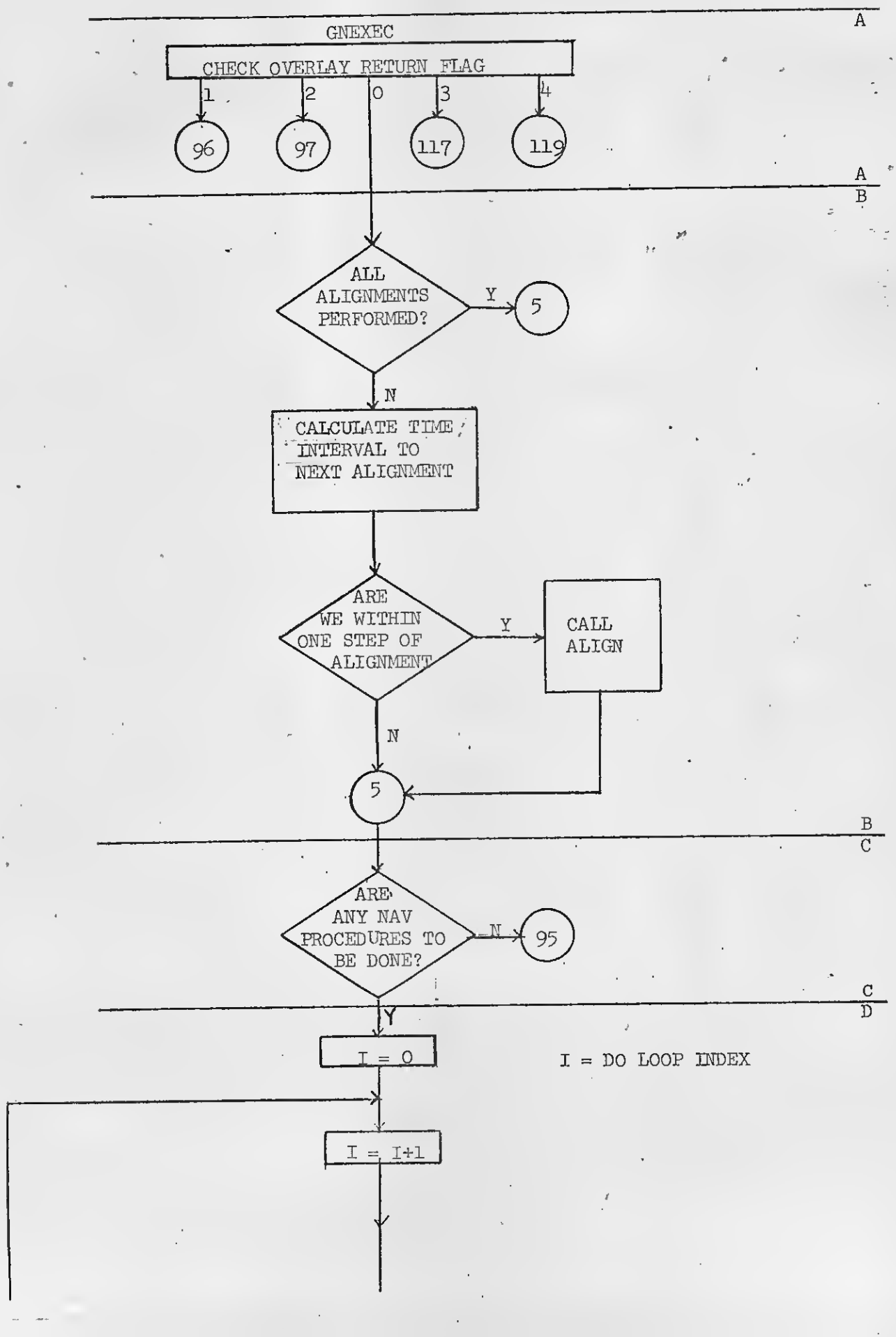
ON NEXT PASS AFTER ONE ON WHICH TGN WAS LESS THAN STEP, TGN WILL BE ZERO. AREA RO IS THEN VISITED AS A RESULT OF THE INSTRUCTION WHICH ASKS IF TGN IS LESS THAN 1 SECOND.
STEP SYNCH FLAG IS INCREMENTED TO INDICATE NEXT PASS IS 'EVENING' STEP.
P(1) SET EQUAL TO T2

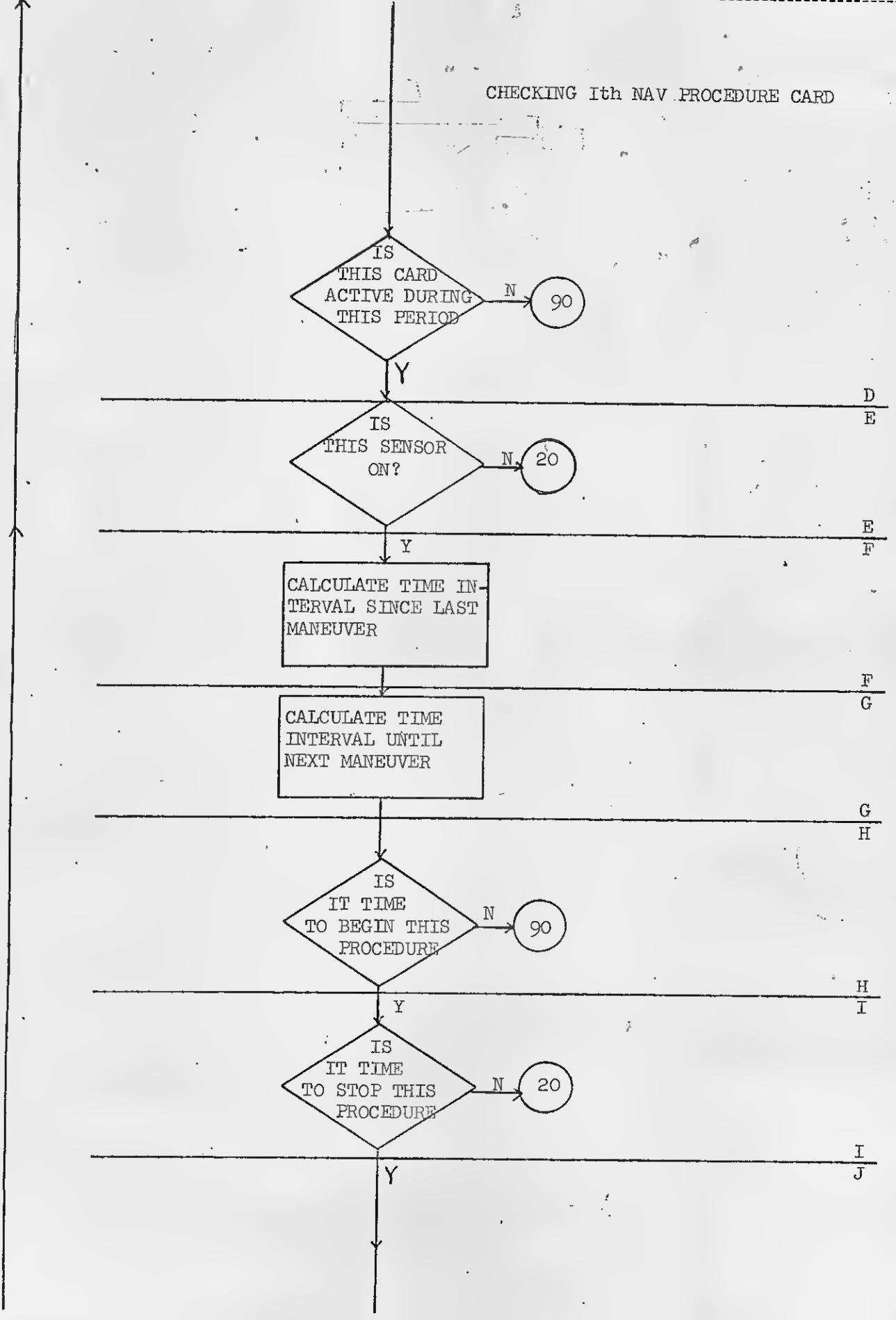
AREA R1

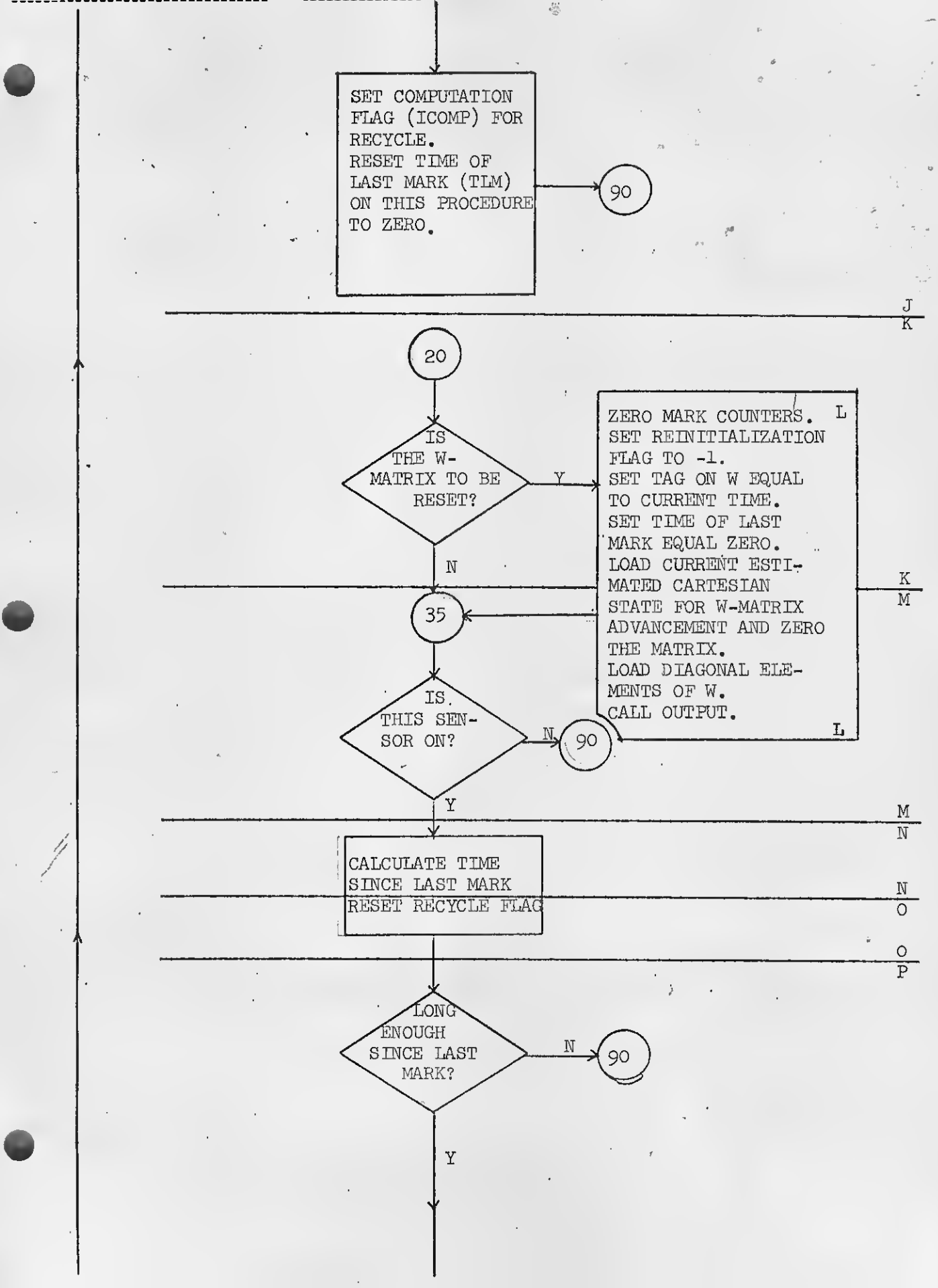
CHECK IS MADE TO SEE IF MANEUVER HAS BEEN COMPUTED. IF SO, PROCEED DIRECTLY TO APPLICATION INSTRUCTIONS.
IF MANEUVER HAS NOT BEEN DEFINED, SEQUENCE OF COMPUTATIONAL INSTRUCTIONS IS PERFORMED IDENTICAL TO AREA Q.

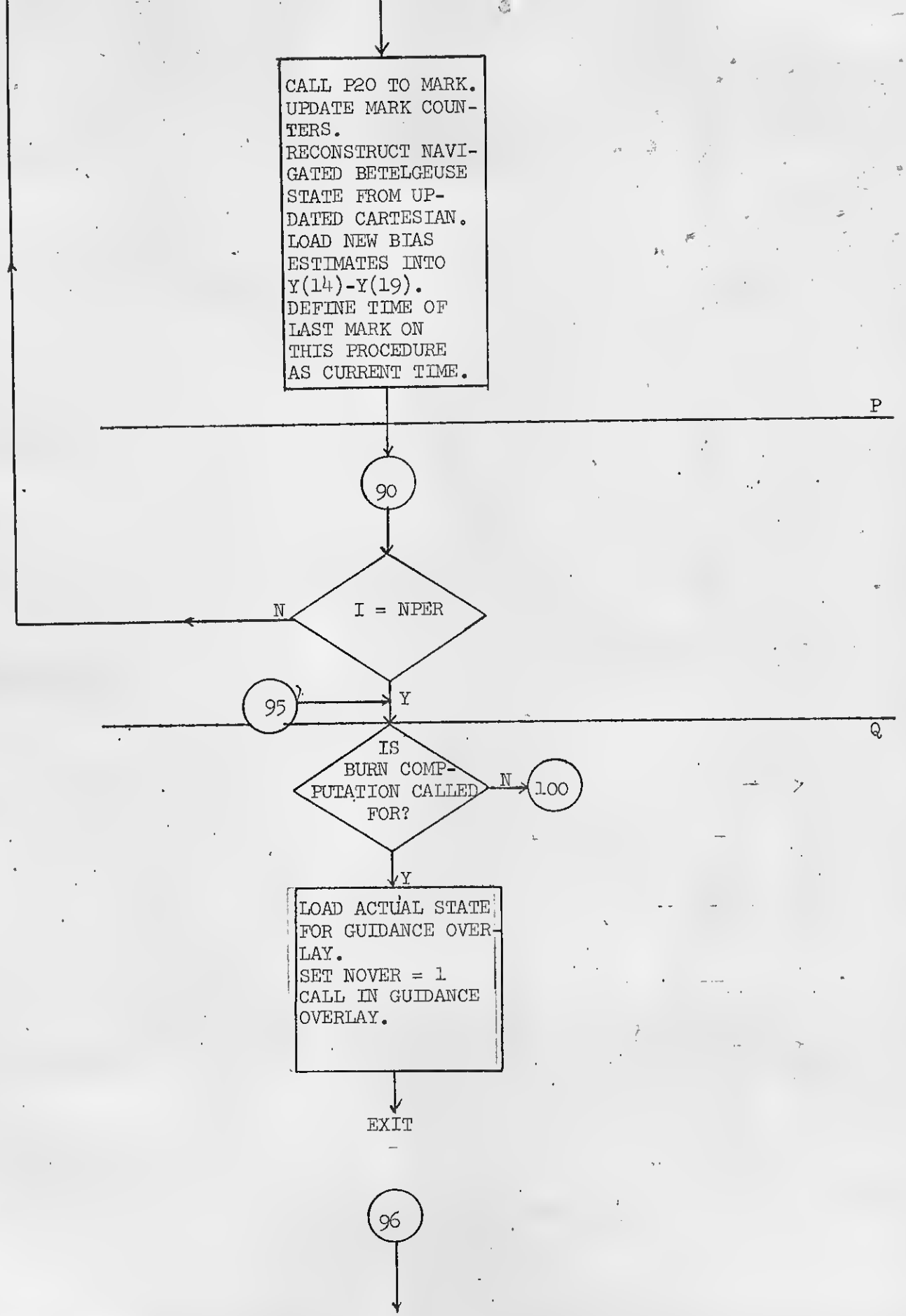
AREA R2

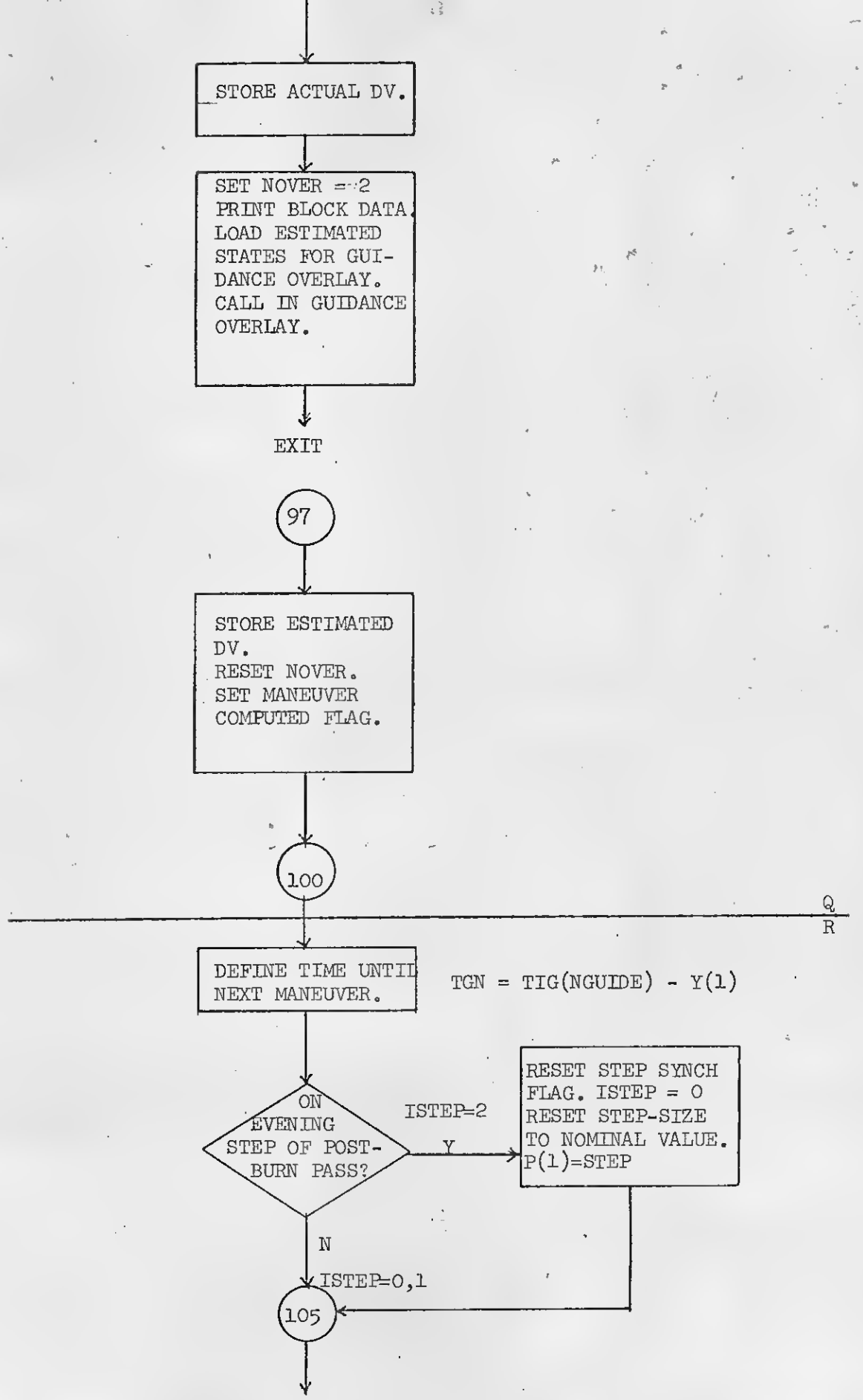
INCREMENT NGUIDE
CALL DELTAV APPLICATION ROUTINE.
STORE STATES AT MANEUVER
EXIT ROUTINE.

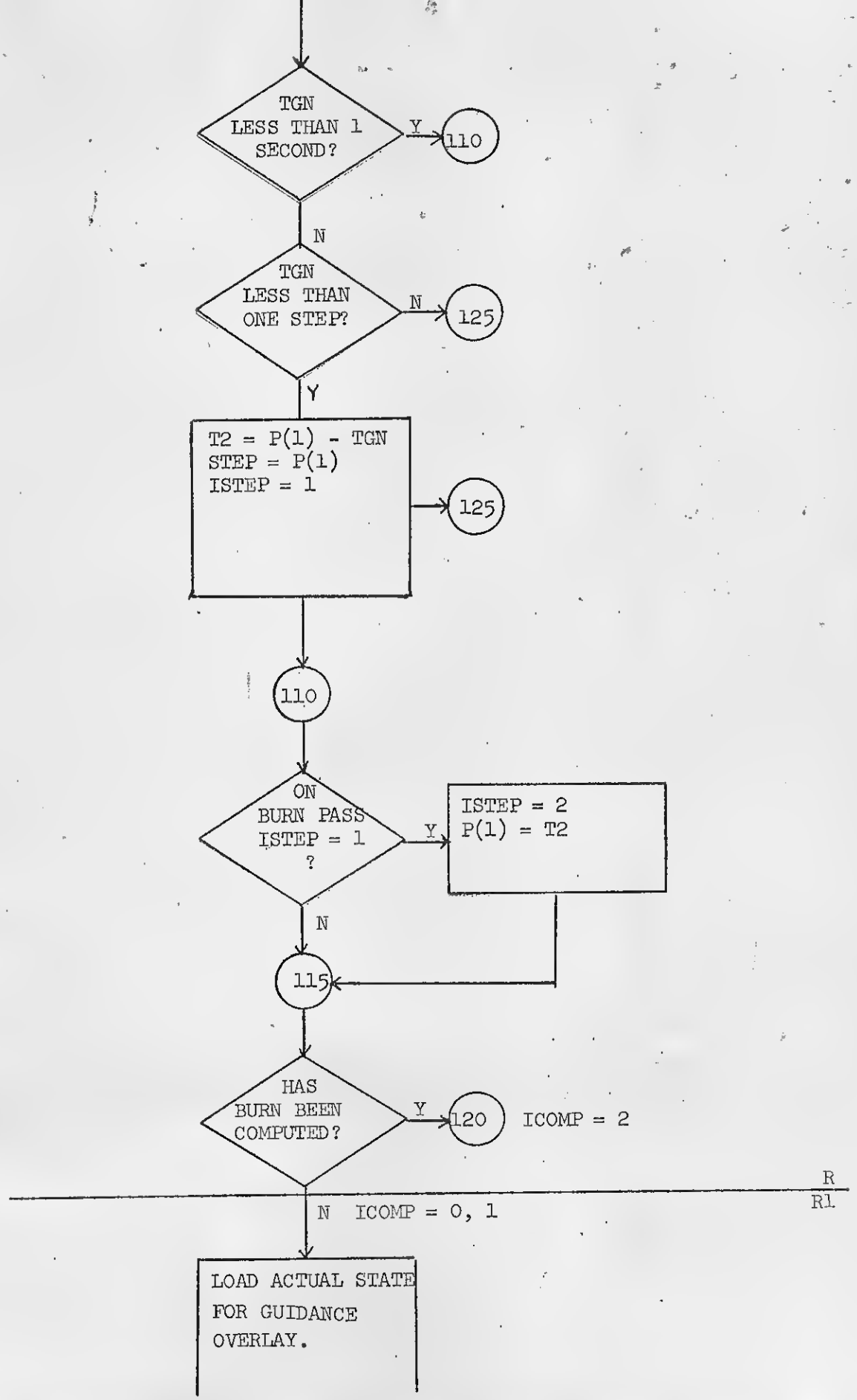

A
GNEXEC
CHECK OVERLAY RETURN FLAG
1
2
0
3
4
96
97
117
119
A
B
ALL ALIGNMENTS PERFORMED?
Y
5
N
CALCULATE TIME INTERVAL TO NEXT ALIGNMENT
ARE WE WITHIN ONE STEP OF ALIGNMENT
Y
CALL ALIGN
N
5
B
C
ARE ANY NAV PROCEDURES TO BE DONE?
N
95
C
D
Y
I = 0
I = I+1
I = DO LOOP INDEX

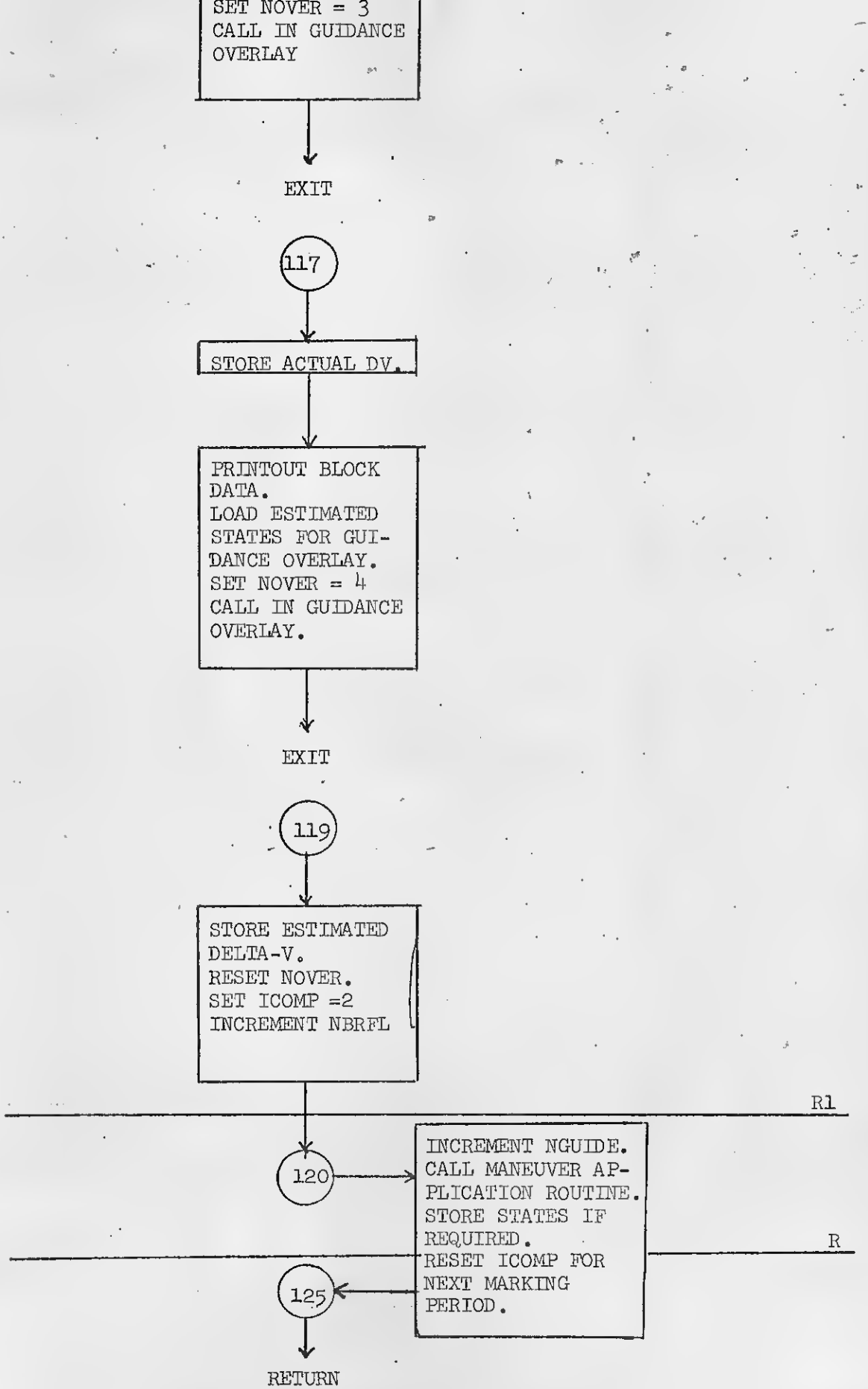

CHECKING Ith NAV PROCEDURE CARD
IS THIS CARD ACTIVE DURING THIS PERIOD
N
90
Y
D
E
IS THIS SENSOR ON?
N
20
E
F
Y
CALCULATE TIME INTERVAL SINCE LAST MANEUVER
F
G
CALCULATE TIME INTERVAL UNTIL NEXT MANEUVER
G
H
IS IT TIME TO BEGIN THIS PROCEDURE
N
90
H
I
Y
IS IT TIME TO STOP THIS PROCEDURE
N
20
I
J
Y


SET COMPUTATION FLAG (ICOMP) FOR RECYCLE.
RESET TIME OF LAST MARK (TLM) ON THIS PROCEDURE TO ZERO.
90
J
K
20
IS THE W-MATRIX TO BE RESET?
Y
N
ZERO MARK COUNTERS.
SET REINITIALIZATION FLAG TO -1.
SET TAG ON W EQUAL TO CURRENT TIME.
SET TIME OF LAST MARK EQUAL ZERO.
LOAD CURRENT ESTIMATED CARTESIAN STATE FOR W-MATRIX ADVANCEMENT AND ZERO THE MATRIX.
LOAD DIAGONAL ELEMENTS OF W.
CALL OUTPUT.
L
L
K
M
35
IS THIS SENSOR ON?
N
90
Y
M
N
CALCULATE TIME SINCE LAST MARK
RESET RECYCLE FLAG
N
O
O
P
LONG ENOUGH SINCE LAST MARK?
N
90
Y


CALL P20 TO MARK.
UPDATE MARK COUNTERS.
RECONSTRUCT NAVIGATED BETELGEUSE STATE FROM UPDATED CARTESIAN.
LOAD NEW BIAS ESTIMATES INTO Y(14)-Y(19).
DEFINE TIME OF LAST MARK ON THIS PROCEDURE AS CURRENT TIME.
P
90
I = NPER
N
95
Y
Q
IS BURN COMPUTATION CALLED FOR?
N
100
Y
LOAD ACTUAL STATE FOR GUIDANCE OVERLAY.
SET NOVER = 1
CALL IN GUIDANCE OVERLAY.
EXIT
96


STORE ACTUAL DV.
SET NOVER = 2
PRINT BLOCK DATA.
LOAD ESTIMATED STATES FOR GUIDANCE OVERLAY.
CALL IN GUIDANCE OVERLAY.
EXIT
97
STORE ESTIMATED DV.
RESET NOVER.
SET MANEUVER COMPUTED FLAG.
100
Q
R
DEFINE TIME UNTIL NEXT MANEUVER.
TGN = TIG(NGUIDE) - Y(1)
ON EVENING STEP OF POST-BURN PASS?
ISTEP=2
Y
RESET STEP SYNCH FLAG. ISTEP = 0
RESET STEP-SIZE TO NOMINAL VALUE.
P(1)=STEP
N
ISTEP=0,1
105


TGN LESS THAN 1 SECOND?
Y
110
N
TGN LESS THAN ONE STEP?
N
125
Y
T2 = P(1) - TGN
STEP = P(1)
ISTEP = 1
125
110
ON BURN PASS ISTEP = 1 ?
Y
ISTEP = 2
P(1) = T2
N
115
HAS BURN BEEN COMPUTED?
Y
120
ICOMP = 2
R
R1
N ICOMP = 0, 1
LOAD ACTUAL STATE FOR GUIDANCE OVERLAY.


SET NOVER = 3
CALL IN GUIDANCE OVERLAY
EXIT
117
STORE ACTUAL DV.
PRINTOUT BLOCK DATA.
LOAD ESTIMATED STATES FOR GUIDANCE OVERLAY.
SET NOVER = 4
CALL IN GUIDANCE OVERLAY.
EXIT
119
STORE ESTIMATED DELTA-V.
RESET NOVER.
SET ICOMP =2
INCREMENT NBRFL
R1
120
INCREMENT NGUIDE.
CALL MANEUVER APPLICATION ROUTINE.
STORE STATES IF REQUIRED.
RESET ICOMP FOR NEXT MARKING PERIOD.
R
125
RETURN

```
      SUBROUTINE GNEXEC                                                 GNEXEC     2
C                                                                       GNEXEC     3
C     SUBROUTINE TO CONTROL THE EXECUTION OF NAVIGATION PROCEDURES,     GNEXEC     4
C     GUIDANCE COMPUTATIONS AND MANEUVER APPLICATIONS                   GNEXEC     5
C                                                                       GNEXEC     6
      COMMON VAR                                                        GNEXEC     7
      DIMENSION VAR(5600), Y(100), DYDX(100), Q(100), FIRSTY(100)       GNEXEC     8
     *,         NTEGER(100), D(100), P(5000)                           GNEXEC     9
      EQUIVALENCE (VAR(1),Y(1))                                         GNEXEC    10
     *,           (VAR(101),DYDX(1))                                   GNEXEC    11
     *,           (VAR(201),Q(1))                                      GNEXEC    12
     *,           (VAR(301),FIRSTY(1))                                 GNEXEC    13
     *,           (VAR(401),NTEGER(1))                                 GNEXEC    14
     *,           (VAR(501),D(1))                                      GNEXEC    15
     *,           (VAR(601),P(1))                                      GNEXEC    16
      DIMENSION SAVE(950), BLK(700), DATA(350), COV(24,24)              GNEXEC    17
      EQUIVALENCE (P(350),SAVE(1))                                      GNEXEC    18
     *,           (P(1300),BLK(1))                                     GNEXEC    19
     *,           (P(4074),DATA(1))                                    GNEXEC    20
     *,           (P(4424),COV(1,1))                                   GNEXEC    21
      DIMENSION QQ(4), SIG(4), C(10), REFMAT(3,3), XNBN(3), YNBN(3)     GNEXEC    22
     *,         ZNBN(3), NE(10), NM(10), DTL(10), DTN(10), DTM(10)     GNEXEC    23
     *,         USP(10), USV(10), UTP(10), UTV(10), SR(10), SRD(10)    GNEXEC    24
     *,         SO(10), SC1(10), SC2(10), NW(10), TLM(10), NS(3)       GNEXEC    25
     *,         ZTZ(4), SZ(4), TALIGN(10), XNBE(3), YNBE(3), ZNBE(3)   GNEXEC    26
     *,         X(18), WE(18,27)                                       GNEXEC    27
      EQUIVALENCE (SAVE(1),QQ(1)),          (SAVE(5),SIG(1))          GNEXEC    28
     *,           (SAVE(9),C(1)),          (SAVE(19),REFMAT(1,1))    GNEXEC    29
     *,           (SAVE(28),XNBN(1)),      (SAVE(31),YNBN(1))        GNEXEC    30
     *,           (SAVE(34),ZNBN(1)),      (SAVE(37),NE(1))          GNEXEC    31
     *,           (SAVE(47),NM(1)),        (SAVE(57),DTL(1))         GNEXEC    32
     *,           (SAVE(67),DTN(1)),       (SAVE(77),DTM(1))         GNEXEC    33
     *,           (SAVE(87),USP(1)),       (SAVE(97),USV(1))         GNEXEC    34
     *,           (SAVE(107),UTP(1)),      (SAVE(117),UTV(1))        GNEXEC    35
     *,           (SAVE(127),SR(1)),       (SAVE(137),SRD(1))        GNEXEC    36
     *,           (SAVE(147),SO(1)),       (SAVE(157),SC1(1))        GNEXEC    37
     *,           (SAVE(167),SC2(1)),      (SAVE(177),NW(1))         GNEXEC    38
     *,           (SAVE(187),TLM(1)),      (SAVE(197),NS(1))         GNEXEC    39
     *,           (SAVE(200),ZTZ(1)),      (SAVE(204),SZ(1))         GNEXEC    40
     *,           (SAVE(208),TALIGN(1)),   (SAVE(218),NALIGN)        GNEXEC    41
     *,           (SAVE(229),XNBE(1)),     (SAVE(232),YNBE(1))       GNEXEC    42
     *,           (SAVE(235),ZNBE(1)),     (SAVE(258),X(1))          GNEXEC    43
     *,           (SAVE(276),WE(1,1))                                GNEXEC    44
```

```
      EQUIVALENCE (C(1),TW)                                          GNEXEC     45
     *,           (C(8),STEP)                                        GNEXEC     46
     *,           (C(9),T2)                                          GNEXEC
     *,           (C(10),TGN)                                        GNEXEC
      EQUIVALENCE (NTEGER(12),NPER)                                  GNEXEC     49
     *,           (NTEGER(29),NGUIDE)                                GNEXEC     50
     *,           (NTEGER(30),ICOMP)                                 GNEXEC     51
     *,           (NTEGER(31),ISTEP)                                 GNEXEC     52
     *,           (NTEGER(32),NOVER)                                 GNEXEC     53
     *,           (NTEGER(33),LIGN)                                  GNEXEC     54
     *,           (NTEGER(35),NGATE)                                 GNEXEC     55
     *,           (NTEGER(36),NBRFL)                                 GNEXEC     56
```

SUBROUTINE GNEXEC COC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
      DIMENSION DU(15), DV(15), DW(15), TFI(15)                      GNEXEC     57
      EQUIVALENCE (P(2141),TFI(1))                                   GNEXEC     58
     *,           (P(2156),DU(1))                                    GNEXEC     59
     *,           (P(2171),DV(1))                                    GNEXEC     60
     *,           (P(2186),DW(1))                                    GNEXEC     61
      IF(NOVER.GT.0) GO TO(96,97,117,119) NOVER                      GNEXEC     62
C                                                                    GNEXEC     63
C     CHECK IF TIME TO PERFORM AN ALIGNMENT                          GNEXEC     64
      IF(LIGN.GE.NALIGN) GO TO 5                                     GNEXEC     65
      KLIGN=LIGN+1                                                   GNEXEC     66
      DTLIGN=ABS(Y(1) - TALIGN(KLIGN))                               GNEXEC     67
      IF(DTLIGN.GE.P(1)) GO TO 5                                     GNEXEC     68
      CALL ALIGN                                                     GNEXEC     69
    5 CONTINUE                                                       GNEXEC     70
C     CHECK IF ANY NAVIGATION PROCEDURES CALLED FOR ON THIS RUN      GNEXEC     71
      IF(NPER.EQ.0) GO TO 95                                         GNEXEC     72
C     CYCLE THROUGH LIST OF DEFINED PROCEDURES                       GNEXEC     73
      DO 90 I=1,NPER                                                 GNEXEC     74
C     CHECK IF PROCEDURE APPLICABLE TO THIS PREMANEUVER PERIOD       GNEXEC     75
      IF(NE(I).NE.NGUIDE) GO TO 90                                   GNEXEC     76
C     CHECK IF THIS SENSOR IS OFF                                    GNEXEC     77
        IF(NM(I).EQ.0) GO TO20                                       AUTOW      1
C     DEFINE TIME SINCE LAST TIG                                     GNEXEC     79
      IF(NGUIDE.EQ.1) TGL=Y(1)                                       GNEXEC     80
```

```
C     DEFINE TIME UNTIL NEXT TIG                                  G     GNEXEC  82
      TGN=TFI(NGUIDE) - Y(1)                                      G     GNEXEC  83
C     CHECK IF IT IS TIME TO BEGIN THIS PROCEDURE                 H     GNEXEC  84
      IF(TGL.LT.DTL(I)) GO TO 90                                  H     GNEXEC  85
C     CHECK IF THIS PROCEDURE IS TO BE TERMINATED                 I     GNEXEC  86
      IF(TGN.GT.OTN(I)) GO TO 20                                  I     GNEXEC  87
C     SET MANEUVER COMPUTATION FLAG AND RESET W INITIALIZATION FLAG     GNEXEC  88
      IF(ICOMP.NE.2) ICOMP=1                                      J     GNEXEC  89
      TLM(I)=0.                                                         GNEXEC  90
      GO TO 90                                                    J     GNEXEC  91
   20 CONTINUE                                                    K     GNEXEC  92
C     CHECK IF W IS TO BE REINITIALIZED                                 GNEXEC  93
      IF(NW(I).NE.0) GO TO 35                                     K     GNEXEC  94
C     ZERO MARK COUNTERS                                          L     GNEXEC  95
      NS(1)=0                                                           GNEXEC  96
      NS(2)=0                                                           GNEXEC  97
      NS(3)=0                                                           GNEXEC  98
C     SET W REINITIALIZED FLAG                                          GNEXEC  99
      NW(I)=-1                                                          GNEXEC 100
C     DEFINE TIME TAG ON W                                              GNEXEC 101
      TW=Y(1)                                                           GNEXEC 102
      TLM(I)=0.                                                         GNEXEC 103
C     STORE INITIAL UNIT VECTORS FOR ADVW                               GNEXEC 104
      CALL UVEC(Y(38),Y(39),Y(40),C(2))   NOT USED CAN                  GNEXEC 105
      CALL UVEC(Y(44),Y(45),Y(46),C(5))   BE DELETED                    GNEXEC 106
C     ZERO THE W-MATRIX                                                 GNEXEC 107
      DO 25 J=1,18                                                      GNEXEC 108
C     STORE STATE FOR ADVW                                              GNEXEC 109
      X(J)=Y(37+J)                                                      GNEXEC 110
      DO 25 K=1,27                                                      GNEXEC 111

SUBROUTINE GNEXEC          CDC 6600 FTN V3.0-P308 OPT=1  08/29/72  11.32.27.

   25 WE(J,K)=0.                                                        GNEXEC 112
C     LOAD THE DIAGONAL ELEMENTS OF W                                   GNEXEC 113
      DO 30 J=1,3                                                       GNEXEC 114
      WE(J,J)=USP(I)                                                    GNEXEC 115
      WE(J+3,J+3)=USV(I)                                                GNEXEC 116
      WE(J+6,J+6)=UTP(I)                                                GNEXEC 117
```

```
      WE(J+14,J+14)=SO(I)                                          GNEXEC  119
      WE(13,13)=SR(I)                                              GNEXEC  120
      WE(14,14)=SRO(I)                                             GNEXEC  1
      WE(17,17)=SC1(I)                                             GNEXEC  1..
      WE(18,18)=SC2(I)                                             GNEXEC  123
      CALL POPW                                          L         GNEXEC  124
   35 CONTINUE                                           M         GNEXEC  125
        IF(NM(I).EQ.0) GO TO 90                          M         AUTOW     2
C     DEFINE TIME SINCE LAST MARK                        N         GNEXEC  126
      DTLM=Y(1) - TLM(I) + 1.                            N         GNEXEC  127
C     SINCE MARKING IN PROGRESS, RESET MANEUVER COMPUTATION FLAG   GNEXEC  128
      ICOMP=0                                            O         GNEXEC  129
C     CHECK IF SUFFICIENT TIME HAS ELAPSED FOR ANOTHER MARK        GNEXEC  130
      IF(DTLM.LT.DTM(I)) GO TO 90                        P         GNEXEC  131
C     CALL P20 TAKE AND INCORPORATE MARK                           GNEXEC  132
      CALL P20(Y(1),Y(38),NM(I))                                   GNEXEC  133
C     UPDATE MARK COUNTER                                          GNEXEC  134
      INS=NM(I)                                                    GNEXEC  135
      IF(NM(I).EQ.4) NS(3)=NS(3) + 1                               GNEXEC  136
      IF(NM(I).EQ.4) NS(2)=NS(2) + 1                               GNEXEC  137
      IF(NM(I).LE.3) NS(INS)=NS(INS) + 1                           GNEXEC  138
C     RECONSTRUCT NAVIGATED BETELGEUSE VECTOR AFTER HAVING TAKEN MARK  GNEXEC  139
      CALL CART2(Y(38),Y(2))                                       GNEXEC  140
      DO 85 J=1,6                                                  GNEXEC  141
      Y(J+13) = Y(J+49)                                            GNEXEC  142
   85 CONTINUE                                                     GNEXEC  143
      TLM(I)=Y(1)                                        P         GNEXEC  144
   90 CONTINUE                                                     GNEXEC  145
   95 CONTINUE                                                     GNEXEC  146
C     CHECK IF MANEUVER COMPUTATION FLAG IS SET          Q         GNEXEC  147
      IF(ICOMP.NE.1) GO TO 100                                     GNEXEC  148
C     LOAD ENVIRONMENT, CALL COMPUTATIONS, SET RETURN FLAG         GNEXEC  149
      P(2001)=Y(1)                                                 GNEXEC  150
      DO 98 J=1,12                                                 GNEXEC  151
   98 P(2001+J)=Y(19+J)                                            GNEXEC  152
      NOVER=1                                                      GNEXEC  153
      CALL DUMMY2                                                  GNEXEC  154
   96 CONTINUE                                                     GNEXEC  155
      CALL STORE1                                                  GNEXEC  156
      NOVER=2                                                      GNEXEC  157
      P(2001)=Y(1)                                                 GNEXEC  158
      CALL POPW                                                    GNEXEC  159
      DO 99 J=1,12                                                 GNEXEC  160
   99 P(2001+J)=Y(1+J)                                             GNEXEC  161
      CALL DUMMY2                                                  GNEXEC  162
   97 CONTINUE                                                     GNEXEC  163
      CALL STORE2                                                  GNEXEC  164
      NOVER=0                                                      GNEXEC  165
```

```
C       INCREMENT NGUIDE                                                GNEXEC  166
        IF(NBRFL.GT.0) NGATE=NGATE + 1                                  GNEXEC  167
C       SET COMPUTATIONS PERFORMED FLAG                                 GNEXEC  168
        ICOMP=2                                                         GNEXEC  169
  100   CONTINUE                                                  Q     GNEXEC  170
C       DEFINE TIME TO NEXT BURN                                  R     GNEXEC  171
        TGN=TFI(NGUIDE) - Y(1)                                          GNEXEC  172
C       CHECK IF FINAL PASS THROUGH MANEUVER APPLICATION SEQUENCE       GNEXEC  173
        IF(ISTEP.NE.2) GO TO 105                                        GNEXEC  174
C       RESET ISTEP AND CHANGE TO ORIGINAL STEP-SIZE                    GNEXEC  175
        ISTEP=0                                                         GNEXEC  176
        P(1)=STEP                                                       GNEXEC  177
  105   CONTINUE                                                        GNEXEC  178
C       CHECK IF TIME FOR MANEUVER APPLICATION                          GNEXEC  179
        IF(ABS(TGN).LE.1.) GO TO 110                                    GNEXEC  180
C       CHECK IF MANEUVER IS LESS THAN ONE STEP AWAY                    GNEXEC  181
        IF(ABS(TGN).GT.P(1)) GO TO 125                                  GNEXEC  182
C       SET STEP-SIZE EQUAL TIME-TO-GO, STORE STEP                      GNEXEC  183
        T2=P(1) - TGN                                                   GNEXEC  184
        STEP=P(1)                                                       GNEXEC  185
        P(1)=TGN                                                        GNEXEC  186
        ISTEP=1                                                         GNEXEC  187
        GO TO 125                                                       GNEXEC  188
  110   CONTINUE                                                  RO    GNEXEC  189
        IF(ISTEP.NE.1) GO TO 115                                        GNEXEC  190
        ISTEP=2                                                         GNEXEC  191
        P(1)=T2                                                         GNEXEC  192
  115   CONTINUE                                                        GNEXEC  193
C       CHECK IF MANEUVER HAS BEEN COMPUTED                             GNEXEC  194
        IF(ICOMP.EQ.2) GO TO 120                                  RO    GNEXEC  195
C       CALL  MANEUVER COMPUTATIONS FOR ENVIRONMENT AND NAVIGATED STATES GNEXEC  196
        P(2001)=Y(1)                                              R1    GNEXEC  197
        DO 116 J=1,12                                                   GNEXEC  198
  116   P(2001+J)=Y(19+J)                                               GNEXEC  199
        NOVER=3                                                         GNEXEC  200
C       LOAD GUIDANCE OVERLAY                                           GNEXEC  201
              CALL DUMMY2                                               GNEXEC  202
  117   CONTINUE                                                        GNEXEC  203
        CALL STORE1                                                     GNEXEC  204
        P(2001)=Y(1)                                                    GNEXEC  205
        CALL POPW                                                       GNEXEC  206
        DO 118 J=1,12                                                   GNEXEC  207
```

```
      NOVER=4                                                    GNEXEC   209
C     LOAD GUIDANCE OVERLAY                                      GNEXEC   210
            CALL DUMMY2                                          GNEXEC   2__
  119 CONTINUE                                                   GNEXEC   2__
      CALL STORE2                                                GNEXEC   213
      NOVER=0                                                    GNEXEC   214
      ICOMP=2                                                    GNEXEC   215
C     INCREMENT NGUIDE                                           GNEXEC   216
      IF(NBRFL.GT.0) NGATE=NGATE + 1                        R1   GNEXEC   217
  120 CONTINUE                                                   GNEXEC   218
C     PERFORM MANEUVER                                      R2   GNEXEC   219
      NGUIDE=NGUIDE+1                                            GNEXEC   220
```

UBROUTINE GNEXEC CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
      NV=NGUIDE - 1                                              GNEXEC   221
      CALL DELTAV(DU(NV),DV(NV),DW(NV))                          GNEXEC   222
      CALL STORE3                                                GNEXEC   223
      ICOMP=0                                               R2   GNEXEC   224
  125 CONTINUE                                                   GNEXEC   225
      RETURN                                                     GNEXEC   226
      END                                                        GNEXEC   227
```

DELTA-V

ROUTINE DELTAV

ROUTINE APPLIES MANEUVER BASED ON ESTIMATED STATE COMPUTATION TO ESTIMATED S/C STATE. ALSO COMPUTES EFFECT OF ACCELEROMETER SCALE FACTOR ERROR, PLATFORM MISALIGNMENT AND CUTOFF UNCERTAINTY ON APPLICATION TO ACTUAL STATE. VECTORS AND MATRICES INVOLVED IN THESE CALCULATIONS ARE:

DXD	NOMINAL DELTAV IN LOCAL VERTICAL FRAME. THIS DELTAV IS COMPUTED FROM ESTIMATED STATE AND IS THE ONBOARD ESTIMATE OF THE APPLIED BURN.
SD	VECTOR OF ACCELEROMETER SCALE FACTOR ERRORS.
S	SCALE FACTOR DISTURBANCE MATRIX. Hold be correlated
VN	VECTOR OF CUTOFF UNCERTAINTY ERROR.
UX,UY,UZ	UNIT VECTORS OF THE ESTIMATED LOCAL VERTICAL FRAME.
DUM^1	TRANSFORMATION FROM BRF TO ESTIMATED LOCAL VERTICAL FRAME.
$DUMT^1$	TRANSFORMATION FROM ESTIMATED LOCAL VERTICAL FRAME TO BRF. (= DUM^{1T})
REFMAT	TRANSFORMATION FROM BRF TO ESTIMATED PLATFORM AXES.
DVI	NOMINAL DELTAV IN BRF.
DVSM	DELTAV IN ESTIMATED PLATFORM FRAME WITH SCALE FACTOR ERRORS APPLIED.
DVSME	DVSM + VN
GAMD	MATRIX OF PLATFORM DRIFT ERROR ANGLES.
DUM^2	S x REFMAT
DUM^3	GAMD x REFMAT
$DUMT^2$	$REFMAT^T$ x $GAMD^T$
DVIE	DELTAV IN BRF DISTURBED BY SCALE FACTOR ERROR, CUT-OFF UNCERTAINTY AND PLATFORM MISALIGNMENT.

$$DVIE = REFMAT^T \times GAMD^T \times [S \times REFMAT \times DUMT \times DXD + VN]$$

AREA A READ ARGUMENT DELTAV INTO OPERATING ARRAY.

AREA B IF NGUIDE IS STILL EQUAL TO ITS STARTING VALUE, THIS IS THE FIRST MANEUVER APPLICATION. IN THIS CASE, VISIT RANDOM NUMBER GENERATOR TO DEFINE SCALE FACTOR ERRORS FOR THIS CYCLE. OTHERWISE, PROCEED TO DEFINE CUTOFF ERROR IN AREA C.

AREA C DEFINE DIFFERENT RANDOM CUTOFF ERROR FOR EACH MANEUVER.

AREA D COMPUTE TRANSFORMATION FROM ESTIMATED LOCAL VERTICAL FRAME TO INERTIAL (BRF).
TRANSFORM LOCAL VERTICAL DELTAV'S TO BRF.

AREA E CONSTRUCT SCALE FACTOR MATRIX. THIS IS THE IDENTITY MATRIX WITH SCALE FACTOR ERRORS ADDED TO DIAGONAL.

AREA F REFMAT MATRIX IS TRANSFORM TO ESTIMATED PLATFORM AXES. EFFECT OF AREA F IS TO CONVERT BRF DELTAV TO ESTIMATED PLATFORM, MULTIPLY BY SCALE FACTOR ERRORS AND ADD CUT-OFF ERROR.

AREA G PLATFORM DRIFT ANGLES ARE D1,D2,D3. EFFECT OF AREA G IS TO TRANSFORM DELTAV TO ACTUAL PLATFORM FRAME AND RETURN FROM ACTUAL PLATFORM FRAME TO BRF.

AREA H CONVERT PERTURBED DELTAV TO INERTIAL FRAME.

AREA I ADVANCE W-MATRIX TO TIME OF BURN. THIS IS NECESSARY BECAUSE ESTIMATED STATE IS CHANGED BY APPLICATION OF DELTAV. SINCE ADVW USES PREVIOUS VALUE OF ESTIMATED STATE, THIS STATE MUST REFLECT THE MANEUVER. X, THE ADVW STORED STATE, WILL BE UPDATED IN AREA K.

AREA J CONSTRUCT A CARTESIAN FORM OF THE ACTUAL STATE.
ADD THE ACTUAL APPLIED DELTAV TO ACTUAL STATE.
ADD THE NOMINAL APPLIED DELTAV TO ESTIMATED STATE.

AREA K RECONSTRUCT ACTUAL BETELGEUSE STATE.
RECONSTRUCT ESTIMATED BETELGEUSE STATE.
UPDATE THE ADVW STORED STATE.

AREA L CONSTRUCT TRANSFORM TO LOCAL VERTICAL.
TRANSFORM ACTUAL APPLIED DELTAV TO LOCAL VERTICAL.
PRINT DELTAV APPLIED MESSAGE.

```
      SUBROUTINE DELTAV(DU,DV,DW)                                       DELTV     2
C                                                                       DELTV     3
      COMMON VAR                                                        DELTV     4
C                                                                       DELTV     5
      DIMENSION VAR(5600), Y(100),DXO(3)                                DELTV     6
     *,P(5000), NTEGER(100), BLK(700)                                   DELTV     7
     *,          VN(3),S(3,3)                                          DELTV     8
     *,          UX(3), UY(3), UZ(3), DUM(3,3), DUMT(3,3)              DELTV     9
     *,DVI(3), DVSM(3), DVSME(3), DVIE(3)                               DELTV    10
     *,GAMO(3,3), XE(12)                                                DELTV    11
     *,          SAVE(950)                                             DELTV    12
     *,REFMAT(3,3)                                                      DELTV    13
C     SUBROUTINE TO APPLY A LOCAL HORIZONTAL DELTA-V TO A BETEL VECTOR  DELTV    14
C                                                                       DELTV    15
C                                                                       DELTV    16
      EQUIVALENCE (VAR(1),Y(1))                                         DELTV    17
     *,           (VAR(401),NTEGER(1))                                 DELTV    18
     *,           (VAR(601),P(1))                                      DELTV    19
      EQUIVALENCE (P(350),SAVE(1))                                      DELTV    20
     *,           (P(1300),BLK(1))                                     DELTV    21
      DIMENSION X(18)                                                   DELTV    22
      EQUIVALENCE (SAVE(258),X(1))                                      DELTV    23
      EQUIVALENCE (BLK(4),S(1,1)) ,(BLK(13),VN(1))                      DELTV    24
     *,(BLK(16),UX(1)), (BLK(19),UY(1)), (BLK(22),UZ(1))                DELTV    25
     *,(BLK(25),DUM(1,1)), (BLK(34),DUMT(1,1)), (BLK(43),GAMO(1,1))     DELTV    26
     *,(BLK(52),DVSM(1)), (BLK(55),DVSME(1)), (BLK(58),DVIE(1))         DELTV    27
     *,(BLK(61),XE(1))                                                  DELTV    28
     *,(NTEGER(29),NGUIDE)                                              DELTV    29
      EQUIVALENCE (SAVE(19),REFMAT(1,1))                                DELTV    30
     *,(Y(98),D1), (Y(99),D2), (Y(100),D3)                              DELTV    31
C                                                                       DELTV    32
      COMMON/DELV/VARS,VARA,NFAMA,SD(3)                                 DELTV    33
C                                                                       DELTV    34
C     READ IN VELOCITIES                                          A     DELTV    35
      DXO(1)=DU                                                         DELTV    36
      DXO(2)=DV                                                         DELTV    37
      DXO(3)=DW.                                                  A     DELTV    38
C                                                                 B     DELTV    39
C     CHECK IF THIS IS THE FIRST MANEUVER OF THIS RUN                   DELTV    40
      NMAN=NGUIDE - NTEGER(3)                                           DELTV    41
      IF(NMAN.NE.1) GO TO 5                                             DELTV    42
C     CREATE SCALE FACTOR ERRORS FOR THIS RUN                           DELTV    43
```

```
      SD(2)=UNURN(0,NFAMA,1.,VARS)                                    DELTV  45
      SD(3)=UNURN(0,NFAMA,1.,VARS)                              B     DELTV  46
    5 CONTINUE                                                  C     DELTV
      DEFINE APPLICATION ERROR                                        DELTV
      VN(1)=UNURN(0,NFAMA,0.,VARA)                                    DELTV  49
      VN(2)=UNURN(0,NFAMA,0.,VARA)                                    DELTV  50
      VN(3)=UNURN(0,NFAMA,0.,VARA)                              C     DELTV  51
C     COMPUTE DELTAV IN ASSUMED REFERENCE FRAME                 D     DELTV  52
      CALL UVEC(Y(38),Y(39),Y(40),UX)                                 DELTV  53
      CALL UCROSS(UX,Y(41),UZ)                                        DELTV  54
      CALL UCROSS(UZ,UX,UY)                                           DELTV  55
      CALL TRN(UX,UY,UZ,DUM,DUMT)                                     DELTV  56
```

SUBROUTINE DELTAV CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
      CALL MATMUL(DUMT,DXD,DVI,3,3,1)                           D     DELTV  57
C     COMPUTE ACTUAL DELTAV APPLIED                             E     DELTV  58
C     COMPUTE SCALE FACTOR MATRIX                                     DELTV  59
      DO 10 I=1,3                                                     DELTV  60
      DO 10 J=1,3                                                     DELTV  61
      S(I,J)=0.                                                       DELTV  62
      IF(I.EQ.J) S(I,J)=SD(I)                                         DELTV  63
   10 CONTINUE                                                  E     DELTV  64
      CALL MATMUL(S,REFMAT,DUM,3,3,3)                           F     DELTV  65
      CALL MATMUL(DUM,DVI,DVSM,3,3,1)                                 DELTV  66
      CALL MATADD(DVSM,VN,DVSME,3,1,1)                          F     DELTV  67
C     COMPUTE PLATFORM MISALIGNMENT MATRIX                      G     DELTV  68
      CALL MAT(D3,D2,D1,1,3,2,GAMD)                                   DELTV  69
      CALL MATMUL(GAMD,REFMAT,DUM,3,3,3)                              DELTV  70
      CALL MATRAN(DUM,3,3,DUMT)                                 G     DELTV  71
      CALL MATMUL(DUMT,DVSME,DVIE,3,3,1)                        H     DELTV  72
      CALL ADVW(Y(1),Y(38))                                     I     DELTV  73
C     CONSTRUCT CARTESIAN ENVIRONMENT VECTOR                    J     DELTV  74
      CALL CART1(XE,Y(20))                                            DELTV  75
      DO 15 J=1,3                                                     DELTV  76
      XE(J+3)=XE(J+3) + DVIE(J)                                       DELTV  77
   15 Y(J+40)=Y(J+40) + DVI(J)                                  J     DELTV  78
C     RECONSTRUCT BETELGEUSE STATES                             K     DELTV  79
      CALL CART2(XE,Y(20))                                            DELTV  80
      CALL CART2(Y(38),Y(2))                                          DELTV  81
```

```
      DO 12 I=1,18                                                    DELTV     82
      X(I) = Y(37+I)                                                  DELTV     83
 12   CONTINUE                                                  K     DELTV     84
      CALL TRN(UX,UY,UZ,DUM,DUMT)                               L     DELTV     85
      CALL MATMUL(DUM,DVIE,DVI,3,3,1)                                 DELTV     86
      PRINT 100, (DXD(I),I=1,3), (DVI(I),I=1,3)                 L     DELTV     87
100   FORMAT(/1X,35HTHE NAVIGATION BURN ESTIMATE IS DU=,F10.3, 5H  DV=, DELTV     88
     1F10.3, 5H  DW=,F10.3,/1X,35HTHE ACTUAL COMPONENTS WERE      DU=,  DELTV     89
     2F10.3, 5H  DV=,F10.3, 5H  DW=,F10.3)                            DELTV     90
      RETURN                                                          DELTV     91
      END                                                             DELTV     92
```

STORE 1

ROUTINE STORE1

SUBROUTINE STORES DATA AT TIME OF MANEUVER APPLICATION FOR MANEUVERS SELECTED BY THE DATA STORAGE CONTROL CARD, SEE PAGE 5, CARD #4.

AREA A — IF ROUTINE IS ENTERED AT STORE1, CALL IS FOR THE PURPOSE OF STORING DELTA-V'S FROM ACTUAL STATE COMPUTATION. SET N=26 TO SELECT 25th ELEMENT OF 31 ELEMENT MANEUVER DATA VECTOR. (N=26 SELECTS 25th ELEMENT BECAUSE FIRST 31 ELEMENT ARRAY STARTS AT DATA(2)).

AREA B — IF ROUTINE IS ENTERED AT STORE2, CALL IS FOR PURPOSE OF STORING DELTA-V'S FROM ESTIMATED STATE COMPUTATION. SET N=29 TO SELECT 28th ELEMENT OF 31 ELEMENT ARRAY.

AREA C — CHECK NST(NGUIDE)=1 TO SEE IF DATA FROM THIS MANEUVER SHOULD BE STORED. IF NOT, EXIT ROUTINE.

AREA D — NSTAT INDEX KEEPS TRACK OF THE NUMBER OF 31-ELEMENT DATA VECTORS SO FAR STORED. THIS NUMBER MAY NOT EXCEED 8.
K IS THE STARTING LOCATION IN THE DATA ARRAY FOR THE NEW INFORMATION.
CALL TO VEC LOADS DU, DV, DW INTO DATA(K,K+1,K+2).

AREA E — IF ROUTINE IS ENTERED AT STORE3, CALL IS FOR THE PURPOSE OF STORING ACTUAL AND ESTIMATED STATES AT TIME OF BURN APPLICATION. SINCE NGUIDE HAS ALREADY BEEN INCREMENTED IN GNEXEC, DEFINE K AS NGUIDE-1, FOR THE MANEUVER JUST PERFORMED.
CHECK IF DATA FROM THIS MANEUVER SHOULD BE STORED. IF NOT, EXIT ROUTINE.

AREA F — CREATE CARTESIAN FORM OF ACTUAL BETELGEUSE STATE.
DEFINE STARTING LOCATION IN DATA ARRAY.
LOAD ACTUAL AND ESTIMATED CARTESIAN STATES.

AREA G — INCREMENT NSTAT TO SHOW ANOTHER STORED MANEUVER.

AREA H — STORE TIME OF MANEUVER IN LAST ELEMENT OF 31 ELEMENT ARRAY JUST WRITTEN.

```
      SUBROUTINE STORE1                                                 DELTV   93
                                                                        DELTV
C     SUBROUTINE TO STORE ASSORTED PARAMETERS IN THE DATA ARRAY         DELTV   95
C                                                                       DELTV   96
      COMMON VAR                                                        DELTV   97
C                                                                       DELTV   98
      DIMENSION VAR(5600)                                               DELTV   99
     *,         Y(100)                                                  DELTV  100
     *,         P(5000)                                                 DELTV  101
     *,         NTEGER(100)                                             DELTV  102
     *,         DATA(350)                                               DELTV  103
     *,         NST(15)                                                 DELTV  104
     *,         DU(15),DV(15),DW(15)                                    DELTV  105
     *,         XE(12)                                                  DELTV  106
C                                                                       DELTV  107
      EQUIVALENCE (VAR(1),Y(1))                                         DELTV  108
     *,           (VAR(401),NTEGER(1))                                 DELTV  109
     *,           (VAR(601),P(1))                                      DELTV  110
     *,           (VAR(934),NSTAT)                                     DELTV  111
     *,           (VAR(935),NST(1))                                    DELTV  112
     *,           (P(4074),DATA(1))                                    DELTV  113
     *,           (NTEGER(29),NGUIDE)                                  DELTV  114
     *,           (P(2156),DU(1)),(P(2171),DV(1)),(P(2186),DW(1))      DELTV  115
C                                                                       DELTV  116
      N=26                                                          A   DELTV  117
      GO TO 5                                                       A   DELTV  118
      ENTRY STORE2                                                  B   DELTV  119
      N=29                                                          B   DELTV  120
    5 CONTINUE                                                          DELTV  121
      IF(NST(NGUIDE).NE.1) GO TO 30                                 C   DELTV  122
      K=NSTAT*31 + N                                                D   DELTV  123
      CALL VEC(DU(NGUIDE),DV(NGUIDE),DW(NGUIDE),DATA(K))                DELTV  124
      GO TO 30                                                      D   DELTV  125
      ENTRY STORE3                                                  E   DELTV  126
      K=NGUIDE - 1                                                      DELTV  127
      IF(NST(K).NE.1) GO TO 30                                      E   DELTV  128
      CALL CART1(XE,Y(200))                                         F   DELTV  129
      J=NSTAT*31 + 1                                                    DELTV  130
      DO 20 I=1,12                                                      DELTV  131
      DATA(I+J)=XE(I)                                                   DELTV  132
   20 DATA(I+J+12)=Y(I+37)                                          F   DELTV  133
      NSTAT=NSTAT + 1                                               G   DELTV  134
      J=NSTAT*31 + 1                                                H   DELTV  135
      DATA(J)=Y(1)                                                  H   DELTV  136
   30 CONTINUE                                                          DELTV  137
      RETURN                                                            DELTV  138
      END                                                               DELTV  139
```

OUTDAT

ROUTINE OUTDAT

THIS ROUTINE DUMPS THE DATA(350) ARRAY AT THE END OF EACH MONTE CARLO CYCLE. THIS PROVIDES A VISUAL INSPECTION OF THE DATA BEING WRITTEN ONTO THE LOCAL MASS STORAGE FILE.

AREA A WRITE OUT THE CURRENT VALUE OF THE MONTE CARLO RUN INDEX.

AREA B DUMP THE 8 MANEUVER DATA VECTORS.

AREA C SAME AS AREA A.

AREA D DUMP THE 10 PLATFORM ALIGNMENT DATA VECTORS.

AREA E DUMP THE USER SPECIFIED PORTION OF DATA.

```
      SUBROUTINE OUTDAT                                                 DELTV    140
C     SUBROUTINE TO OUTPUT STORED DATA FROM EACH STATISTICS RUN         DELTV    141
C                                                                       DELTV    142
      COMMON VAR                                                        DELTV    143
      DIMENSION VAR(5600), P(5000), DATA(350)                           DELTV    144
      EQUIVALENCE (VAR(601),P(1))                                       DELTV    145
      EQUIVALENCE (P(4074),DATA(1))                                     DELTV    146
C                                                                       DELTV    147
      NRUN=DATA(1)                                                  A   DELTV    148
      WRITE(6,100) NRUN                                             A   DELTV    149
C                                                                       DELTV    150
      DO 5 I=1,31                                                   B   DELTV    151
    5 WRITE(6,105) DATA(I+  1),DATA(I+ 32),DATA(I+ 63),DATA(I+ 94)      DELTV    152
     5*,           DATA(I+125),DATA(I+156),DATA(I+187),DATA(I+218)   B   DELTV    153
C                                                                       DELTV    154
      WRITE(6,110) NRUN                                             C   DELTV    155
C                                                                       DELTV    156
      DO 10 I=1,7                                                   D   DELTV    157
   10 WRITE(6,115) DATA(I+250),DATA(I+257),DATA(I+264)                  DELTV    158
     *,             DATA(I+271),DATA(I+278),DATA(I+285)                DELTV    159
     *,             DATA(I+292),DATA(I+299),DATA(I+306)                DELTV    160
     *,             DATA(I+313)                                   D    DELTV    161
C                                                                       DELTV    162
      WRITE(6,120) (DATA(I),I=321,350)                              E   DELTV    163
  100 FORMAT(1H1,50X,32HVECTOR/MANEUVER DATA FOR RUN NO.,I4,10H FOLLOWS DELTV    164
     *- ,/)                                                             DELTV    165
  105 FORMAT(1X,8E15.6)                                                 DELTV    166
  110 FORMAT(/50X,29HALIGNMENT HISTORY FOR RUN NO.,I4,10H FOLLOWS- ,/)  DELTV    167
  115 FORMAT(1X,10E13.4)                                                DELTV    168
  120 FORMAT(/44X,47HASSORTED OTHER JUNK KNOWN ONLY TO USER FOLLOWS-,   DELTV    169
     *3(/1X,8E15.6),/1X,6E15.6)                                         DELTV    170
      RETURN                                                            DELTV    171
      END                                                               DELTV    172
```

POPOUT

ROUTINE POPOUT

SUBROUTINE HAS TWO FUNCTIONS:
(1) COMPUTES ACTUAL, ESTIMATED AND MEASURED RELATIVE PARAMETERS. MEASURED RELATIVE PARAMETERS ARE SUPPLIED TO NAVIGATION FILTER DURING MARKING PROCESS. POPOUT MUST BE VISITED PERIODICALL WHENEVER MARKING IS IN PROGRESS, EVEN IF NO PRINTED OUTPUT IS DESIRED.
(2) COMPUTES, ORGANIZES AND PRINTS OUT THE AAP STANDARD DATA BLOCK.

AREA A — IF ENTRY IS THROUGH CALL TO POPOUT, IT IS FOR THE NORMAL PURPOSES DESCRIBED ABOVE. DEPENDING ON VALUE OF P(9) AND ELAPSED TIME SINCE LAST EXECUTION OF THE PRINT INSTRUCTIONS, PRINT MAY OR MAY NOT RESULT. IF ENTRY IS THROUGH CALL TO POPW, IT IS FOR THE PURPOSE OF A BLOCK DATA PRINT AT A TIME OF MANEUVER COMPUTATION OR RESETTING W. IN THIS CASE, PRINT IS DESIRED REGARDLESS OF THE TIME SINCE LAST PRINT. SETTING OF NSKIP=1 CAUSES ELAPSED TIME CHECK AT PRINT INSTRUCTIONS TO BE DISABLED.

AREA B — CALLS TO SETUP CREATE AN OUTPUT FORM OF THE BETELGEUSE VECTOR FOR ESTIMATED AND ACTUAL STATES.

AREA C — ERROR IN SENSOR BIAS ESTIMATE IS DEFINED AS ESTIMATED MINUS ACTUAL. ANGLE BIASE ERRORS ARE CONVERTED TO MILLIRADIANS BEFORE OUTPUT.

AREA D — CREATE A CARTESIAN FORM OF THE ACTUAL BETELGEUSE STATE.

AREA E — MEASUREMENT FRAME IS 'NATURAL GEOMETRY' FRAME OF RELATIVE MEASUREMENTS.

AREA F — MFTMAT IS THE TRANSFORMATION MATRIX FROM BRF CARTESIAN FRAME TO LINE-OF-SIGHT (MEASUREMENT) FRAME. FOR PURPOSES OF COORDINATE TRANSFORMATION, THE W-MATRIX TRANSFORMS AS AN ERROR VECTOR (REFERENCE 1). BY THE DEFINITION OF THE ESTIMATED CARTESIAN STATE:

$$\underline{X}_N = [\underline{R}_{S/C}, \underline{V}_{S/C}, \underline{R}_{TGT}, \underline{V}_{TGT}, \underline{K}_S]$$

$\underline{X}_N$ = ESTIMATED STATE $\underline{K}_S$ = ESTIMATED SENSOR BIASES

THE ERROR VECTOR IS AN ARRAY OF STATE ERRORS IN THE SAME SEQUENCE. SINCE THE POSITIONS AND VELOCITIES ARE ALL EXPRESSED IN THE SAME BRF FRAME, THEY HAVE THE SAME TRANSFORMATION TO THE MEASUREMENT FRAME:

$$\underline{MFTRN} = \begin{bmatrix} \underline{MFTMAT} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & \underline{MFTMAT} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & \underline{MFTMAT} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & \underline{MFTMAT} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} \end{bmatrix}.$$

WHERE MFTMAT IS THE MATRIX WHOSE ROWS ARE THE UNIT VECTORS OF THE MEASUREMENT FRAME IN THE BRF. SINCE MFTRN TRANSFORMS AN ERROR VECTOR, IT TRANSFORMS THE W-MATRIX. ALL SUBSEQUENT COMPUTATIONS INVOLVING THE W-MATRIX WILL USE THE MEASUREMENT FRAME W, DENOTED WM.

AREA G TRANSFORM W TO MEASUREMENT FRAME: WM = MFTRN x W

AREA H BY THE DEFINITION OF W, $WW^T=E$, THE COVARIANCE OF STATE ERRORS. AS THE DIAGONAL ELEMENTS OF WM ARE THE VARIANCES OF STATE ERRORS, THEIR SQUARE-ROOT IS THE STANDARD DEVIATION. AREA H COMPUTES THE DIAGONAL ELEMENTS OF $WM\,WM^T$ AND FINDS THEIR SQUARE ROOTS.

AREA I COMPUTE THE COVARIANCE OF RELATIVE STATE ERRORS. LET

$$\underline{r} = \underline{R}_{TGT} - \underline{R}_{S/C}$$

$$\underline{v} = \underline{V}_{TGT} - \underline{V}_{S/C}$$

THEN

$$\begin{bmatrix} \underline{r} \\ \underline{v} \end{bmatrix} = \begin{bmatrix} -I_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & -I_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} \end{bmatrix} \underline{X}_N$$

$$= [\underline{OK}]\,\underline{X}_N$$

FROM WHICH IT NECESSARILY FOLLOWS THAT

$$\underline{RWM}_{6x18} = \underline{OK}_{6x18} \times \underline{WM}_{18x18}$$

IS THE W-MATRIX OF RELATIVE STATE ERRORS, AND

$$\underline{REM} = \underline{RWM} \times \underline{RWM}^T$$

IS THE COVARIANCE OF RELATIVE ERRORS.

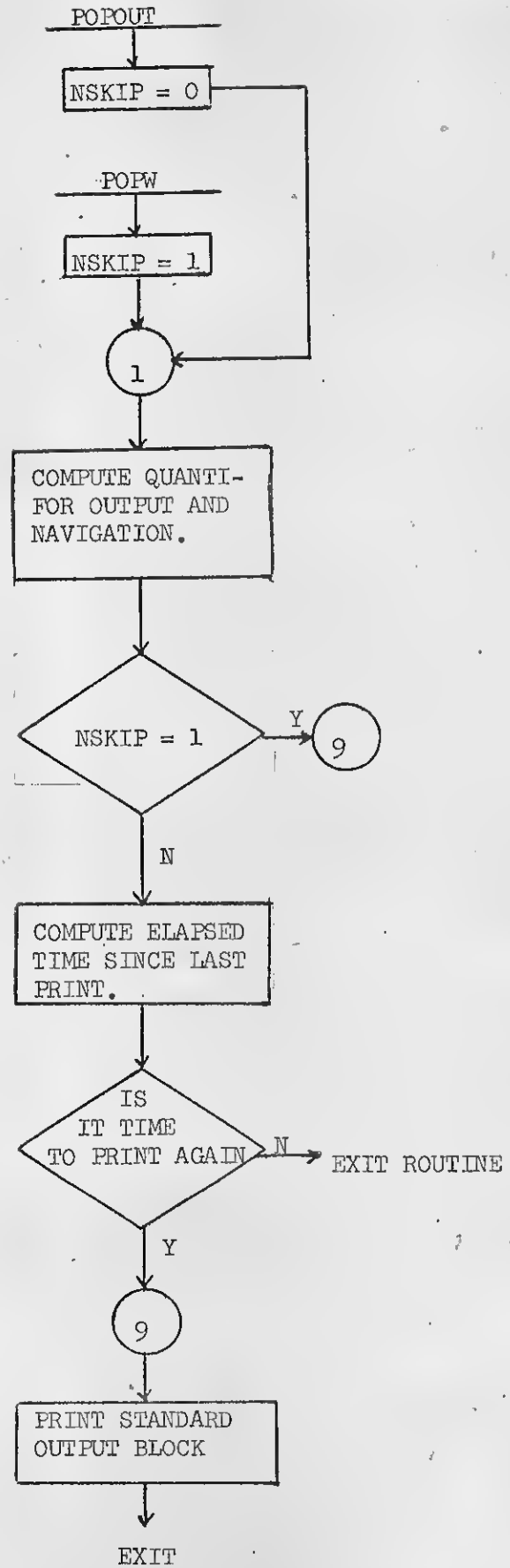
AREA K — CURRENT CARTESIAN ERROR VECTOR IS ROTATED TO MEASUREMENT FRAME.
CURRENT RELATIVE ERROR VECTOR IS COMPUTED.

AREA L — COMPUTES RELATIVE QUANTITIES FOR NAVIGATION AND OUTPUT.
CALL REF WITH ESTIMATED CARTESIAN S/C STATE TO DEFINE ACTUAL LOCAL VERTICAL UNIT VECTORS, ACTUAL NAV BASE UNIT VECTORS, ESTIMATED LOCAL VERTICAL VECTORS AND ESTIMATED NAV BASE VECTORS. CURRENT NAV BASE UNIT VECTORS ARE DEFINED AS THE LINE-OF-SIGHT FRAME UNIT VECTORS (SAME AS MEASUREMENT FRAME).
CALL REL WITH ACTUAL STATE AND ACTUAL NAV BASE VECTORS TO COMPUTE R, RDOT, AZ, EL , ACTUAL RELATIVE PARAMETERS. AZ, EL ARE ANGLES DEFINED WITH RESPECT TO NAV BASE UNIT VECTORS, NOT LOCAL VERTICAL.
CALL GARBAGE TO CREATE MEASURED RELATIVE PARAMETERS FROM ACTUAL RELATIVE PARAMETERS. RQE VECTOR USED FOR THIS COMPUTATION IS OVERWRITTEN WITH NEXT INSTRUCTION. THIS CALL TO GARBAGE ADDS NOISE TO RQE VECTOR AND STORES IT IN QQ.
CALL REL AGAIN WITH ACTUAL STATE AND ACTUAL LOCAL VERTICAL UNIT VECTORS TO DEFINE ACTUAL VALUE OF LOCAL VERTICAL AZ AND EL.
CALL REL AGAIN WITH ESTIMATED STATE AND ESTIMATED LOCAL VERTICAL UNIT VECTORS TO DEFINE ESTIMATED VALUE OF LOCAL VERTICAL AZ AND EL, ESTIMATED R, RDOT.
LOAD MEASURED VALUES INTO OUTPUT ARRAY.
CONVERT OUTPUT ANGLES TO DEGREES.

AREA M — CONVERT BIAS SIGMA FROM AREA H COMPUTATION TO MILLIRADIANS.

AREA N — FORGET IT. NOONE SEEMS SURE WHAT THIS MEANS OR IF IT IS CORRECT. IT DOES NOT SEEM TO BEAR ON THE OPERATION OF THE FILTER AND I HAVE BEEN UNABLE TO EXTRACT ANYTHING USEFUL FROM IT.

AREA O — IF ENTRY WAS THROUGH POPW, GO AROUND PRINT INTERVAL CHECK.


POPOUT
NSKIP = 0
POPW
NSKIP = 1
1
COMPUTE QUANTI-
FOR OUTPUT AND
NAVIGATION.
NSKIP = 1
Y
9
N
COMPUTE ELAPSED
TIME SINCE LAST
PRINT.
IS
IT TIME
TO PRINT AGAIN
N
EXIT ROUTINE
Y
9
PRINT STANDARD
OUTPUT BLOCK
EXIT

```
                SUBROUTINE POPOUT                                       POPOUT     2
C                                                                       POPOUT     3
C       OUTPUT SUBROUTINE FOR NAVIGATED STATE                           POPOUT     4
C                                                                       POPOUT     5
C       FOLLOWING AREA IS BLANK COMMON FOR BETELGEUSE                   POPOUT     6
            COMMON VAR                                                  POPOUT     7
C       +++++                                                           POPOUT     8
        REAL MFTMAT, MFTRN                                              POPOUT     9
C       +++++                                                           POPOUT    10
        DIMENSION VAR(5600), Y(100), DYDX(100), Q(100), FIRSTY(100)     POPOUT    11
     *,           NTEGER(100), O(100), P(5000)                          POPOUT    12
        EQUIVALENCE (VAR(1),Y(1))                                       POPOUT    13
     *,             (VAR(101),OYDX(1))                                  POPOUT    14
     *,             (VAR(201),Q(1))                                     POPOUT    15
     *,             (VAR(301),FIRSTY(1))                                POPOUT    16
     *,             (VAR(401),NTEGER(1))                                POPOUT    17
     *,             (VAR(501),O(1))                                     POPOUT    18
     *,             (VAR(601),P(1))                                     POPOUT    19
        DIMENSION SAVE(950), BLK(700), OATA(350), COV(24,24)            POPOUT    20
        EQUIVALENCE (P(350),SAVE(1))                                    POPOUT    21
     *,             (P(1300),BLK(1))                                    POPOUT    22
     *,             (P(4074),OATA(1))                                   POPOUT    23
     *,             (P(4424),COV(1,1))                                  POPOUT    24
        DIMENSION QQ(4), SIG(4), C(10), REFMAT(3,3), XNBN(3), YNBN(3)   POPOUT    25
     *,           ZNBN(3), NE(10), NM(10), OTL(10), OTN(10), OTM(10)    POPOUT    26
     *,           USP(10), USV(10), UTP(10), UTV(10), SR(10), SRO(10)   POPOUT    27
     *,           SO(10), SC1(10), SC2(10), NW(10), TLM(10), NS(3)      POPOUT    28
     *,           ZTZ(4), SZ(4), TALIGN(10), XNBE(3), YNBE(3), ZNBE(3)  POPOUT    29
     *,           X(18), WE(18,27)                                      POPOUT    30
     *,           XLVE(3),YLVE(3),ZLVE(3),XLVN(3),YLVN(3),ZLVN(3)       NOSHIT     1
        EQUIVALENCE (SAVE(1),QQ(1)),          (SAVE(5),SIG(1))          POPOUT    31
     *,             (SAVE(9),C(1)),          (SAVE(19),REFMAT(1,1))    POPOUT    32
     *,             (SAVE(28),XNBN(1)),      (SAVE(31),YNBN(1))        POPOUT    33
     *,             (SAVE(34),ZNBN(1)),      (SAVE(37),NE(1))          POPOUT    34
     *,             (SAVE(47),NM(1)),        (SAVE(57),OTL(1))         POPOUT    35
     *,             (SAVE(67),OTN(1)),       (SAVE(77),OTM(1))         POPOUT    36
     *,             (SAVE(87),USP(1)),       (SAVE(97),USV(1))         POPOUT    37
     *,             (SAVE(107),UTP(1)),      (SAVE(117),UTV(1))        POPOUT    38
     *,             (SAVE(127),SR(1)),       (SAVE(137),SRO(1))        POPOUT    39
     *,             (SAVE(147),SO(1)),       (SAVE(157),SC1(1))        POPOUT    40
     *,             (SAVE(167),SC2(1)),      (SAVE(177),NW(1))         POPOUT    41
     *,             (SAVE(187),TLM(1)),      (SAVE(197),NS(1))         POPOUT    42
```

```
     *,              (SAVE(208),IALIGN(1)),   (SAVE(218),NALIGN)                POPOUT     44
     *,              (SAVE(229),XNBE(1)),     (SAVE(232),YNBE(1))               POPOUT     45
     *,              (SAVE(235),ZNBE(1)),     VE(258),X(1))                     POPOUT     4
     *,              (SAVE(276),WE(1,1))                                        POPOUT     4
     *,              (NTEGER(30),ICOMP)                                         POPOUT     48
      EQUIVALENCE (SAVE(762),XLVE(1))                                           NOSHIT      2
     *,          (SAVE(765),YLVE(1))                                            NOSHIT      3
     *,          (SAVE(768),ZLVE(1))                                            NOSHIT      4
     *,          (SAVE(771),XLVN(1))                                            NOSHIT      5
     *,          (SAVE(774),YLVN(1))                                            NOSHIT      6
     *,          (SAVE(777),ZLVN(1))                                            NOSHIT      7
C                                                                               POPOUT     49
```

SUBROUTINE POPOUT CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
      EQUIVALENCE (C(1),TW)                                                     POPOUT     50
C                                                                               POPOUT     51
      DIMENSION OUT(200)                                                        POPOUT     52
C                                                                               POPOUT     53
      EQUIVALENCE (BLK(1),OUT(1))                                               POPOUT     54
      EQUIVALENCE (NTEGER(42),NLINE)                                            POPOUT     55
C                                                                               POPOUT     56
C                                                                               POPOUT     57
      EQUIVALENCE (P(7),CGO)                                                    POPOUT     58
     *,           (P(8),CRO)                                                    POPOUT     59
C                                                                               POPOUT     60
      DIMENSION XNB(12), XEB(12), BIASN(6), BIASE(6), DBIAS(6), XSNC(6)         POPOUT     61
     *,         XTNC(6), XSEC(6), XTEC(6), DXSC(6), DXTC(6), RXEC(6)           POPOUT     62
     *,         DXSM(6), DXTM(6), DRXM(6), SIGEM(18), REM(6,6), RQN(4)         POPOUT     63
     *,         RQE(4), RQM(4)                                                 POPOUT     64
C                                                                               POPOUT     65
      EQUIVALENCE (OUT(1),XNB(1))                                               POPOUT     66
     *,           (OUT(13),XEB(1))                                             POPOUT     67
     *,           (OUT(25),BIASN(1))                                           POPOUT     68
     *,           (OUT(31),BIASE(1))                                           POPOUT     69
     *,           (OUT(37),DBIAS(1))                                           POPOUT     70
     *,           (OUT(43),XSNC(1))                                            POPOUT     71
     *,           (OUT(49),XTNC(1))                                            POPOUT     72
     *,           (OUT(55),XSEC(1))                                            POPOUT     73
     *,           (OUT(61),XTEC(1))                                            POPOUT     74
```

```
     *,              (OUT(67),OXSC(1))                                  POPOUT     75
     *,              (OUT(73),OXTC(1))                                  POPOUT     76
     *,              (OUT(79),RXEC(1))                                  POPOUT     77
     *,              (OUT(85),DXSM(1))                                  POPOUT     78
     *,              (OUT(91),DXTM(1))                                  POPOUT     79
     *,              (OUT(97),DRXM(1))                                  POPOUT     80
     *,              (OUT(103),SIGEM(1))                                POPOUT     81
     *,              (OUT(121),REM(1,1))                                POPOUT     82
      EQUIVALENCE    (OUT(157),RON(1))                                  POPOUT     83
     *,              (OUT(161),RQE(1))                                  POPOUT     84
     *,              (OUT(165),RQM(1))                                  POPOUT     85
C                      400)                                            POPOUT     86
C       OUT(1-12)        NAVIGATED BETELGEUSE                          POPOUT     87
C       OUT(13-24)       ENVIRONMENT BETELGEUSE                        POPOUT     88
C       OUT(25-30)       NAVIGATED BIASES                              POPOUT     89
C       OUT(31-36)       ENVIRONMENT BIASES                            POPOUT     90
C       OUT(37-42)       NAVIGATED MINUS ENVIRONMENT BIASES            POPOUT     91
C       OUT(43-48)       NAVIGATED CARTESIAN S/C                       POPOUT     92
C       OUT(49-54)       NAVIGATED CARTESIAN TGT                       POPOUT     93
C       OUT(55-60)       ENVIRONMENT CARTESIAN S/C                     POPOUT     94
C       OUT(61-66)       ENVIRONMENT CARTESIAN TGT                     POPOUT     95
C       OUT(67-72)       NAVIGATED MINUS ENVIRONMENT CARTESIAN S/C      POPOUT     96
C       OUT(73-78)       NAVIGATED MINUS ENVIRONMENT CARTESIAN TGT      POPOUT     97
C       OUT(79-84)       TGT MINUS S/C ENVIRONMENT CARTESIAN            POPOUT     98
C       OUT(85-90)       NAVIGATED MINUS ENVIRONMENT MEASUREMENT S/C    POPOUT     99
C       OUT(91-96)       NAVIGATED MINUS ENVIRONMENT MEASUREMENT TGT    POPOUT    100
C       OUT(97-102)      NAVIGATED MINUS ENVIRONMENT RELATIVE STATE     POPOUT    101
C                        VECTOR, MEASUREMENT                           POPOUT    102
C       OUT(103-120)     ONE-SIGMA ERRORS MEASUREMENT                  POPOUT    103
C       OUT(121-156)     RELATIVE COVARIANCE MATRIX MEASUREMENT        POPOUT    104
```



```
C       OUT(157-160)     RELATIVE QUANTITIES NAVIGATED                 POPOUT    105
C       OUT(161-164)     RELATIVE QUANTITIES ENVIRONMENT               POPOUT    106
C       OUT(165-168)     RELATIVE QUANTITIES OBSERVED                  POPOUT    107
C                                                                      POPOUT    108
      DIMENSION XM(3), YM(3), ZM(3), MFTMAT(3,3), DUM(3,3)             POPOUT    109
     *,         MFTRN(18,18), WM(18,27), OK(6,18), RWM(6,27)          POPOUT    110
     *,         RWMI(27,6)                                            POPOUT    111
```

```
C                                                                       POPOUT  112
C                                                                       POPOUT  113
      NSKIP = 0                                                      A  POPOUT  11
      GO TO 1                                                           POPOUT  11
      ENTRY POPW                                                        POPOUT  116
      NSKIP = 1                                                         POPOUT  117
    1 CONTINUE                                                       A  POPOUT  118
C                                                                    B  POPOUT  119
C SET UP THE NAVIGATED STATE VECTOR IN THE BETELGEUSE FRAME             POPOUT  120
      CALL SETUP(Y(2),XNB(1))                                           POPOUT  121
C SET UP THE ENVIRONMENT STATE VECTOR IN THE BETELGEUSE FRAME           POPOUT  122
      CALL SETUP(Y(20),XEB(1))                                       B  POPOUT  123
C CREATE THE NAVIGATED AND ENVIRONMENT BIAS ARRAYS AND THEIR DIFFERENCE POPOUT  124
      DO 5 I=1,6                                                     C  POPOUT  125
      BIASN(I) = Y(I+13)                                                POPOUT  126
      BIASE(I) = Y(I+31)                                                POPOUT  127
      DBIAS(I) = BIASN(I) - BIASE(I)                                    POPOUT  128
      IF(I.EQ.3.OR.I.EQ.4) DBIAS(I) = DBIAS(I)*1000.0                C  POPOUT  129
    5 CONTINUE                                                          POPOUT  130
C CREATE THE NAVIGATED STATE VECTOR IN THE CARTESIAN FRAME               POPOUT  131
      DO 10 I=1,6                                                       POPOUT  132
      XSNC(I) = Y(I+37)                                                 POPOUT  133
      XTNC(I) = Y(I+43)                                                 POPOUT  134
   10 CONTINUE                                                          POPOUT  135
C ROTATE ENVIRONMENT STATE VECTOR FROM THE BETELGEUSE TO CARTESIAN FRAME POPOUT  136
      CALL CART1(OUT(55),Y(20))                                      D  POPOUT  137
C CALCULATE THE ERROR VECTOR BETWEEN THE NAVIGATED AND ENVIRONMENT STATE POPOUT  138
C VECTORS IN THE CARTESIAN FRAME                                        POPOUT  139
      DO 15 I=1,6                                                       POPOUT  140
      DXSC(I) = XSNC(I) - XSEC(I)                                       POPOUT  141
      DXTC(I) = XTNC(I) - XTEC(I)                                       POPOUT  142
   15 CONTINUE                                                          POPOUT  143
C IN THE CARTESIAN FRAME CALCULATE THE RELATIVE STATE VECTOR BETWEEN THE POPOUT  144
C ENVIRONMENT STATES OF THE S/C AND TGT                                 POPOUT  145
      DO 20 I=1,6                                                       POPOUT  146
      RXEC(I) = XTEC(I) - XSEC(I)                                       POPOUT  147
   20 CONTINUE                                                          POPOUT  148
C                                                                       POPOUT  149
C*************************************************************       E*********** POPOUT  150
C                                                                    E  POPOUT  151
C CREATE THE TRANSFORMATION MATRIX WHICH ROTATES A VECTOR FROM THE       POPOUT  152
C CARTESIAN TO THE MEASUREMENT FRAME                                    POPOUT  153
C                                                                       POPOUT  154
C     YM = UNIT(RANGE)                                                  POPOUT  155
C     ZM = UNIT(RS X RANGE)                                             POPOUT  156
C     XM = UNIT(YM X ZM)                                                POPOUT  157
C                                                                       POPOUT  158
```

```
      CALL UCROSS(XSEC(1),YM,ZM)                                        POPOUT  160
      CALL UCROSS(YM,ZM,XM)                                             POPOUT  161
      CALL TRN(XM,YM,ZM,MFTMAT,DUM)                                E    POPOUT  162
C                                                                  F    POPOUT  163
C CREATE THE MATRIX MFTRN WHICH ROTATES THE W MATRIX FROM THE CARTESIAN POPOUT  164
C TO THE MEASUREMENT FRAME                                              POPOUT  165
C                                                                       POPOUT  166
C ZERO THE LOCATIONS FOR THE MATRIX MFTRN                               POPOUT  167
      DO 25 I=1,18                                                      POPOUT  168
      DO 25 J=1,18                                                      POPOUT  169
      MFTRN(I,J) = 0.0                                                  POPOUT  170
   25 CONTINUE                                                          POPOUT  171
C SET UP MFTRN AS AN (18 X 18) MATRIX COMPOSED OF 4 (3 X 3) MFTMATS AND POPOUT  172
C 2 (3 X 3) IDENTITY MATRICES ALONG THE DIAGONAL                        POPOUT  173
      DO 30 I=3,12,3                                                    POPOUT  174
      L = I - 3                                                         POPOUT  175
      DO 30 J=1,3                                                       POPOUT  176
      DO 30 K=1,3                                                       POPOUT  177
      MFTRN((J+L),(K+L)) = MFTMAT(J,K)                                  POPOUT  178
   30 CONTINUE                                                          POPOUT  179
      DO 35 I=13,18                                                     POPOUT  180
      MFTRN(I,I) = 1.0                                                  POPOUT  181
   35 CONTINUE                                                     F    POPOUT  182
C ROTATE THE W MATRIX INTO THE MEASUREMENT FRAME                   G    POPOUT  183
      CALL MATMUL(MFTRN,WE,WM,18,18,27)                            G    POPOUT  184
C CALCULATE THE ONE-SIGMA POSITION AND VELOCITY ERRORS OF THE COVARIANCE POPOUT  185
C MATRIX E IN THE MEASUREMENT FRAME                                H    POPOUT  186
      DO 40 I=1,18                                                      POPOUT  187
      EO = 0.0                                                          POPOUT  188
      DO 40 J=1,27                                                      POPOUT  189
      EO = EO + WM(I,J)**2                                              POPOUT  190
      IF(J.EQ.27) SIGEM(I) = SQRT(EO)                                   POPOUT  191
   40 CONTINUE                                                     H    POPOUT  192
C                                                                  I    POPOUT  193
C CREATE THE OK MATRIX TO BE USED IN CALCULATING THE RELATIVE W MATRIX  POPOUT  194
C IN THE MEASUREMENT FRAME                                              POPOUT  195
      DO 45 I=1,6.                                                      POPOUT  196
      DO 50 J=1,18                                                      POPOUT  197
      OK(I,J) = 0.0                                                     POPOUT  198
   50 CONTINUE                                                          POPOUT  199
      OK(I,I) = 1.0                                                     POPOUT  200
      OK(I,(I+6)) = -1.0                                                POPOUT  201
   45 CONTINUE                                                     F
```

```
C                                                                       POPOUT   203
C CREATE THE RELATIVE W MATRIX IN THE MEASUREMENT FRAME          J      POPOUT   204
      CALL MATMUL(OK,WM,RWM,6,18,27)                                    POPOUT   205
C CREATE THE RELATIVE COVARIANCE MATRIX IN THE MEASUREMENT FRAME         POPOUT   206
      CALL MATRAN(RWM,6,27,RWMT)                                        POPOUT   207
      CALL MATMUL(RWM,RWMT,REM,6,27,6)                           J      POPOUT   208
C REPLACE THE VARIANCES ALONG THE DIAGONAL OF THE RELATIVE COVARIANCE     POPOUT   209
C MATRIX WITH THEIR ONE-SIGMAS                                           POPOUT   210
      DO 55 I=1,6                                                       POPOUT   211
      REM(I,I) = SQRT(ABS(REM(I,I)))                                    POPOUT   212
   55 CONTINUE                                                          POPOUT   213
C CALCULATE THE CORRELATION COEFFICIENTS AND PLACE THEM IN THE LOWER      POPOUT   214
```

SUBROUTINE POPOUT CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
C TRIANGLE OF THE SYMMETRIC COVARIANCE MATRIX                            POPOUT   215
      DO 60 I=1,6                                                       POPOUT   216
      IF(REM(I,I).EQ.0.0) GO TO 60                                      POPOUT   217
      DO 59 J=1,6                                                       POPOUT   218
      IF(J.GE.I) GO TO 59                                               POPOUT   219
      REM(I,J) = REM(J,I)/(REM(I,I)*REM(J,J))                           POPOUT   220
   59 CONTINUE                                                          POPOUT   221
   60 CONTINUE                                                          POPOUT   222
C                                                                  K     POPOUT   223
C ROTATE THE ERROR VECTOR BETWEEN THE NAVIGATED AND ENVIRONMENT          POPOUT   224
C TRAJECTORIES FROM THE CARTESIAN TO THE MEASUREMENT FRAME               POPOUT   225
      CALL MATMUL(MFTRN,OUT(67),OUT(85),18,18,1)                        POPOUT   226
C IN THE MEASUREMENT FRAME, CALCULATE THE DIFFERENCE BETWEEN THE         POPOUT   227
C RELATIVE STATE VECTORS OF THE NAVIGATED AND ENVIRONMENT TRAJECTORIES   POPOUT   228
      DO 61 I=1,6                                                       POPOUT   229
      DRXM(I) = OXTM(I) - OXSM(I)                                       POPOUT   230
   61 CONTINUE                                                    K     POPOUT   231
C                                                                        POPOUT   232
C*********************************************************************** POPOUT   233
C                                                                  L     POPOUT   234
C DEFINE THE CURRENT NAVIGATION BASE ATTITUOE FOR THE AZIMUTH AND        POPOUT   235
C ELEVATION ANGLE COMPUTATION                                            POPOUT   236
      CALL PEF(OUT(43))                                                 POPOUT   237
C COMPUTE THE ENVIRONMENT RELATIVE QUANTITIES                            POPOUT   238
      CALL REL(OUT(55),ROE,XNRE,YNRE,ZNRE)                              POPOUT   239
```

```
C ADD NOISE TO ENVIRONMENT OBSERVABLES                                  POPOUT   240
      CALL GARBAGE(ROE)                                                 POPOUT   241
      CALL REL(OUT(55),RQE,XLVE,YLVE,ZLV                                NOSHIT
C COMPUTE THE NAVIGATED RELATIVE QUANTITIES                             POPOUT   24_
      CALL REL(OUT(43),RON,XLVN,YLVN,ZLVN)                              NOSHIT     9
      RADEG = 57.2957795                                                POPOUT   244
      DO 65 I=1,4                                                       POPOUT   245
C ADD BIASES TO NAVIGATED RELATIVE QUANTITIES                           POPOUT   246
      RON(I) = RON(I) + BIASN(I)                                        POPOUT   247
C SET UP THE MEASURED RELATIVE QUANTITIES FOR OUTPUT                    POPOUT   248
      ROM(I) = OO(I)                                                    POPOUT   249
C CONVERT AZIMUTH AND ELEVATION ANGLES TO DEGREES                       POPOUT   250
      IF(I.LT.3) GO TO 65                                               POPOUT   251
      ROM(I) = ROM(I)*RADEG                                             POPOUT   252
      RON(I) = RON(I)*RADEG                                             POPOUT   253
      ROE(I) = ROE(I)*RADEG                                             POPOUT   254
 65   CONTINUE                                                          POPOUT   255
C OUTPUT ONE-SIGMA ERRORS IN THE AZIMUTH AND ELEVATION ANGLES AS        POPOUT   256
C MILLIRADIANS                                                          POPOUT   257
      SIGEM(15) = SIGEM(15)*1000.                                       POPOUT   258
      SIGEM(16) = SIGEM(16)*1000.                                       POPOUT   259
C                                                                       POPOUT   260
*************************************************************************POPOUT   261
C                                                                       POPOUT   262
C CALCULATE THE PRODUCT OF THE UNIVERSAL GRAVITY CONSTANT WITH THE MASS POPOUT   263
C OF THE EARTH                                                          POPOUT   264
      GM = CGO*CRO**2                                                   POPOUT   265
C ADVANCE ESTIMATED TARGET TO ENVIRONMENT TARGET                        POPOUT   266
      CALL OPEN1(XTNC,XTEC,STT)                                         POPOUT   267
      CALL OPEN2(XTNC,GM,-STT,RTT,RDTT,VHTT)                            POPOUT   268
```

EROUTINE POPOUT COC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
C                                                                       POPOUT   269
C ADVANCE ESTIMATED SPACECRAFT TO ENVIRONMENT TARGET                    POPOUT   270
      CALL OPEN1(XSNC,XTEC,STS)                                         POPOUT   271
      CALL OPEN2(XSNC,GM,-STS,RTS,RDTS,VHTS)                            POPOUT   272
C                                                                       POPOUT   273
C ADVANCE ENVIRONMENT SPACECRAFT TO ENVIRONMENT TARGET                  POPOUT   274
```

```
      CALL OPEN2(XSEC,GM,-STSA,RTSA,RDTSA,VHTSA)                        PDPOUT  276
C COMPUTE CURVILINEAR ERRORS                                            PDPOUT  277
      RTE = RSS(XTEC(1),XTEC(2),XTEC(3))                                PDPOUT  278
      RDTE = DOT(XTEC(1),XTEC(4))/RTE                                   PDPOUT  279
      VTE = RSS(XTEC(4),XTEC(5),XTEC(6))                                PDPOUT  280
      VHTE = SQRT(VTE**2 - RDTE**2)                                     PDPOUT  281
C                                                                       PDPOUT  282
      SE = (STSA - STS + STT)*RTE                                       PDPOUT  283
      RE = (RTE - RTSA) - (RTT - RTS)                                   PDPOUT  284
      VHE = (VHTE - VHTSA) - (VHTT - VHTS)                              PDPOUT  285
      RDE = (RDTE - RDTSA) - (RDTT - RDTS)                              PDPOUT  286
      PHID = VHTE/RTE                                                   PDPOUT  287
      DVDH = 1.5*PHID*PE                                                PDPOUT  288
      VCE = VHE + DVDH                                            N     PDPOUT  289
      IF(NSKIP.EQ.1) GO TO 9                                      O     PDPOUT  290
C                                                                       PDPOUT  291
C     CHECK IF SUFFICIENT TIME HAS ELAPSED FOR A PRINT                  PDPOUT  292
      TLP=Y(1) - P(10)                                                  PDPOUT  293
      IF(TLP.LT.P(9)) GO TO 200                                         PDPOUT  294
      P(10)=Y(1)                                                        PDPOUT  295
C                                                                       PDPOUT  296
    9 CONTINUE                                                          PDPOUT  297
C     CHECK FOR TIME TO PAGEHEAD                                        PDPOUT  298
      IF(NLINE.LT.58) GO TO 11                                          PDPOUT  299
      PRINT 95                                                          PDPOUT  300
      NLINE=0                                                           PDPOUT  301
   11 CONTINUE                                                          PDPOUT  302
C                                                                       PDPOUT  303
C*********************************************************************  PDPOUT  304
C                                                                       PDPOUT  305
C                       PRINT OUT THE GOODIES                           PDPOUT  306
C                                                                       PDPOUT  307
      IF(NLINE.EQ.29) PRINT 90                                          PDPOUT  308
C OUTPUT THE NAVIGATED AND ENVIRONMENT BETELGEUSE STATES                PDPOUT  309
      PRINT 100, Y(1)                                                   PDPOUT  310
      PRINT 105                                                         PDPOUT  311
      PRINT 110, (XNB(I),I=1,12), (XEB(I),I=1,12)                       PDPOUT  312
C OUTPUT THE NAVIGATED AND ENVIRONMENT CARTESIAN STATES                 PDPOUT  313
      PRINT 115, Y(1)                                                   PDPOUT  314
      PRINT 120                                                         PDPOUT  315
      PRINT 125, (XSNC(I),I=1,6), (XTNC(I),I=1,6)                       PDPOUT  316
     *,          (XSEC(I),I=1,6), (XTEC(I),I=1,6)                       PDPOUT  317
C OUTPUT CARTESIAN STATE ERRORS IN THE MEASUREMENT FRAME                PDPOUT  318
      PRINT 130                                                         PDPOUT  319
      PRINT 135                                                         PDPOUT  320
      PRINT 140, (DXSM(I),I=1,6), (DXTM(I),I=1,6)                       PDPOUT  321
C OUTPUT RELATIVE STATE ERRORS, RELATIVE PARAMETERS, MARKS, COVARIANCE  PDPOUT  322
C OF RELATIVE ERRORS IN THE MEASUREMENT FRAME, AND INERTIAL STANDARD    PDPOUT  323
```

```
C OEVIATIONS                                                                    POPOUT   324
      PRINT 145                                                                 POPOUT   325
      PRINT 150, (OBIAS(I),I=1,6), (ORXM(I),I=1,6)                              POPOUT   326
      PRINT 155                                                                 POPOUT   327
      PRINT 160, SE, VHE, VCE, RE, RDE                                          POPOUT   328
      PRINT 165, ICOMP, ZTZ(1)                                                  POPOUT   329
     *,           (RON(I),I=1,4), NS(1), ZTZ(2)                               POPOUT   330
     *,           (RQE(I),I=1,4), NS(2), ZTZ(3), (REM(1,I),I=1,6)             POPOUT   331
     *,           (ROM(I),I=1,4), NS(3), ZTZ(4), (REM(2,I),I=1,6)             POPOUT   332
     *,                                          (REM(3,I),I=1,6)             POPOUT   333
     *,                                  SZ(1), (REM(4,I),I=1,6)              POPOUT   334
     *,           (SIGEM(I),I=1,6),      SZ(2), (REM(5,I),I=1,6)              POPOUT   335
     *,           (SIGEM(I),I=7,12),     SZ(3), (REM(6,I),I=1,6)              POPOUT   336
     *,           (SIGEM(I),I=13,18),    SZ(4)                                POPOUT   337
      NLINE = NLINE + 29                                                        POPOUT   338
C                                                                               POPOUT   339
C*****************************************************************************POPOUT   340
C                                                                               POPOUT   341
C                        FORMAT STATEMENTS                                      POPOUT   342
C                                                                               POPOUT   343
   90 FORMAT(//)                                                                POPOUT   344
   95 FORMAT(1H1)                                                               POPOUT   345
  100 FORMAT(53X,17HBETELGEUSE STATE ,F8.1)                                     POPOUT   346
  105 FORMAT(12X,117HRAD VEC  LONGITUDE LATITUDE ALT RATE  HOR VEL  HEA         POPOUT   347
     *DING     X-BAR      Y-BAR      Z-BAR      U-BAR      V-BAR      W-BA      POPOUT   348
     *R)                                                                        POPOUT   349
  110 FORMAT(1X,10HNAV       ,F9.0,2F9.3,F9.2,F10.2,F9.3,3F11.0,3F10.2,        POPOUT   350
     *      /1X,10HACT       ,F9.0,2F9.3,F9.2,F10.2,F9.3,3F11.0,3F10.2)        POPOUT   351
  115 FORMAT(/53X,17HCARTESIAN STATE  ,F8.1)                                    POPOUT   352
  120 FORMAT(7X,123HXS(BRF)    YS(BRF)    ZS(BRF)   XSD(BRF)  YSD(BRF)        POPOUT   353
     * ZSD(BRF)   XT(BRF)    YT(BRF)    ZT(BRF)   XTD(BRF)  YTD(BRF)  ZT      POPOUT   354
     *D(BRF))                                                                   POPOUT   355
  125 FORMAT(1X,3HNAV,3F11.0,3F10.2,3F11.0,3F10.2/                              POPOUT   356
     *      1X,3HACT,3F11.0,3F10.2,3F11.0,3F10.2)                              POPOUT   357
  130 FORMAT(/42X,43HCARTESIAN STATE ERRORS IN MEASUREMENT FRAME)               POPOUT   358
  135 FORMAT(7X,123HXSE(MF)    YSE(MF)    ZSE(MF)   XSDE(MF)  YSDE(MF)        POPOUT   359
     * ZSDE(MF)   XTE(MF)    YTE(MF)    ZTE(MF)   XTDE(MF)  YTDE(MF)  ZT      POPOUT   360
     *DE(MF))                                                                   POPOUT   361
  140 FORMAT(4X,3F11.0,3F10.2,3F11.0,3F10.2)                                    POPOUT   362
  145 FORMAT(/56X,21HRELATIVE STATE ERRORS,/6X,124H=====================       POPOUT   363
     *=======BIAS ESTIMATION=====================**********POSITI             POPOUT   364
     *ON AND VELOCITY**********/10X,120HRANGE      R-RATE                     POPOUT   365
```

```
     *(MF)   XRDE(MF)   YRDE(MF)   ZRDE(MF))                                    POPOUT    367
 150 FORMAT(6X,F9.2,2F11.3,3F10.3,3F11.0,3F10.2)                                POPOUT    368
     FORMAT(74X,49HO-P       S-DOT    SD-      L      DELTA-H  DELH-DOT    ) POPOUT    3
 160 FORMAT(16X,52HRELATIVE PARAMETERS          MARKS                          POPOUT    3.
     *  ,F10.0,F10.3,F11.3,F11.0,F10.3)                                        POPOUT    371
 165 FORMAT(10X,40HRANGE     R-RATE  AZIMUTH REL ELEV  MFLG,I2,6H  UR          POPOUT    372
     *  F5.1/1X,6HNAV    ,F9.0,F10.2,2F9.3,6H  RAD ,I2,6H  URD ,F6.1,13X,      POPOUT    373
     *  35HCOVARIANCE OF RELATIVE ERRORS (MF)  ,/1X,6HACT    ,F9.0,F10.2,     POPOUT    374
     *  2F9.3,6H  VHF ,I2,6H  UAZ ,F6.1,3H  X,6E10.3,/1X,6HMEAS. ,F9.0,       POPOUT    375
     *F10.2,2F9.3,5H  OPT,I3,6H  UEL ,F6.1,3H  Y,F10.5,  5E10.3,/64X,3H       POPOUT    376
     *  Z,2F10.5,          4E10.3,/8X,50HINERTIAL STANDARD DEVIATIONS          POPOUT    377
     *              SR  ,F5.3,4H  XO, 3F10.5,            3E10.3/1X,4HS/C     POPOUT    378
```

UBROUTINE POPOUT COC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
     *,3F8.0,3F7.2,8H    SRO ,F5.3,4H  YO, 4F10.5,              2E10.3,/     POPOUT    379
     *1X,4HTGT ,3F8.0,3F7.2,8H    SAZ ,F5.3,4H  ZD, 5F10.5,                   POPOUT    380
     *E10.3,/1X,4HBIAS,F8.0,2F8.3,3F7.2,8H    SEL ,F5.3)                      POPOUT    381
 200 CONTINUE                                                                  POPOUT    382
           RETURN                                                              POPOUT    383
           END                                                                 POPOUT    384
```

CART

ROUTINE CART1

ENTRY CART1 CONVERTS A STANDARD BETELGEUSE VECTOR (S/C AND TGT) TO A STANDARD CARTESIAN VECTOR IN THE BRF FRAME (NORTH POINTING, EARTH CENTERED, INERTIAL). AS USUAL, BETELGEUSE LONGITUDE IS A POSITIVE ROTATION ABOUT THE SOUTH POLE.

ENTRY CART2 IS THE FUNCTIONAL INVERSE OF CART1.

```
      SUBROUTINE CART1(X,Y)                                                  CART
C                                                                            CART     3
      OIMENSION X(12),Y(12)                                                  CART     4
C                                                                            CART     5
C     SUBROUTINE TO CONVERT A BETEL VECTOR TO PLANET CENTERED INERTIAL       CART     6
C     Y=BETELGEUSE                                        X=CARTESIAN        CART     7
      CMU=COS(Y(2))                                                          CART     8
      SMU=SIN(Y(2))                                                          CART     9
      CLA=COS(Y(3))                                                          CART    10
      SLA=SIN(Y(3))                                                          CART    11
C                                                                            CART    12
      X(1)=  Y(1)*CLA*CMU                                                    CART    13
      X(2)= -Y(1)*CLA*SMU                                                    CART    14
      X(3)=  Y(1)*SLA                                                        CART    15
      X(4)=  Y(7)*CLA*CMU-Y(8)*SMU+Y(9)*SLA*CMU                              CART    16
      X(5)= -Y(7)*CLA*SMU-Y(8)*CMU-Y(9)*SLA*SMU                              CART    17
      X(6)=  Y(7)*SLA-Y(9)*CLA                                               CART    18
      X(7)=  (Y(1)+Y(4))*CLA*CMU-Y(5)*SMU+Y(6)*SLA*CMU                       CART    19
      X(8)= -(Y(1)+Y(4))*CLA*SMU-Y(5)*CMU-Y(6)*SLA*SMU                       CART    20
      X(9)=  (Y(1)+Y(4))*SLA-Y(6)*CLA                                        CART    21
      X(10)= (Y(7)+Y(10))*CLA*CMU-(Y(8)+Y(11))*SMU+(Y(9)+Y(12))*SLA*CMU      CART    22
      X(11)=-(Y(7)+Y(10))*CLA*SMU-(Y(8)+Y(11))*CMU-(Y(9)+Y(12))*SLA*SMU      CART    23
      X(12)= (Y(7)+Y(10))*SLA-(Y(9)+Y(12))*CLA                               CART    24
      RETURN                                                                 CART    25
      ENTRY CART2                                                            CART    26
C     SUBROUTINE TO CONVERT PLANET CENTERED INERTIAL TO A BETEL VECTOR       CART    27
C                                                                            CART    28
      Y(1)=RSS(X(1),X(2),X(3))                                               CART    29
      Y(2)=-ATAN2(X(2),X(1))                                                 CART    30
      Y(3)=ASIN(X(3)/Y(1))                                                   CART    31
      CMU=COS(Y(2))                                                          CART    32
      SMU=SIN(Y(2))                                                          CART    33
      CLA=COS(Y(3))                                                          CART    34
      SLA=SIN(Y(3))                                                          CART    35
      Y(4)= (X(7)-X(1))*CLA*CMU-(X(8)-X(2))*CLA*SMU+(X(9)-X(3))*SLA          CART    36
      Y(5)=-(X(7)-X(1))*SMU-(X(8)-X(2))*CMU                                  CART    37
      Y(6)= (X(7)-X(1))*SLA*CMU-(X(8)-X(2))*SLA*SMU-(X(9)-X(3))*CLA          CART    38
      Y(7)= X(4)*CLA*CMU-X(5)*CLA*SMU+X(6)*SLA                               CART    39
      Y(8)=-X(4)*SMU-X(5)*CMU                                                CART    40
      Y(9)= X(4)*SLA*CMU-X(5)*SLA*SMU-X(6)*CLA                               CART    41
      Y(10)= (X(10)-X(4))*CLA*CMU-(X(11)-X(5))*CLA*SMU+(X(12)-X(6))*SLA      CART    42
      Y(11)=-(X(10)-X(4))*SMU-(X(11)-X(5))*CMU                               CART    43
      Y(12)= (X(10)-X(4))*SLA*CMU-(X(11)-X(5))*SLA*SMU-(X(12)-X(6))*CLA      CART    44
      RETURN                                                                 CART    45
      END                                                                    CART    46
```

SETY

```
ATAN2      REAL      2 LIBRARY      29
COS        REAL      1 LIBRARY       7          9          31         33
RSS        REAL      3              28
SIN        REAL      1 LIBRARY       8         10          32         34

ICS
AM LENGTH     430B      280
```

```
SUBROUTINE  SETY                                  COC 6600 FTN V3.0-P308 OPT=1  08/29/72  11.32.27.

                      SUBROUTINE SETY(Y)                                          SETY       2
      C                                                                           SETY       3
      C     SUBROUTINE TO COMPUTE  STATE VECTORS OF INTEREST-                     SETY       4
      C     Y(1)=TIME                                                           SETY       5
      C     Y(2)-Y(19)  NAVIGATED BETELGEUSE STATE                              SETY       6
      C     Y(20)-Y(37) ENVIRONMENT BETELGEUSE STATE                            SETY       7
      C     Y(38)-Y(55) NAVIGATED CARTESIAN STATE                               SETY       8
      C     Y(56)-Y(67) L.T. S/C ERROR STATE                                    SETY       9
      C     Y(68)-Y(79) L.T. TGT ERROR STATE                                    SETY      10
      C     Y(80)-Y(97) CARTESIAN DEVIATION VECTOR                              SETY      11
      C                                                                         SETY      12
                      DIMENSION Y(100),XE(12),XCS(12),XCT(12),RXVT(3,3)         SETY      13
     1                ,XS(12),XI(12)
```

ROUTINE SETY

THIS ROUTINE IS LEFT OVER FROM AN EARLY VERSION OF THE NAVIGATION PROGRAM. IT HAS ONLY ONE OPERATIVE INSTRUCTION AND CALLS TO SETY MAY BE REPLACED BY THAT INSTRUCTION WHEREVER THEY ARE FOUND IN THE PROGRAM.

AREA A — CALL TO CART1 IS ONLY OPERATIVE INSTRUCTION IN ROUTINE. LOADS BETELGEUSE ESTIMATED STATE INTO Y(38)-Y(49) AS CARTESIAN VECTOR.

```
C
C
C     CONSTRUCT CARTESIAN VECTORS FOR NAV AND ENV STATES                A
            CALL CART1(Y(38),Y(2))
        RETURN                                                            A
        ENTRY FAKE
            CALL CART1(XE,Y(20))
C       DEFINE CARTESIAN DIFFERENCE VECTOR
            CALL MATADD(XE,Y(38),Y(80),12,1,2)
C       LOAD UPPER CART. DEV. VECTOR WITH ESTIMATED CONSTANTS
C       CONSTRUCT PSEUDO-BETELGEUSE VECTORS FOR NAV AND ENV
            DO 10 I=1,6
            Y(I+91)=Y(I+31)-Y(I+13)
            XS(I)=Y(I+37)
            XS(I+6)=XE(I)
            XT(I)=Y(I+43)
            XT(I+6)=XE(I+6)
   10       CONTINUE
            CALL CART2(XS,XCS)
            CALL CART2(XT,XCT)
C           CONSTRUCT TRANSFORM TO LOCAL TANGENT FOR S/C
            CALL TBG(XCS(7),XCS(8),XCS(9),RXVT)
            CALL MATMUL(RXVT,XCS( 4),Y(62),3,3,1)
            CALL MATMUL(RXVT,XCS( 7),Y(59),3,3,1)
            CALL MATMUL(RXVT,XCS(10),Y(65),3,3,1)
C       CONSTRUCT TRANSFORM TO LOCAL TANGENT FOR TGT
            CALL TBG(XCT(7),XCT(8),XCT(9),RXVT)
            CALL MATMUL(RXVT,XCT( 4),Y(68),3,3,1)
            CALL MATMUL(RXVT,XCT( 7),Y(77),3,3,1)
            CALL MATMUL(RXVT,XCT(10),Y(71),3,3,1)
            DO 30 I=1,3
            Y(I+55)=XCS(I)
            Y(I+73)=XCT(I)
   30       CONTINUE
            RETURN
            END
```

```
SETY    15
SETY    16
SETY    17
SETY    18
SETY    19
AUTOW    3
AUTOW    4
SETY    20
SETY    21
SETY    22
SETY    23
SETY    24
SETY    25
SETY    26
SETY    27
SETY    28
SETY    29
SETY    30
SETY    31
SETY    32
SETY    33
SETY    34
SETY    35
SETY    36
SETY    37
SETY    38
SETY    39
SETY    40
SETY    41
SETY    42
SETY    43
SETY    44
SETY    45
SETY    46
SETY    47
SETY    48
SETY    49
```

REL

ROUTINE REL

SUBROUTINE REL COMPUTES RANGE, R-RATE, AZIMUTH AND ELEVATION FROM A TWO-VEHICLE CARTESIAN VECTOR AND THREE ORTHONORMAL VECTORS. CONTRARY TO COMMENT CARD, UNIT VECTORS NEED NOT BE NAV BASE VECTORS (SEE AREA L, ROUTINE POPOUT).

```
      SUBROUTINE REL(Y,S,REF1,REF2,REF3)                                REL     2
C                                                                       REL     3
      DIMENSION Y(12), S(4)                                             REL     4
C                                                                       REL     5
      COMMON VAR                                                        REL     6
      DIMENSION VAR(5600), P(5000), SAVE(950), BLK(700)                 REL     7
      EQUIVALENCE (VAR(601),P(1))                                       REL     8
      EQUIVALENCE (P(350),SAVE(1))                                      REL     9
     *,           (P(1300),BLK(1))                                      REL    10
      DIMENSION REF1(3), REF2(3), REF3(3)                               REL    11
C                                                                       REL    12
      DIMENSION DX(6),DXNB(6),UR(3)                                     REL    13
C                                                                       REL    14
C     SUBROUTINE TO COMPUTE RANGE, RANGE RATE, AZIMUTH AND ELEVATION IN REL    15
C     CURRENT NAVIGATION BASE COORDINATES                               REL    16
C                                                                       REL    17
      DO 5 I=1,6                                                        REL    18
    5 DX(I)=Y(I+6)-Y(I)                                                 REL    19
      CALL UVEC(DX(1),DX(2),DX(3),UR)                                   REL    20
      S(1)=DOT(UR,DX(1))                                                REL    21
      S(2)=DOT(UR,DX(4))                                                REL    22
      S(3)=ATAN(DOT(UR,REF3)/DOT(UR,REF2))                              REL    23
      S(4)=ASIN(DOT(UR,REF1))                                           REL    24
      RETURN                                                            REL    25
      END                                                               REL    26
```

SETUP

ROUTINE SETUP

SETS UP OUTPUT FORM OF BETELGEUSE STATE.
CONVERTS LONGITUDE AND LATITUDE TO DEGREES.
COMPUTES S/C HORIZONTAL SPEED.
DEFINES HEADING ANGLE AS A POSITIVE ROTATION ABOUT S/C RADIUS VECTOR FROM DUE WEST.

```
      SUBROUTINE SETUP(Y,OUT)                                          SETUP      2
C                                                                      SETUP      3
C     SETS UP A STANDARD BETEL OUTPUT                                  SETUP      4
C                                                                      SETUP      5
      DIMENSION Y(12),OUT(12)                                          SETUP      6
C                                                                      SETUP      7
      OUT( 1)=Y(1)                                                     SETUP      8
      OUT( 2)=Y(2)*57.2957795                                          SETUP      9
      OUT( 3)=Y(3)*57.2957795                                          SETUP     10
      OUT( 4)=Y(7)                                                     SETUP     11
      OUT(5)=RSS(Y(8),Y(9),0.)                                         SETUP     12
      OUT(6)=ATAN2(Y(9),Y(8))*57.2957795                               SETUP     13
      DO 5 I=1,3                                                       SETUP     14
      OUT(I+6)=Y(I+3)                                                  SETUP     15
    5 OUT(I+9)=Y(I+9)                                                  SETUP     16
      RETURN                                                           SETUP     17
      END                                                              SETUP     18
```

GARBAGE

ROUTINE GARBAGE

SENSOR NOISE MODEL. CALCULATES BIASES AND NOISE FROM A A PREUDO-RANDOM NORMAL DISTRIBUTION. LOADS RELATIVE PARAMETER MEASURED VALUE LOCATION FOR BVEC.

AREA A — CHECK TO SEE IF THIS IS FIRST PASS THROUGH THIS ROUTINE ON THIS MONTE-CARLO CYCLE. IF NOT, GO TO NOISE COMPUTATIONS. OTHERWISE, COMPUTE VALUES FOR RANGE, R-RATE, AZ AND EL BIASES.
LOAD BIASES INTO ACTUAL BETELGEUSE STATE, Y(32)-Y(35).

AREA B — STANDARD DEVIATION COMPUTATIONS. FOR RANGE AND R-RATE, SIGMA IS A FRACTION OF THE PARAMETER VALUE, DOWN TO A MINIMUM. AREA B SELECTS MAX OF PARAMETER FRACTION AND ITS MINIMUM. THIS COMPUTATION IS PERFORMED EACH PASS THROUGH GARBAGE.

AREA C — THIS INSTRUCTION ASSURES THAT THE RANGE MEASUREMENT IS A POSITIVE NUMBER. AT SMALL RANGES, QQ(1) COULD BECOME NEGATIVE AFTER THE ADDITION OF NOISE AND BIASES TO S(1).

```
      SUBROUTINE GARBAGE(S)                                                 GARBAGE    2
C                                                                           GARBAGE    3
      DIMENSION  S(4)                                                       GARBAGE    4
C                                                                           GARBAGE    5
      COMMON VAR                                                            GARBAGE    6
      DIMENSION VAR(5600), NTEGER(100), P(5000), SAVE(950), BLK(700)        GARBAGE    7
     *,         Y(100)                                                      GARBAGE    8
      EQUIVALENCE (VAR(1),Y(1))                                             GARBAGE    9
     *,           (VAR(401),NTEGER(1))                                     GARBAGE   10
     *,           (VAR(601),P(1))                                          GARBAGE   11
      EQUIVALENCE (NTEGER(29),NGUIDE)                                       GARBAGE   12
      EQUIVALENCE (P(350),SAVE(1))                                          GARBAGE   13
     *,           (P(1300),BLK(1))                                         GARBAGE   14
      DIMENSION QQ(4), SIG(4), C(10), REFMAT(3,3), XNBN(3), YNBN(3)         GARBAGE   15
     *,         ZNBN(3), NE(10), NM(10), OTL(10), OTN(10), OTM(10)         GARBAGE   16
     *,         USP(10), USV(10), UTP(10), UTV(10), SR(10), SRD(10)        GARBAGE   17
     *,         SO(10), SC1(10), SC2(10), NW(10), TLM(10), NS(3)           GARBAGE   18
     *,         ZTZ(4), SZ(4), TALIGN(10), XNBE(3), YNBE(3), ZNBE(3)       GARBAGE   19
     *,         X(18), WE(18,27)                                           GARBAGE   20
      EQUIVALENCE (SAVE(1),QQ(1)),          (SAVE(5),SIG(1))              GARBAGE   21
     *,           (SAVE(9),C(1)),          (SAVE(19),REFMAT(1,1))        GARBAGE   22
     *,           (SAVE(28),XNBN(1)),      (SAVE(31),YNBN(1))            GARBAGE   23
     *,           (SAVE(34),ZNBN(1)),      (SAVE(37),NE(1))              GARBAGE   24
     *,           (SAVE(47),NM(1)),        (SAVE(57),OTL(1))             GARBAGE   25
     *,           (SAVE(67),OTN(1)),       (SAVE(77),OTM(1))             GARBAGE   26
     *,           (SAVE(87),USP(1)),       (SAVE(97),USV(1))             GARBAGE   27
     *,           (SAVE(107),UTP(1)),      (SAVE(117),UTV(1))            GARBAGE   28
     *,           (SAVE(127),SR(1)),       (SAVE(137),SRD(1))            GARBAGE   29
     *,           (SAVE(147),SO(1)),       (SAVE(157),SC1(1))            GARBAGE   30
     *,           (SAVE(167),SC2(1)),      (SAVE(177),NW(1))             GARBAGE   31
     *,           (SAVE(187),TLM(1)),      (SAVE(197),NS(1))             GARBAGE   32
     *,           (SAVE(200),ZTZ(1)),      (SAVE(204),SZ(1))             GARBAGE   33
     *,           (SAVE(208),TALIGN(1)),   (SAVE(218),NALIGN)            GARBAGE   34
     *,           (SAVE(229),XNBE(1)),     (SAVE(232),YNBE(1))           GARBAGE   35
     *,           (SAVE(235),ZNBE(1)),     (SAVE(258),X(1))              GARBAGE   36
     *,           (SAVE(276),WE(1,1))                                      GARBAGE   37
C                                                                           GARBAGE   38
      EQUIVALENCE (C(1),TW)                                                 GARBAGE   39
C                                                                           GARBAGE   40
C                                                                           GARBAGE   41
      COMMON/GARB/RVAR,RVARMIN,VVAR,VVARMIN,VARAZ,VAREL,NFAMV               GARBAGE   42
     *,           BR,BV,BAZ,BEL,NFAMB                                      GARBAGE   43
```

```
C                                                                       GARBAGE    45
C     CHECK IF INITIAL VALUES FO BIASES HAVE BEEN SET IN            A   GARBAGE    46
      IF(VAR(1).GT.1.) GO TO 5                                          GARBAGE
      BR=UNURN(0,NFAMB,0.,BRO)                                          GARBAGE
      BV=UNURN(0,NFAMB,0.,BVO)                                          GARBAGE    49
      BAZ=UNURN(0,NFAMB,0.,BAZO)                                        GARBAGE    50
      BEL=UNURN(0,NFAMB,0.,BELO)                                        GARBAGE    51
      Y(32) = BR                                                        GARBAGE    52
      Y(33) = BV                                                        GARBAGE    53
      Y(34) = BAZ                                                       GARBAGE    54
      Y(35) = BEL                                                   A   GARBAGE    55
    5 CONTINUE                                                          GARBAGE    56
```

SUBROUTINE GARBAGE CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
C     COMPUTE MEASUREMENT NOISE AND AOD BIASES                      B   GARBAGE    57
      SIG(1)=AMAX1(S(1)*RVAR,RVARMIN)                                   GARBAGE    58
      SIG(2)=AMAX1(ABS(S(2)*VVAR),VVARMIN)                              GARBAGE    59
      SIG(3)=VARAZ                                                      GARBAGE    60
      SIG(4)=VAREL                                                  B   GARBAGE    61
      QQ(1) = S(1) + UNURN(0,NFAMV,BR,SIG(1))                           GARBAGE    62
      QQ(1) = ABS(QQ(1))                                            C   GARBAGE    63
      QQ(2) = S(2) + UNURN(0,NFAMV,BV,SIG(2))                           GARBAGE    64
      QQ(3) = S(3) + UNURN(0,NFAMV,BAZ,SIG(3))                          GARBAGE    65
      QQ(4) = S(4) + UNURN(0,NFAMV,BEL,SIG(4))                          GARBAGE    66
      IF(VAR(1).LT.5.) PRINT 100,BR,BV,BAZ,BEL,(SIG(I),I=1,4)           GARBAGE    67
  100 FORMAT(/1X,8E15.6)                                                GARBAGE    68
      RETURN                                                            GARBAGE    69
      END                                                               GARBAGE    70
```

ALIGN

ROUTINE ALIGN

SIMPLE-MINDED PLATFORM MODEL. CREATES DRIFT RATES AT INITIAL PASS, MISALIGNMENT BIASES EVERY ALIGNMENT.

AREA A — CHECK THE ALIGNMENT COUNTER TO SEE IF THIS PASS IS FIRST ALIGNMENT. IF IT IS VISIT UNURN TO CREATE INITIAL PLATFORM DRIFT RATES. OTHERWISE, PASS TO MISALIGNMENT COMPUTATION.

AREA B — INCREMENT THE ALIGNMENT COUNTER.

AREA C — CALCULATE THE STARTING LOCATION IN DATA ARRAY FOR DATA FROM THIS ALIGNMENT.
STORE THREE RATES AND THREE INITIAL MISALIGNMENTS.

AREA D — REFER TOTAL PLATFORM DRIFT TO TIME OF ALIGNMENT. ON INPUT, THE FIRST ALIGNMENT MAY BE SPECIFIED AS HAVING OCCURRED AT A TIME PREVIOUS TO THE START OF THE RUN: FOR EXAMPLE, TALIGN(1) = -600.SPECIFIES AN ALIGNMENT 10 MINUTES BEFORE START OF RUN. AREA D CALCULATES TOTAL DRIFT TO PRESENT TIME.

AREA E — CONVERT DRIFT RATE FROM RAD/SEC TO MR/HR AND INITIAL MISALIGNMENT FROM RAD TO MR FOR OUTPUT.

AREA F — STORE TIME OF ALIGNMENT.

AREA G — COMPUT REFMAT. THIS IS DEFINED AS THE ESTIMATED TRANSFORMATION MATRIX FROM BRF TO PLATFORM AXES. APOLLO USAGE DEFINES "NOMINAL" ALIGNMENT AS PLATFORM AXES COINCIDENT WITH LOCAL VERTICAL UNIT VECTORS AT TALIGN.

```
      SUBROUTINE ALIGN                                                  ALIGN     2
C                                                                       ALIGN     3
C     SUBROUTINE TO SIMULATE THE PERFORMANCE OF A PLATFORM ALIGNMENT    ALIGN     4
C                                                                       ALIGN     5
      COMMON VAR                                                        ALIGN     6
      DIMENSION VAR(5600), Y(100), DYDX(100), Q(100), FIRSTY(100)       ALIGN     7
     *,         NTEGER(100), O(100), P(5000)                           ALIGN     8
      EQUIVALENCE (VAR(1),Y(1))                                         ALIGN     9
     *,           (VAR(101),OYOX(1))                                   ALIGN    10
     *,           (VAR(201),O(1))                                      ALIGN    11
     *,           (VAR(301),FIRSTY(1))                                 ALIGN    12
     *,           (VAR(401),NTEGER(1))                                 ALIGN    13
     *,           (VAR(501),O(1))                                      ALIGN    14
     *,           (VAR(601),P(1))                                      ALIGN    15
      DIMENSION SAVE(950), BLK(700), OATA(350), COV(24,24)              ALIGN    16
      EQUIVALENCE (P(350),SAVE(1))                                      ALIGN    17
     *,           (P(1300),BLK(1))                                     ALIGN    18
     *,           (P(4074),OATA(1))                                    ALIGN    19
     *,           (P(4424),COV(1,1))                                   ALIGN    20
      DIMENSION QQ(4), SIG(4), C(10), REFMAT(3,3), XNBN(3), YNBN(3)     ALIGN    21
     *,         ZNBN(3), NE(10), NM(10), OTL(10), OTN(10), OTM(10)     ALIGN    22
     *,         USP(10), USV(10), UTP(10), UTV(10), SR(10), SRD(10)    ALIGN    23
     *,         SO(10), SC1(10), SC2(10), NW(10), TLM(10), NS(3)       ALIGN    24
     *,         ZTZ(4), SZ(4), TALIGN(10), XNBE(3), YNBE(3), ZNBE(3)   ALIGN    25
     *,         X(18), WE(18,27)                                       ALIGN    26
      EQUIVALENCE (SAVE(1),QQ(1)),          (SAVE(5),SIG(1))          ALIGN    27
     *,           (SAVE(9),C(1)),          (SAVE(19),REFMAT(1,1))    ALIGN    28
     *,           (SAVE(28),XNBN(1)),      (SAVE(31),YNBN(1))        ALIGN    29
     *,           (SAVE(34),ZNBN(1)),      (SAVE(37),NE(1))          ALIGN    30
     *,           (SAVE(47),NM(1)),        (SAVE(57),OTL(1))         ALIGN    31
     *,           (SAVE(67),OTN(1)),       (SAVE(77),OTM(1))         ALIGN    32
     *,           (SAVE(87),USP(1)),       (SAVE(97),USV(1))         ALIGN    33
     *,           (SAVE(107),UTP(1)),      (SAVE(117),UTV(1))        ALIGN    34
     *,           (SAVE(127),SR(1)),       (SAVE(137),SRD(1))        ALIGN    35
     *,           (SAVE(147),SO(1)),       (SAVE(157),SC1(1))        ALIGN    36
     *,           (SAVE(167),SC2(1)),      (SAVE(177),NW(1))         ALIGN    37
     *,           (SAVE(187),TLM(1)),      (SAVE(197),NS(1))         ALIGN    38
     *,           (SAVE(200),ZTZ(1)),      (SAVE(204),SZ(1))         ALIGN    39
     *,           (SAVE(208),TALIGN(1)),   (SAVE(218),NALIGN)        ALIGN    40
     *,           (SAVE(229),XNBE(1)),     (SAVE(232),YNBE(1))       ALIGN    41
     *,           (SAVE(235),ZNBE(1)),     (SAVE(258),X(1))          ALIGN    42
     *,           (SAVE(276),WE(1,1))                                ALIGN    43
      DIMENSION PPIFT(3), PATE(3), PI(3), PREFI(3), UX(3), UY(3), UZ(3) ALIGN    44
```

```
     *             DUMMY(3,3)                                                  ALIGN     45
      EQUIVALENCE (Y(98),DRIFT(1))                                             ALIGN     46
     *,           (DYDX(98),RATE(1))                                          ALIGN     47
     *,           (NTEGER(33),LIGN)                                           ALIGN     48
      EQUIVALENCE (BLK(1),RT(1))                                               ALIGN     49
     *,           (BLK(4),DRFT(1))                                            ALIGN     50
     *,           (BLK(7),UX(1))                                              ALIGN     51
     *,           (BLK(10),UY(1))                                             ALIGN     52
     *,           (BLK(13),UZ(1))                                             ALIGN     53
C                                                                             ALIGN     54
C     PLATFORM MODEL IS 3-GIMBAL WITH INITIAL 1-SIGMA BIASES AND DRIFT        ALIGN     55
C     AS DEFINED BELOW                                                        ALIGN     56
```

SUBROUTINE ALIGN CDC 6600 FTN V3.0-P308 DPT=1 08/29/72 11.32.27.

```
C                                                                             ALIGN     57
      COMMON/ALIG/GDR,ALIGNB,NFAMB                                            ALIGN     58
C                                                                             ALIGN     59
C     CHECK IF THIS IS FIRST ALIGNMENT                                    A   ALIGN     60
      IF(LIGN.GT.0) GO TO 10                                                  ALIGN     61
C     SET UP INITIAL DRIFT RATES FOR THIS RUN                                 ALIGN     62
      DO 5 I=1,3                                                              ALIGN     63
    5 RATE(I)=UNURN(0,NFAMB,0.,GDR)                                           ALIGN     64
   10 CONTINUE                                                            A   ALIGN     65
      LIGN=LIGN + 1                                                       B   ALIGN     66
C     CONSTRUCT ALIGNMENT BIASES                                              ALIGN     67
      DO 15 I=1,3                                                             ALIGN     68
   15 DRIFT(I)=UNURN(0,NFAMB,0.,ALIGNB)                                       ALIGN     69
C     REFER INTEGRATED DRIFT TO TIME OF ALIGNMENT AND SCAL OUTPUTS            ALIGN     70
      DT=Y(1) - TALIGN(LIGN)                                                  ALIGN     71
      DO 20 I=1,3                                                         C   ALIGN     72
C     CALCULATE INDEX TO STORE ALIGNMENT DATA IN ARRAY                        ALIGN     73
      MM=(LIGN-1)*7 + 250 + I                                                 ALIGN     74
      DATA(MM)=RATE(I)                                                        ALIGN     75
      DATA(MM+3)=DRIFT(I)                                                 C   ALIGN     76
      DRIFT(I)=DRIFT(I) + DT*RATE(I)                                      D   ALIGN     77
      RT(I)=RATE(I)*3600000.                                              E   ALIGN     78
   20 DRFT(I)=DRIFT(I)*1000.                                              E   ALIGN     79
      MM=LIGN*7+250                                                       F   ALIGN     80
      DATA(MM)=TALIGN(LIGN)                                                   ALIGN     81
```

```
C                                                                       ALIGN     .82
C     COMPUTE REFMAT AT TALIGN, DEFINED AS THE NOMINAL-          G      ALIGN     83
C               SMX= UNIT(R)                                          ALIGN     84
C               SMY= UNIT(R X V)                                      ALIGN     85
C               SMZ= UNIT(SMX X SMY)                                  ALIGN     86
C                                                                     ALIGN     87
      CALL UVEC(Y(38),Y(39),Y(40),UX)                                 ALIGN     88
      CALL UCROSS(UX,Y(41),UY)                                        ALIGN     89
      CALL UCROSS(UX,UY,UZ)                                           ALIGN     90
      CALL TRN(UX,UY,UZ,REFMAT,DUMMY)                            G    ALIGN     91
      PRINT 100, LIGN, (DRFT(I),I=1,3), (RT(I),I=1,3)                 ALIGN     92
  100 FORMAT(/14X,24HTHIS IS ALIGNMENT NUMBER,I5,23H  BIASES AND RATES ALIGN     93
     1ARE-,6F10.5)                                                    ALIGN     94
      RETURN                                                          ALIGN     95
      END                                                             ALIGN     96
```

ROUTINE REF

COMPUTES UNIT VECTORS OF ACTUAL LOCAL VERTICAL, ACTUAL NAV BASE, ESTIMATED LOCAL VERTICAL AND ESTIMATED NAV BASE. ACTUAL AND ESTIMATED QUANTITIES DIFFER BY PLATFORM DRIFT. CURRENT NAV BASE VECTORS ARE DEFINED AS THE LINE-OF-SIGHT SYSTEM. SUBROUTINE IS ALWAYS CALLED WITH ESTIMATED CARTESIAN STATE.

AREA A CALCULATE ESTIMATED LOCAL VERTICAL VECTORS.

AREA A1 CALCULATE ESTIMATED LINE-OF-SIGHT VECTORS.

AREA B CALCULATE THE TOTAL PLATFORM DRIFT MATRIX, GAMD. GAMD IS THE TRANSFORMATION FROM ACTUAL TO ESTIMATED PLATFORM AXES. LET $\underline{V}$ BE ANY ESTIMATE OF A VECTOR IN THE BRF FRAME AND $\underline{V}^*$ BE ITS VALUE AFTER APPLICATION OF PLATFORM ERRORS:

$$\underline{V}_P = \underline{REFMAT} \times \underline{V} \qquad \text{VECTOR IN ESTIMATED PLATFORM FRAME}$$

$$\underline{V}^*_P = \underline{GAMD}^T \times \underline{V}_P \qquad \text{VECTOR IN ACTUAL PLATFORM FRAME}$$

$$\underline{V}^* = \underline{REFMAT}^T \times \underline{V}^*_P \qquad \text{DISTURBED VECTOR IN BRF FRAME}$$

$$= \underline{REFMAT}^T \times \underline{GAMD}^T \times \underline{REFMAT} \times \underline{V}$$

$$\doteq (\underline{GAMD} \times \underline{REFMAT})^T \times \underline{REFMAT} \times \underline{V}$$

CALCULATE $\underline{GAMD} \times \underline{REFMAT}$.

CALULATE $(\underline{GAMD} \times \underline{REFMAT})^T$.

CALCULATE $(\underline{GAMD} \times \underline{REFMAT})^T \times \underline{REFMAT}$.

AREA C APPLY THE JUST COMPUTED MATRIX TO THE ESTIMATED LOCAL VERTICAL VECTORS TO GET THE ACTUAL VECTORS. APPLY THE SAME MATRIX TO THE ESTIMATED NAV BASE VECTORS TO GET THE ACTUAL NAV BASE VECTORS.

```
      SUBROUTINE REF(Y)                                                 REF     2
C                                                                       REF     3
C     SUBROUTINE TO COMPUTE THE ACTUAL (XNBE,YNBE,ZNBE) AND ASSUMED     REF     4
C     (XNBN,YNBN,ZNBN) UNIT VECTORS OF THE NAVIGATION BASE AXES IN      REF     5
C     THE BASIC REFERENCE FRAME                                         REF     6
C     SUBROUTINE ALSO COMPUTES UNIT VECTORS OF THE ASSUMED (XLVN,YLVN,  NOSHIT  10
C     ZLVN) AND ACTUAL (XLVE,YLVE,ZLVE) LOCAL VERTICAL IN-PLANE FRAME   NOSHIT  11
C                                                                       REF     7
      DIMENSION Y(12)                                                   REF     8
C                                                                       REF     9
      COMMON VAR                                                        REF    10
      DIMENSION VAR(5600), P(5000), SAVE(950), BLK(700)                 REF    11
      EQUIVALENCE (VAR(601),P(1))                                       REF    12
      EQUIVALENCE (P(350),SAVE(1))                                      REF    13
     *,           (P(1300),BLK(1))                                      REF    14
      DIMENSION REFMAT(3,3), XNBN(3), YNBN(3), ZNBN(3)                  REF    15
      EQUIVALENCE (SAVE(19),REFMAT(1,1))                                REF    16
     *,           (SAVE(28),XNBN(1))                                    REF    17
     *,           (SAVE(31),YNBN(1))                                    REF    18
     *,           (SAVE(34),ZNBN(1))                                    REF    19
      DIMENSION OUM(3,3), OUMT(3,3), DICKUP(3,3), GAMO(3,3), YB(100)    REF    20
     *,         XLVE(3),YLVE(3),ZLVE(3),XLVN(3),YLVN(3),ZLVN(3)        NOSHIT  12
      EQUIVALENCE (P(1),YB(1))                                          REF    21
      EQUIVALENCE (YB(98),O1)                                           REF    22
     *,           (YB(99),O2)                                          REF    23
     *,           (YB(100),O3)                                         REF    24
      EQUIVALENCE (BLK(201),OUM(1,1))                                   REF    25
     *,           (BLK(210),OUMT(1,1))                                 REF    26
     *,           (BLK(219),GAMO(1,1))                                 REF    27
     *,           (BLK(228),DICKUP(1,1))                               REF    28
     *,         (SAVE(762),XLVE(1))                                    NOSHIT  13
     *,         (SAVE(765),YLVE(1))                                    NOSHIT  14
     *,         (SAVE(768),ZLVE(1))                                    NOSHIT  15
     *,         (SAVE(771),XLVN(1))                                    NOSHIT  16
     *,         (SAVE(774),YLVN(1))                                    NOSHIT  17
     *,         (SAVE(777),ZLVN(1))                                    NOSHIT  18
C                                                                       REF    29
C     SUBROUTINE COMPUTES TRANSFORM FROM B.R.F. TO NAV. BASE FRAME      REF    30
C                                                                       REF    31
*     CURRENT NAV BASE IS LINE-OF-SIGHT SYSTEM DEFINED BY               NOSHIT  19
*                                                                       NOSHIT  20
*     UNIT(Y) = UNIT(RT - RS)                                           NOSHIT  21
*     UNIT(X) = UNIT(XNBN X ZLVN)
```

```
*     UNIT(Z) = UNIT(XNBN X YNBN)                                        NOSHIT     23
*                                                                        NOSHIT     24
      CALCULATE LOCAL VERTICAL UNIT VECTO                          A     NOSHIT
                                                                         NOSHIT     2
      CALL UVEC(Y(1),Y(2),Y(3),XLVN)                                     NOSHIT     27
      CALL UCROSS(XLVN,Y(4),ZLVN)                                        NOSHIT     28
      CALL UCROSS(ZLVN,XLVN,YLVN)                                        NOSHIT     29
C                                                                  A     NOSHIT     30
C     COMPUT NAV BASE UNIT VECTORS                                       NOSHIT     31
C                                                                        NOSHIT     32
      DX1 = Y(7) - Y(1)                                                  NOSHIT     33
      DX2 = Y(8) - Y(2)                                                  NOSHIT     34
```

SUBROUTINE REF COC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
      DX3 = Y(9) - Y(3)                                                  NOSHIT     35
      CALL UVEC(DX1,DX2,DX3,YNBN)                                        NOSHIT     36
      CALL UCROSS(YNBN,ZLVN,XNBN)                                        NOSHIT     37
      CALL UCROSS(XNBN,YNBN,ZNBN)                                        NOSHIT     38
C                                                                        NOSHIT     39
C                                                                        REF        42
C     COMPUTE THE ACTUAL VALUES OF THESE UNIT VECTORS IN BRF             REF        43
C                                                                  B     REF        44
      CALL MAT(D3,D2,D1,1,3,2,GAMD)                                      REF        45
      CALL MATMUL(GAMD,REFMAT,DUM,3,3,3)                                 REF        46
      CALL MATRAN(DUM,3,3,DUMT)                                          REF        47
      CALL MATMUL(DUMT,REFMAT,DICKUP,3,3,3)                        B     REF        48
      DO 10 I=1,7,3                                                C     REF        49
        CALL MATMUL(DICKUP,SAVE(I+770),SAVE(I+761),3,3,1)                NOSHIT     40
10    CALL MATMUL(DICKUP,SAVE(I+27),SAVE(I+228),3,3,1)             C     REF        50
      RETURN                                                             REF        51
      END                                                                REF        52
```

ROUTINE P20

SUBEXECUTIVE FOR NAVIAGATION MARKS. ADVANCES COVARIANCE, TAKES MARK, RETURNS UPDATED STATE.

AREA A THIS AREA IS LEFT OVER FROM THE ORIGINAL VERSION AND WAS NEVER ACTIVE.

AREA B READS UPDATED VECTOR BACK INTO ARGUMANT LOCATIONS.

```
      SUBROUTINE P20(T,Y,N)                                              P20    2
C                                                                        P20    3
      COMMON VAR                                                         P20    4
      DIMENSION VAR(5600), P(5000), SAVE(950), BLK(700)                  P20    5
      EQUIVALENCE (VAR(601),P(1))                                        P20    6
      EQUIVALENCE (P(350),SAVE(1))                                       P20    7
     *,           (P(1300),BLK(1))                                       P20    8
      DIMENSION REFMAT(3,3), X(18)                                       P20    9
      EQUIVALENCE (SAVE(19),REFMAT(1,1))                                 P20   10
     *,           (SAVE(258),X(1))                                       P20   11
C                                                                        P20   12
      DIMENSION Y(18)                                                    P20   13
C     SUBROUTINE TO CONTROL THE TAKING OF A NAVIGATION MARK              P20   14
C                                                                        P20   15
C     ADVANCE W TO CURRENT TIME                                          P20   16
      CALL ADVW(T,Y)                                                     P20   17
C     COMPUTE CUPRENT REFMAT                                        A    P20   18
C     CALL REF(REFMAT)                                              A    P20   19
C     CALCULATE AND INCORPORATE UPDATE                                   P20   20
      CALL BVEC(N)                                                       P20   21
      DO 30 I=1,18                                                  B    P20   22
   30 Y(I)=X(I)                                                     B    P20   23
      RETURN                                                             P20   24
      END                                                                P20   25
```

ADVW

ROUTINE ADVW

ADVANCES SQUARE-ROOT MATRIX OF THE COVARIANCE. SEE REFERENCE 1 FOR DISCUSSION OF THE METHOD.

AREA A — DEFINE FINAL TIME OF W AS CURRENT TIME.
SAVE INPUT VALUE OF NUMBER OF INTEGRATED VARIABLES.
SAVE INPUT VALUE OF INTEGRATION STEP SIZE.
SET INTEGRATED VARIABLES = 13 FOR RK. SINCE BOTTOM SIX ROWS OF W ARE ASSOCIATED WITH ESTIMATED CONSTANTS, THEY NEED NOT BE INTEGRATED. RK WILL INTEGRATE TIME AND FIRST TWELVE ROWS OF W, ASSOCIATED WITH S/C AND TGT STATES.

AREA B — DEFINE TGO AS CURRENT TIME (TWF) MINUS TIME TAG ON W.
IF TGO LESS THAN 1 SECOND, GO TO AREA H. DO NOT INTEGRATE.

AREA C — TAKE EACH COLUMN OF W AND UPDATE IT:
DEFINE TGO AS BEFORE.
ADD COLUMN TO SAVED VALUE OF ESTIMATED STATE.
CONVERT TO BETELGEUSE VECTOR FOR INTEGRATION.
ENTER RKW FOR ADVANCEMENT OF ONE STEP.

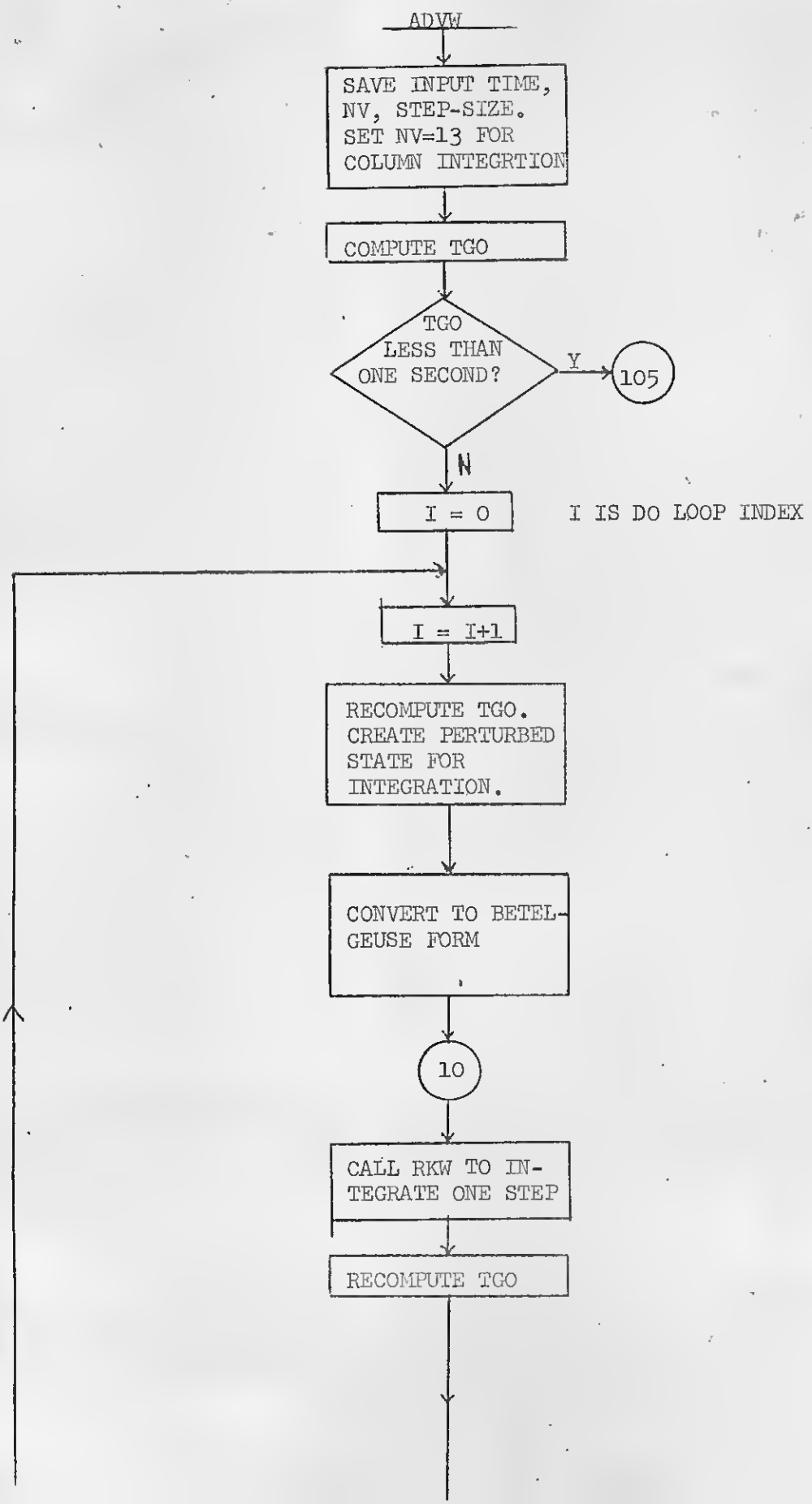
AREA D — DECREMENT TGO BY ONE STEP.
IF TGO LARGER THAN ONE STEP, CALL RKW.
IF TGO LESS THAN ONE SECOND, DO NEXT COLUMN.

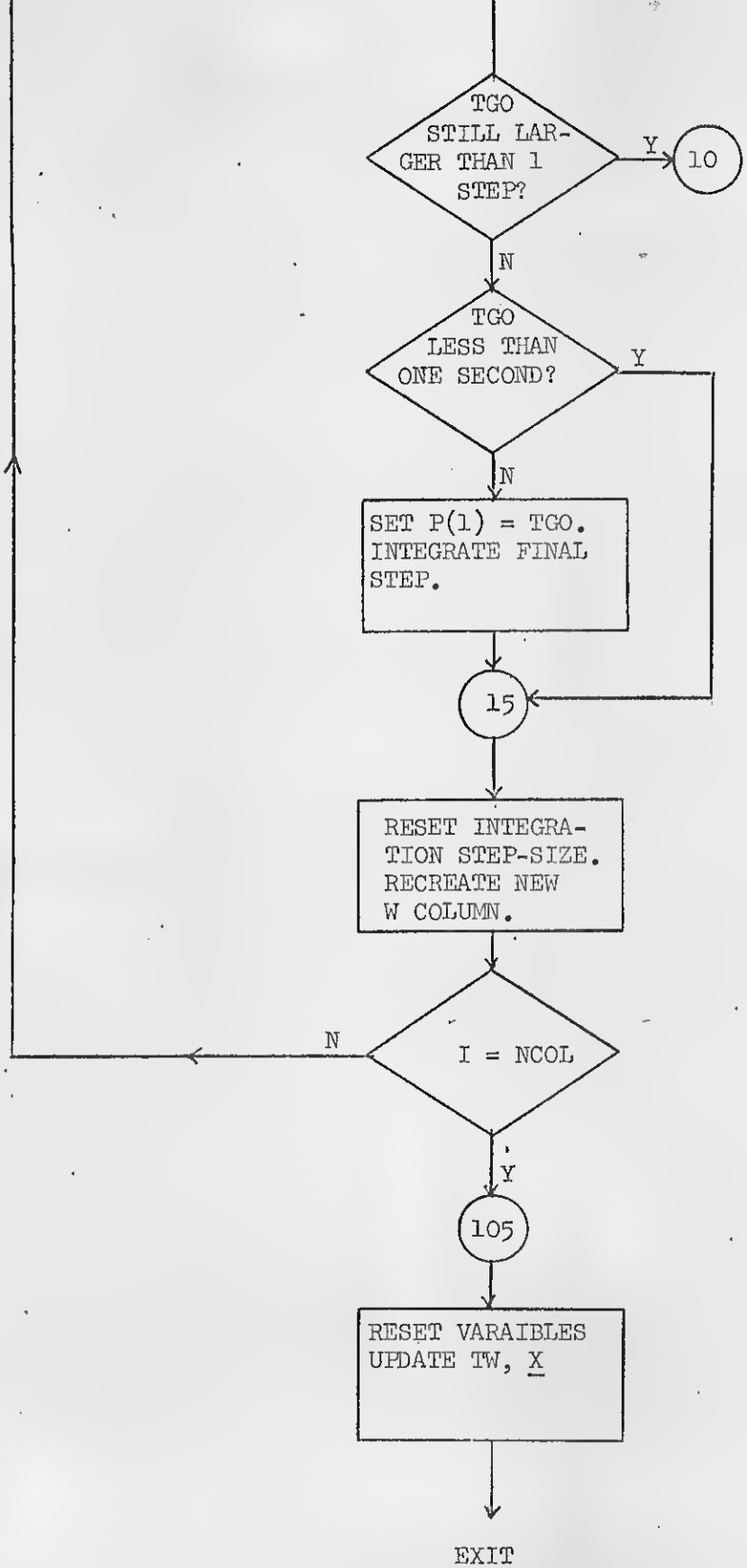
AREA E — SET STEP EQUAL TGO FOR FINAL PASS THROUGH RKW ON THIS COLUMN.
CALL RKW TO BRING COLUMN TO TWF.

AREA F — RESET STEPSIZE FOR NEXT COLUMN.

AREA G — RECONSTRUCT CARTESIAN VECTOR.
SUBTRACT OFF CURRENT ESTMATE TO FORM NEW COLUMN OF W.
GO ON TO NEXT COLUMN.

AREA H — ALL COLUMNS OF W ARE NOW UPDATED.
TO BE ON THE SAFE SIDE, REDEFINE PROGRAM TIME.
UPDATE TIME TAG ON W.
UPDATE SAVED STATE FOR NEXT PASS THROUGH ADVW. THIS WILL BE UPDATED AGAIN BY THE NAVIGATION MEASUREMENTS.


ADVW
SAVE INPUT TIME, NV, STEP-SIZE. SET NV=13 FOR COLUMN INTEGRTION
COMPUTE TGO
TGO LESS THAN ONE SECOND?
Y
105
N
I = 0
I IS DO LOOP INDEX
I = I+1
RECOMPUTE TGO. CREATE PERTURBED STATE FOR INTEGRATION.
CONVERT TO BETEL-GEUSE FORM
10
CALL RKW TO IN-TEGRATE ONE STEP
RECOMPUTE TGO


TGO STILL LARGER THAN 1 STEP?
Y
10
N
TGO LESS THAN ONE SECOND?
Y
N
SET P(1) = TGO. INTEGRATE FINAL STEP.
15
RESET INTEGRATION STEP-SIZE. RECREATE NEW W COLUMN.
I = NCOL
N
Y
105
RESET VARAIBLES UPDATE TW, X
EXIT

```
      SUBROUTINE ADVW(T,XF)                                              ADVW      2
C                                                                        ADVW      3
      COMMON VAR                                                         ADVW      4
      DIMENSION VAR(5600), Y(100), NTEGER(100), P(5000), SAVE(950),      ADVW      5
     *          BLK(700)                                                ADVW      6
      EQUIVALENCE (VAR(1),Y(1))                                          ADVW      7
     *,           (VAR(401),NTEGER(1))                                  ADVW      8
     *,           (VAR(601),P(1))                                       ADVW      9
      EQUIVALENCE (P(350),SAVE(1))                                       ADVW     10
     *,           (P(1300),BLK(1))                                      ADVW     11
C                                                                        ADVW     12
      DIMENSION UR1(3),UR2(3),UR(3)                                      ADVW     13
      DIMENSION D(10), X(18), WE(18,27)                                  ADVW     14
      EQUIVALENCE (SAVE(9),D(1))                                         ADVW     15
     *,           (SAVE(258),X(1))                                      ADVW     16
     *,           (SAVE(276),WE(1,1))                                   ADVW     17
C                                                                        ADVW     18
      EQUIVALENCE (D(1),TW)                                              ADVW     19
C                                                                        ADVW     20
      EQUIVALENCE (D(2),UR1(1)),(D(5),UR2(1))                            ADVW     21
     *,           (BLK(649),UR(1))                                      ADVW     22
C                                                                        ADVW     23
      DIMENSION XOP(12),XF(18),XFP(12)                                   ADVW     24
C     EQUIVALENCE (BLK(1),XOP(1)),(BLK(13),XFP(1))                       ADVW     25
      EQUIVALENCE (NTEGER(9),NCOL)                                       ADVW     26
C                                                                        ADVW     27
      COL = NCOL                                                         ADVW     28
C     SAVE CURRENT TIME NO. INT. VAR. STEP SIZE, SET I.V.=13             ADVW     29
      TWF=T                                                          A   ADVW     30
      NV=NTEGER(6)                                                       ADVW     31
      STEP=P(1)                                                          ADVW     32
      NTEGER(6)=13                                                   A   ADVW     33
C     CHECK TGO                                                      B   ADVW     34
      TGO=TWF-TW                                                         ADVW     35
      IF(ABS(TGO).LT.1.) GO TO 105                                   B   ADVW     36
C     ADVANCE W BY COLUMNS                                           C   ADVW     37
      DO 100 I=1,NCOL                                                    ADVW     38
      TGO=TWF - TW                                                       ADVW     39
C     CREATE PERTURBED STATE                                             ADVW     40
      DO 5 J=1,12                                                        ADVW     41
    5 XOP(J) = X(J) + WE(J,I)*SQRT(COL)                                  ADVW     42
C     CONVERT TO BETELGEUSE SYSTEM                                       ADVW     43
      CALL CART2(XOP,Y(2))
```

```
   10   CALL RKW                                                    C     ADVW    45
        TGO=TGO - P(1)                                              D     ADVW    46
        IF(TGO.GE.P(1)) GO TO 10                                          ADVW    47
        IF(ABS(TGO).LT.1.) GO TO 15                                 D     ADVW    48
        P(1)=TGO                                                    E     ADVW    49
        CALL RKW                                                    E     ADVW    50
   15   P(1)=STEP                                                   F     ADVW    51
C       RECTIFY RESULTING STATE AND COMPUTE NEW COLUMN OF W         G     ADVW    52
        CALL CART1(XFP,Y(2))                                              ADVW    53
        DO 20 J=1,12                                                      ADVW    54
   20   WF(J,I) = (XFP(J) - XF(J))/SQRT(COL)                        G     ADVW    55
  100   CONTINUE                                                          ADVW    56
```

WORKS

UBROUTINE ADVW CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
  105   CONTINUE                                                          ADVW    57
        Y(1)=TWF                                                    H     ADVW    58
        TW=TWF                                                            ADVW    59
        NTEGER(6)=NV                                                      ADVW    60
        DO 110 I=1,18                                                     ADVW    61
  110   X(I)=XF(I)                                                  H     ADVW    62
        RETURN                                                            ADVW    63
        END                                                               ADVW    64
```

DVEC

ROUTINE BVEC

COMPUTES GEOMETRY VECTORS AS SPECIFIED BY CALLING ROUTINE. CALLS FILTER TO UPDATE STATE AND COVARIANCE. SEE REFERENCE 1 FOR DISCUSSION.

AREA RR COMPUTES RANGE RATE MEASUREMENT GEOMETRY VECTOR:

$\underline{r}$ = [DX(1), DX(2), DX(3)] RELATIVE POSITION VECTOR

$\underline{v}$ = [DX(4), DX(5), DX(6)] RELATIVE VELOCITY VECTOR

$\underline{UR} = UNIT(\underline{r})$

RC = RANGE

$$\underline{B}_{RD} = \begin{bmatrix} -\underline{UR} \times (\underline{v} \times \underline{UR})/RC \\ -\underline{UR} \\ \underline{UR} \times (\underline{v} \times \underline{UR})/RC \\ \underline{UR} \end{bmatrix}$$

AREA A CALCULATE THE ESTIMATED RELATIVE POSITION VECTOR.

AREA B DEFINE CURRENT ESTIMATED RANGE-RATE AS $\underline{UR}^T\underline{v}$ PLUS RANGE-RATE BIAS ESTIMATE.

AREA C DEFINE CURRENT ESTIMATED RANGE AS $\underline{UR}^T\underline{r}$ PLUS RANGE BIAS ESTIMATE.

AREA D COMPUTE $\underline{UR} \times (\underline{v} \times \underline{UR})$ USING PORTIONS OF GEOMETRY VECTOR AS SCRATCH PAD.

AREA E LOAD B AS DEFINED ABOVE (AREA RR). B(14) IS THE PARTIAL OF R-RATE MEASUREMENT WITH RESPECT TO A BIAS ESTIMATE ERROR: SET EQUAL TO 1.

AREA F DEFINE MEASUREMENT RESIDUAL (DQ) AS MEASURED R-RATE MINUS ESTIMATED.
DEFINE R-RATE SIGMA AS MAX OF R-RATE FRACTION AND MINIMUM VALUE.

AREA F — SET MEASUREMENT TYPE FLAG FOR FILTER
UPDATE STATE WITH R-RATE MEASUREMENT (CALL FILTER)

AREA R — COMPUTES RANGE MEASUREMENT GEOMETRY VECTOR:

$$\underline{B}_R = \begin{bmatrix} -\underline{UR} \\ \underline{0}_3 \\ \underline{UR} \\ \underline{0}_3 \end{bmatrix}$$

AREA A — COMPUTE CURRENT RANGE UNIT VECTOR.

AREA B — DEFINE ESTIMATED RANGE AS IN AREA RR(C)

AREA C — LOAD B AS DEFINED ABOVE. B(13) IS THE PARTIAL OF A RANGE MEASUREMENT WITH RESPECT TO A RANGE BIAS ESTIMATE ERROR: SET EQUAL TO 1.

AREA D — DEFINE MEASUREMENT RESIDUAL AS MEASURED RANGE MINUS ESTIMATED RANGE.
DEFINE RANGE SIGMA AS MAX OF RANGE FRACTION AND MINIMUM VALUE.
SET MEASUREMENT TYPE FLAG FOR FILTER.
UPDATE STATE WITH RANGE MEASUREMENT (CALL FILTER)

AREA E — IF THIS IS A RANGE-ONLY (VHF) MARK, EXIT ROUTINE.

AREA OP — COMPUTES AZIMUTH AND ELEVATION MEASUREMENT GEOMETRY VECTORS:

REF1, REF2, REF3 — ESTIMATED NAV BASE UNIT VECTORS.

$$AZ = \text{ATAN}[(\underline{UR}^T\underline{REF3})/(\underline{UR}^T\underline{REF2})]$$

$$EL = \text{ASIN}[\underline{UR}^T\underline{REF1}]$$

THESE DEFINITIONS ARE NOT UNIQUE, BUT THEY MUST BE THE SAME AS THOSE IN SUBROUTINE REL.

AREA OP FOR AZ: $RCEL = \underline{UR}^T \underline{r}[\cos(EL)]$ PROJECTION OF RANGE INTO REF2, REF3 PLANE.

$$\underline{B}_{AZ} \begin{bmatrix} -\text{UNIT}(\underline{REF1} \times \underline{UR})/RCEL \\ \underline{0}_3 \\ \text{UNIT}(REF1 \times UR)/RCEL \\ \underline{0}_3 \end{bmatrix}$$

FOR EL: $RCEL = \underline{UR}^T \underline{r}$ RANGE MAGNITUDE

$$\underline{B}_{EL} \begin{bmatrix} -\text{UNIT}[\underline{UR} \times (\underline{REF1} \times \underline{UR})]/RCEL \\ \underline{0}_3 \\ \text{UNIT}[\underline{UR} \times (\underline{REF1} \times \underline{UR})]/RCEL \\ \underline{0}_3 \end{bmatrix}$$

AREA A COMPUTE THE CURRENT ESTIMATES OF AZ AND EL AS DEFINED ABOVE, INCLUDING CURRENT ESTIMATES OF BIASES X(15), X(16).

AREA B IF ELEVATION COMPONENT OF OPTICS MARK, GO TO AREA F TO COMPUTE GEOMETRY VECTOR.

AREA C COMPUTE AZIMUTH GEOMETRY VECTOR AS ABOVE.

AREA D DEFINE MEASUREMENT RESIDUAL.
DEFINE AZ SIGMA

AREA E DEFINE RANGE PROJECTION TERM.
GO TO AREA I TO LOAD B AND TAKE MARK.

AREA F DEFINE ELEVATION GEOMETRY VECTOR AS DESCRIBED ABOVE.

AREA G — DEFINE ELEVATION MEASUREMENT RESIDUAL.
DEFINE ELEVATION SIGMA.

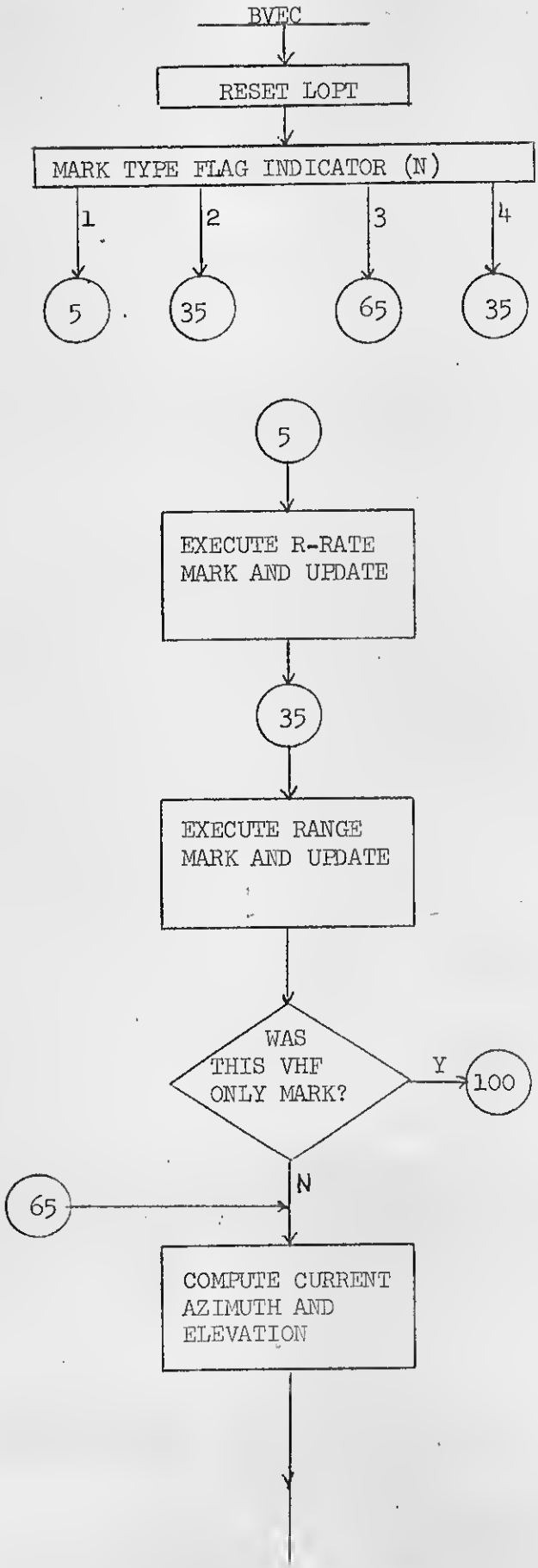
AREA H — DEFINE RANGE PROJECTION TERM.

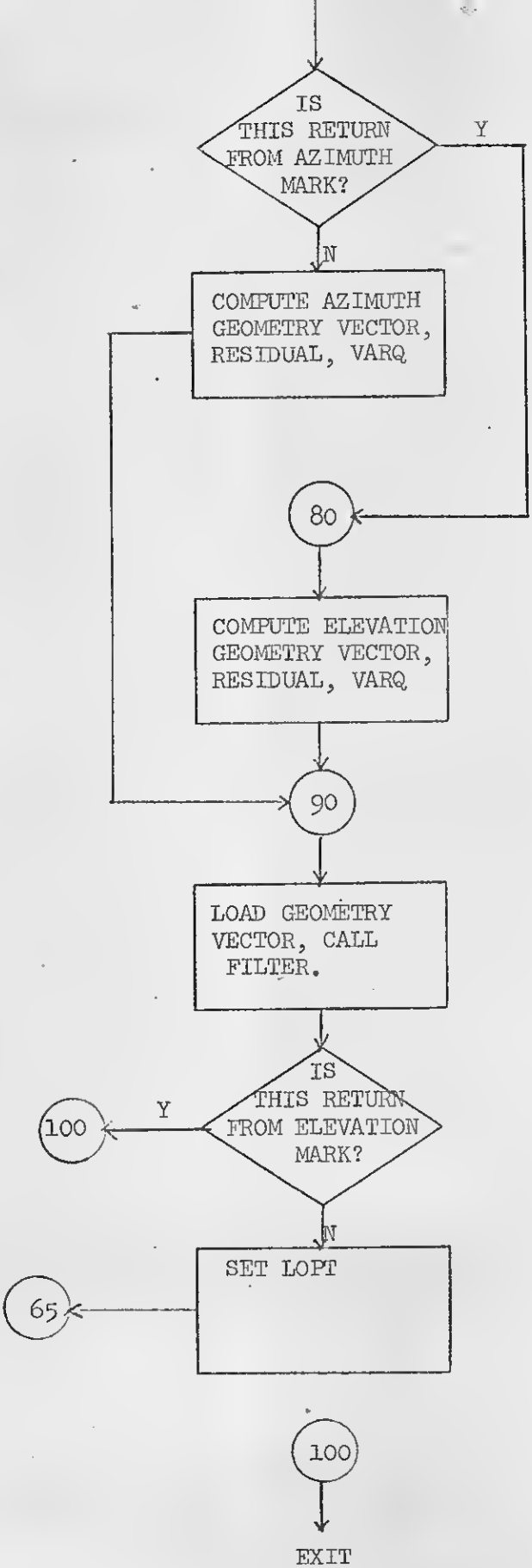
AREA I — LOAD B WITH GEOMETRY VECTOR AS DESCRIBED ABOVE.

AREA J — B(16) IS PARTIAL OF ELEVATION ANGLE MEASUREMENT WITH RESPECT TO BIAS ESTIMATE ERROR. IF THIS IS ELEVATION COMPONENT, SET B(16)=1.
B(15) IS PARTIAL OF AZIMUTH ANGLE MEASUREMENT WITH RESPECT TO BIAS ESTIMATE ERROR. IF THIS IS AZIMUTH COMPONENT, SET B(15)=1.

AREA K — SET APPROPRIATE MEASUREMENT TYPE FLAG FOR FILTER.
CALL FILTER TO TAKE MARK.

AREA L — IF ELEVATION MARK HAS JUST BEEN PROCESSED, EXIT ROUTINE.
LOPT FLAG WILL BE RESET ON NEXT ENTRY TO BVEC.
IF THIS WAS NOT AN ELEVATION MARK, GO TO AREA OP TO PERFORM THIS MEASUREMENT.


BVEC
RESET LOPT
MARK TYPE FLAG INDICATOR (N)
1
2
3
4
5
35
65
35
5
EXECUTE R-RATE MARK AND UPDATE
35
EXECUTE RANGE MARK AND UPDATE
WAS THIS VHF ONLY MARK?
Y
100
N
65
COMPUTE CURRENT AZIMUTH AND ELEVATION


IS THIS RETURN FROM AZIMUTH MARK?
Y
N
COMPUTE AZIMUTH GEOMETRY VECTOR, RESIDUAL, VARQ
80
COMPUTE ELEVATION GEOMETRY VECTOR, RESIDUAL, VARQ
90
LOAD GEOMETRY VECTOR, CALL FILTER.
IS THIS RETURN FROM ELEVATION MARK?
Y
100
N
SET LOPT
65
100
EXIT

```
      SUBROUTINE BVEC(N)                                                BVEC      2
C                                                                       BVEC      3
C     SUBROUTINE TO COMPUTE GEOMETRY VECTORS FOR NAVIGATION MARKS       BVEC      4
C                                                                       BVEC      5
      COMMON VAR                                                        BVEC      6
      DIMENSION VAR(5600), NTEGER(100), P(5000), SAVE(950), BLK(700)    BVEC      7
      EQUIVALENCE (VAR(401),NTEGER(1))                                  BVEC      8
     *,           (VAR(601),P(1))                                       BVEC      9
      EQUIVALENCE (NTEGER(29),NGUIDE)                                   BVEC     10
      EQUIVALENCE (P(350),SAVE(1))                                      BVEC     11
     *,           (P(1300),BLK(1))                                      BVEC     12
      DIMENSION OQ(4), SIG(4), C(10), REFMAT(3,3), XNBN(3), YNBN(3)     BVEC     13
     *,         ZNBN(3), NE(10), NM(10), DTL(10), OTN(10), OTM(10)     BVEC     14
     *,         USP(10), USV(10), UTP(10), UTV(10), SR(10), SRD(10)    BVEC     15
     *,         SO(10), SC1(10), SC2(10), NW(10), TLM(10), NS(3)       BVEC     16
     *,         ZTZ(4), SZ(4), TALIGN(10), XNBE(3), YNBE(3), ZNBE(3)   BVEC     17
     *,         X(18), WE(18,27)                                       BVEC     18
      EQUIVALENCE (SAVE(1),OO(1)),          (SAVE(5),SIG(1))          BVEC     19
     *,           (SAVE(9),C(1)),          (SAVE(19),REFMAT(1,1))    BVEC     20
     *,           (SAVE(28),XNBN(1)),      (SAVE(31),YNBN(1))        BVEC     21
     *,           (SAVE(34),ZNBN(1)),      (SAVE(37),NE(1))          BVEC     22
     *,           (SAVE(47),NM(1)),        (SAVE(57),DTL(1))         BVEC     23
     *,           (SAVE(67),OTN(1)),       (SAVE(77),DTM(1))         BVEC     24
     *,           (SAVE(87),USP(1)),       (SAVE(97),USV(1))         BVEC     25
     *,           (SAVE(107),UTP(1)),      (SAVE(117),UTV(1))        BVEC     26
     *,           (SAVE(127),SR(1)),       (SAVE(137),SRD(1))        BVEC     27
     *,           (SAVE(147),SO(1)),       (SAVE(157),SC1(1))        BVEC     28
     *,           (SAVE(167),SC2(1)),      (SAVE(177),NW(1))         BVEC     29
     *,           (SAVE(187),TLM(1)),      (SAVE(197),NS(1))         BVEC     30
     *,           (SAVE(200),ZTZ(1)),      (SAVE(204),SZ(1))         BVEC     31
     *,           (SAVE(208),TALIGN(1)),   (SAVE(218),NALIGN)        BVEC     32
     *,           (SAVE(229),XNBE(1)),     (SAVE(232),YNBE(1))       BVEC     33
     *,           (SAVE(235),ZNBE(1)),     (SAVE(258),X(1))          BVEC     34
     *,           (SAVE(276),WE(1,1))                                BVEC     35
      DIMENSION REF1(3), REF2(3), REF3(3)                               BVEC     36
      EQUIVALENCE (XNBN(1),REF1(1))                                     BVEC     37
     *,           (YNBN(1),REF2(1))                                     BVEC     38
     *,           (ZNBN(1),REF3(1))                                     BVEC     39
C                                                                       BVEC     40
      EQUIVALENCE (C(1),TW)                                             BVEC     41
C                                                                       BVEC     42
      DIMENSION B(18), DX(6), UR(3), WV(18), BZ(27), WS(27,18)          BVEC     43
```

```
      EQUIVALENCE (BLK(1),B(1))                                          BVEC    45
     *,           (BLK(19),DX(1))                                        BVEC    46
     *,           (BLK(25),UR(1))                                        BVEC
     *,           (BLK(28),VARQ)                                         BVEC    4o
     *,           (BLK(29),DO)                                           BVEC    49
     *,           (BLK(30),NINS)                                         BVEC    50
     *,           (BLK(31),WV(1))                                        BVEC    51
     *,           (BLK(49),BZ(1))                                        BVEC    52
     *,           (BLK(76),WS(1))                                        BVEC    53
C                                                                        BVEC    54
      COMMON/BV/RVAR,RVARMIN,VVAR,VVARMIN,VARAZ,VAREL                    BVEC    55
C                                                                        BVEC    56
```

UBROUTINE BVEC CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
      LOPT=0                                                             BVEC    57
C     CHECK IF RADAR (N=1), VHF (N=2), OR OPTICS (N=3) MARK IS TO DONE   BVEC    58
      GO TO(5,35,65,35) N                                                BVEC    59
C                                                                        BVEC    60
    5 CONTINUE                                                       RR  BVEC    61
C     COMPUTE RELATIVE STATE                                             BVEC    62
      DO 10 I=1,6                                                        BVEC    63
   10 DX(I)=X(I+6)-X(I)                                                  BVEC    64
C     COMPUTE UNIT RANGE VECTOR                                      A   BVEC    65
      CALL UVEC(DX(1),DX(2),DX(3),UR)                                    BVEC    66
C     COMPUTE CURRENT ESTIMATED RANGE RATE                           A   BVEC    67
      PDC=DOT(UR,DX(4)) + X(14)                                      B   BVEC    68
C     COMPUTE CURRENT ESTIMATED RANGE                                C   BVEC    69
      RC=DOT(UR,DX(1)) + X(13)                                       C   BVEC    70
C     COMPUTE RANGE RATE GEOMETRY VECTOR                             D   BVEC    71
      CALL CROSS(DX(4),UR,B(4))                                          BVEC    72
      CALL CROSS(UR,B(4),B(1))                                       D   BVEC    73
C     LOAD B                                                         E   BVEC    74
      DO 15 I=1,3                                                        BVEC    75
      B(I)=-B(I)/RC                                                      BVEC    76
      B(I+3)=-UR(I)                                                      BVEC    77
      B(I+6)=-B(I)                                                       BVEC    78
      B(I+9)=UR(I)                                                       BVEC    79
      B(I+12)=0.                                                         BVEC    80
```

```
      B(14)=1.                                                          E
C     DEFINE VARQ AND DQ                                                F
      DQ = QQ(2) - RDC
      VARQ=AMAX1(ABS(RDC*VVAR),VVARMIN)**2
      NINS=2
C     CALCULATE AND INCORPORATE UPDATE
      CALL FILTER                                                       F
 35   CONTINUE                                                          R
C     COMPUTE RELATIVE STATE FOR VHF MEASUREMENT
      DO 40 I=1,6
 40   DX(I)=X(I+6)-X(I)
C     COMPUTE CURRENT UNIT RANGE VECTOR                                 A
      CALL UVEC(DX(1),DX(2),DX(3),UR)
C     COMPUTE CURRENT RANGE ESTIMATE                                    A
      RC=DOT(UR,DX(1)) + X(13)                                          B
C     LOAD B                                                            C
      DO 45 I=1,3
      B(I)=-UR(I)
      B(I+3)=0.
      B(I+6)=UR(I)
      B(I+9)=0.
      B(I+12)=0.
 45   B(I+15)=0.
      B(13)=1.                                                          C
C     DEFINE VARQ, DQ                                                   D
      DQ = QQ(1) - RC
      VARQ=AMAX1(RC*RVAR,RVARMIN)**2
      NINS=1
C     CALCULATE AND INCORPORATE MARK
      CALL FILTER                                                       D
```

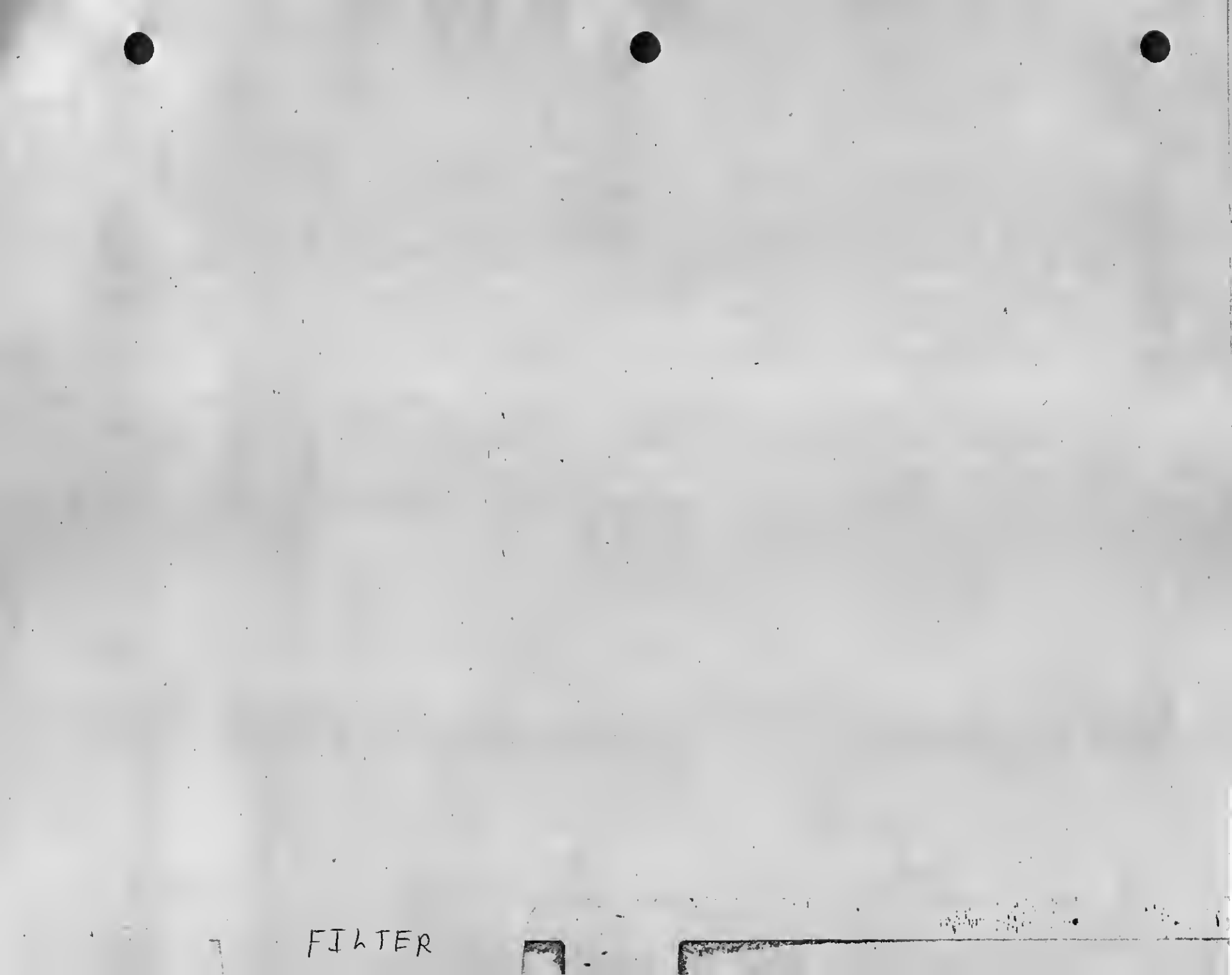
```
BVEC    82
BVEC    83
BVEC
BVEC    85
BVEC    86
BVEC    87
BVEC    88
BVEC    89
BVEC    90
BVEC    91
BVEC    92
BVEC    93
BVEC    94
BVEC    95
BVEC    96
BVEC    97
BVEC    98
BVEC    99
BVEC   100
BVEC   101
BVEC   102
BVEC   103
BVEC   104
BVEC   105
BVEC   106
BVEC   107
BVEC   108
BVEC   109
BVEC   110
BVEC   111
```

SUBROUTINE BVEC CDC 6600 FTN V3.0-P308 OPT=1 08/29/72 11.32.27.

```
C     CHECK IF THIS WAS A VHF MARK OR SECOND PART OF RADAR MARK          E
      IF(N.EQ.2) GO TO 100                                              E
 65   CONTINUE                                                          OP
C     COMPUTE RELATIVE STATE FOR OPTICS MARK
      DO 70 I=1,6
 70   DX(I)=X(I+6) - X(I)
```

```
BVEC   112
BVEC   113
BVEC   114
BVEC   116
BVEC   117
BVEC   118
```

```
      CALL UVEC(DX(1),DX(2),DX(3),UR)                                    BVEC  120
C     COMPUTE CURRENT AZIMUTH ESTIMATE                              A    BVEC  121
      AZC=ATAN(DOT(UR,REF3)/DOT(UR,REF2)) + X(15)                        BVEC  122
      COMPUTE CURRENT ESTIMATEQ ELEVATION                                BVEC  123
      ELC=ASIN(DOT(UR,REF1)) + X(16)                                A    BVEC  124
C     CHECK IF ELEVATION COMPONENT IS CURRENTLY BEING PROCESSED          BVEC  125
      IF(LOPT.EQ.1) GO TO 80                                        B    BVEC  126
C     COMPUTE AZIMUTH GEOMETRY VECTOR                               C    BVEC  127
      CALL UCROSS(REF1,UR,B(1))                                     C    BVEC  128
C     DEFINE VARQ, QQ AND PROJECTION TERM                           D    BVEC  129
      QQ = GQ(3) - AZC                                                   BVEC  130
      VARQ=VARAZ**2                                                 D    BVEC  131
      RCEL=DOT(UR,UX(1))*COS(ELC)                                   E    BVEC  132
C     LOAD B AND INCORPORATE MARK                                        BVEC  133
      GO TO 90                                                           BVEC  134
 80   CONTINUE                                                      F    BVEC  135
C     DEFINE ELEVATION GEOMETRY VECTOR                                   BVEC  136
      CALL UCROSS(REF1,UR,B(4))                                          BVEC  137
      CALL UCROSS(UR,B(4),B(1))                                     F    BVEC  138
C     DEFINE VARQ, QQ AND PROJECTION TERM                           G    BVEC  139
      QQ = GQ(4) - ELC                                                   BVEC  140
      VARQ=VAPEL**2                                                 G    BVEC  141
      RCEL=DOT(UR,UX(1))                                            H    BVEC  142
 90   CONTINUE                                                      I    BVEC  143
C     LOAD B                                                             BVEC  144
      DO 95 I=1,3                                                        BVEC  145
      B(I)=-B(I)/RCEL                                                    BVEC  146
      B(I+3)=0                                                           BVEC  147
      B(I+6)=-B(I)                                                       BVEC  148
      B(I+9)=0.                                                          BVEC  149
      B(I+12)=0.                                                         BVEC  150
 95   B(I+15)=0.                                                    I    BVEC  151
      IF(LOPT.EQ.1) B(16)=1.                                        J    BVEC  152
      IF(LOPT.EQ.0) B(15)=1.                                        J    BVEC  153
      IF(LOPT.EQ.1) NINS=4                                          K    BVEC  154
      IF(LOPT.EQ.0) NINS=3                                          K    BVEC  155
      CALL FILTER                                                        BVEC  156
C     CHECK IF THIS WAS THE AZIMUTH COMPONENT OF THE OPTICS MARK         BVEC  157
      IF(LOPT.EQ.1) GO TO 100                                       L    BVEC  158
      LOPT=1                                                             BVEC  159
      GO TO 65                                                           BVEC  160
100   CONTINUE                                                      L    BVEC  161
      RETURN                                                             BVEC  162
      END                                                                BVEC  163
```


FILTER

ROUTINE FILTER

COMPUTES OPTIMAL WEIGHTING VECTOR FOR MEASUREMENT, UPDATES STATE AND COVARIANCE. FOR DISCUSSION, SEE REFERENCE 1.

WE = COVARIANCE SQUARE-ROOT MATRIX

$WS = WE^T$

$\underline{BZ} = WS \times \underline{B}$

$A = VARQ + \underline{BZ}^T \underline{BZ}$ — TOTAL A PRIORI UNCERTAINTY IN MEASUREMENT.

$ZTZ = \underline{BZ}^T \underline{BZ}$ — TOTAL A PRIORI UNCERTAINTY IN MEASUREMENT DUE TO STATE UNCERTAINTY.

$SZ = [VARQ/A]^{\frac{1}{2}}$ — CONFIDENCE LEVEL OF A PRIORI ESTIMATE OF MEASUREMENT.

$\underline{WV} = WE \times \underline{BZ}$ — WEIGHTING VECTOR.

$\underline{X} = \underline{X} + (DQ/A)\underline{WV}$ — STATE UPDATE EQUATION.

$WE = WE - (1/VARB)\underline{WV}\ \underline{BZ}^T$ — COVARIANCE UPDATE EQUATION.

$VARB = A(1. + SZ)$

AREA A — CALCULATE TRANSPOSE OF W.
CALCULATE Z-VECTOR.

AREA B — CALCULATE A.

AREA C — CALCULATE ZTZ FOR THIS MARK COMPONENT (FOR POPOUT)

AREA D — CONVERT ZTZ TO MR IF ANGLE MARK.
CALCULATE SZ FOR POPOUT.

AREA E — CALCULATE WEIGHTING VECTOR.

AREA F — CALCULATE COVARIANCE UPDATE FACTOR.

AREA G — UPDATE STATE AND COVARIANCE.

```
      SUBROUTINE FILTER                                                  FILTER     2
C                                                                        FILTER     3
C     SUBROUTINE TO PROCESS A NAVIGATION MARK                            FILTER     4
C                                                                        FILTER     5
      COMMON VAR                                                         FILTER     6
      DIMENSION VAR(5600), P(5000), SAVE(950), BLK(700)                  FILTER     7
     *,         NTEGER(100)                                             FILTER     8
      EQUIVALENCE (VAR(601),P(1))                                        FILTER     9
     *,          (VAR(401),NTEGER(1))                                   FILTER    10
      EQUIVALENCE (P(350),SAVE(1))                                       FILTER    11
     *,          (P(1300),BLK(1))                                       FILTER    12
      DIMENSION ZTZ(4), SZ(4), X(18), WE(18,27)                          FILTER    13
      EQUIVALENCE (SAVE(200),ZTZ(1))                                     FILTER    14
     *,          (SAVE(204),SZ(1))                                      FILTER    15
     *,          (SAVE(258),X(1))                                       FILTER    16
     *,          (SAVE(276),WE(1,1))                                    FILTER    17
     *,          (NTEGER(9),NCOL)                                       FILTER    18
C                                                                        FILTER    19
C                                                                        FILTER    20
      DIMENSION B(18), DX(6), UR(3), WV(18), BZ(27), WS(27,18)           FILTER    21
     *,         EBAR(18)                                                FILTER    22
C                                                                        FILTER    23
      EQUIVALENCE (BLK(1),B(1))                                          FILTER    24
     *,          (BLK(19),DX(1))                                        FILTER    25
     *,          (BLK(25),UR(1))                                        FILTER    26
     *,          (BLK(28),VARQ)                                         FILTER    27
     *,          (BLK(29),DQ)                                           FILTER    28
     *,          (BLK(30),NINS)                                         FILTER    29
     *,          (BLK(31),WV(1))                                        FILTER    30
     *,          (BLK(49),BZ(1))                                        FILTER    31
     *,          (BLK(76),WS(1))                                        FILTER    32
     *,          (BLK(562),EBAR(1))                                     FILTER    33
C                                                                        FILTER    34
C     COMPUTE Z-VECTOR                                                   FILTER    35
      CALL MATRAN(WE,18,27,WS)                                       A   FILTER    36
      CALL MATMUL(WS,B,BZ,27,18,1)                                   A   FILTER    37
      A=VARQ                                                         B   FILTER    38
      DO 5 I=1,27                                                        FILTER    39
    5 A=A + BZ(I)*BZ(I)                                              B   FILTER    40
C     COMPUTE STATE UNCERTAINTY OF OBSERVABLE AND REDUCTION FACTOR       FILTER    41
      ZTZ(NINS)=SQRT(A-VARQ)                                         C   FILTER    42
      IF(NINS.GT.2) ZTZ(NINS)=ZTZ(NINS)*1000.                        D   FILTER    43
      SZ(NINS)=SQRT(VARQ/A)                                          D
```

```
      CALL MATMUL(WE,DZ,WV,18,27,1)                                    E
      VARB=A*(1. + SQRT(VARQ/A))                                       F
      DO 10 I=1,18                                                     G
      X(I)=X(I) + WV(I)*DQ/A
      DO 10 J=1,27
10    WE(I,J)=WE(I,J) - WV(I)*BZ(J)/VARB                               G
      RETURN
      END
```

```
FILTER    45
FILTER    46
FILTER    47
FILTER    48
FILTER    49
FILTER    50
FILTER    51
FILTER    52
```

OPEN1 -OPENS

6.0 REFERENCES

1. EVERYTHING YOU ALWAYS WANTED TO KNOW... (ENCLOSURE)
2. BATTIN, R. H., ASTRONAUTICAL GUIDANCE, NEW YORK, 1964

EVERYTHING YOU ALWAYS WANTED TO KNOW ABOUT KALMAN FILTERING
(AREN'T YOU GLAD YOU ASKED)

I. Fundamental definitions and important properties of vectors and matrices

A matrix A is said to be an nxm array when it has n rows and m columns.

An nxl matrix is called a (column) vector

A lxn matrix is called a (row) vector

If n = m, A is square.

a_{ij} is a representative element of A from the i^{th} row and j^{th} column

Define $\underline{b}$ an arbritrary nxl (vector)
A an arbritrary nxn (square)
B an arbritrary nxm (rectangular)

Further $\underline{0} = \begin{bmatrix} 0 \\ 0 \\ - \\ - \\ - \\ 0 \end{bmatrix}$ (nxl) $\qquad I = \begin{bmatrix} 1 & 0 & 0 & - & - & 0 \\ 0 & 1 & 0 & - & - & 0 \\ 0 & 0 & 1 & - & - & 0 \\ - & & & - & & - \\ - & & & & - & - \\ 0 & 0 & 0 & - & - & 1 \end{bmatrix}$ (nxn)

$A\underline{0} = \underline{0} \qquad AI = A$

For $A = [a_{ij}]$, $A^T = [a_{ji}]$ transpose of A

A^{-1} is the matrix, if it exists, which has the property that

$A^{-1}A = AA^{-1} = I$, inverse of A

Properties of Matrices

If A is a square matrix, and if for all $\underline{b} \neq \underline{0}$,

$\underline{b}^T A \underline{b} > 0$ A is positive definite (p.d.)

$\underline{b}^T A \underline{b} < 0$ A is negative definite (n.d.)

If, for any $\underline{b} \neq \underline{0}$,

$\underline{b}^T A \underline{b} = 0$ A is indefinite

If $A = A^T$ A is symmetric

$A^T = A^{-1}$ A is orthogonal

If A, B are square

$$(AB)^T = B^T A^T$$
$$(AB)^{-1} = B^{-1} A^{-1}$$

Properties of Vectors

If for a set $\{\underline{b}_i\}$ of nxl vectors

$$\sum_i c_i \underline{b}_i = \underline{0} \Rightarrow c_i = 0$$

The $\underline{b}_i$ are said to be linearly independent. If there are n members of the set $\{\underline{b}_i\}$, the set is complete with respect to the space of nxl vectors. If the set $\{\underline{b}_i\}$ is independent and complete, any arbitrary nxl vector, $\underline{e}$, may be expressed as

$$\underline{e} = \sum_i c_i \underline{b}_i$$

Given two vectors, $\underline{b} \neq \underline{0}$, $\underline{c} \neq \underline{0}$, if

$\underline{b}^T \underline{c} = 0$ $\underline{b}$ and $\underline{c}$ are orthogonal

If

$\underline{b}^T \underline{b} = 1$ $\underline{b}$ is normal

Eigenvalues, eigenvectors, functions of a matrix

If there exists a set of vectors $\underline{L}_i$ and corresponding numbers λ_i such that

$$A\underline{L}_i = \lambda_i \underline{L}_i$$

$\underline{L}_i$ is said to be an eigenvector of A corresponding to eigenvalve λ_i.

If A is definite ($\underline{b}'A\underline{b} \neq 0$ for all $\underline{b} \neq \underline{0}$) it has no $\lambda_i = 0$:

Suppose for some $\underline{b}_i$, $\lambda_i = 0$

$$\underline{b}_i^T A \underline{b}_i = \lambda_i \underline{b}_i^T \underline{b}_i$$
$$= 0$$

contrary to the assumption.

In addition, if A is definite, its eigenvectors are a complete linearly independent set. In this case, A may be represented as

$$A = P \Lambda P^{-1}$$

$$P = \left[\underline{L}_1 \; \underline{L}_2 \; - \; - \; \underline{L}_n \right]$$

square matrix of eigenvectors

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & - & - & 0 \\ 0 & \lambda_2 & - & - & 0 \\ - & & - & & 1 \\ - & & & - & - \\ 0 & 0 & - & - & \lambda_n \end{bmatrix}$$

diagonal matrix of eigenvalues

Furthermore, the function

$$f(A) = \sum_i c_i A^i$$

exists and converges to

$$f(A) = P \begin{bmatrix} \sum c_i \lambda_1^{\,i} & 0 & - & - & 0 \\ 0 & \sum c_i \lambda_2^{\,i} & - & - & 0 \\ - & - & & & - \\ - & - & - & & - \\ 0 & - & 0 & - & \sum c_i \lambda_n^{\,i} \end{bmatrix} P^{-1}$$

$$= P \, f(\Lambda) \, P^{-1}$$

This may be demonstrated by noting that

$$AP = \left[\lambda_1 \underline{L}_i \quad \lambda_2 \underline{L}_2 \; - \; - \; \lambda_n \underline{L}_n \right] = P\Lambda$$

$$A^i P = \left[\lambda_1^i \; \underline{L}_1 \lambda_2^i \; \underline{L}_2 \; - \; - \; \lambda_n^i \; \underline{L}_n \right] = P\Lambda^i$$

$$A^i = P\Lambda^i P^{-1}$$

The normalized eigenvectors of a symmetric matrix are orthonormal. Let $\{\underline{e}_i\}$ be the normalized e. v. of E, a symmetric matrix

$$\underline{e}_j^T E \underline{e}_i = \lambda_i \underline{e}_j^T \underline{e}_i$$

$$e_j^T E \underline{e}_i = (E^T \underline{e}_j)^T \underline{e}_i$$

$$= (E \underline{e}_j)^T \underline{e}_i$$

$$= \lambda_j \underline{e}_j^T \underline{e}_i$$

$$e_j^T E \underline{e}_i - \underline{e}_j^T E \underline{e}_i = 0 = (\lambda_i - \lambda_j) \underline{e}_j^T \underline{e}_i$$

If $\lambda_i \neq \lambda_j$ it is seen that $\underline{e}_j^T \underline{e}_i = 0$. If $\lambda_i = \lambda_j$, one may construct a new vector, $\underline{e}_j^*$ which has an eigenvalue λ_i and is orthogonal to $\underline{e}_i$:

$$\underline{e}_j^* = \quad (\underline{e}_j - (\underline{e}_i^T \underline{e}_j) \underline{e}_i$$

$$\underline{e}_i^T e_j^* = \underline{e}_i^T \underline{e}_j - (\underline{e}_i^T \underline{e}_j) \underline{e}_i^T \underline{e}_i$$

$$= \underline{e}_i^T \underline{e}_j - (\underline{e}_i^T \underline{e}_j) = 0$$

Since $\underline{e}_i^T \underline{e}_i = 1$ (normalized). This may be done for every repeated eigenvalue until a complete orthonormal set is constructed.

II. Conventional usages in vector and matrix calculus

Differentiation with respect to a scalar:

$$\frac{d}{dt}A = \dot{A} = \left[\frac{d}{dt}a_{ij}\right]$$

Differentiation with respect to a vector:

$$\underline{X} = \{X_i\} \qquad \text{nx1 vector}$$

$$\Phi = \Phi(\underline{X}) \qquad \text{a scalar}$$

For a scalar

$$\frac{\partial \Phi}{\partial \underline{X}} = \left[\frac{\partial \Phi}{\partial X_i}\right] = \begin{bmatrix} \frac{\partial \Phi}{\partial X_1} \\ \frac{\partial \Phi}{\partial X_2} \\ - \\ - \\ \frac{\partial \Phi}{\partial X_n} \end{bmatrix}$$

For a vector $\underline{R} = \{R_i\}$

$$\frac{\partial \underline{R}}{\partial \underline{X}} = \begin{bmatrix} \frac{\partial R_1}{\partial X_1} & \frac{\partial R_2}{\partial X_1} & - & - & - & \frac{\partial R_n}{\partial X_1} \\ \frac{\partial R_1}{\partial X_2} & \frac{\partial R_2}{\partial X_2} & - & - & - & \frac{\partial R_n}{\partial X_2} \\ - & - & - & - & - & - \\ - & - & - & - & - & - \\ \frac{\partial R_1}{\partial X_n} & \frac{\partial R_2}{\partial X_n} & & & & \frac{\partial R_n}{\partial X_n} \end{bmatrix}^T$$

$$= \begin{bmatrix} \frac{\partial R_1}{\partial X_1} & \frac{\partial R_1}{\partial X_2} & - & - & - & \frac{\partial R_1}{\partial X_n} \\ \frac{\partial R_2}{\partial X_1} & \frac{\partial R_2}{\partial X_2} & - & - & - & \frac{\partial R_2}{\partial X_n} \\ - & - & & & & \\ - & - & & & & \\ - & - & & & & \\ \frac{\partial R_n}{\partial X_1} & \frac{\partial R_n}{\partial X_2} & - & - & - & \frac{\partial R_n}{\partial X_n} \end{bmatrix}$$

$$= \left\{ \frac{\partial}{\partial \underline{X}} (\underline{R}^T) \right\}^T$$

Integration of a matrix:

$$\int A dt = \left[\int a_{ij} dt \right]$$

Differentials:

$$\delta ( ) = \frac{\partial ( )}{\partial \underline{X}} \delta \underline{X}$$

Examples:

$$\delta\,\Phi = \frac{\partial \Phi}{\partial \underline{X}}\,\delta \underline{X}$$

$$\delta R = \frac{\partial \underline{R}}{\partial \underline{X}}\,\delta \underline{X} = \left\{ \frac{\partial}{\partial \underline{X}}\,\underline{R}^{T} \right\}^{T} \delta \underline{X}$$

III. Eyeball statistics

Let X be a discrete random variable with sample described by the distribution $\{X_i\}$

$$\text{Average of } X = \overline{X} = \frac{1}{N} \sum_i X_i$$

$$\text{Variance of } X = \text{var}\,(X) = \sigma_X^{\,2} = \frac{1}{N} \sum_i (X_i - \overline{X})^2 = \overline{(X_i - \overline{X})^2}$$

$$\text{Standard deviation of } X = \sigma X = \left[\text{var}(X) \right]^{1/2}$$

If X is a continuous variable, the distribution is characterized by $f(\xi)$, the probability per unit interval that $X = \xi$. In this case,

$$\overline{X} = \int_{-\infty}^{\infty} \xi f(\xi) d\xi$$

$$\text{Var}(X) = \int_{-\infty}^{\infty} (\xi - \overline{X})^2 f(\xi)\, d\xi$$

For the vector $\underline{X}$, a joint distribution function is defined $f(\underline{\xi})$

$$\overline{\underline{X}} = \int_v \underline{\xi}\, f(\underline{\xi})\, d\xi_1 d\xi_2 - - d\xi_n$$

$$\text{COV}(\underline{X}) = \overline{\left[(\underline{X}-\overline{\underline{X}})(\underline{X}-\overline{\underline{X}})^T\right]} \text{ covariance matrix of } X$$

$$= \int_v (\underline{\xi}-\overline{\underline{X}})(\underline{\xi}-\overline{\underline{X}})^T f(\underline{\xi})\, d\xi_1\, d\xi_2 - - d\xi_n$$

$$= \begin{bmatrix} \sigma_{x_1}^2 & \sigma_{x_1x_2} & - & - & - & \sigma_{x_1x_n} \\ \sigma_{x_2x_1} & \sigma_{x_2}^2 & - & - & - & \sigma_{x_2x_n} \\ - & - & & & & \\ - & - & & & & \\ - & - & & & & \\ \sigma_{x_nx_1} & \sigma_{x_nx_2} & - & - & - & \sigma_{x_n}^2 \end{bmatrix}$$

where $\sigma_{x_ix_j}$ = covariance of x_i, x_j

$$= \int_v (\xi_i-\bar{x}_i)(\xi_j-\bar{x}_j)\, f(\underline{\xi})\, d\xi_1 d\xi_2 - - d\xi_n$$

If $\sigma_{x_ix_j} = 0$, x_i and x_j are said to be linearly independent in the statistical sense. It does not imply general functional independence.

The function

$$\rho_{ij} = \frac{\sigma_{x_ix_2}}{\sqrt{\sigma_{x_i}^2}\ \sqrt{\sigma_{x_j}^2}}$$

is called the correlation coefficient of x_i to x_j.

For physical systems, covariance matrices are virtually always positive definite.

Let $\underline{e} = \underline{X} - \overline{\underline{X}}$ and

$$E = \overline{\underline{e}\underline{e}^T} = \mathrm{COV}(X)$$

E is symmetric:

$$E^T = \overline{\underline{e}\underline{e}}^T = \overline{(\underline{e}\underline{e}^T)}^T = \overline{\underline{e}\underline{e}^T} = E$$

The matrix of normalized eigenvectors of E is orthogonal, i.e. $P^T = P^{-1}$:

Since E is symmetric, its eigenvectors are orthogonal.

Some properties of determinants thrown in for good measure.

$|A| \neq 0$ if $\underline{b}^T A \underline{b} \neq 0$ for all $\underline{b} \neq \underline{0}$

$$|A^T| = |A|$$

$$|A^{-1}| = |A|^{-1}$$

If $|AB| = |A||B|$

If P is orthogonal, $|P| = \pm 1$

$$1 = |I| = |P^T P| = |P^T||P| = |P|^2$$

$$|P| = \pm 1$$

The determinant of a matrix is the product of its eigenvalues:

$$A = P \Lambda P^{-1}$$

$$|A| = |P \Lambda P^{-1}|$$

$$= |P||\Lambda||P^{-1}|$$

$$= |P||P|^{-1}|\Lambda|$$

$$= |\Lambda|$$

$$= \lambda_1 \lambda_2 - - - \lambda_n$$

IV. Statistical State Estimation

Some Philosophical Remarks on the Nature of the Problem

It is worth noting that from a physical measurement point of view, such quantities as the position and velocity of a particle are not available to direct observation. They are abstract mathematical constructs, evolving according to an empirically developed law and should not be confused with "real" aspects of the system. They are useful only in so far as they serve to predict the observables of the system, i.e. scalar quantities such as distance, angular measure and speed; the only aspects of the system to which the measurement process has direct access.

Mechanics considers a system "noise free" if it is possible to argue that, in principle, all that is needed for a perfect state determination is a perfect sensor. In addition, the assumption is usually made in solving particular problems that the "equations of motion" are perfectly known. In most engineering problems, neither of these assumptions is true. Even were they true, a further practical difficulty arises in the attempt to find the "state" variables as a function of the observables, as the observables are generally transcendental functions of the state variables and time, and therefore not amenable to algebraic solution.

The problem in state estimation is therefore two-fold:

1. Circumvent the mathematical difficulties associated with determination of the system state in terms of the observables, and (2) do so in a manner with reflects the imperfect character of the measurements and knowledge of the system law of motion. In order to do this, the attitude will be adopted that the measuring device is an inseparable part of the system in the dynamic sense: its properties effect the determination of the system state.

The system under study will be taken as two particles in orbit around a planet, between which measurements of range, range rate, and relative direction may be made with an appropriate sensor. Any biases which may effect a sensor measurement are classified as state variables. Otherwise, all noise on the system (state as well as sensor) will be considered as random with zero mean. No assumption about its statistical distribution will be made (Fig. I). Also, for purposes of specific treatment, a measurement coordinate frame is depicted in Fig. II:

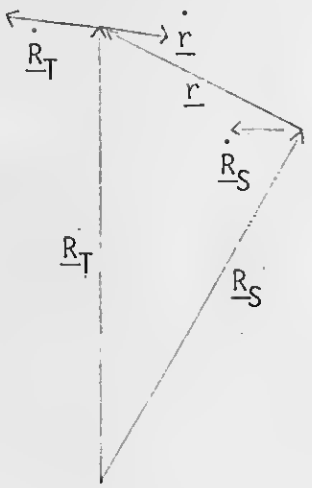


FIGURE I

$\underline{R}_S$ = spacecraft position vector

$\dot{\underline{R}}_S$ = spacecraft velocity vector

$\underline{R}_T$ = target position vector

$\dot{\underline{R}}_T$ = target velocity vector

$\underline{r}$ = relative position vector

$\dot{\underline{r}}$ = relative velocity vector

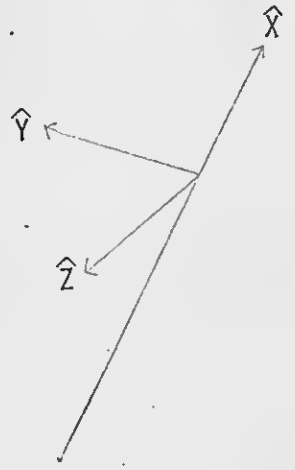


FIGURE II

$\hat{X}$ = unit $(\underline{R}_S)$

$\hat{Y}$ = unit $\underline{R}_S \times (\underline{R}_S \times \dot{\underline{R}}_S)$

$\hat{Z}$ = unit $(\underline{R}_S \times \dot{\underline{R}}_S)$

Total state of system

$$X = \begin{bmatrix} \underline{R}_S \\ \dot{\underline{R}}_S \\ \underline{R}_T \\ \dot{\underline{R}}_T \\ \underline{\beta} \end{bmatrix}$$

$\underline{\beta}$ estimated constants

In order to solve for the state in terms of observable quantities, a method of differential corrections will be devised. Consider an observable of the system, Q

$$Q = Q\,(\underline{X})$$

and its expansion about a reference state, $\underline{X}_R$:

$$Q(\underline{X}) = Q(\underline{X}_R) + \frac{\partial Q}{\partial \underline{X}}\bigg|^T_{\underline{X}_R}(\underline{X}-\underline{X}_R) + (\underline{X}-\underline{X}_R)^T\left\{\frac{\partial}{\partial \underline{X}}\left(\frac{\partial Q}{\partial \underline{X}}\right)^T\right\}^T_{\underline{X}_R}(\underline{X}-\underline{X}_R) + \cdots$$

assume that $\underline{X}_R$ is chosen sufficiently close to $\underline{X}$ that derivative terms higher than first are neglible:

$$Q(\underline{X}) = Q(\underline{X}_R) + \frac{\partial Q}{\partial \underline{X}}\bigg|^T_{\underline{X}_R}(\underline{X}-\underline{X}_R)$$

defining $\delta Q = Q(\underline{X}) - Q(\underline{X}_R)$, $\delta \underline{X} = (\underline{X}-\underline{X}_R)$

4.1 $$\delta Q = \frac{\partial Q}{\partial \underline{X}}\bigg|^T_{\underline{X}_R}\delta \underline{X}$$

Conventional usage will now be followed and such variations will be considered as truncated expansions, whenever the notation $\delta(\ .\ )$ is used. The quantity

$$\frac{\partial Q}{\partial \underline{X}}\bigg|_{\underline{X}_R}$$

is called the geometry vector or mapping vector and will henceforth be denoted

$$\underline{b}_Q = \frac{\partial Q}{\partial \underline{X}}\bigg|_{\underline{X}_R} \qquad \underline{b}_Q \quad (n\times 1)$$

so that 4.1 becomes

4.2 $$\delta Q = \underline{b}_Q^T\,\delta \underline{X}$$

Equation 4.2 is the desired linear relationship between the observable Q and the state $\underline{X}$. As there are n elements of $\delta\underline{X}$ to determine, n independent measurements, δQ, will be required. Since these cannot normally be simultaneously obtained, it will be necessary to investigate the time history of $\delta\underline{X}$. For gravitational field, the derivative, $\dot{\underline{X}}$ is a vector function of $\underline{X}$:

4.3 $$\frac{d}{dt}\underline{X} = \underline{f}(\underline{X}) \qquad \underline{f}_{nx1}$$

for example, in the Keplerian case with $\underline{X}$ as previously defined

4.4 $$\frac{d}{dt}\underline{X} = \frac{d}{dt}\begin{bmatrix} \underline{R}_S \\ \dot{\underline{R}}_S \\ \underline{R}_T \\ \dot{\underline{R}}_T \\ \underline{\beta} \end{bmatrix} = \begin{bmatrix} \emptyset & I & \emptyset & \emptyset & \emptyset \\ -\frac{\mu}{R_S^3}I & \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & I & \emptyset \\ \emptyset & \emptyset & -\frac{\mu}{R_T^3}I & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \end{bmatrix} \begin{bmatrix} \underline{R}_S \\ \dot{\underline{R}}_S \\ \underline{R}_T \\ \dot{\underline{R}}_T \\ \underline{\beta} \end{bmatrix} \qquad \emptyset = [0]_{3x3} \quad I = I_{3x3}$$

If $\underline{X}$ and its derivatives with respect to state elements and time are continuous, the operations $\frac{d}{dt}$ and $\delta(\ )$ are interchangeable, i.e.

$$\frac{d}{dt}\delta(\ ) = \delta\frac{d}{dt}(\ )$$

Taking the variation of 4.3

4.5a $$\delta\left(\frac{d}{dt}\underline{X}\right) = \frac{d}{dt}\delta\underline{X} = \delta(\underline{f}(\underline{X})) = \left\{\frac{\partial}{\partial\underline{X}}\underline{f}^T\right\}\delta\underline{X} = \underline{F}\delta\underline{X}$$

For the Keplerian field of 4.4

$$F = \left\{\frac{\partial}{\partial \underline{X}}\underline{f}^T\right\}^T = \begin{bmatrix} \emptyset & I & \emptyset & \emptyset & \emptyset \\ [G]_S & \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & I & \emptyset \\ \emptyset & \emptyset & [G]_T & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset & \emptyset \end{bmatrix}_{nxn}$$

$$[G]_S = \frac{\mu}{\underline{R}_S^5}\left(3\underline{R}_S\underline{R}_S^T - R_S^2 I\right)$$

$$[G]_T = \frac{\mu}{\underline{R}_T^5}\left(3\underline{R}_T\underline{R}_T^T - R_T^2 I\right)$$

Therefore

4.5b $$\dot{\delta \underline{X}} = F\delta\underline{X}$$

This equation may be integrated numerically. A more useful solution takes the form of

4.6 $$\delta\underline{X}(t) = \Phi(t,t_o)\,\delta\underline{X}(t_o) \qquad \Phi_{nxn}$$

Φ is called the state transition matrix. Differentiating 4.6 and substituting for $\delta\underline{X}$ and $\dot{\delta\underline{X}}$ in 4.5b:

$$\dot{\delta\underline{X}} = \dot{\Phi}\delta\underline{X}(t_o) = F\,\Phi\,\delta\underline{X}_o$$

By inspection then

4.7 $$\dot{\Phi} = F\Phi, \text{ subject to } \Phi(t_o,t_o) = I$$

For a noise-free system, the problem is now solved. A deviation at t can be referenced to any other time. Suppose a set of n measurements has been made times t_1, t_2 - - t_n.

$$\delta Q_1 = \underline{b}_Q^T(t_1)\ \delta \underline{X}(t_1) = \underline{b}_Q^T(t_1)\,\Phi\,(t_1,t_n)\,\delta \underline{X}_n$$

$$\delta Q_2 = \underline{b}_Q^T(t_2)\ \delta \underline{X}(t_2) = \underline{b}_Q^T(t_2)\,\Phi\,(t_2,t_n)\,\delta \underline{X}_n$$

$$\vdots \qquad\qquad \vdots$$

4.8a

$$\delta Q_n = \underline{b}_Q^T(t_n)\,\delta \underline{X}(t_n) = \underline{b}_Q^T(t_n)\,\Phi\,(t_n,t_n)\ \delta \underline{X}_n$$

Let $\underline{h}^T_i = \underline{b}_Q^T\ \Phi(t_i,t_n)$, then

4.8b

$$\begin{bmatrix} \delta Q_1 \\ \delta Q_2 \\ \vdots \\ \delta Q_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} \underline{h}_1^T \\ \underline{h}_2^T \\ \vdots \\ \underline{h}_n^T \end{bmatrix}_{n\times n} \delta \underline{X}_n(t_n)$$

4.9a

$$\delta \underline{X}(t_n) = \begin{bmatrix} \underline{h}_1^T \\ \underline{h}_2^T \\ \vdots \\ \underline{h}_n^T \end{bmatrix}^{-1} \begin{bmatrix} \delta Q_1 \\ \delta Q_2 \\ \vdots \\ \delta Q_n \end{bmatrix}$$

If more than n measurements are made, say m (>n), the solution has the form

$$\delta \underline{X}(t_m) = \left\{ \left[\underline{h}_i^T\right]^T \left[\underline{h}_i^T\right] \right\}^{-1} \left[\underline{h}_i^T\right]^T \left[\delta Q_i\right]$$

4.9b

$$= \left\{ \begin{bmatrix} \underline{h}_2^T \\ - \\ - \\ \underline{h}_m^T \end{bmatrix} \begin{bmatrix} \underline{h}_2^T \\ - \\ - \\ \underline{h}_m^T \end{bmatrix} \right\}^{-1} \begin{bmatrix} \underline{h}_2^T \\ - \\ - \\ \underline{h}_m^T \end{bmatrix} \begin{bmatrix} \delta Q_2 \\ - \\ - \\ \delta Q_m \end{bmatrix}$$

Equation 4.9a is called the deterministic solution of $\delta \underline{X}(t_n)$, 4.9b is the unweighted least squares solution of $\delta \underline{X}(t_n)$. If statistical information on the relative quality of various measurements is available, a more virtuous estimate might be obtained by multiplying each side of 4.8 by the relative weights; customarily used is the covariance of measurement noise for the m measurements, C :

4.9c $$\delta \underline{X}(t_n) = \left[H^T[C]H\right]^{-1} H^T[C] \overrightarrow{\delta Q} \qquad H = \begin{bmatrix} \underline{h}_1^T \\ \underline{h}_2^T \\ - \\ - \\ \underline{h}_n^T \end{bmatrix}$$

This is called the weighted least squares estimate of $\delta X(t_n)$. An estimator of the type 4.9c is also called a batch estimator. Normally, a set of data $\overrightarrow{\delta Q}$ will be processed several times, each time using the estimate, $\delta \underline{X}$, to compute a new $\underline{X}_R^*$:

$$\underline{X}_R^* = \underline{X}_R + \delta \underline{X}$$

whereupon new values of $Q(\underline{X}_R^*)$ and hence $\delta Q(t_i)$, $\underline{b}_Q(t_i)$ and $\Phi(t_i,t_n)$ are computed and used in the next pass. $\delta \underline{X}$ converges to near zero and the assumption that higher order derivatives are negligible is accurately fulfilled. A number of defects with the technique make it unsuitable for

many applications: all data from a set of measurements must be stored, as well as the H matrix, which may become quite large; a large matrix inversion is required which is slow and sometimes numerically difficult or inaccurate; in most cases, several "passes" are needed for complete convergence. For these reasons, a technique of sequential estimation has been developed, which avoids these problems, as will be seen. First, an investigation into the propagation of state errors is required.

Let $\underline{X}_E$ be an estimate of X. Then

$$\delta \underline{X}_E = \underline{X}_E - \underline{X}_R$$

Since it has been found that

$$\delta \underline{X}(t) = \Phi \, \delta \underline{X}(t_o)$$

then

$$\delta \underline{X}_E(t) = \Phi \, \delta \underline{X}_E(t_o)$$

and

$$\underline{e}(t) = \delta \underline{X}_E(t) - \delta \underline{X}(t) = \Phi \left[\delta \underline{X}_E(t_o) - \delta \underline{X}(t_o) \right]$$
$$= \Phi \, \underline{e}_o$$

Furthermore

$$4.10 \qquad E(t) = \overline{\underline{e}(t)\underline{e}(t)^T} = \overline{\Phi \underline{e}(t_o)\underline{e}(t_o)^T \Phi^T} = \Phi E(t_o) \, \Phi^T$$

Demonstrating the evolution of the covariance of state errors. An nxn symmetric matrix may be expressed as the sum of n linearly independent forms of the kind

$$E = \frac{1}{n}\left[\underline{e}_1\underline{e}_1^T + \underline{e}_2\underline{e}_2^T = - - + \underline{e}_n\underline{e}_n^T\right] = \frac{1}{\sqrt{n}}\left[\underline{e}_1\ \underline{e}_2 - - -\ \underline{e}_n\right]\begin{bmatrix}\underline{e}_1^T\\ \underline{e}_2^T\\ -\\ -\\ \underline{e}_n^T\end{bmatrix}\frac{1}{\sqrt{n}}$$

Define $W = \frac{1}{\sqrt{n}}\left[\underline{e}_1\ \underline{e}_2 - -\ \underline{e}_n\right]$ and thus

$$E = WW^T$$

By inspection of 4.10,

$$E(t) = W(t)W(t)^T = \Phi E(t_o)\,\Phi^T = \Phi\, W(t_o)W(t_o)^T\Phi^T$$

4.11 $$W(t) = \Phi\, W(t_o)$$

Physically, W is a set of linearly independent vectors drawn from the sample space of E which have the property that

$$\frac{1}{n}\sum_{i=1}^{n}\underline{e}_i\underline{e}_i^T = \int_V \underline{\xi}\,\underline{\xi}^T f(\underline{\xi})\, d\xi_1\, d\xi_2 - - d\xi_n$$

That is, a finite, discrete sample from a continuous distribution, having the particular property that their simple average produces the covariance defined by the distribution function $f(\xi)$. They "typify" the distribution.

Sequential Estimator (Kalman Filter)

The sequential estimator has the form

4.12 $$\delta\hat{\underline{X}}(t_n) = \delta\hat{\underline{X}}'(t_n) + \underline{w}_n\,(\delta\tilde{Q}(t_n) - \delta\hat{Q}'(t_n))$$

where

$$\delta\hat{\underline{X}}'(t_n) = \Phi(t_n, t_{n-1})\ \delta\hat{\underline{X}}(t_{n-1})$$

$$\delta\hat{Q}' = \underline{b}_Q^T(t_n)\ \delta\hat{\underline{X}}'$$

$$\delta\tilde{Q} = \tilde{Q} - Q(\underline{X}_R)$$

$\tilde{Q}$ = measured value of Q

$\underline{w}$ an nx1 vector to be determined

Equation 4.12 proposes that if $\delta\hat{\underline{X}}'$ is an estimate of $\delta\underline{X}$, a better estimate of $\delta\underline{X}$ is the liner combination of $\delta\hat{\underline{X}}'$ with a weighting vector, $\underline{w}$, multiplied by the difference between the measured value of δQ ($\delta\tilde{Q}$) and the expected value of δQ ($\delta\hat{Q}'$) based on $\delta\hat{\underline{X}}'$. To determine $\underline{w}$, it is necessary to consider what constitutes a "better" estimate of $\delta\underline{X}$. Clearly, in a particular circumstance, no direct knowledge of the existing error is possible, and it will be necessary to deal with some average function of the error. Since in general the state error distribution is unknown, one must work with the mean error and the covariance of errors. Minimizing the average squared error seems to be the most physically meaningful objective. To this end, use 4.12 to construct an expression for the state error:

$$\begin{aligned}
\underline{e}(t_n) = \delta\underline{X}_n - \delta\hat{\underline{X}}_n &= \delta\underline{X}_n - \delta\hat{\underline{X}}_n' - \underline{w}_n(\delta\tilde{Q}_n - \delta Q_n') \\
&= \hat{\underline{e}}_n' - \underline{w}_n(\underline{b}_Q^T\,\delta\underline{X} + \alpha_Q - \underline{b}_Q^T\,\delta\hat{\underline{X}}_n') \\
&= \hat{\underline{e}}_n' - \underline{w}_n\underline{b}_Q^T\,(\delta\underline{X} - \delta\hat{\underline{X}}_n') - \alpha_Q\underline{w}_n \\
&= \hat{\underline{e}}_n' - \underline{w}_n\underline{b}_Q^T\,\underline{e}_n' - \alpha_Q\underline{w}_n \\
&= (I - \underline{w}\underline{b}_Q^T)\,\underline{e}_n' - \alpha_Q\underline{w}_n
\end{aligned}$$

4.13

Where

$$\tilde{\delta Q} = \tilde{Q}(\underline{X}) - Q(\underline{X}_R)$$

$$= Q(\underline{X}) + \alpha_Q - Q(\underline{X}_R)$$

$$= \text{measured value of } \delta Q$$

and

$$\alpha_Q = \text{random measurement noise}$$

By direct calculation, the covariance of state errors E is

4.14a
$$E = \overline{\underline{ee}^T} = \overline{(I - \underline{wb}_Q^T)\,\underline{e}_n\underline{e}_n^T(I - \underline{wb}_Q^T)^T + \alpha_Q^2\,\underline{www}^T} - \overline{(I - \underline{wb}_Q^T)\,\underline{e}_n\underline{w}^T\alpha_Q - \alpha_Q\underline{we}_n^T\,(I - \underline{b}_Q\underline{w}^T)^T}$$

Since previous state errors are uncorrelated with current measurement errors, the terms involving

$$\overline{\underline{e}_n \alpha_Q}, \qquad \overline{\alpha_Q \underline{e}_n^T},$$

average to zero. Hence the expression for the covariance is

4.14b $$E = (I - \underline{w}\,\underline{b}_Q^T)\, E_r' \,(I - \underline{w}\,\underline{b}_Q^T)^T + \overline{\alpha_Q^2}\,\underline{w}\,\underline{w}^T$$

The diagonal elements of E, E' are the respective average squared errors in the components of $\underline{X}_E$. Hence, by calculus of variations $\underline{w}$ will be chosen so as to minimize the elements of E, including its diagonal: For an extremum of E with respect to $\underline{w}$

$$\delta E = \left[\left(- \delta \underline{w}\, \underline{b}_Q^T E' \,(I - \underline{w}\,\underline{b}_Q^T)\right] + \overline{\alpha_Q^2}\,\delta \underline{w}\,\underline{w}^T\right.$$

$$+ \left[(I - \underline{w}\,\underline{b}_Q^T)^T E'\, \underline{b}_Q\, (-\delta \underline{w}^T)\right] + \overline{\alpha_Q^2}\,\underline{w}\delta\underline{w}^T$$

$$= \left\{ \left[-E'\underline{b}_Q + (\underline{b}_Q^T E'\, \underline{b}_Q + \overline{\alpha_Q^2})\, \underline{w}\right]\delta\, \underline{w}^T \right\}^T + \left[-E'\underline{b}_Q + (\underline{b}_Q^T E_n' \underline{b}_Q + \overline{\alpha_Q^2})\, \underline{w}\right]\delta\underline{w}^T$$

$$= [0] \quad \text{for an extremum}$$

This condition will apparently be met if

$$\left[-E'\underline{b}_Q + (\underline{b}_Q^T\, E'\underline{b}_Q + \overline{\alpha_Q^2})\, \underline{w}\right] = \underline{0}$$

which implies

4.15 $$\underline{w} = \frac{E'\underline{b}_Q}{\overline{\alpha_Q^2} + \underline{b}^T E' \underline{b}}$$

This is the Kalman Filter. It remains to show that extremum of E for which $\underline{w}$ has this value is a minimum:

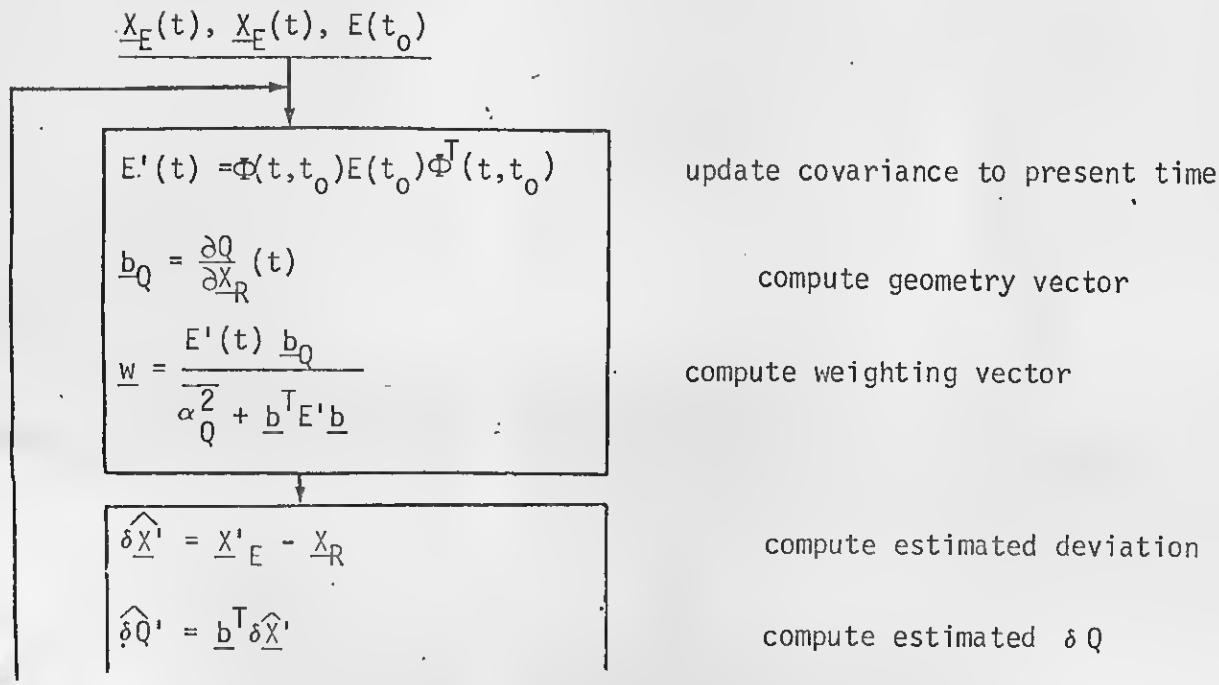
Using the expression for δE:

$$\delta^2 E = (\underline{b}_Q^T E' \underline{b}_Q + \overline{\alpha_Q^2})(\delta \underline{w} \delta \underline{w}^T + \delta \underline{w} \delta \underline{w}^T)$$

It is clearly apparent that

$$(\underline{b}_Q^T E' \underline{b}_Q + \overline{\alpha_Q^2}) > 0$$

always since E' is positive definite. Further more, the diagonal terms of $\delta \underline{w}\ \delta \underline{w}^T$ are positive for all real variations, $\delta \underline{w}$. The corresponding second variations of E, those of the mean squared error, are therefore positive and minimum. Figure III presents an operational flow chart of the filter.



$$\delta\tilde{Q} = \tilde{Q} - Q(\underline{X}_R) \qquad \text{compute measured } \delta Q$$

$$\delta\hat{\underline{X}} = \delta\hat{\underline{X}}' + \underline{w}(\delta\tilde{Q} - \delta\hat{Q}') \qquad \text{compute update to estimates } \delta\underline{X}$$

$$\underline{X}_E = \underline{X}_E' + \delta\hat{\underline{X}} \qquad \text{update } \underline{X}_E$$

$$E = (I - \underline{w}\,\underline{b}_Q^T)E'(I - \underline{w}\,\underline{b}_Q^T)^T + \underline{w}\,\underline{w}^T\,\alpha^2_Q \qquad \text{update } E$$

Figure III.

V. Further development and insights.

Propagation of Errors

For a Keplerian force field, the state transition matrix, which is a solution to the equation.

$$\dot{\Phi}(t,t_o) = [F]\Phi(t,t_o) \qquad \Phi(t_o,t_o) = I$$

has the property that

$$\Phi^T J\,\Phi = J \qquad J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} \qquad J^T J = I \qquad J^2 = -I$$

For as single vehicle, Φ is '6x6 and from 4.5b

$$\underline{F} = \begin{bmatrix} \phi & I \\ G & 0 \end{bmatrix} \qquad G = \frac{\mu}{R^5}\,(3\underline{R}\,\underline{R}^T - R^2 I)$$

To prove this property, note that at t_o

$$\Phi^T J \Phi = \underline{I}\,\underline{J}\,\underline{I} = \underline{J} \qquad \Phi(t_o,t_o) = \underline{I}$$

Evaluating

$$\frac{d}{dt}(\Phi^T \underline{J} \Phi) = \dot{\Phi}^T \underline{J} \Phi + \Phi^T \underline{J} \dot{\Phi}$$

$$= \Phi^T F^T \underline{J} \Phi + \Phi^T \underline{J} F \Phi$$

$$= \Phi^T \left[\underline{F}^T \underline{J} + \underline{J}\, \underline{F} \right] \Phi$$

$$\underline{F}^T \underline{J} = \begin{bmatrix} 0 & I \\ G & 0 \end{bmatrix} \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} = \begin{bmatrix} -I & 0 \\ 0 & G \end{bmatrix} = -\underline{J}F$$

Therefore $\underline{F}^T \underline{J} + \underline{J}\, \underline{F} = [0]$ and

$$\frac{d}{dt}(\Phi^T J \Phi) = [0] \Rightarrow \Phi^T J \Phi = \text{constant} = J$$

Now note that $|\underline{J}| = 1$ and therefore

$$\left| \Phi^T \Phi \underline{J} \right| = \left| \Phi^T \right| \left| \underline{J} \right| \left| \Phi \right|$$

$$= |\Phi|^2 \, 1$$

$$= |J| = 1$$

For a Keplerian field the propagation of E is described by

$$E(t) = \Phi\, E(t_0)\, \Phi^T$$

and its determinant by

$$\left|\underline{E}(t)\right| = \left|\Phi\, E(t_o)\, \Phi^T\right|$$

$$= \left|\Phi\right|^2 \left|E(t_o)\right|$$

$$= \left|E(t_o)\right|$$

The physical interpretation of this fact is taken to be that the total "amount" of error in the system remains bounded in time for a Keplerian force field.

Practical Computational Techniques

It usually happens that operations performed on E by virtue of equation 4.10 or 4.14b will cause it to be no longer symmetric due to computational inaccuracies. Should this happen, it no longer represents a covariance matrix and something must be done. One method is to perform the replacement operation

$$E = \frac{1}{2}(E + E^T)$$

which will symmetrize E in any digital computer worthy of the name. An alternative approach, which offers considerable other computational advantages will be constructed. Making use of the fact that any covariance matrix may be represented as a linear combination of product transposes, define

$$E' = \frac{1}{N}\left[\underline{e}_1\underline{e}_1^T + \underline{e}_2\underline{e}_2^T - - - \underline{e}_N\underline{e}_N^T\right]$$

$$= \frac{1}{\sqrt{N}}\left[\underline{e}_1\ \underline{e}_2 - - \underline{e}_N\right] \qquad \left[\underline{e}_1\ \underline{e}_2 - - \underline{e}_N\right]^T \frac{1}{\sqrt{N}}$$

$$= W'W'^T$$

$$W = \frac{1}{\sqrt{N}}\left[\underline{e}_1\ \underline{e}_2 - - \underline{e}_N\right]$$

Let

$$\underline{z}_Q = W'^T \underline{b}_Q$$

and rewrite equation 4.15 as

$$5.1 \qquad \underline{W} = \left(\frac{W' \underline{z}_Q}{\overline{\alpha_Q^2} + \underline{z}_Q^T \underline{z}}\right)$$

Equation 4.14b, the update equation for E, requires careful consideration:

Substituting for $\underline{w}$:

$$WW^T = \left(I - \frac{W'\underline{z}\,\underline{b}^T}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}}\right) W'W'^T \left(I - \frac{W'\underline{z}\,\underline{b}^T}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}}\right)^T + \frac{W'\underline{z}\,\underline{z}^T W'^T}{(\overline{\alpha_Q^2} + \underline{z}^T\underline{z})^2}\,\overline{\alpha_Q^2}$$

$$= \left(W' - \frac{W'\underline{z}\,\underline{z}^T}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}}\right)\left(W'^T - \frac{\underline{z}\,\underline{z}^T W'^T}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}}\right) + \frac{W'\underline{z}\,\underline{z}^T W'^T}{(\overline{\alpha_Q^2} + \underline{z}^T\underline{z})^2}\,\overline{\alpha_Q^2}$$

$$= W'\left[\left(I - \frac{\underline{z}\,\underline{z}^T}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}}\right)\left(I - \frac{\underline{z}\,\underline{z}^T}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}}\right) + \frac{\underline{z}\,\underline{z}^T\,\overline{\alpha_Q^2}}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}}\right] W'^T$$

$$= W'\left[\left(I - \frac{2\underline{z}\,\underline{z}^T}{\overline{\alpha_Q^2} + \underline{z}^T\underline{z}} + \frac{\underline{z}\,\underline{z}^T(\underline{z}^T\underline{z})}{(\overline{\alpha_Q^2} + \underline{z}^T\underline{z})^2} + \frac{\underline{z}\,\underline{z}^T\,(\overline{\alpha_Q^2})}{(\overline{\alpha_Q^2} + \underline{z}^T\underline{z})^2}\right)\right] W'^T$$

$$= W' \left(I - \frac{\underline{z}\,\underline{z}^T}{\alpha_Q^2 + \underline{z}^T\underline{z}}\right) W'^T$$

Look for a matrix of the form

$$\left(I - \frac{K\,\underline{z}\,\underline{z}^T}{\alpha_Q^2 + \underline{z}^T\underline{z}}\right)$$

such that

$$\left(I - \frac{K\underline{z}\,\underline{z}^T}{\alpha_Q^2 + \underline{z}^T\underline{z}}\right)^2 = \left(I - \frac{\underline{z}\,\underline{z}^T}{\alpha_Q^2 + \underline{z}^T\underline{z}}\right)$$

$$I - \frac{2K\underline{z}\,\underline{z}^T}{\alpha_Q^2 + \underline{z}^T\underline{z}} + \frac{(K^2\underline{z}^T\underline{z})\,\underline{z}\,\underline{z}^T}{(\alpha_Q^2 + \underline{z}^T\underline{z})^2} = I - \frac{\underline{z}\,\underline{z}^T}{\alpha_Q^2 + \underline{z}^T\underline{z}}$$

$$\left(-\frac{2K}{a} + \frac{1}{a} + \frac{K^2\underline{z}^T\underline{z}}{a^2}\right) \underline{z}\,\underline{z}^T = [0]$$

$$K^2 - \frac{2a}{\underline{z}^T\underline{z}}\,K + \frac{a}{\underline{z}^T\underline{z}} = 0$$

$$K = \frac{\frac{2a}{\underline{z}^T\underline{z}} - \sqrt{\frac{4a^2}{(\underline{z}^T\underline{z})^2} - \frac{4a}{\underline{z}^T\underline{z}}}}{2}$$

$$= \frac{a}{\underline{z}^T\underline{z}} \left(1 - \sqrt{1 - \frac{\underline{z}^T\underline{z}}{a}}\right)$$

where $a = \underline{z}^T\underline{z} + \overline{\alpha_Q^2}$. This may be simplified by letting

$$c = \frac{1}{K} = \frac{\frac{\underline{z}^T\underline{z}}{a}}{\left(1 + \sqrt{1 - \frac{\underline{z}^T\underline{z}}{a}}\right)} = \frac{\frac{\underline{z}^T\underline{z}}{a}\left(1 + \sqrt{1 - \frac{\underline{z}^T\underline{z}}{a}}\right)}{(1)^2 - \left(\sqrt{1 - \frac{\underline{z}^T\underline{z}}{a}}\right)^2}$$

$$= 1 + \sqrt{1 - \frac{\underline{z}^T\underline{z}}{a}}$$

$$= 1 + \sqrt{\frac{\overline{\alpha_Q^2}}{a}}$$

5.2

Putting 1/c in for K

$$WW^T = W'\left(I - \frac{\underline{z}\,\underline{z}^T}{a}\right) W'^T$$

$$= W'\left(I - \frac{\underline{z}\,\underline{z}^T}{ca}\right)^2 W'^T$$

Therefore

5.3 $$W = W'\left(I - \frac{\underline{z}\,\underline{z}^T}{ca}\right)$$

The advantages to working with W rather than E are several:

(1) Since $WW^T = E$, any function of W of the form $\underline{z}^T\underline{z}$ or $W\underline{z}$ implies a symmetric (positive definite) E.

(2) The updating equation for W is simpler than that for E and involves fewer computations.

(3) Since W is a construction of representative error vectors $\{\underline{e}_i\}$, its propagation in time is more easily discussed.

Noting that $E(t) = \Phi(t,t_o)\, E_o\, \Phi^T(t,t_o)$, it is found that

$$E(t) = W(t)W(t) = \Phi\, W_o\, W_o^T\, \Phi^T$$

5.4a $$W(t) = \Phi(t,t_o)\, W(t_o)$$

Also

$$\dot{W}(t) = \Phi\, W_o = F\, \Phi\, W_o$$

5.4b $$= FW$$

A third method of advancing W is to note that by definition

$$\underline{e}_i(t_o) = \underline{X}_E^i(t_o) - \underline{X}(t_o)$$

$\underline{X}_E$ = estimated state

$\underline{X}$ = actual state

therefore

$$\underline{e}_i(t) = \underline{X}_E^i(t) - \underline{X}(t)$$

Let

$$\underline{X}_E^i(t_o) = \underline{X}_E(t_o) + \underline{e}_i(t_o)$$

$$\underline{e}_i(t) = \underline{X}_E^i(t) - \underline{X}_E(t)$$

where $\underline{X}_E^i(t)$ and $\underline{X}_E(t)$ are the respective vectors at to integrated to time t. This constitutes a more exact solution to the problem of propagating E and makes no linearizing assumptions as were made to obtain 4.5b, which is the basis for equations 5.4a and 5.4b.

Definition of the Reference State

For practical purposes, the reference state is generally taken to be the estimated state at each instant. In this case

$$\delta\hat{\underline{X}}' = \hat{\underline{X}}' - \underline{X}_R$$

$$= \hat{\underline{X}}' - \hat{\underline{X}}' = \underline{0}$$

$$\delta\hat{Q}' = \underline{b}_Q^T \, \delta\hat{\underline{X}}' = 0$$

$$\delta\tilde{Q} = \tilde{Q} - Q(\underline{X}_R)$$

$$= \tilde{Q} - Q(\hat{\underline{X}}')$$

$$\delta\hat{\underline{X}} = \underline{w}\,\delta\tilde{Q} = \underline{w}\,(\tilde{Q} - Q(\hat{\underline{X}}'))$$

$$\hat{\underline{X}} = \hat{\underline{X}}' + \delta\hat{\underline{X}}$$

This definition guarantees that as the differential correction process occurs, $\underline{X}_R$ is sufficiently close to $\underline{X}$ so that the linearizing assumptions of Sec. IV are valid.

Computation of the Geometry ($\underline{b}$) Vecter

For a range measurement

$$r = \left[\underline{r}^T\underline{r}\right]^{1/2} = \left[(\underline{R}_T - \underline{R}_S)^T(\underline{R}_T - \underline{R}_S)\right]^{1/2}$$

$$\delta r = 1/2 \left[\underline{r}^T \underline{r}\right]^{-1/2} (\delta \underline{r}^T \underline{r}) + 1/2 \left[\underline{r}^T \underline{r}\right]^{-1/2} (\underline{r}^T \delta \underline{r}) + \beta_r$$

$$= \left[\underline{r}^T \underline{r}\right]^{-1/2} \underline{r}^T \delta \underline{r} + \beta_r$$

$$= \hat{\underline{r}}^T \delta \underline{r} = \hat{\underline{r}}^T \left[\delta \underline{X}_T - \delta \underline{X}_S\right] + \beta_r$$

Where β_r is the range measurement bias, and $\hat{\underline{r}}$ = unit $(\underline{r})$. Therefore,

$$\underline{b}_r = \begin{bmatrix} -\hat{\underline{r}} \\ \underline{0} \\ \hat{\underline{r}} \\ \underline{0} \\ \underline{k}_r \end{bmatrix} \qquad \text{for } \underline{k}_r = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

For a range-rate measurement

$$\dot{r} = \hat{\underline{r}}^T \dot{\underline{r}} = \frac{\underline{r}^T}{r} \dot{\underline{r}}$$

$$\delta \dot{r} = \frac{\delta \underline{r}^T}{r} \dot{\underline{r}} - \frac{\underline{r}^T \dot{\underline{r}}}{r^2} \delta r + \frac{\underline{r}^T}{r} \delta \dot{\underline{r}} + \beta_{\dot{r}}$$

$$= \frac{\delta \underline{r}^T}{r} \dot{\underline{r}} - \frac{\underline{r}^T \dot{\underline{r}}}{r^2} (\hat{\underline{r}} \cdot \delta \underline{r}) + \frac{\underline{r}^T}{r} \delta \dot{\underline{r}} + \beta_{\dot{r}}$$

$$= \frac{1}{r}\left[\dot{\underline{r}} - (\hat{\underline{r}}^T \dot{\underline{r}})\, \hat{\underline{r}}\right]^T \delta \underline{r} + \hat{\underline{r}}\, \delta \dot{\underline{r}} + \beta_{\dot{r}}$$

$$= -\frac{1}{r}\left[\hat{\underline{r}} \times (\hat{\underline{r}} \times \dot{\underline{r}})\right]^T \delta \underline{r} + \hat{\underline{r}}\, \delta \dot{\underline{r}} + \beta_{\dot{r}}$$

The geometry vector for range-rate is accordingly

$$\underline{b}_{\dot{r}} = \begin{bmatrix} +\frac{1}{r}\left[\hat{\underline{r}} \times (\hat{\underline{r}} \times \underline{\dot{r}})\right] \\ -\hat{\underline{r}} \\ \frac{1}{r}\left[\hat{\underline{r}} \times (\hat{\underline{r}} \times \underline{\dot{r}})\right] \\ +\hat{\underline{r}} \\ \underline{k}_{\dot{r}} \end{bmatrix} \qquad \underline{k}_{\dot{r}} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

For azimuth and altitude measurements, defined according to Fig. II as

$$AZ = TAN^{-1}\left(\frac{\hat{\underline{z}} \cdot \hat{\underline{r}}}{\hat{\underline{Y}} \cdot \hat{\underline{r}}}\right)$$

$$EL = SIN^{-1}\left(\hat{\underline{x}} \cdot \hat{\underline{r}}\right)$$

Some tedious algebraic hack produces

$$\underline{b}_{AZ} = \begin{bmatrix} -(\hat{\underline{x}} \times \underline{r}) / \left|\hat{\underline{x}} \times \underline{r}\right|^2 \\ \underline{0} \\ (\hat{\underline{x}} \times \hat{\underline{r}}) / \left|\hat{\underline{x}} \times \underline{r}\right|^2 \\ \underline{0} \\ \underline{k}_{AZ} \end{bmatrix} \qquad \underline{k}_{AZ} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\underline{b}_{EL} = \begin{bmatrix} -\dfrac{|\hat{\underline{x}} \times \hat{\underline{r}}|}{r} (\hat{\underline{r}} \times \underline{b}_{AZ}) \\ \underline{0} \\ \dfrac{|\hat{\underline{x}} \times \hat{\underline{r}}|}{r} (\hat{\underline{r}} \times \underline{b}_{AZ}) \\ \underline{0} \\ \underline{k}_{EL} \end{bmatrix} \qquad \underline{k}_{EL} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

For these specific cases, the bias sub-vector of the state is defined as

$$\underline{\beta} = \begin{bmatrix} \beta r \\ \dot{\beta r} \\ \beta AZ \\ \beta EL \end{bmatrix}$$

Assorted comments about the weighting vector, $\underline{w}$.

$$\underline{w} = \frac{E'\underline{b}}{\overline{\alpha_Q^2} + \underline{b}^T E' \underline{b}} = \frac{W'\underline{z}}{\overline{\alpha_Q^2} + \underline{z}^T \underline{z}}$$

Examine $\underline{b}^T E' \underline{b}$:

$$\underline{b}^T E' \underline{b} = \underline{b}^T \overline{E' E'^T} \underline{b}$$

$$= \overline{\underline{b}^T \underline{e}' \underline{e}'^T \underline{b}}$$

$$= \overline{U_Q^{2'}}$$

where $\underline{b}^T\underline{e} = U_Q$, the a' priori uncertainty in Q due to state errors. Since $\overline{\alpha_Q^2}$ is the variance of Q due to measurement noise, the term

$$\overline{\alpha_Q^2} + \underline{b}^T E' \underline{b} = \overline{\alpha_Q^2} + \underline{z}^T\underline{z}$$
$$= \overline{\alpha_Q^2} + \overline{U_Q^{2'}}$$

is the total variance of Q due to measurement noise and state uncertainty. Now compute the estimate $\delta\hat{Q}$, the estimate of δQ based on $\delta\hat{\underline{x}}$:

$$\delta\hat{\underline{X}} = \delta\hat{\underline{X}}' + \underline{w}\,(\delta\tilde{Q} - \delta\hat{Q}')$$

$$\delta\hat{Q} = \underline{b}^T\,\delta\hat{\underline{X}}' + \underline{b}^T\underline{w}\,(\delta\tilde{Q} - \delta\hat{Q}')$$

$$= \delta\hat{Q}' + \frac{\underline{b}^T\underline{W}\underline{z}}{\overline{\alpha_Q^2} + \overline{U_Q^{2'}}}\,(\delta\tilde{Q} - \delta\hat{Q}')$$

$$= \delta\hat{Q}' + \frac{\underline{z}^T\underline{z}}{\overline{\alpha_Q^2} + \overline{U_Q^{2'}}}\,(\delta\tilde{Q} - \delta\hat{Q}')$$

$$= \delta\hat{Q}' + \frac{\overline{U_Q^2}}{\overline{\alpha_Q^2} + \overline{U_Q^{2'}}}\,(\delta\tilde{Q} - \delta\hat{Q}')$$

Finally

$$\delta\hat{Q} = \delta\hat{Q}' \left(\frac{\overline{\alpha_Q^2}}{\overline{\alpha_Q^2} + \overline{U_Q^{2'}}}\right) + \delta\tilde{Q} \left(\frac{\overline{U_Q^2}}{\overline{\alpha_Q^2} + \overline{U_Q^{2'}}}\right)$$

Let the ratio of measurement noise to state uncertainty be denoted

$$\rho' = \overline{\alpha_Q^2} \,/\, \overline{U_Q^{2'}}$$

so that

$$\delta\hat{Q} = \delta\hat{Q}' \left(\frac{\rho'}{1+\rho'}\right) + \delta\tilde{Q} \left(\frac{1}{1+\rho'}\right)$$

Obviously if measurement noise variance, $\overline{\alpha_Q^2}$, is very small compared to the state uncertainty of δQ, $\overline{U_Q^2}$, we have $\rho' \sim 0$ and

$$\delta\hat{Q} \simeq \delta\tilde{Q} \quad \underline{\text{estimate equal to measured}}$$

On the other hand, if the measurement noise is large compared to the state uncertainty $1/\rho' \sim 0$ and

$$\delta\hat{Q} \simeq \delta\hat{Q}' \quad \underline{\text{estimate equal to previous estimate}}$$

Now look at the reduction of the state uncertainty of δQ, $\overline{U_Q^2}$, resulting from a measurement. From 5.3

$$W = W' \frac{(I - \underline{z}\,\underline{z}^T)}{ca} \qquad a = \overline{\alpha_Q^2} + \overline{U_Q^2}$$

$$c = 1 - \sqrt{\overline{\alpha_Q^2}/a}$$

$$\overline{U_Q^2} = \underline{z}^T\underline{z} = \underline{b}^T WW^T\underline{b}$$

$$= \underline{b}^T W' \frac{(I - \underline{z}\,\underline{z}^T)}{ca} \frac{(I - \underline{z}\,\underline{z}^T)}{ca} W'^T\underline{b}$$

$$= \underline{b}^T W' \frac{(I - \underline{z}\,\underline{z}^T)}{a} W'^T\underline{b}$$

by definition

$$= \underline{z}^T \frac{(I - \underline{z}\,\underline{z}^T)}{a} \underline{z}$$

$$= \overline{U_Q^{2'}} - \frac{(\overline{U_Q^2})^2}{a}$$

$$= \overline{U_Q^{2'}}\left(1 - \frac{\overline{U_Q^2}}{a}\right)$$

$$= \overline{U_Q^{2'}} \frac{(a - \overline{U_Q^2})}{a}$$

$$= \overline{U_Q^{2'}} \left(\frac{\overline{\alpha_Q^2}}{\overline{\alpha_Q^2} + \overline{U_Q^2}}\right)$$

$$= \overline{U_Q^{2'}} \left(\frac{\rho'}{1+\rho'}\right)$$

Again, as noted before, if $\rho' \simeq 0$,

$$\overline{U_Q^2} \simeq 0$$

i.e. the state uncertainty of δQ is greatly reduced. If the measurement noise is large compared to the state uncertainty, $\rho'/(1+\rho') \sim 1$ and

$$\overline{U_Q^2} \simeq \overline{U_Q^{2'}}$$

This tallies with the expectation that if the measurement process is very noisy compared to the state errors, the average state errors are little reduced by the fixes. The case of a range measurement is particularly illustrative:

The form

$$\overline{U_r^2} = Z^T Z = \underline{b}_r^T W W^T \underline{b}_r = \underline{b}_r^T E \underline{b}_r = \overline{\underline{b}_r^T \underline{e}\, \underline{e}^T \underline{b}_r}$$

is the average squared state error along the line of sight. Typically, a good range sensor has 1 at random noise of about 10 feet on a measurement, thus

$$\overline{\alpha^2_r} = 100 \text{ ft}^2$$

If as is typical, the line of sight relative state errors are on the order of 5000 ft, we have

$$\overline{U_r^2} = 2{:}5 \times 10^7 \text{ ft}^2$$

$$\rho \simeq 4 \times 10^{-6}$$

$$\rho/1 + \rho \simeq (1 - \rho) = \rho - \rho^2$$

$$\simeq \rho$$

$$= 4 \times 10^{-6}$$

Across a mark, therefore

$$\delta\hat{Q} = \delta\hat{Q}' \left(\frac{\rho'}{1 + \rho'} \right) + \delta\tilde{Q} \left(\frac{1}{1 + \rho'} \right)$$

$$\simeq \delta\hat{Q}' \left(4 \times 10^{-6} \right) + \delta\tilde{Q} \left(1 - 4 \times 10^{-6} \right)$$

$$= \delta\hat{Q}' \left(.000004 \right) + \delta\tilde{Q} \left(.999996 \right)$$

The state uncertainty is reduced to

$$\overline{U_r^2} = \overline{U_r^{2'}} \left(\frac{\rho'}{1 + \rho'} \right)$$

$$\simeq \overline{U_r^{2'}} \left(4 \times 10^{-6} \right)$$

$$= 2.5 \times 10^{7} \left(4 \times 10^{-6} \right)$$

$$= 100 \text{ ft}^2$$

$$\simeq \overline{\alpha_Q^2}$$

Note that the reduction coefficient,

$$\left(\frac{\rho}{1 + \rho} \right)$$

is always ≤ 1 which guarantees that the measurement process never results in a greater $\overline{U_Q^2}$ after a mark than before it. Suppose in the interval between marks $\overline{U_Q^2}$ does not change much. Then given $\overline{U_Q^2}(t_o)$ we have

$$\rho(t_o) = \frac{\overline{\alpha_Q^2}}{\overline{U_Q^2}(t_o)} = \frac{\overline{\alpha_Q^2}}{\overline{U_Q^2}(t_o)\left(\frac{\rho'}{1+\rho'}\right)} = \frac{\rho'}{\rho'/(1+\rho')} = 1 + \rho'$$

where $\rho'(t_o) \dfrac{\overline{\alpha_Q^2}}{\overline{U_Q^{2'}}(t_o)}$

If, by assumption, ρ changes little between marks, after n marks

$$\rho(t_i) \simeq \rho(t_0) = 1 + \rho'(t_0)$$

$$\rho(t_i) \simeq 1 + \rho'(t_i) = 1 + 1 + \rho' = 2 + \rho'(t_0)$$

$$\rho(t_n) \simeq (n + 1) + \rho'(t_0)$$

thus
$$\frac{\rho(t_n)}{1 + \rho(t_0)} = \frac{(n+1) + \rho'(t_0)}{1 + n+1 + \rho'(t_0)}$$

$$\simeq \frac{n + 1}{1 + n + 1} \qquad n + 1 \gg \rho'$$

$$\simeq 1 \qquad n + 1 \gg 1$$

And

$$\overline{U_Q^2}\,(t_n) \simeq \overline{U_Q^{2'}}\,(t_n)$$

I.e. the uncertainty in the direction of $\underline{b}_Q$ is not much reduced after a large number of marks. When this condition has occured, the filter is said to be "saturated". Since E is a real symmetric matrix, it eigenvectors are orthogonal, and the matrix of its normalized eigenvectors is an orthonormal matrix. Hence, as noted in Sec. I, E has a diagonal representation in terms of its eigenvalues as

$$E = P \Lambda P^T \qquad P\,P^T = P^T P = I$$

The matrix of orthonormal vectors, ρ, may be considered as defining a co-ordinate rotation in the space of n dimensions; or alternatively, a change of variable in that space, to a new set

$$\underline{\xi} = P^T \delta X$$

Cast $\underline{w}$ into the new variables

$$\underline{w}_\xi = P^T \underline{w}_X = \frac{P^T E \underline{b}}{\overline{\alpha_Q^2} + \overline{U_Q^2}}$$

$$= \frac{P^T E \, P P^T \underline{b}}{\overline{\alpha_Q^2} + \overline{U_Q^2}}$$

$$= \frac{\Lambda \, \underline{b}_\xi}{\overline{\alpha_Q^2} + \overline{U_Q^2}} \qquad \underline{b}_\xi = P^T \underline{\xi}$$

Note that

$$\underline{b}_\xi = P^T \frac{\partial Q}{\partial \underline{X}} = \frac{\partial Q}{\partial \underline{\xi}}$$

Also

$$\overline{U_Q^2} = \underline{b}^T E \underline{b} = \underline{b}_\xi^T \Lambda \, \underline{b}_\xi^T = \sum \left(\frac{\partial Q}{\partial \underline{\xi}}\right)^2_i \sigma_i^2 =$$

$$(\underline{w}_{\xi})_i = \frac{\frac{\partial Q}{\partial \xi_i}\sigma_i^2}{\overline{\alpha_Q^2} + \sum\left(\frac{\partial Q}{\partial \xi}\right)_j^2 \sigma_j}$$

$$= \frac{\partial Q}{\partial \xi_i}\left[\frac{\overline{\alpha_Q^2}}{\sigma_i^2} + \left(\frac{\partial Q}{\partial \xi_i}\right)^2 + \sum_{j\neq i}\left(\frac{\partial Q}{\partial \xi_j}\right)^2\left(\frac{\sigma_j}{\sigma_i}\right)^2\right]^{-1}$$

$$= \left(\frac{\partial Q}{\partial \xi_i}\right)^{-1}\left\{1+\left(\frac{\partial Q}{\partial \xi_i}\right)^{-2}\left[\frac{\overline{\alpha_Q^2}}{\sigma_i^2} + \sum_{j\neq i}\left(\frac{\partial Q}{\partial \xi_j}\right)^2\left(\frac{\sigma_j}{\sigma_i}\right)^2\right]\right\}^{-1}$$

$$= \left(\frac{\partial \xi_i}{\partial Q}\right)\left\{1 + \left(\frac{\partial \xi_i}{\partial Q}\right)^2\left[\frac{\overline{\alpha_Q^2}}{\sigma_i^2} + \sum_{j\neq i}\left(\frac{\partial Q}{\partial \xi_j}\right)^2 \cdot \frac{\sigma_j^2}{\sigma_i^2}\right]\right\}^{-1}$$

Where σ_i^2 is the i^{th} eigenvalue of $\underline{E}$, the variance of ξ_i. In the event that $\sigma_i >> \sigma_j$ for all j and $\sigma_i >> \dot{\alpha}_Q^2$ the inverse in brackets approximates 1:

$$(\underline{w}_{\xi}) \simeq \frac{\partial \xi_i}{\partial Q}$$

The updating equation in terms of the new variables is

$$\delta\hat{\underline{\xi}} = P^T \delta\hat{\underline{X}} = P^T \delta\hat{\underline{X}}' + P^T\underline{W}(\delta\tilde{Q} - \delta\hat{Q}')$$

$$= \delta\hat{\xi}' + \underline{W}(\delta\tilde{Q} - \delta\hat{Q}')$$

For the case where $\delta\hat{\underline{\xi}}' = \underline{0}$ as described previously, the update for the $i\underline{th}$ component of $\delta\hat{\underline{\xi}}$ is

$$(\delta\hat{\underline{\xi}})_i \simeq \left(\frac{\partial \xi}{\partial Q}\right)_i \delta\tilde{Q}$$

as expected. Thus thus true meaning of $\underline{w}$ is this:

> It's sort of the inverse partial of $\underline{X}$ with respect to Q, weighted by assorted variances of the measurement and state uncertainties, and conditioned by the extent to which $\underline{b}$ is in the direction of the expected error; sort of . . .

Bandsito
Pundup
Prox

2.9 Rendezvous Analysis

2.9.1 Introduction

Concentric Flight Plan is the name given to the technique of rendezvous developed during Program Gemini and now in use for Apollo. It is the fruition of attempts to construct a plan which offers simplicity of operation, high reliability of achievement and fuel economy. As a matter of basic philosophy, it has been assumed that the spacecraft crew would participate in the rendezvous activities to the extent of flying the vehicle, evaluating the progress of the trajectory and (when necessary) computing backup solutions for the maneuvers.

Early studies by Flight Procedures Branch and others resulted in the identification of transfer elevation angle (η_{TPI}) and transfer interval ($\overline{\omega t}$) as critical parameters characterizing the shape of various intercept trajectories. Further work resulted in choices of $\overline{\omega t}$ and η_{TPI} to reconcile conflicting requirements of fuel economy, error propagation and ease of operation. As a logical extension of this work, the coelliptical trajectory was settled on as the standard pre-transfer condition. Since the time of arrival at a given elevation angle can be varied by changing differential altitude in the coelliptical phase, it is possible to control the transfer time for a given η_{TPI} , placing it so as to satisfy requirements of lighting, navigation, and ground tracking. By fixing η_{TPI} and $\overline{\omega t}$ in advance and choosing a coelliptical pre-transfer condition, the shape of the rendezvous trajectory is held constant throughout from various approaches, so that the crew can hope to become familiar with its development during training. Such familiarity enables them to monitor its progress, detect off-nominal conditions and develop a high degree of proficiency in execution. The predictions of digital analysis and simulation with regard to trajectory behavior and fuel consumption appear to have been borne out in detail on the Gemini and Apollo programs.

A principle activity of Flight Procedures Branch during Projects Gemini and Apollo has been the construction and testing of rendezvous maneuver charts. These are provided as an independent source of rendezvous maneuver solutions, requiring only the most fundamental information - range, range rate, elevation angle - available directly from the radar and platform. In order to perform the required analysis and verification, several large

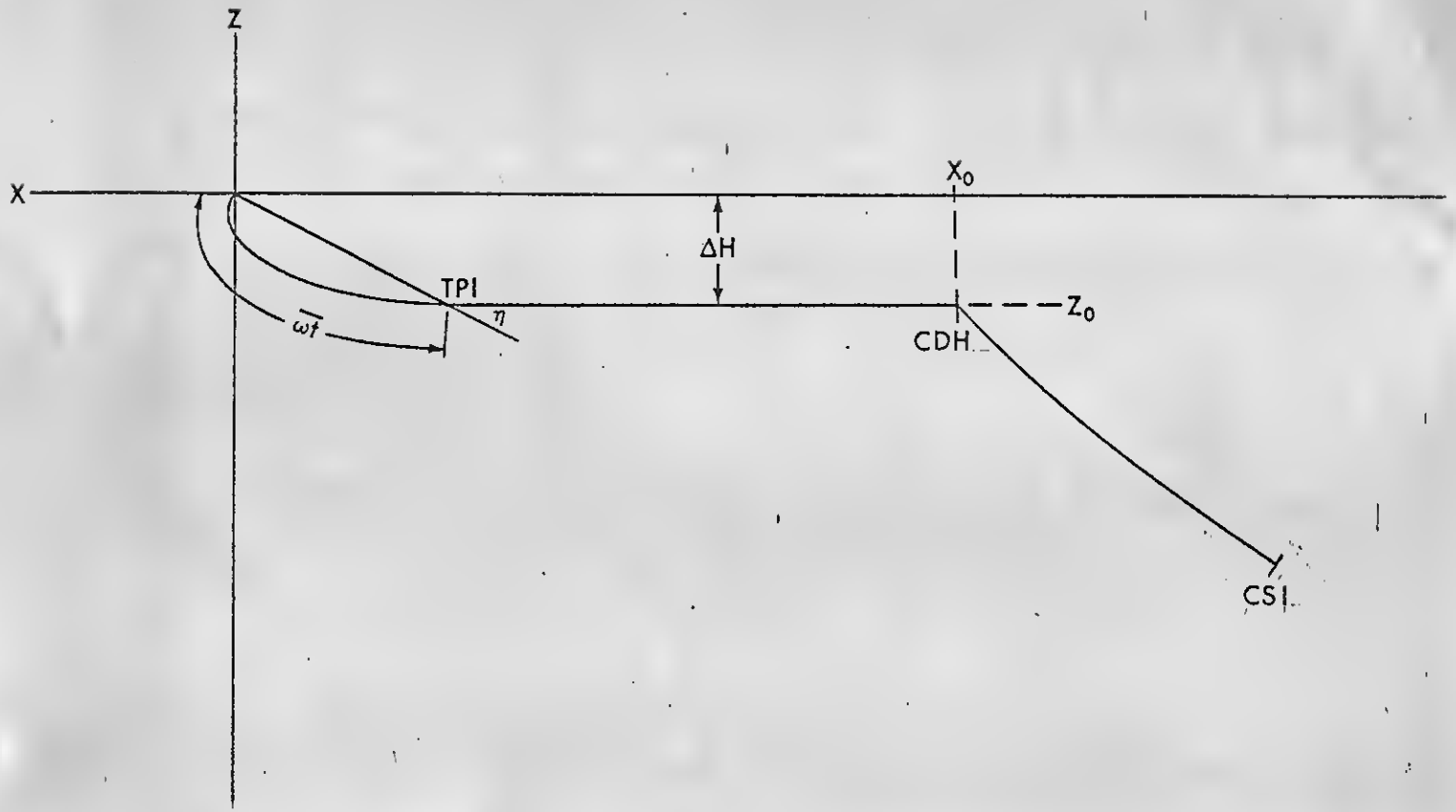


FIGURE 1 — CONCENTRIC FLIGHT PLAN

digital programs capable of solving for various maneuvers and providing dispersed trajectories have been constructed. Among the original techniques developed may be noted the derivation of maneuver functions for TPI and CDH, and the iterative use of the Clohessy-Wiltshire equations to compute transfer maneuver.

2.9.2 Development of Concentric Flight Plan

2.9.2.1 Transfer Phase

In order to clearly display the relative motion of the rendezvous trajectory, solve the linearized equations of relative motion (Eq. 4.1 - 9) for the required relative velocity

$$\dot{\mathscr{L}}_{TPI} = -B^{-1}(\bar{\omega}t)\, A(\bar{\omega}t)\, \mathscr{L}_{TPI} \qquad 2.1 - 1$$

therefore, in the intercept

$$\dot{\mathscr{L}}(t) = \left[A(\omega t) - B(\omega t) B^{-1}(\bar{\omega}t) A(\bar{\omega}t)\right] \mathscr{L}_{TPI} \qquad 2.1 - 2$$

These equations have the form

$$\begin{aligned} x(t) &= k_1(t)\, x_{TPI} + k_2(t)\, z_{TPI} \\ z(t) &= h_1(t)\, x_{TPI} + h_2(t)\, \dot{z}_{TPI} \end{aligned} \qquad 2.1 - 3$$

Since the shape of the trajectory in time is characterized by the ratio of x to z (Figure I),

$$\frac{x(t)}{z(t)} = \cot(\eta) = \frac{k_1 x_{TPI} + k_2 z_{TPI}}{h_1 x_{TPI} + h_2 z_{TPI}} = \frac{k_1 \cot(\eta_{TPI}) + k_2}{h_1 \cot(\eta_{TPI}) + h_2} \qquad 2.1 - 4$$

This equation clearly shows that the shape of an intercept is completely specified by choosing $\bar{\omega}t$ and η_{TPI}. By defining the parameters in advance for all intercepts, the shape is thus fixed, and learning is facilitated.

Practical choices of $\bar{\omega}t$ and η_{TPI} must reconcile fuel economy and ease of control. Actual simulation and flight experience has resulted in a choice of $\bar{\omega}t$ at about 130°. Shorter transfer intervals tend to be more costly in terms of transfer and braking ΔV and longer ones suffer from deleterious propagation of initial errors in estimate of transfer ΔV into miss distances at intercept. It also has proven possible to pick η_{TPI} such that the apparent inertial motion of the target in the latter part of the intercept is near zero and so that the transfer

from a coelliptical orbit is along the line of sight. For lunar orbit, this is about 26.6° and for earth orbit 27.5°. The practical advantage of this choice is that the terminal braking procedure is particularly simple: the pilot thrusts so as to null any apparent inertial motion of the target normal to the line of sight.

2.9.2.2 Coelliptical Phase

It is generally desired to constrain the time of transfer so as to satisfy operational constraints such as lighting and tracking. For given η_{TPI}, this reduces to the problem of bringing about the appearance of this angle at a selected instant. By far the simplest pre-transfer condition (with standard approach conditions) which allows this is the co-elliptical trajectory, wherein the differential altitude is constant throughout the phase. Again, from the equations 4.1 - 8a, for coellipticity

$$x(t)=x_o+\tfrac{3}{2}z_o\omega t$$
$$z(t)=z_o=\Delta h \qquad 2.2 - 1$$

or

$$\cot(\eta)=\cot(\eta_o)+\tfrac{3}{2}\omega t \qquad 2.2 - 2$$

Since at transfer $\eta(t) = \eta_{TPI}$, it is required that

$$\cot\eta_o=\cot\eta_{TPI}-\tfrac{3}{2}\omega t_{TPI}$$

The η_o point at which a maneuver is performed to bring about the co-elliptical condition is customarily designated CDH (Constant Delta-H). It is here implied that some maneuver(s) has been performed prior to CDH so as to bring about the required $\cot(\eta_o)= x_{CDH}/z_{CDH}$. In the Apollo program, this maneuver is called CSI (Concentric Sequence Initiation) and is customarily performed $\frac{1}{2}$ revolution before the CDH point. One should note that at first order, the appropriate value of η_o is a function only of the time from CDH to TPI.

The values of η_{TPI} and $\overline{\omega t}$ now in use represent compromise choices and may be inappropriate under some special conditions. If, for example, it is desired to rendezvous under circumstances where the transfer maneuver is subject to errors, the effect of these may be minimized by shortening $\overline{\omega t}$, as was done for the GT-10 passive rendezvous.

2.9.2.3 Targeting Phase

For a given approximate time of CDH, it is necessary to arrange

x_{CDH} , z_{CDH} such that their ratio ($\cot(\eta_{CDH})$) has the correct value. As there are two degrees of freedom and only one condition to be satisfied, the restriction can be made in several ways. If a value of z_{CDH} is given and the time of CSI fixed, the value of x_{CDH} is constrained and can be obtained by one two-axis maneuver at CSI or two single-axis maneuvers at different times. Alternatively, by letting z_{CDH} go unrestrained, the ratio can be obtained by means of a single one-axis maneuver. For reasons of efficiency, this is currently defined to be a horizontal maneuver n/2 periods before CDH. Under this plan, the value of Δh is a variable. A maneuver added to the sequence which has as its objective changing the trajectory so as to bring about a certain value of Δh is called a height adjust (N_H) maneuver.

It may be here noted that the out-of-plane problem can be shown to be uncoupled (to second order) from the in-plane rendezvous problem. Therefore, out-of-plane solutions may be handled separately in computation and application. In practice, as soon as the out-of-plane motion has been established, a maneuver is performed in conjunction with a scheduled in-plane maneuver (such as CSI) which has the effect of nulling the current out-of-plane rate, thus forcing a node to occur $\frac{1}{4}$ rev later. On arrival at this node, the velocity is again nulled, placing the active vehicle in-plane. This sequence may be repeatedly performed as better information is obtained until a satisfactory condition exists. Small residual errors in the out-of-plane direction are easily handled during terminal braking.

2.9.2.4 Operational Constraints

Considerations which determine the timing and arrangement of maneuvers are ground tracking, spacecraft-target visibility and maintenance of the trajectory plan. The most critical of these requirements constrain the time of transfer, which must occur so that the target is visible during tracking, and so that suitable tracking periods are available to each vehicle.. A detailed analysis of the requirements is contained in MSC Internal Note CF-R-69-6. In order to maintain the accuracy of the TPI maneuver and minimize the effects

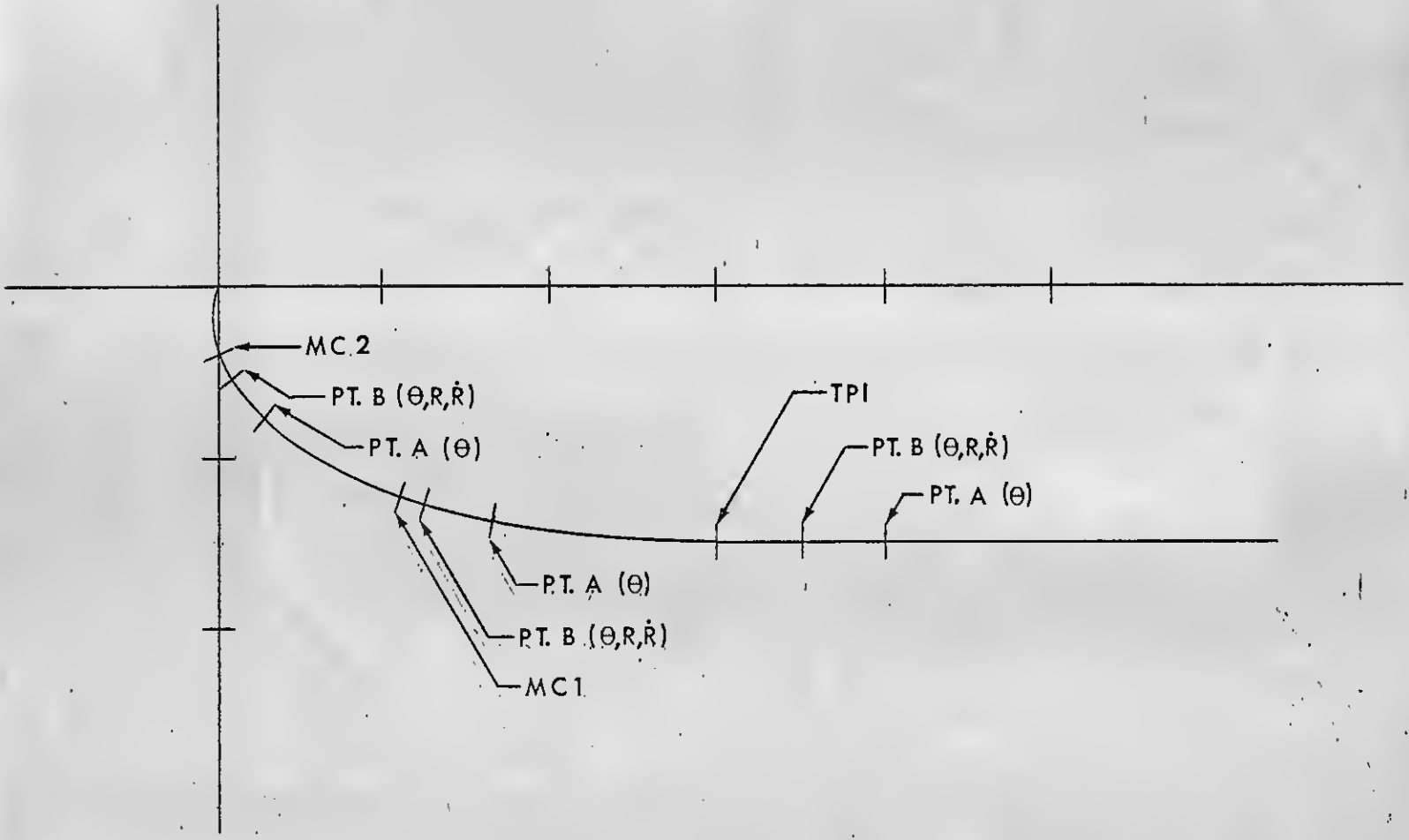


FIGURE 11

of trajectory estimation errors, it is generally undesirable for Δh to be smaller than some value. If an analysis of the likely trajectory dispersions indicates this may happen, or that it may grow unacceptably large, an N_H maneuver may be inserted into the flight plan to minimize these effects.

2.9.3 Backup Charts

2.9.3.1 TPI Backups

The method employed in computing a backup TPI solution is that of differencing the actual conditions observed just prior to TPI from the conditions required at TPI to achieve rendezvous $\overline{\omega t}$ of orbit travel later. The observables required for this technique are range (R) and range rate ($\dot{R}$) at a fixed time before TPI and two measurements of relative elevation angle (θ) at fixed times before TPI. The solution obtained is resolved into a velocity component along the line-of-sight (ΔV_{LOS}) at TPI and a velocity component normal to the line-of-sight (ΔV_{NOR}) at TPI. Figure II shows the measurement geometry.

The algorithms used to compute the velocity along the line-of-sight and the velocity normal to the line-of-sight are

$$\Delta V_{LOS} = \left[\frac{\dot{R}_{B(N)} + \Delta V_{LOS(N)}}{R_{B(N)}}\right] R_B - \dot{R}_B$$

$$\Delta V_{NOR} = \left[\frac{\Delta V_{NOR(N)}}{R_{B(N)}} - \frac{\Delta\theta_{(N)} - \Delta\theta}{\Delta t}\right] R_B \qquad 3.1 - 1$$

where

$$\Delta\theta_{(N)} = \theta_{B(N)} - \theta_{A(N)}$$
$$\Delta\theta = \theta_B - \theta_A$$

and Δt is the time between points A and B. The subscripts A and B indicate the time at which an observable was measured, A being at a time earlier than B. The subscripts N indicate those quantities which are referenced to the nominal trajectory.

This approach depends on the fact that the catchup rate ($\dot{X}$) of the active vehicle with respect to the passive vehicle is very nearly constant for coelliptic orbits, and is given by

$$\dot{x} = \tfrac{3}{2}\omega z_o$$

where ω is orbital angular rate and z_o is the differential altitude between the two vehicles (see the derivation in section 2.9.4).

It can be shown in the following manner that range rate is a function of catchup rate and relative elevation angle.

Letting

$$R = \sqrt{x^2 + z^2}$$

and differentiating, we get

$$\dot{R} = [\dot{x}x + \dot{z}z]/R \qquad 3.1 - 2$$

But, under the assumption that we are coelliptic, $\dot{z} = 0$ and $x/R \simeq \cos\theta$ so

$$\dot{R} = \dot{x}\cos\theta \qquad 3.1 - 3$$

Similarly, one can show that $\dot{\theta}$ is a function of elevation angle:

$$\theta = \tan^{-1}[z/x]$$

$$\dot{\theta} = [x\dot{z} - z\dot{x}]/R^2 \qquad 3.1 - 4$$

But $\dot{z}=0$, $\dot{x}=\frac{3}{2}\omega z_o$, $z=z_o$ so

$$\dot{\theta} = -\frac{3}{2}\omega\left(\frac{z_o}{R}\right)^2 = -\frac{3}{2}\omega\sin^2(\theta) \qquad 3.1 - 5$$

Hence, the values of θ, $\dot{R}$, and $\dot{\theta}$ at B can be used to infer $\dot{R}$ and $\dot{\theta}$ at TPI. These predicted values can thus be differenced from the required values to get a TPI solution.

The backup chart is graphical in nature and the data used to plot it are computed using a digital routine which generates the orbital parameters for a set of trajectories covering the region of expected dispersions about the nominal trajectory. The outputs of the routine are $(\dot{R}/R)_{REQ}$ and θ_{REQ} as functions of θ_B (the subscript $_{REQ}$ denoting the required values of the variables). For each of the θ_B's under consideration, coelliptic orbits are generated and advanced back in time by the appropriate Δt to get θ_A and hence, $\Delta\theta$. The trajectories are then advanced to TPI and the transfer maneuver is computed (see section 2.9.4.3), ΔV_{LOS} and ΔV_{NOR}. Then

$$\dot{R}/R = [\dot{R}_B + \Delta V_{LOS}]/R_B \qquad 3.1 - 6a$$

and

$$\Delta\theta_{REQ} = \Delta\theta - \left[\frac{\Delta V_{NOR}}{R_B}\right]\Delta t \qquad 3.1 - 6b$$

are computed and stored.

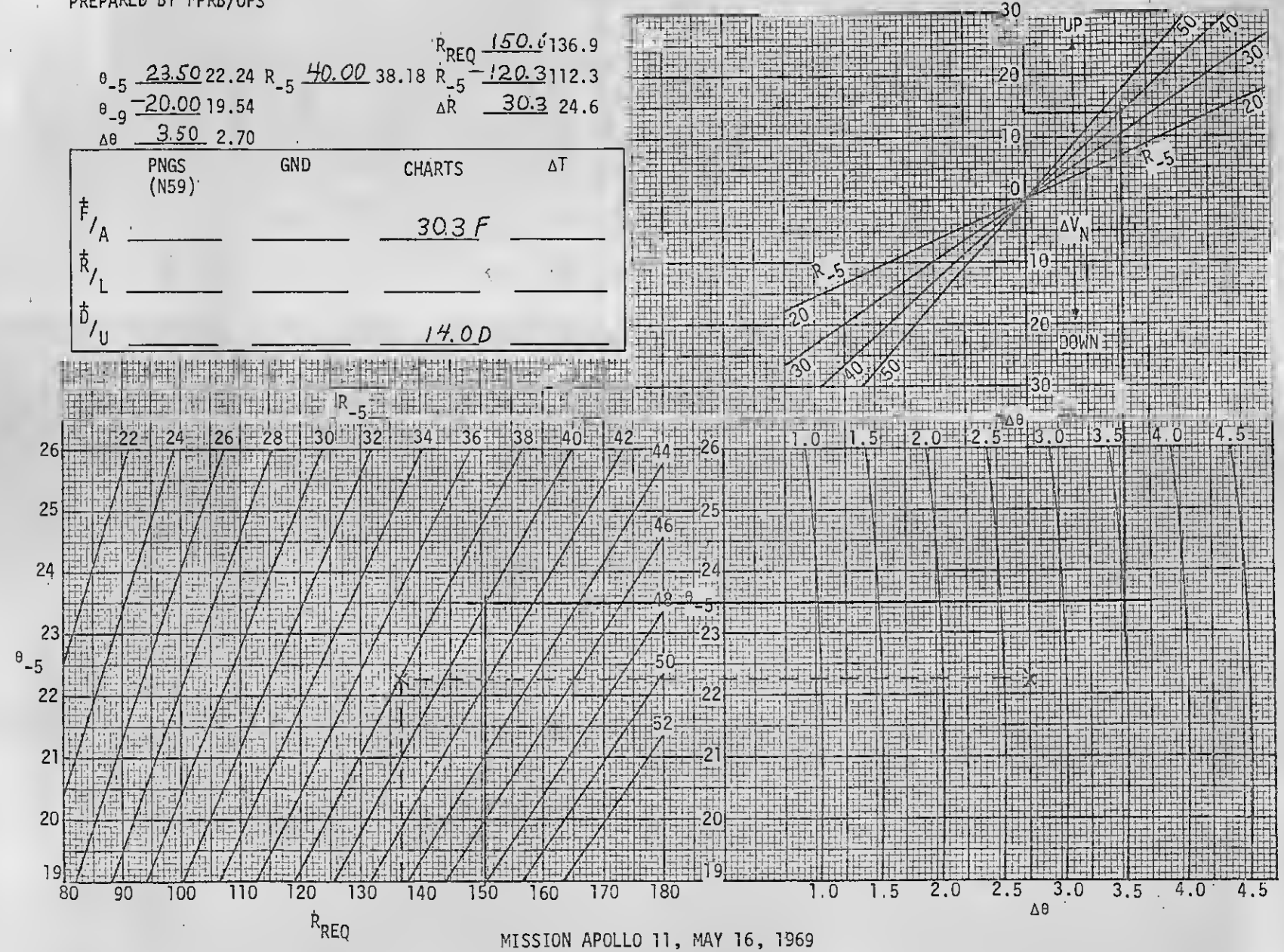
G MISSION TERMINAL PHASE INITIATION

PREPARED BY FPRB/OPS

θ_{-5}	23.50	22.24	R_{-5}	40.00	38.18
θ_{-9}	−20.00	19.54			
$\Delta\theta$	3.50	2.70			

R_{REQ}	150.0	136.9			
R_{-5}	−120.3	112.3			
ΔR	30.3	24.6			

	PNGS (N59)	GND	CHARTS	ΔT
†F/A			30.3 F	
†R/L				
†D/U			14.0 D	



MISSION APOLLO 11, MAY 16, 1969

FIGURE III

Graphically, the solution for ΔV_{LOS} is obtained by taking the product of $(\dot{R}/R)_{REQ}$ and R_B to get $\dot{R}_{REQ}$. $\dot{R}_B$ is then subtracted from $\dot{R}_{REQ}$ to get ΔV_{LOS}. Likewise, $\Delta\theta_{REQ}$ is subtracted from the actual $\Delta\theta$ and multiplied by $R_B/\Delta t$ to get ΔV_{NOR}. Figure III presents the TPI chart as flown on Apollo 11.

It should be noted here that the midcourse correction (MCC) charts are equivalent to TPI charts with the following exceptions:

1. The active vehicle is assumed to be on a collision course with respect to the passive vehicle rather than in a coelliptic orbit as in the nominal case.

2. The MCC burn is assumed to occur at the instant of the second measurement point, rather than a fixed time later.

With the exception of the above assumptions, the MCC charts are generated and used in exactly the same manner as the TPI chart. Later work on this problem has resulted in a TPI table, utilizing R, $\dot{R}$, θ, and the interval (Δt) between the last measurement and TPI.

2.9.3.2 CSI Backups

Since the ΔV for CSI is not, (in general) even approximately available as a function of the observables, it is necessary to adopt a somewhat different approach than that of the last section. Of the standard mathematical techniques for approximating an unknown function; the simplest is the Taylor Series. For CSI, only the in-plane problem is to be solved; therefore, four independent measurements will serve to constrain the problem. The simplest possibility is an equation of the form

$$\Delta V_{CSI} = \sum_n a_n q_1^n + b_n q_2^n + c_n q_3^n + d_n q_4^n \qquad 3.2 - 1$$

i.e., an uncoupled power series in the observables q_i. Should this assumption fail, it may be necessary to look for higher order cross terms in particular cases. For the situation where CSI occurs as the result of a nominal ascent from the lunar surface, and similar trajectories, this assumption works well.

To determine the constants a_n, b_n, c_n, d_n, a set of trajectories

spanning the envelope of expected dispersions is required, together with the ΔV_{CSI} for each case. In the solution of this problem, measurement time is held fixed to remove it as a variable, and the values of the observables, R and/or $\dot{R}$ are read at each measurement point. When these are substituted into 3.2 - 1, a system of simultaneous algebraic equations result

$$\begin{bmatrix} \Delta V_1 \\ \Delta V_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \Delta V_K \end{bmatrix} = \begin{bmatrix} q_1 & q_1^2 & \cdot & \cdot & q_2 & q_2^2 & \cdot & \cdot & q_3 & q_3^2 & \cdot & \cdot & q_4 & q_4^2 & \cdot & \cdot \\ \cdot & & & & & & & & & & & & & & & \\ \cdot & & & & & & & & & & & & & & & \\ \cdot & & & & & & & & & & & & & & & \\ \cdot & & & & & & & & & & & & & & & \\ \cdot & & & & & & & & & & & & & & & \\ \cdot & & & & & & & & & & & & & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ d_1 \\ d_2 \\ \cdot \end{bmatrix}$$

This system is generally well over determined, and writing

$$\vec{\Delta V} = Q\vec{C}$$

by the usual theory (2.9.4.6)

$$\vec{C} = [Q'Q]^{-1}Q'\vec{\Delta V}$$

is the least squares fit of $\vec{C}$ to the system. Having determined these constants, a table of partial sums is constructed where each entry is the partial sum of a particular value of a q_i for various values. Such a table is presented in Figure IV. For this case, the measurements are $\dot{R}$ at CSI -30^m, -20^m, -10^m and a range at -10^m. As each measurement is obtained, its corresponding factor is looked up in the table and logged in the space provided. At the last observation, these factors are summed to give the ΔV_{CSI}.

2.9.3.3 CDH Backups

Early work on the CDH problem by Flight Procedures Branch elicited the property that range rate in a near-coelliptical orbit is closely representable by a sinusoid of the form

$$\dot{R} = \dot{R}_o + \dot{R}_m \cos(\phi + \Delta\omega t) \qquad 3.3 - 1$$

By appeal to the linearized relative equation it can also be shown that

D MISSION CDH CHART

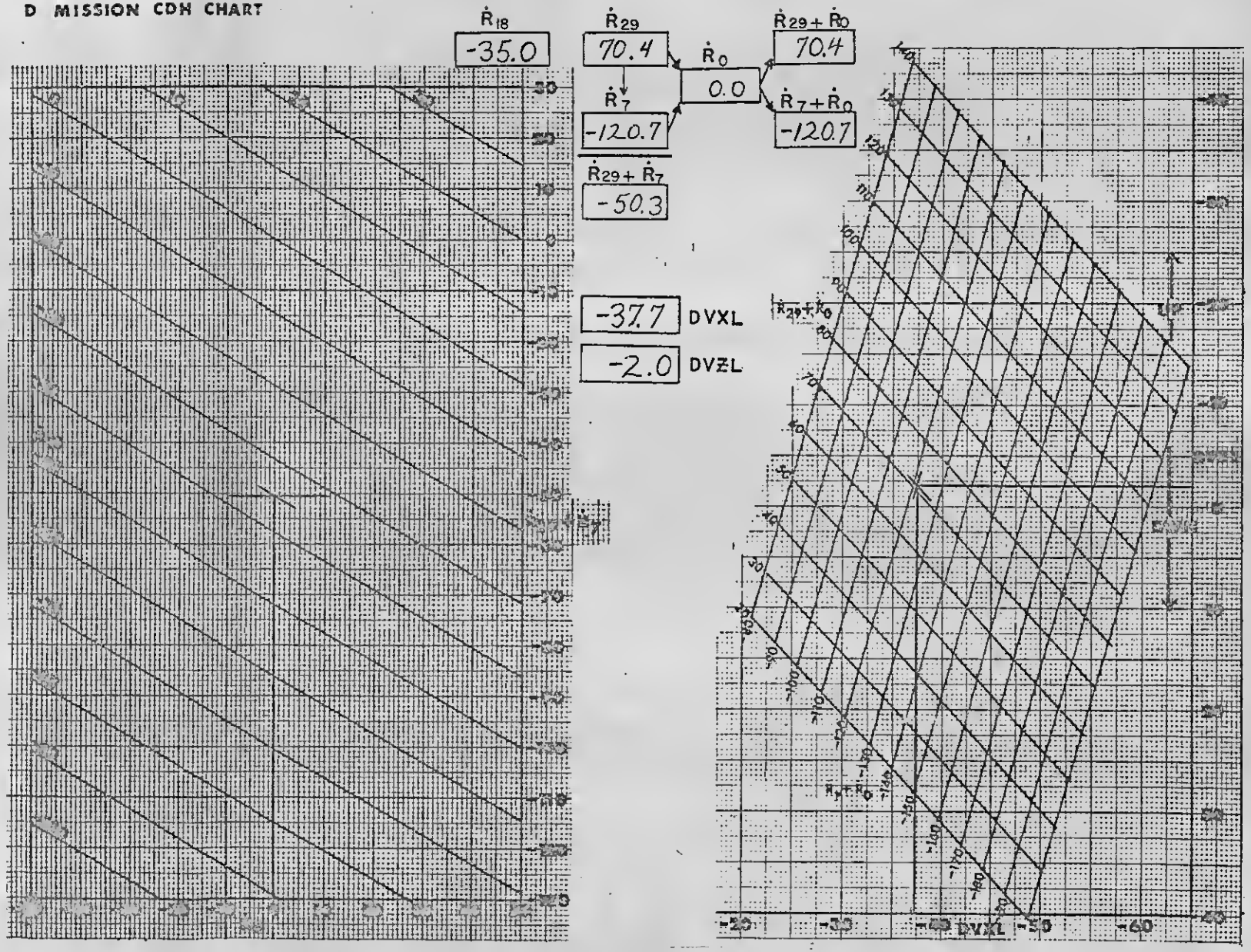


FIGURE V

$\dot{R}1$	F1	$\dot{R}2$	F2	$\dot{R}3$	F3	R3	F4
-240.0	247.3	-140.0	254.7	-70.0	72.4	120.0	15.2
-241.0	248.4	-141.0	256.6	-71.0	73.4	121.0	15.5
-242.0	249.4	-142.0	258.4	-72.0	74.4	122.0	15.7
-243.0	250.5	-143.0	260.2	-73.0	75.5	123.0	15.9
-244.0	251.5	-144.0	262.1	-74.0	76.5	124.0	16.1
-245.0	252.5	-145.0	263.9	-75.0	77.5	125.0	16.3
-246.0	253.6	-146.0	265.7	-76.0	78.6	126.0	16.5
-247.0	254.6	-147.0	267.6	-77.0	79.6	127.0	16.7
-248.0	255.7	-148.0	269.4	-78.0	80.6	128.0	16.9
-249.0	256.7	-149.0	271.3	-79.0	81.7	129.0	17.1
-250.0	257.8	-150.0	273.1	-80.0	82.7	130.0	17.3
-251.0	258.8	-151.0	274.9	-81.0	83.7	131.0	17.5
-252.0	259.9	-152.0	276.8	-82.0	84.8	132.0	17.7
-253.0	260.9	-153.0	278.6	-83.0	85.8	133.0	17.9
-254.0	262.0	-154.0	280.4	-84.0	86.9	134.0	18.1
-255.0	263.0	-155.0	282.3	-85.0	87.9	135.0	18.4
-256.0	264.1	-156.0	284.1	-86.0	88.9	136.0	18.6
-257.0	265.1	-157.0	286.0	-87.0	90.0	137.0	18.8
-258.0	266.2	-158.0	287.8	-88.0	91.0	138.0	19.0
-259.0	267.2	-159.0	289.7	-89.0	92.0	139.0	19.2
-260.0	268.3	-160.0	291.5	-90.0	93.1	140.0	19.4
-261.0	269.3	-161.0	293.4	-91.0	94.1	141.0	19.6
-262.0	270.4	-162.0	295.2	-92.0	95.2	142.0	19.8
-263.0	271.4	-163.0	297.0	-93.0	96.2	143.0	20.0
-264.0	272.5	-164.0	298.9	-94.0	97.2	144.0	20.2
-265.0	273.5	-165.0	300.7	-95.0	98.3	145.0	20.5
-266.0	274.6	-166.0	302.6	-96.0	99.3	146.0	20.7
-267.0	275.7	-167.0	304.4	-97.0	100.4	147.0	20.9
-268.0	276.7	-168.0	306.3	-98.0	101.4	148.0	21.1
-269.0	277.8	-169.0	308.1	-99.0	102.4	149.0	21.3
-270.0	278.8	-170.0	310.0	-100.0	103.5	150.0	21.5
-271.0	279.9	-171.0	311.9	-101.0	104.5	151.0	21.7
-272.0	281.0	-172.0	313.7	-102.0	105.6	152.0	21.9
-273.0	282.0	-173.0	315.6	-103.0	106.6	153.0	22.1
-274.0	283.1	-174.0	317.4	-104.0	107.7	154.0	22.4
-275.0	284.2	-175.0	319.3	-105.0	108.7	155.0	22.6
-276.0	285.2	-176.0	321.1	-106.0	109.7	156.0	22.8
-277.0	286.3	-177.0	323.0	-107.0	110.8	157.0	23.0
-278.0	287.4	-178.0	324.9	-108.0	111.8	158.0	23.2
-279.0	288.4	-179.0	326.7	-109.0	112.9	159.0	23.4
-280.0	289.5	-180.0	328.6	-110.0	113.9	160.0	23.6
-281.0	290.6	-181.0	330.4	-111.0	115.0	161.0	23.8
q_1	$\Sigma a_n q_1^n$	q_2	$\Sigma b_n q_2^n$	q_3	$\Sigma c_n q_3^n$	q_4	$\Sigma d_n q_4^n$

CSI BACKUP TABLE
MISSION G

TIME (Min)		NOMINAL
-30 $\dot{R}1$	-278.0	(-283.3)
-20 $\dot{R}2$	-170.0	(-173.9)
-10 $\dot{R}3$	-91.0	(- 94.0)
-10 R3	149.0	(154.1)
F1	287.4	(293.0)
+F3	94.1	(97.2)
	381.5	(390.2)
-F2	310.0	(-317.3)
	71.5	(72.9)
-F4	21.3	(- 22.4)
	50.2	(50.5)
+ΔΔVCSI	0.0	(0.0)
ΔVCSI	50.2	(50.5)

PREPARED by FPr8/OPS

MISSION APOLLO 11, MAY 16, 1969

FIGURE IV

$$\Delta V_x = -\frac{1}{A}\dot{R}\cos(\psi) \qquad \psi = \phi + 2\Delta\omega t + \alpha$$

$$\Delta V_z = \frac{2}{A}\dot{R}\sin(\psi) \qquad 3.3 - 2$$

where $A \simeq 4$ and $\Delta\omega t$ is the interval between measurements, α the interval between the last measurement and CDH. Since the CDH maneuver depends only on the relative velocity and current Δh, 3 independent measurements suffice to solve the problem as reflected by 3.3 - 1. From the equations 3.3 - 1 and 3.3 - 2, a nomographic solution of the CDH problem may be constructed as presented in Figure V. The somewhat lengthy derivation of the actual results is presented in Section 2.9.4.2.

Later analysis directed at the solution of the CSI problem has resulted in the techniques of the last section being applied to the construction of CDH backup tables. Their preparation and use is exactly the same as for CSI.

2.9.3.4 Performance Analysis

Once backup charts have been generated, a statistical analysis is done to determine exactly how well they can be expected to perform. Data for the analysis are generated with a routine which executes a large number of rendezvous, and calculates statistical data on the parameters of interest.

The runs for the analysis generally start approximately 40 minutes prior to CSI, and are run through intercept. A total of 300 runs are generally made, broken up into four groups, each run having randomly dispersed initial conditions. The first of these groups consists of 100 runs, made with all applicable random errors, biases, and drifts. It had previously been determined that 100 runs would yield statistically meaningful results. This was done by plotting some of the randomly distributed variables and noting the shape of the resulting bell curve. Studies of this type also indicated that 50 runs would be the absolute minimum number that could be made, and still yield meaningful results.

A second group also consists of 100 runs. These runs are identical to the first set, except for the fact that braking and line-of-

TABLE 1

Differences Between Chart Solutions With and Without Errors and Conic Solutions

Maneuver	Average		Mean		Standard Deviation	
	Set D ft/sec	Set A ft/sec	Set D ft/sec	Set A ft/sec	Set D ft/sec	Set A ft/sec
*CSI $\Delta\Delta V_H$	0.0	.77	0.0	- .02	0.0	.95
CDH $\Delta\Delta V_V$	1.11	1.54	1.11	1.05	1.21	1.83
CDH $\Delta\Delta V_H$	.25	.46	- .25	- .33	.34	.60
TPI $\Delta\Delta V_{LOS}$	1.34	2.40	-1.04	- .76	1.45	3.02
TPI $\Delta\Delta V_N$	.37	2.41	- .29	- .44	.54	3.12
MCC1 $\Delta\Delta V_{LOS}$	1.59	2.65	-1.59	-1.83	.91	2.80
MCC1 $\Delta\Delta V_N$	1.34	2.16	1.34	1.47	.37	2.26
MCC2 $\Delta\Delta V_{LOS}$	.84	1.48	- .82	- .97	.59	1.48
MCC2 $\Delta\Delta V_N$	.50	1.29	.50	.85	.16	1.77

The data in Table 3-3 listed under Set D represents the theoretical error inherent in the charts, while the data listed under Set A represents the total expected error, including theoretical error, system errors, and execution errors.

*The value for CSI $\Delta\Delta V$ represents the difference between CSI ΔV computed with sensor and reading errors, and the value for CSI ΔV computed without sensor and reading errors.

sight control are omitted. This is done in order to obtain data on miss distance at closest approach.

The remaining two sets consist of 50 runs each, and are made without any random errors, biases, or drifts. The purpose of these two sets was to get baseline data on chart performance. Again, the second set was identical to the first, with the exception that braking and line-of-sight control were omitted to establish data on miss distance.

During the analysis, one of the basic parameters which was looked at, was the accuracy of the chart solutions. Table 1[1] is representative of the type of data which were derived. In addition, data concerning miss distance at closest approach, total fuel used in the rendezvous, and arrival time at TPI were also derived.

2.9.3.5 Onboard Rendezvous Evaluation

Evaluation of the progress of the rendezvous is one of the primary crew functions in manual spaceflight. Much of the analysis done by Flight Procedures Branch has been directed toward the provision of "rule of thumb" statements about the behavior of maneuver solutions following trajectory dispersions. For this purpose, the linearized equations of Section 2.9.4 constitute a powerful analytical tool. Let the coordinate system of Section 2.9.4 be fixed on the nominal spacecraft so that motion of the actual to nominal may be compared. As an example, consider an insertion dispersed behind the nominal phase angle (up range distance). To first order, the equations say that this dispersion will propagate to a similar off-nominal position at each point, in particular at CDH. If the phase angle was too large, the S/C is too far up range and a larger Δh will be necessary to arrive at η_{TPI} at the right time. For a nominal rendezvous profile, this will result in a lower ΔV_{CSI}, since CSI is raising pericythion from about 10 nm to about 45 nm. Similar remarks apply to a horizontal overspeed at insertion. The equations show that after nearly 1 rev, i.e., at CDH, an overspeed will place the spacecraft up range of its

1. MSC Internal Note No. CF-R-69-20, Apollo Mission F Performance Analysis of Rendezvous Charts, April 29, 1969.

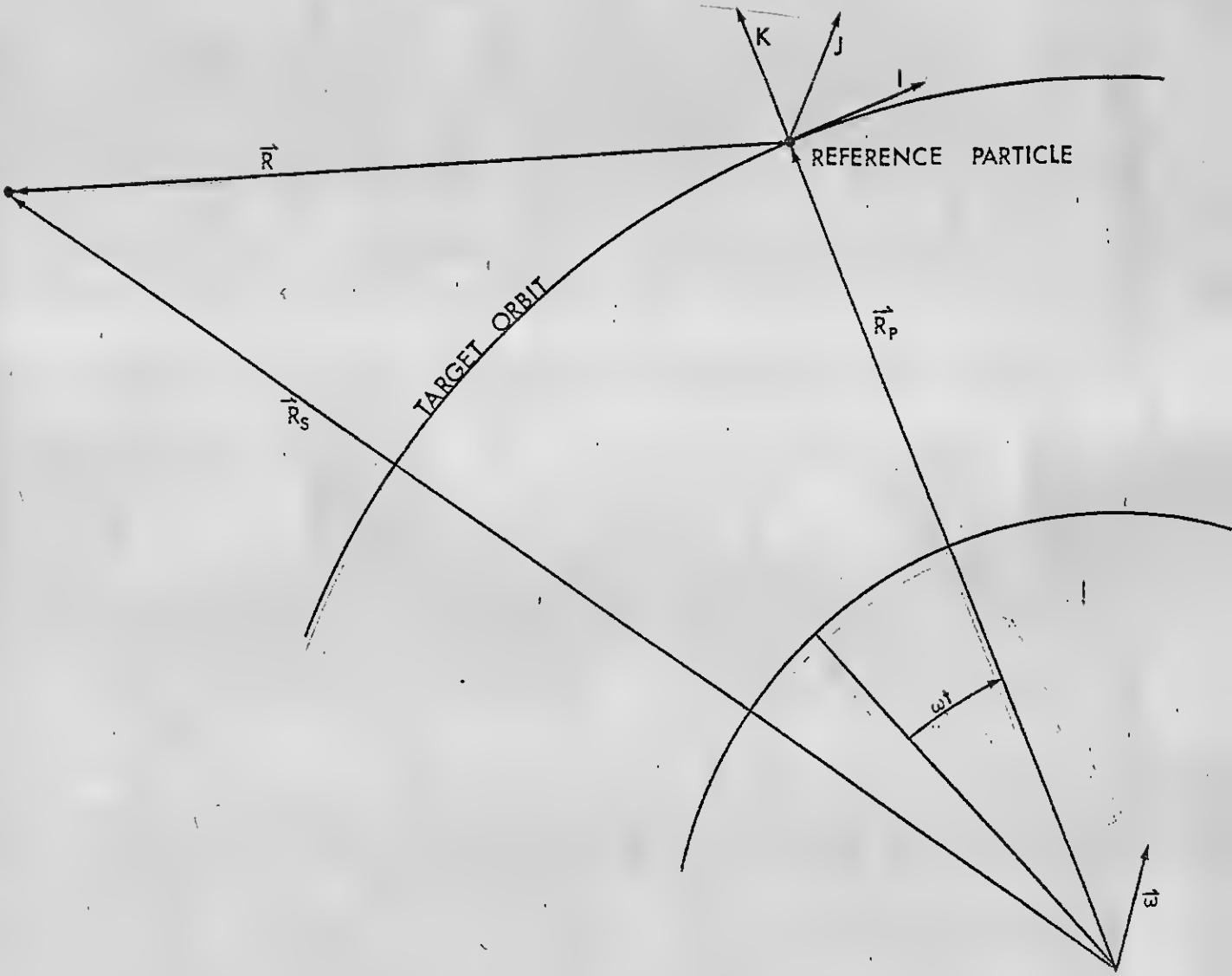


FIGURE VI — TWO PARTICLES IN ORBIT AROUND A SPHERICAL PLANET

nominal position. Converse statements apply to too small an insertion phase angle or an insertion underspeed. If the insertion error is in altitude rate, the actual spacecraft after one rev is nearly coincidental with the nominal one, but has an altitude rate error nearly equal to that at insertion. Thus the vertical component of the CDH maneuver will be perturbed to remove it, but the resulting Δh will be little affected. Provided the pilot has information on the dispersions resulting from a particular case, he can infer the trend of his maneuver solutions in comparison to nominal.

By similar means, the effect of an incorrect CSI maneuver in arrival time at η_{TPI} may be gauged. Since CSI is strictly horizontal, one need only consider dispersions in this axis and it is instantly apparent that an overburn causes late arrival and conversely. For lunar orbit, this is about 4 min/fps.

2.9.4 Mathematical and Technical Appendix

2.9.4.1 The Linearized Relative Equations

All rendezvous problems have in common two basic requirements; (1) knowledge of the relative motion between the interceptor and target, and (2) a plan of maneuvers for the interceptor which results in a terminal condition of zero relative velocity at a small distance.

In order to facilitate analysis and understanding of the rendezvous problem, it is useful to develop a set of equations describing the relative motion of one vehicle with respect to another when they are reasonably close. This follows the standard treatment, and two assumptions will be adherred to in the discussion:

1. The orbit of the reference particle is near circular.
2. The distance between them is small compared to the radius vector of the reference particle.

In Fig. IV, construct a coordinate system fixed on the reference particle: $\hat{K} = \vec{R}_P/|\vec{R}_P|$, $\hat{J} = \hat{K} \times \vec{V}_P / |\hat{K} \times \vec{V}_P|$, $\hat{I} = \hat{J} \times \hat{K}$ and defining

$$\mathbb{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \Omega = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

and noting Ω and R_P constant by assumption 1., the derivative operator for such a vector is

$$\frac{d\vec{A}}{dt} = \left[1\frac{\partial}{\partial t} + \Omega\right]\vec{A}$$

$$\frac{d^2\vec{A}}{dt^2} = \left[1\frac{\partial^2}{\partial t^2} + 2\Omega\frac{\partial}{\partial t} + \Omega^2\right]\vec{A}$$

Using these equations and writing R_s with respect to R_P find Newton's Law for particle S

$$\ddot{\vec{R}}_S = \frac{\partial^2\vec{R}_S}{\partial t^2} + 2\Omega\frac{\partial\vec{R}_S}{\partial t} + \Omega^2\vec{R}_S = -\frac{\mu}{R_S^3}\vec{R}_S \qquad \vec{R}_S = \begin{bmatrix} x \\ y \\ z+R_P \end{bmatrix} \qquad 4.1 - 2$$

$$= \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} + 2\omega\begin{bmatrix} \dot{z} \\ 0 \\ -\dot{x} \end{bmatrix} - \omega^2\begin{bmatrix} x \\ 0 \\ z+R_P \end{bmatrix} = -\frac{\mu}{R_S^3}R_S = -\frac{\mu}{R_P^3}\left(\frac{R_P}{R_S}\right)^3\vec{R}_S \qquad 4.1 - 3$$

Now examine

$$\left(\frac{R_S}{R_P}\right)^{-3} = \frac{\left[x^2+y^2+(z+R_P)^2\right]^{-\frac{3}{2}}}{R_P^{-3}} = \left[\left(\frac{x}{R_P}\right)^2 + \left(\frac{y}{R_P}\right)^2 + \left(1+\frac{z}{R_P}\right)^2\right]^{-\frac{3}{2}} \simeq 1 - \frac{3z}{R_P} \qquad 4.1 - 4$$

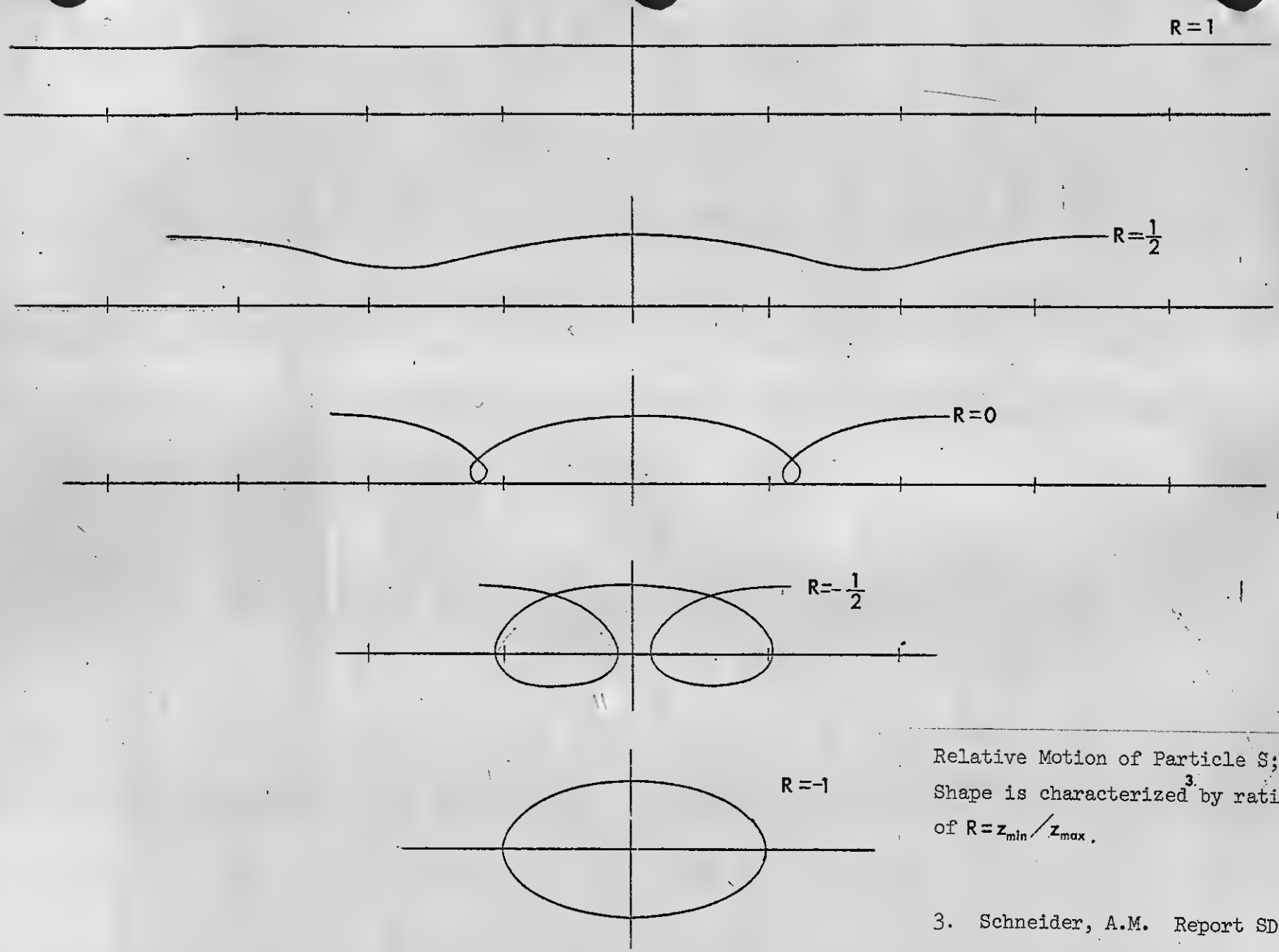
if terms of second order are ignored by assumption 2. This is the first of two approximations to be made in the interest of obtaining a linear system. Careful note should be taken of the implied limitations on the result. The second approximation is gotten by assuming terms of the form

$$\frac{3xz}{R_P} \quad , \quad \frac{3yz}{R_P} \quad , \quad \frac{3z^2}{R_P}$$

are also negligble, thus to get

$$\left(\frac{R_P}{R_S}\right)^3\vec{R}_S \simeq \begin{bmatrix} x \\ y \\ R_P-3z \end{bmatrix} \qquad 4.1 - 5$$

further identify $-\frac{\mu}{R_P^3} = -\omega^2$ and write



Relative Motion of Particle S;
Shape is characterized[3] by ratio
of $R = z_{min} / z_{max}$.

3. Schneider, A.M. Report SDC-1-69

FIGURE VII

$$\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix}+2\omega\begin{bmatrix}\dot{x}\\ 0\\ -\dot{x}\end{bmatrix}+\omega^2\begin{bmatrix}0 & -X\\ \not{0} & 0\\ -(z+R_p)\end{bmatrix}=-\omega^2\begin{bmatrix}x\\ y\\ (z+R_p)-3z\end{bmatrix} \qquad 4.1-6$$

and finally get the equations of equations

$$\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix}+2\omega\begin{bmatrix}\dot{z}\\ 0\\ -\dot{x}\end{bmatrix}+\omega^2\begin{bmatrix}0\\ y\\ -3z\end{bmatrix}=0 \qquad 4.1-7$$

These simultaneous equations are readily integrated to give

$$\begin{bmatrix}x\\ z\end{bmatrix}=\begin{bmatrix}1 & -6(\omega t-\sin\omega t)\\ 0 & (4-3\cos\omega t)\end{bmatrix}\begin{bmatrix}x_o\\ z_o\end{bmatrix}+\begin{bmatrix}\frac{4}{\omega}(\sin\omega t-\frac{3}{4}\omega t) & -\frac{2}{\omega}(1-\cos\omega t)\\ \frac{2}{\omega}(1-\cos\omega t) & \frac{\sin\omega t}{\omega}\end{bmatrix}\begin{bmatrix}\dot{x}_o\\ \dot{z}_o\end{bmatrix} \qquad 4.1-8a$$

$$y=y_o\cos\omega t+\frac{\dot{y}_o}{\omega}\sin\omega t \qquad 4.1-8b$$

Of immediate interest is the fact that the out-of-plane motions are uncoupled from the in-plane. Thus the out-of-plane problem may be treated separately. Writing equations 4.1 - 8a in the form

$$\mathcal{S}=A[\omega t]\mathcal{S}_o+B[\omega t]\dot{\mathcal{S}}_o \qquad 4.1-9$$

and considering two possible initial states for the particle

$$\mathcal{S}_o^1,\dot{\mathcal{S}}_o^1 \quad \text{and} \quad \mathcal{S}_o^2,\dot{\mathcal{S}}_o^2$$

it can be shown that because of the linearity of the equations, the state at any time due for a combination of initial perturbations is the same as the sum of the states at that time due to the perturbations applied separately, i.e.:

$$\begin{aligned}\mathcal{S}&=A\left[\mathcal{S}_o^1+\mathcal{S}_o^2\right]+B\left[\dot{\mathcal{S}}_o^1+\dot{\mathcal{S}}_o^2\right]\\&=\left[A\mathcal{S}_o^1+B\dot{\mathcal{S}}_o^1\right]+\left[A\mathcal{S}_o^2+B\dot{\mathcal{S}}_o^2\right]\\&=\mathcal{S}^1+\mathcal{S}^2\end{aligned}$$

It may be guessed from intuition and stated from experience that the requirement that the reference particle orbit be circular may be relaxed somewhat, provided that x is interpreted as down-range curvilinear distance along the orbit arc, and z as normal distance from the point x to the particle R_s. It should be stressed that even in regions where the assumptions leading to linearity are not strictly true, the equations still provide a useful indication of the relative motion to be expected.

2.9.4.2 CDH Equations

It is desired to relate the velocity maneuvers at CDH to the observable, range rate. This will be measured at selected times before CDH.

From Equation 3.1 - 2 of Section 3.2

$$R=[\dot{x}x + \dot{z}z]/R=\dot{x}\cos\eta + \dot{z}\sin\eta$$

Since at large ranges $\cos\eta \simeq 1$ and $\sin\eta \simeq 0$,

$$\dot{R} \simeq \dot{x} \qquad 4.2 - 1$$

Equations 4.1 - 8a, by defining

$$b=2[2z_o+\dot{x}_o/\omega]$$

$$c=[2\dot{z}_o/\omega-x_o]$$

$$\rho=-\left[(\dot{z}_o/\omega)^2-(2\dot{x}_o/\omega)+3z_o)^2\right]$$

$$\gamma=\tan^{-1}[\dot{z}_o/(2\dot{x}_o+3\omega z_o)]$$

can be written

$$x=c+\tfrac{3}{2}b\varphi-2\rho\sin(\varphi+\gamma) \qquad \varphi=\omega t$$

$$z=b-\rho\cos(\varphi+\gamma)$$

Therefore,

$$\dot{x}=\tfrac{3}{2}b\dot{\varphi}-2\rho\dot{\varphi}\cos(\varphi+\gamma)$$

$$\dot{z}=\rho\dot{\varphi}\sin(\varphi+\gamma) \qquad 4.2 - 2$$

For coellipticity, it is required

$$\dot{x}=\tfrac{3}{2}\omega z_o$$

$$\dot{z}=0$$

So that the components of CDH at any point are

$$\begin{aligned}\Delta V_x = x_{REQ} - x_A &= \tfrac{3}{2}\dot{\varphi}z-\tfrac{3}{2}\dot{\varphi}b+2\rho\dot{\varphi}\cos(\varphi+\gamma)\\ &=\tfrac{3}{2}\dot{\varphi}b-\tfrac{3}{2}\dot{\varphi}\rho\cos(\varphi+\gamma)-\tfrac{3}{2}b\dot{\varphi}+2\rho\dot{\varphi}\cos(\varphi+\gamma)\\ &=\tfrac{1}{2}\dot{\varphi}\rho\cos(\varphi+\gamma)\end{aligned} \qquad 4.2 - 3$$

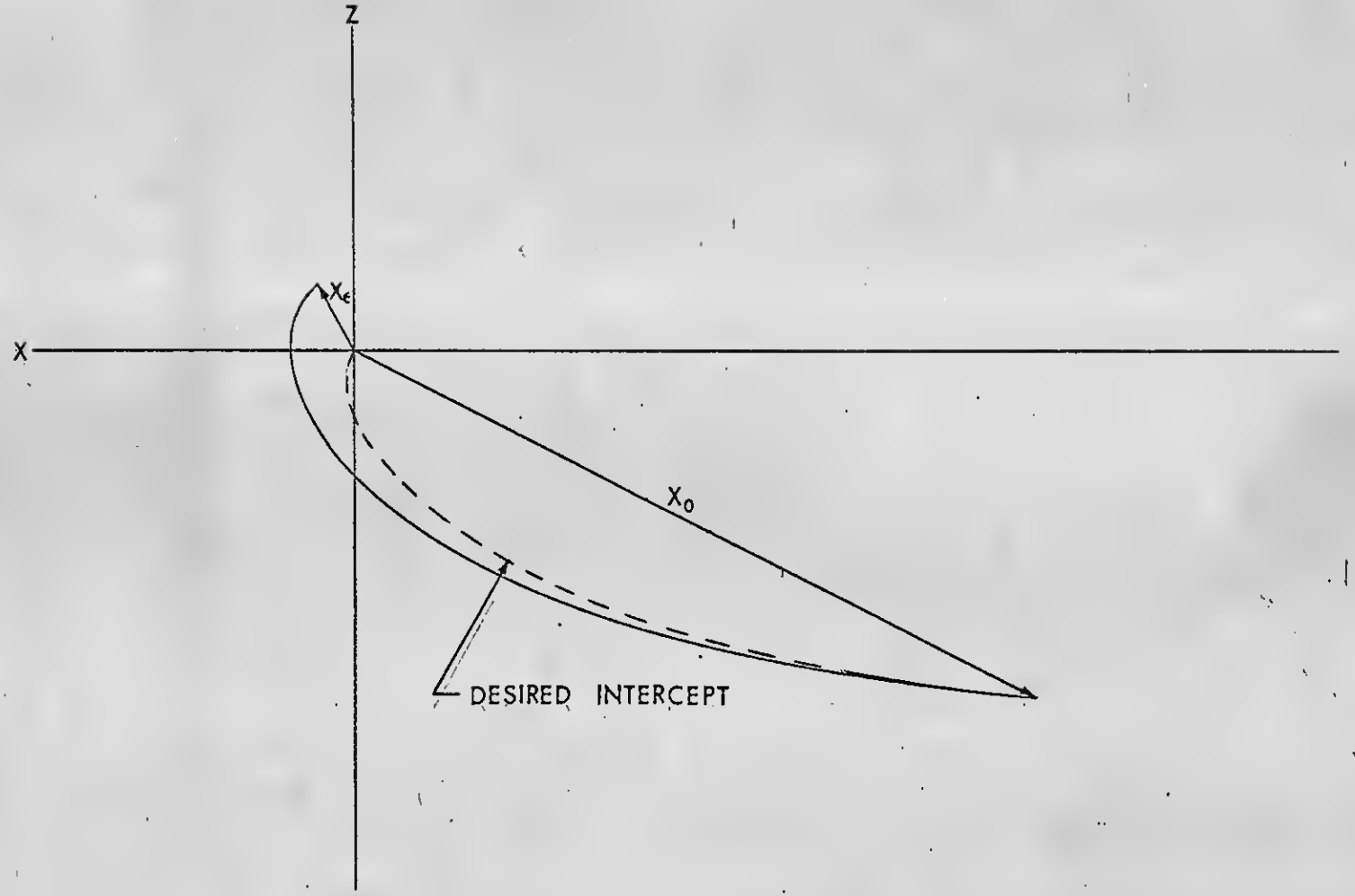


FIGURE VIII RENDEZVOUS PROBLEM

$$\Delta V_z = z_{REQ} - z_A = -\rho\dot{\varphi}\sin(\varphi+\gamma) \qquad 4.2. - 4$$

Taking the first equation of 4.2 - 2 and defining

$$\dot{R}_c = \tfrac{3}{2}b\dot{\varphi}$$

$$\dot{R}_m = -2\rho\dot{\varphi}$$

have

$$\dot{R} = \dot{R}_c + \dot{R}_m \sin(\varphi+\gamma)$$

as the equation to be solved. Taking t=0 at the first measurement noting that three will be required to determine $\dot{R}_c$, $\dot{R}_m$, γ :

$$\dot{R}_o = \dot{R}_c + \dot{R}_m \sin(\gamma)$$

$$\dot{R}_1 = \dot{R}_c + \dot{R}_m \sin(\varphi+\gamma)$$

$$\dot{R}_2 = \dot{R}_c + \dot{R}_m \sin(2\varphi+\gamma)$$

After considerable manipulation, the solution of this simultaneous set can be obtained as

$$\sin\gamma = (\dot{R}_o - \dot{R}_c)/\dot{R}_m$$

$$\dot{R}_c = [\dot{R}_o + \dot{R}_2 - 2\dot{R}_1\cos\varphi]/2(1-\cos\varphi) \qquad \dot{R}_{E_o} = \dot{R}_o - \dot{R}_c$$

$$\dot{R}_m = [\dot{R}_{E_o}^2 + \dot{R}_{E_2}^2 - 2\dot{R}_{E_2}\dot{R}_{E_o}\cos 2\varphi]/\sin 2\varphi \qquad \dot{R}_{E_2} = \dot{R}_2 - \dot{R}_c$$

Therefore, if α is the elapsed central angle between the last measurement and the time of CDH, the maneuvers are given by

$$\Delta V_x = -\tfrac{1}{4}\dot{R}_m \cos(2\varphi+\gamma+\alpha)$$

$$\Delta V_z = \tfrac{1}{2}\dot{R}_m \sin(2\varphi+\gamma+\alpha)$$

2.9.4.3 Digital Computation of Transfer

In general, a transfer problem consists of finding the velocity maneuver required to go between given points subject to various constraints.

It is desired to solve the time of flight problem, in a manner appropriate for use with a digital computer.

As in figure VIII, let a Clohessy-Wiltshire (C-W) frame be attached to the passive particle, with ω, $\dot{X}_o$, X_o given. Then from the last seetion

$$X(t) = A(\omega t)X_o + B(\omega t)\dot{X}_o \qquad 4.3 - 1$$

Where $A(\omega t)$, $B(\omega t)$ are the C-W matrix functions of time. For intercept require $X(t) = 0$ after a time t:

$$A(\omega t)X_o + B(\omega t)\dot{X}_r = 0 \qquad 4.3 - 2$$

$$\dot{X}_r = -B^{-1}(\omega t)A(\omega t)X_o \qquad 4.3 - 3$$

Since maneuvers will be done in the active vehicle local vertical frame, compute

$$\vec{\dot{R}}_s = \partial/\partial t(R_t + X_o) + \omega \times (R_t + X_o)$$

$$= \dot{R}_t \hat{R}_t + \dot{X}_r + \omega \times (\vec{R}_t + X_r)$$

$$= \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \vec{\dot{R}}_s \qquad \phi = \text{Central angle at intercept.}$$

in the local vertical frame of the active vehicle.

If this $\dot{X}_r$ is applied, and the active vehicle precisely advanced along the resulting orbit, it will not, in general, intercept the origin. This is due to the approximate nature of the C-W equations, which results in an X_ϵ residual at the time for intercept. At this juncture, one may proceed in two similar but slightly different ways, both of which will be discussed.

1. Offset Targeting

Note that the general solution of the C-W equations for X_r is

$$\dot{X}_r^1 = B^{-1}(\omega t)X(t) - B^{-1}(\omega t)A(\omega t)X_o \qquad 4.3 - 4$$

In which $X(t)$ was set equal to zero for intercept. Since the solution obtained is known to miss by X_ϵ , retarget for $-X_\epsilon$ and reasonably expect to hit in between X_ϵ and $-X_\epsilon$, i.e., near the origin:

$$\dot{X}_r^2 = -B^{-1}(\omega t)X_\epsilon^1 - B^{-1}(\omega t)A(\omega t)X_o \qquad 4.3 - 5$$

This may again miss by X_ϵ^2 say, and a better solution may be obtained as before by aiming for $-X_\epsilon^2$:

$$\dot{X}_r^3 = -B^{-1}(\omega t)[X_\epsilon^1 + X_\epsilon^2] - B^{-1}(\omega t)A(\omega t)X_o \qquad 4.3 - 6$$

and so on

$$\dot{X}_r^n = -B^{-1}(\omega t)\sum_{i=0}^{n-1} X_\epsilon^i - B^{-1}(\omega t)A(\omega t)X_o \qquad 4.3 - 7$$

where $X_\epsilon^0 = 0$

one may consider this an iteration on initial displacement by noting that the operation A(wt) transforms an initial displacement into a final displacement:

$$X_f = A(\omega t)X_i \qquad 4.3 - 8$$

thus targeting for $\sum X_\epsilon^i$ is the same as perturbing X_o by $A^{-1}(\omega t)\sum X_\epsilon^i$ and leads to the equation

$$\dot{X}_r^n = -B^{-1}(\omega t)A(\omega t)[X_o + A^{-1}\sum_{i=0}^{n-1} X_\epsilon^i] \qquad 4.3 - 9$$

where A^{-1} may be considered a matrix gain factor determining how an intercept error should perturb the initial displacement to effect convergence.

2. Successive C-W Frames

Attach a C-W frame to the active particle and imagine that at intercept, the passive vehicle has a position $-X_\epsilon$ in this frame. Then <u>in this frame</u> compute a velocity maneuver which would send a particle to this point from its origin in an $(\omega t)' = \Delta\nu_s$. Where $\Delta\nu_s$ is the change in the true anomaly of this new frame from transfer to intercept:

$$\dot{X}_\epsilon = -B^{-1}(\omega t)' X_\epsilon$$

Add this correction to the initial guess and prceed as before:

$$\dot{X}_r^n = -\sum_{i=0}^{n-1} B^{-1}(\omega t)_i' X_\epsilon^i - B^{-1}(\omega t)A(\omega t)X_o$$

in effect constituting a new C-W frame at X_ϵ for each attempt, computing a solution to drive a particle to $-X_\epsilon$, and summing these.

Note that the propriety of these two methods lies in the tendency of the C-W equations to give exact results as the distance between the particles approaches zero.

2.9.4.5 CSI/CDH

Considering first the CDH maneuver, there are several ways to rigorously define coellipticity. The one in use for Apollo results in alignment of the semi-major axes and no variation in Δh to first order. In terms of the eccentric anomoly for each vehicle

$$R_1=a_1(1-e_1 \cos E_1)$$

$$R_2=a_2(1-e_2 \cos E_2)$$

therefore

$$\Delta h=R_2-R_1=a_2-a_1+a_1w_1 \cos E_1-a_2e_2 \cos E_2$$

$$=\Delta h_0+(a_1e_1 \cos E_1-a_2e_2 \cos E_2)$$

In order for there to be no variation in Δh, it must be true that $E_1=E_2-\phi$. (ϕ = phase angle) so that when the vehicle radius vectors are coincident $E_2=E_1$:

$$\Delta h_{CDH}=\Delta h_0+(a_1e_1-a_2e_2) \cos E_{CDH}$$

Then if $a_1e_1=a_2e_2$, h will be constant. It can be shown that for given true anomoly of both vehicles the same, the variation in Δh is the order of $e_1^2-e_2^2$.

For CSI, a straight iterative procedure is used wherein a trial velocity for a CSI is varied to compute a numerical partial derivative of change in η_{TPI} with respect to change in ΔV_{CSI}.

2.9.4.6 Multiple Linear Regression

Let an over-determined system of equations be given

$$RC = \Delta V + \epsilon$$

where R is an m x n m > n known matrix and C is an n-row by 1-column. Since the system is overdetermined, $\vec{\epsilon}$ will not in general by zero, hence let us seek to minimize its magnitude:

$$\vec{\epsilon} = R\vec{C} - \Delta\vec{V} \qquad 4.6 - 2$$

$$\epsilon^2 = \vec{\epsilon}'\vec{\epsilon} = (\vec{C}'R' - \Delta\vec{V}')(R\vec{C} - \Delta\vec{V}) \qquad 4.6 - 3a$$

$$= \vec{C}'R'R\vec{C} + (\vec{C}'R'\Delta\vec{V} + \Delta\vec{V}'R\vec{C}) + \Delta\vec{V}'\Delta\vec{V} \qquad 4.6 - 3b$$

An extremum (hopefully minimum) of ϵ^2 will be found when the variation $\delta\epsilon^2$ consequent upon a variation δC is zero. i.e. require:

$$\delta\epsilon^2 = 2[\delta\vec{C}'R'R\vec{C} - \delta\vec{C}'R'\Delta\vec{V}] = 0 \qquad 4.6 - 4$$

since $\delta\Delta\vec{V} = \delta R = 0$. Now note that if α, a scalar is

$$\alpha = \vec{K}'\vec{K} = (\vec{K}'\vec{K})' = \alpha'$$

Since $\delta\epsilon^2$ is a scalar conclude that each term of 4.6 - 4 is scalar and substitute for $\vec{C}'R'R\delta\vec{C}$ and $\Delta\vec{V}'R\delta\vec{C}$ their transposes.

Since the variations δC are arbitrary, we conclude that is minimum if

$$R'R\vec{C} - R'\Delta\vec{V} = 0$$

$$\vec{C} = (R'R)^{-1}R'\Delta\vec{V} \qquad 4.6 - 5$$

2.9.4.7 Digital Programs

Several digital programs have been developed for use in the generation and verification of backup charts and the study of the rendezvous problem. One of these, used for all of the above purposes, is a large multi-purpose Fortran program called Betelgeuse. It has the capability to integrate two vehicles through either Earth or Lunar orbit.

The program utilizes standardized input and output routines. As a normal course of events, approximately 40 different parameters are output at each integration step.

A set of subroutines are available which are designed to solve for each of the maneuvers required in the Concentric

Flight Plan. Included in these is the capability to execute externally supplied ΔV maneuvers. Taken together, these features provide a flexible tool for the study of rendezvous. Additionally, the program has the capability to generate up to 100 consecutive Monte Carlo runs, each with randomly dispersed initial conditions. This feature is used to make the runs which yield data for the construction of backup charts.

Another set of subroutines in the program represent a mechanization of the backup charts. These routines are capable of sampling data during a run, calculating, and applying the required maneuvers. In addition, there is a routine which executes braking and line-of-sight control during the final phase of the rendezvous. Random errors, biases, and drifts are included in each of these subroutines to make the simulation realistic. These routines are used in conjunction with the above mentioned Monte Carlo generator and a set of statistical analysis routines for performance analyses of backup charts.

A second Fortran program is used to generate the coefficients needed for CSI and CDH backup charts, which are based on Maclaurin expansions. The program takes data derived from Betelgeuse runs, and processes it using a multiple linear regression technique, yielding the required coefficients.

A third Fortran program is used to generate data for TPI and mid-course backup charts. It has all of the required equations mechanized, and outputs data which can be directly plotted to yield a backup chart.