

MISSION ANALYSIS PROGRAM FOR SOLAR ELECTRIC PROPULSION (MAPSEP)

CONTRACT NAS8-29666

(Revised) October, 1974

**(NASA-CR-143934) MISSION ANALYSIS PROGRAM
FOR SOLAR ELECTRIC PROPULSION (MAPSEP).
VOLUME 1: ANALYTICAL MANUAL (Martin
Marietta Corp.) 148 p HC \$5.75 CSCI 21C**

N75-31211

**Unclas
G3/20 35799**

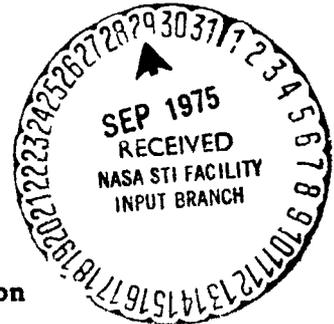
VOLUME I - ANALYTICAL MANUAL

**Prepared by:
P. E. Hong
G. L. Shults
R. J. Boain
K. R. Huling
T. Wilson**

**Planetary Systems Mission Analysis and Operations Section
Denver Division
Martin Marietta Corporation**

For

**National Aeronautics and Space Administration
Marshall Space Flight Center
Huntsville, Alabama**



FOREWORD

MAPSEP (Mission Analysis Program for Solar Electric Propulsion) is a computer program developed by Martin Marietta Aerospace, Denver Division, for the NASA Marshall Space Flight Center under Contract NAS8-29666. MAPSEP contains the basic modes: TOPSEP (trajectory generation), GODSEP (linear error analysis) and SIMSEP (simulation). These modes and their various options give the user sufficient flexibility to analyze any low thrust mission with respect to trajectory performance, guidance and navigation, and to provide meaningful system related requirements for the purpose of vehicle design.

This volume is the first of three and contains the analytical or functional description of MAPSEP. Subsequent volumes relate to operational usage and to program logical flow.

Acknowledgement

The authors wish to express their thanks to Jeanette Wearden for her skill and stamina in typing all three volumes, to Susie Borah for her art work on the flow charts, and to Kathy, Paula, Joan and Marion for their patience and understanding.

TABLE OF CONTENTS

	<u>Page</u>
Foreword	ii
Acknowledgement	iii
Table of Contents	iv
1.0 INTRODUCTION	1
2.0 PROGRAM DESCRIPTION	4
3.0 NOMENCLATURE	10
4.0 TRAJ	13
4.1 Equations of Motion	13
4.2 Trajectory Termination	22
4.3 Trajectory Accuracy	23
4.4 Trajectory Repeatability	25
4.5 State Transition Matrix Generation	27
4.6 Covariance Integration	30
5.0 TRAJECTORY GENERATION - TOPSEP	34
5.1 Nominal Trajectory Propagation	35
5.2 Grid Generation	35
5.3 Targeting and Optimization	36
6.0 LINEAR ERROR ANALYSIS - GODSEP	55
6.1 Augmented State	56
6.2 Covariance Propagation	58
6.3 Measurement Types	63
6.4 Filter	76
6.5 Generalized Covariance	80
6.6 Guidance	82

7.0	TRAJECTORY SIMULATION - SIMSEP	90
7.1	Program Scope and Methods	90
7.2	Definitions and Concepts	93
7.3	Guidance	95
7.3.1	Linear Guidance	96
7.3.2	Linear Impulsive Guidance	97
7.3.3	Low Thrust Linear Guidance	100
7.3.4	Non-Linear Guidance	103
7.4	Simulated Orbit Determination	108
7.5	Thrust Process Noise	109
7.6	Guidance Execution Errors	110
8.0	REFERENCES	112
9.0	APPENDICES	113
9.1	Conic Equations for Position and Velocity in Elliptical and Hyperbolic Orbits (Appendix 1)	113
9.2	A Generalized 4th Order Runge-Kutta Algorithm with Runge's Coefficients (Appendix 2)	115
9.3	Newton's 3rd Order Divided Difference Inter- polation Polynomial (Appendix 3)	117
9.4	Analytical Expressions for Terms in F Matrix (App. 4)	119
9.5	TOPSEP Injection Modeling (Appendix 5)	124
9.5.1	Injection Controls	124
9.5.2	Tug Multiple-Impulse Orbit Transfer	129
9.6	Control Weighting Matrices (Appendix 6)	137
9.7	Integrated State Transition Matrices for Computing the Targeting Sensitivity Matrix (Appendix 7)	140

1. INTRODUCTION

A major requirement for spacecraft systems design is the effective analysis of performance errors and their impact on mission success. This requirement is especially necessary for low thrust missions where thrust errors dominate all spacecraft error sources. Fast, accurate parametric error analyses can only be performed by a computer program which is efficiently constructed, easy to use, flexible, and contains modeling of all pertinent spacecraft and environmental processes. MAPSEP (Mission Analysis Program for Solar Electric Propulsion) is designed to meet these characteristics. It is intended to provide rapid evaluation of guidance, navigation and performance requirements to the degree necessary for spacecraft and mission design.

The baseline design of MAPSEP was taken from a previous study effort (Reference 1). Suitable modifications to the design were made which reflected subsequent operational experience (References 2 and 3) and actual construction and testing of MAPSEP. Considerable knowledge was also gained in the construction and usage of the engineering version of MAPSEP which was actually three separate programs that corresponded to the modes (Figure 1-1): TOPSEP, GODSEP and SIMSEP. Driving considerations in the program design and construction were: realistic vehicle and environment modeling consistent with preliminary vehicle design, flexibility in usage, computational speed and accuracy, minimum core utilization (for turnaround time and operating costs) and maximum growth potential through modularity.

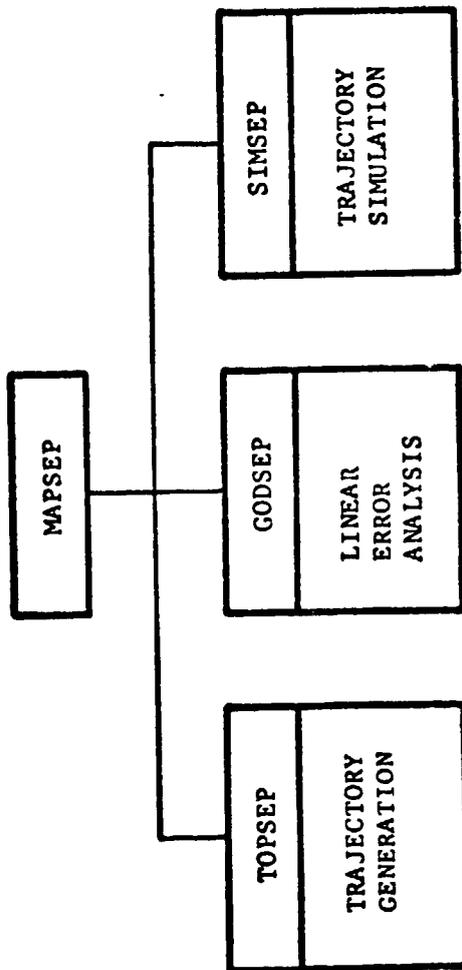


Figure 1-1 MAPSEP Modes

This document is the first of three volumes. Contained herein is a brief introduction to MAPSEP organization and detailed analytical descriptions of all models and algorithms. These include, for example, trajectory and error covariance propagation methods, orbit determination processes, thrust modeling and trajectory correction (guidance) schemes. This analytic background is necessary to fully understand program operation and to maximize program capability with respect to the user.

The second volume is a description of program usage, that is, input, output, recommended operating procedures and sample cases. The third volume is a detailed description of internal MAPSEP structure including macrologic, variable definition, subroutines and logical flow.

2. PROGRAM DESCRIPTION

This section summarizes MAPSEP's function and use, and MAPSEP structure. These areas are discussed in greater detail in the user's manual and programmer's manual, respectively.

As mentioned earlier, MAPSEP is composed of three primary modes. Each mode is intended to serve a given function in the mission design sequence. TOPSEP (Targeting and Optimization for SEP) is used to generate numerically integrated trajectories consistent with dynamic and system constraints. Performance data and related sensitivities are computed in the process of trajectory generation but can also be obtained by parametric application of TOPSEP. Indeed, each mode readily lends itself to parametric use which is a necessary feature for mission and system design studies.

GODSEP (Guidance and Orbit Determination for SEP) is used to perform a linear covariance analysis about a selected reference trajectory, generated by TOPSEP. Various dynamic and measurement related error sources are applied in a statistical sense to compute trajectory error covariances. These covariances, corresponding to estimation uncertainty (knowledge) and to actual trajectory deviations from the nominal (control), are propagated through a sequence of mission events: thruster switching, navigation measurement/state update, trajectory correction (guidance), etc. Thus, GODSEP computes a time history of the ensemble of all expected trajectory errors, and in the process displays such useful system parameters as required thrust control

authority, predicted terminal miss, additional fuel expenditures for off-nominal performance, etc.

SIMSEP (Trajectory SIMulation of SEP) is used in the latter stages of system design. It deterministically simulates the trajectory including the application of discrete dynamic errors. Trajectory corrections are simulated in an operational sense through a thrust update design and execution process. Navigation is simulated by sampling estimation error covariances (generated by GODSEP) prior to each guidance event. By operating SIMSEP in a Monte Carlo fashion, any desired number of simulated missions can be obtained, and estimated singly or collectively in statistical displays.

A fourth "mode", REFSEP (REFerence SEP), is actually an expansion of TOPSEP to provide a greater amount of trajectory and navigation related data for a particular reference trajectory.

Each mode of MAPSEP uses a common trajectory propagation routine, TRAJ. This guarantees trajectory reproducibility among modes. TRAJ integrates the equations of motion in Encke form using a fourth order Runge-Kutta scheme. Covariance propagation and transition matrices are computed by integrating variational equations simultaneously with the equations of motion. An option exists in GODSEP which stores a complete set of trajectory parameters and transition matrices as it is generated by TRAJ onto a magnetic disc called the STM file so that subsequent error analyses will not have to regenerate the data. The information on disc can then be transferred to magnetic tape for permanent storage, if desired. A more limited option is available for TOPSEP and SIMSEP which stores only the initial (input) trajectory data on disc.

Input to MAPSEP is primarily through cards using the NAMELIST feature, with supplementary means depending upon mode and function (Table 2-1). All modes require the \$TRAJ namelist which defines

Mode	INPUT			OUTPUT	
	Namelist	Formatted Cards	Tape (or disc)	Punched Cards	Tape (or disc)
TOPSEP	\$TRAJ \$TOPSEP	None	STM	None	STM GAIN
GODSEP	\$TRAJ \$GODSEP \$GEVENT	Event Data	STM GAIN	States Covariances Guidance	STM GAIN SUMMARY
SIMSEP	\$TRAJ \$SIMSEP \$GUID	None	STM	Statistics	STM GAIN SUMMARY
REFSEP	\$TRAJ	Print Events	STM	None	STM

TABLE 2-1. MAPSEP Input/Output

the nominal trajectory and subsequent mode usage. However, if recycling or case stacking is performed it is not necessary to input \$TRAJ again unless desired. The second namelist required for each mode corresponds to mode peculiar input and bears the name of that particular mode. Additional namelist, formatted cards, and tape input are generally optional. Besides the standard printout associated with MAPSEP auxiliary output can be obtained which will facilitate subsequent runs.

The structure of MAPSEP is organized into three levels of "overlays" which are designed to minimize total computer storage. At any given time, only those routines which are in active use are loaded into the working core of the computer. The main overlay (Figure 2-1) is always in core and contains the main executive, MAPSEP, and all utility routines that are common to the three modes. The primary overlays contain key operating routines of each mode, that is, those routines which are always needed when that particular mode is in use. Also included as a primary overlay is the data initialization routine, DATAM, where \$TRAJ namelist is read, trajectory and preliminary mode parameters are initialized, and appropriate parameters are printed out.

The secondary overlays contain routines which perform various computations during a particular operational sequence. Included are data initialization routines, analogous to DATAM, which operate on mode peculiar input and perform mode initialization. An example of core usage in the changing overlay structure may be provided by a standard error analysis event sequence. Error analysis initialization is performed by the overlay DATAG. Transition matrices are then read from the STM file, the state covariance is propagated to a measurement event, and the overlay MEAS is called, which physically replaces, or overlays, the same core used previously by DATAG. Similarly at a guidance event, overlay TRAJ will replace MEAS to compute target sensitivity matrices and overlay GUID will then

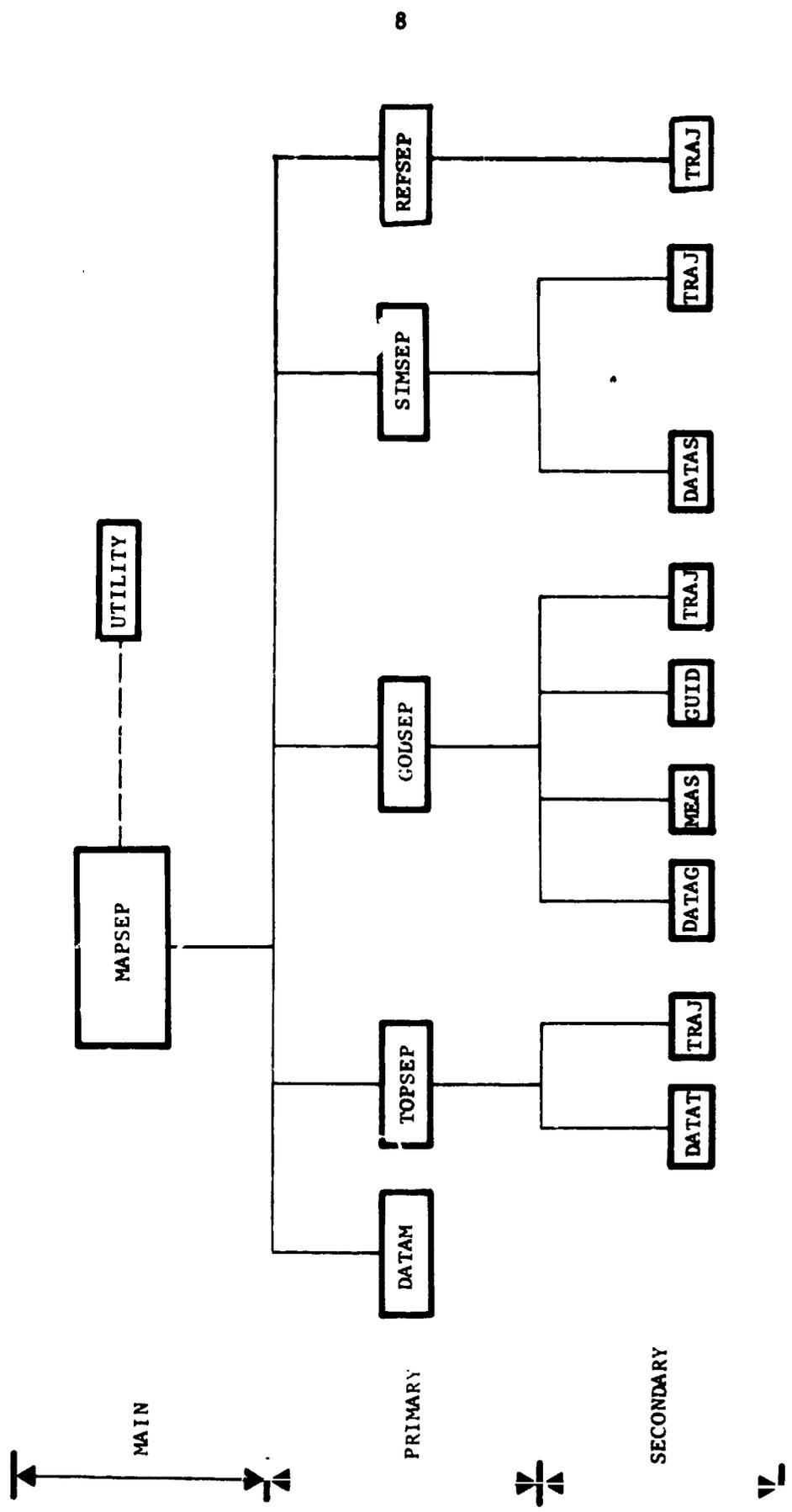


Figure 2-1. OVERLAY STRUCTURE

replace TRAJ to compute guidance corrections. Overlay switching is performed internally and is transparent to the user.

All of the routines and structure of MAPSEP are constructed to minimize core storage (thus reducing turn-around time and computer run cost) yet retain the flexibility needed for broad analysis requirements. Furthermore, routines are built as modular as possible to reduce the difficulties in future modifications and extensions.

3. NOMENCLATURE

The following symbols are used throughout the Analytic Manual, and to a great extent in the User's and Program Manuals. However, deviations from these symbols may occur in localized discussion if required for purposes of clarity.

<u>SYMBOL</u>	<u>DEFINITION</u>
a	propulsive acceleration
c	propulsive exhaust velocity
C_{ij}	cross covariance between i and j parameters
E	target error index
F	dynamic variation matrix
g	performance gradient or thrust transformation matrix
H	observation sensitivity matrix (WRT state)
I	identity matrix
K	filter gain matrix
m	spacecraft mass
P	covariance of state deviations <u>or</u> electrical power <u>or</u> projection operator
P_0	effective power at 1 AU
Q	dynamic (thrust) noise matrix
r	spacecraft position
s	solve-for parameters
S	target sensitivity matrix WRT control parameters
t	time

<u>SYMBOL</u>	<u>DEFINITION</u>
T	event time, <u>or</u> target variables, <u>or</u> thrust
u	dynamic (consider) parameters
u_T	thrust acceleration proportionality (throttlin
U	Control parameters
v	spacecraft velocity <u>or</u> measurement parameters
w	ignore parameters
W	weighting matrix
x	spacecraft state
r	guidance matrix
η	propulsive efficiency
θ	transition matrix of dynamic parameters
μ	gravitational constant
ρ	relative range
σ	standard deviation
τ	correlation time of thrust noise
Φ	transition matrix of augmented state
$\Phi(t_{k+1}, t_k)$	state transition matrix from time t_k to t_{k+1}
ω	thrust noise
<u>SUBSCRIPT</u>	<u>DEFINITION</u>
() _C	state control covariance
() _{k+1, k}	matrix evaluated over time interval t_k to t_{k+1}
() _k	state knowledge covariance

SUBSCRIPT $()_p$ $()_s$ $()_v$ $()_w$ $()_x$ DEFINITION

planet related parameters

solve-for parameters

measurement consider parameters

ignore parameters

spacecraft state parameters

MISCELLANEOUS

G&N

OD

PGM

S/C

SEP

WRT

E[]

 $()^+$ $()^-$ $(\dot{ })$ $(\hat{ })$ DEFINITION

guidance and navigation

orbit determination

Projected Gradient Method

spacecraft

solar electric propulsion

with respect to

expected value operation

post-event value

pre-event value

time derivative

unit vector

4.0 TRAJ

Essential to any program used for performance and navigation analysis is an accurate, but computationally efficient, trajectory propagation routine. This routine must contain realistic models of the dynamic processes acting on and performed by the spacecraft. In MAPSEP, the subroutine TRAJ fulfills this role. TRAJ is designed to be used by the three modes TOPSEP, GODSEP and SIMSEP, and is capable of duplicating the same trajectory in all modes.

The trajectory overlay TRAJ propagates planetary and interplanetary low thrust trajectories, using Encke's formulation of the equations of motion, from any epoch to a termination condition. TRAJ can optionally propagate the state covariance or the state transition matrix along the trajectory for either the basic state or an augmented state. Two of the most important features incorporated into TRAJ are the variable integration step algorithm and trajectory repeatability.

4.1 Equations of Motion

In Encke's formulation, all accelerations other than those due to the gravity of a primary body are called perturbing accelerations. Position (\underline{r}_c) and velocity ($\dot{\underline{r}}_c$) vectors are computed relative to the primary body using two body formula. The deviation vectors from the reference conic position and velocity vectors, $\delta \underline{r}$ and $\delta \dot{\underline{r}}$, respectively, are the direct results of numerically integrating the sum of the perturbing accelerations. The true position and velocity

vectors are, respectively,

$$\underline{r} = \underline{r}_c + \delta \underline{r}$$

$$\dot{\underline{r}} = \dot{\underline{r}}_c + \delta \dot{\underline{r}}$$

Since \underline{r}_c and $\dot{\underline{r}}_c$ can easily be computed from conic formula (See Appendix 1), the only problem is to compute \underline{r} and $\dot{\underline{r}}$.

Let $\delta \ddot{\underline{r}}$ be the acceleration deviation from two body motion, such that $\delta \ddot{\underline{r}}$ is the sum of all perturbing accelerations, e.g., other bodies, thrust, etc.

$$\delta \ddot{\underline{r}} = -\frac{\mu}{r_c^3} \left[f(\alpha) \underline{r} + \delta \underline{r} \right] - \sum_{i=1}^N \frac{\mu_i}{e_i^3} \left[\underline{r} + f(\alpha_i) \underline{r}_i \right] + \underline{a} + \underline{a}_R$$

The first term is the difference between two body and perturbed two body motion. The second term is the sum of the accelerations due to the N perturbing bodies. The third term (\underline{a}) is the acceleration due to thrust. The fourth term (\underline{a}_R) is the acceleration due to radiation pressure.

The first term is computed from the following equations (See Reference 4),

$$f(\alpha) = \frac{\alpha(3 + 3\alpha + \alpha^2)}{1 + (1 + \alpha)^{3/2}}$$

$$\alpha = \frac{(\delta \underline{r} - 2\underline{r}) \cdot \delta \underline{r}}{r^2}$$

where \underline{r} is the true s/c position vector relative to the primary body and μ is the gravitational constant of the primary body.

To compute the second term, the heliocentric position vectors, \underline{r}_i , of the perturbing bodies, are computed from mean analytical orbital elements to obtain

$$\underline{\rho}_i = \underline{r} + \underline{r}_p - \underline{r}_i$$

$$f(\alpha_i) = \alpha_i \left[\frac{3 + 3\alpha_i + \alpha_i^2}{1 + (1 + \alpha_i)^{3/2}} \right]$$

$$\alpha_i = \frac{r}{\rho_i} \left[\frac{r}{\rho_i} - \frac{2 \underline{r} \cdot \underline{\rho}_i}{r \rho_i} \right]$$

where \underline{r}_p is the heliocentric position vector of the primary body.

The acceleration vector due to radiation pressure is

$$\underline{a}_R = \frac{1.024 \times 10^8 \text{ A C}_r}{m r^2} \hat{\underline{r}}$$

where 1.024×10^8 - solar flux constant

- m - instantaneous mass of the s/c
- \underline{r} - heliocentric position vector of the s/c
- A - effective cross sectional area of the s/c
- C_r - coefficient of reflectivity.

The option exists in TRAJ, to include or exclude the effects of

radiation pressure when propagating both planetary and interplanetary trajectories. Before defining the acceleration due to thrust \underline{a} , we will model the power subsystem. There are two power subsystem models used by TRAJ for low thrust. They are solar electric and nuclear electric. The power to the thrusters (P) is

$$P = \begin{cases} P_o \left[\frac{C_1}{r^2} + \frac{C_2}{r^{5/2}} + \frac{C_3}{r^3} + \frac{C_4}{r^{7/2}} + \frac{C_5}{r^4} \right] \exp \left[-P_L(t-t_{DL}) \right] - P_{HK}, & \text{solar electric} \\ P_{\max}, & \text{if } P > P_{\max} \text{ or } r < r_{\min} \\ & \text{solar electric} \\ P_o \exp \left[-P_L(t-t_{DL}) \right] - P_{HK} & \text{nuclear electric} \end{cases}$$

P_o - Power available (at 1 AU for solar, at energization for nuclear electric)

C_1 - (Empirical) Constants defining solar array characteristics

r - Heliocentric position magnitude of the S/C

P_L - Power decay constant

t - Time from epoch

t_{DL} - Time delay

P_{HK} - Housekeeping power

P_{\max} - Maximum allowable solar electric power

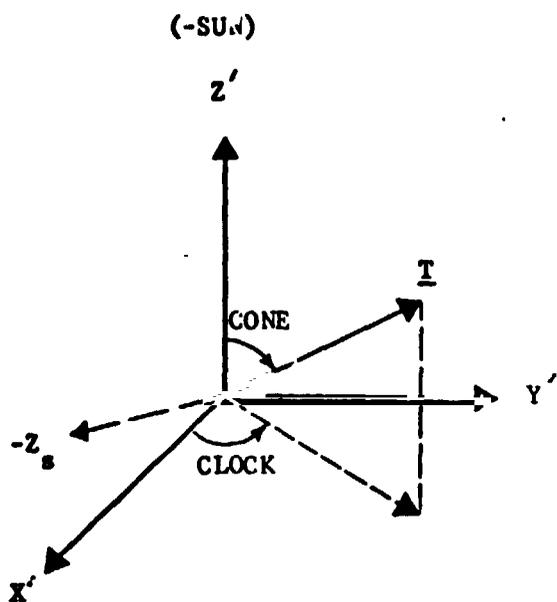
r_{\min} - Heliocentric distance at which P reaches P_{\max}

The exponential term in the solar electric expression describes the degradation of the solar array as a function of time.

The thrusters provide the spacecraft with the ability to maneuver. By changing the orientation of the thrust vector and maintaining that orientation for a long enough time, it is possible to steer the spacecraft and "shape" the trajectory. The thrust controls are defined in terms of constant parameters over a given time segment of the trajectory. Thus the user or mode can specify the following controls for each segment (up to 20):

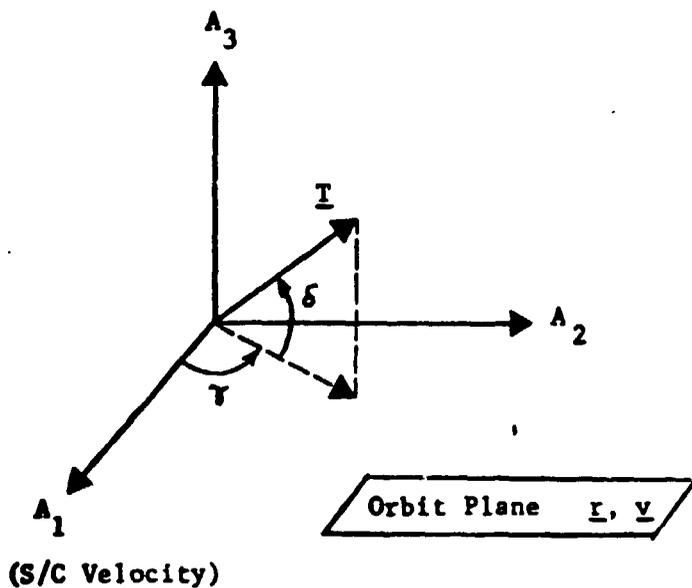
1. Thrust policy: Cone + Clock (sun orientation), In and Out of Plane Angles (orbit plane coordinates), or coast.
2. Segment end time (referenced to launch) of the current segment.
3. Throttling level, u_T
4. Cone angle (or In Plane Angle).
5. Clock Angle (or Out of Plane Angle).
6. Time rate of (4).
7. Time rate of (5).
8. Number of operating thrusters.

The thrust policy is merely thrusting or not thrusting (coasting). During thrust, the user has an option as to the reference system for the acceleration vector. The two reference systems are Cone and Clock Angles, Figure (4-1), and In and Out of Plane Angles, Figure (4-2). The latter is primarily used for Earth orbital applications and the former is primarily used for interplanetary missions.



The spacecraft is assumed to be oriented with the Z' axis in the sun-spacecraft line and the X' axis in a plane containing Z' and Z_s (reference star direction).

Figure (4-1). Cone and Clock Angles



δ - Out of plane angle

τ - In plane angle

Figure (4-2). In and Out of Plane Angles

The acceleration due to thrust is

$$|\underline{a}| = \frac{0.002 \eta P u_T}{m c}$$

.002 - Conversion factor

η - Averaged efficiency of the thrusters

P - Power delivered to the thrusters

u_T - Throttling level

m - Spacecraft mass

c - Exhaust velocity

The mass is numerically integrated from the equation

$$\dot{m} = \frac{-ma}{c}$$

The thrust acceleration vector \underline{a}' relative to the spacecraft for the two reference systems is

$$a'_x = a \cos \text{CLOCK} \sin \text{CONE}$$

$$a'_y = a \sin \text{CLOCK} \sin \text{CONE}$$

$$a'_z = a \cos \text{CONE}$$

for the Cone-Clock system and

$$a'_x = a \cos \zeta \cos \gamma$$

$$a'_y = a \cos \zeta \sin \gamma$$

$$a'_z = a \sin \zeta$$

for the In and Out of Plane system. In order to transform \underline{a}' into \underline{a} , which is in heliocentric ecliptic coordinates, we must define the matrix A such that

$$\underline{a} = A \underline{a}'$$

where

$$A = [\underline{X}' \quad \underline{Y}' \quad \underline{Z}']$$

and

$$\underline{Z}' = \underline{r} / |\underline{r}|$$

$$\underline{Y}' = \underline{r} \times \underline{z}_s / |\underline{r} \times \underline{z}_s|$$

$$\underline{X}' = \underline{Y}' \times \underline{Z}'$$

\underline{r} is the heliocentric spacecraft position vector and \underline{z}_s are the ecliptic direction cosines of a reference star for the Cone/Clock System. For the In/Out of Plane system, A is defined in terms of the position and velocity vectors relative to the primary body. Let \underline{r} be the position vector and \underline{v} be the velocity vector, then A can be defined as

$$A = [\underline{A}_1 \quad \underline{A}_2 \quad \underline{A}_3]$$

where

$$\underline{A}_1 = \underline{v} / |\underline{v}|$$

$$\underline{A}_3 = \frac{\underline{r} \times \underline{v}}{|\underline{r} \times \underline{v}|}$$

and

$$\underline{A}_2 = \underline{A}_3 \times \underline{A}_1$$

Since the trajectory is made up of segments, TRAJ propagates the trajectory using the appropriate set of controls over the interval the controls are in effect. Updates are automatically performed, at the beginning of each new segment.

Now that the perturbing acceleration, $\delta \ddot{\underline{r}}$, is defined, we can obtain $\delta \underline{r}$ and $\delta \dot{\underline{r}}$. To do this, $\delta \ddot{\underline{r}}$ is numerically integrated with a generalized 4th Order Runge-Kutta algorithm for first order differential equations (Appendix 2). We can express $\delta \ddot{\underline{r}}$ as a set of first order equations

$$\delta \dot{\underline{x}} = \begin{bmatrix} \delta \dot{\underline{r}} \\ \delta \ddot{\underline{r}} \end{bmatrix}$$

$\delta \dot{\underline{x}}$ can be numerically integrated to give $\delta \underline{x}(t) = \begin{bmatrix} \delta \underline{r} \\ \delta \dot{\underline{r}} \end{bmatrix}$ when given the following initial conditions:

$$\text{at } t = t_0, \quad \delta \underline{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \text{ and } \underline{r}_c = \underline{r}_0, \quad \dot{\underline{r}}_c = \dot{\underline{r}}_0$$

The propagation of $\delta \underline{r}$ (and $\delta \dot{\underline{r}}$) continues until δr is greater than or equal to some prescribed value δr_{\max} , then "rectification" occurs.

Hence, when $\delta r \geq \delta r_{\max}$ at some time, t ,

$$\text{we reinitialize } \underline{r}_c = \underline{r}(t), \quad \dot{\underline{r}}_c = \dot{\underline{r}}(t) \quad \text{and} \quad \delta \underline{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

and compute a new reference conic orbit. Rectification ensures that the conic is always "close" to the true state. The propagation

continues until δr is again greater than or equal to δr_{\max} , and so on to the end of the trajectory.

The use of constant thrust controls over discrete time intervals makes the trajectory discontinuous in acceleration at the control switching boundaries. The integration is thus done piecewise. That is, the state at the boundary between segments is used as initial conditions (rectification) for the next segment.

4.2 Trajectory Termination

There are four options for terminating trajectories in TRAJ: (1) final time, (2) closest approach to a target body, (3) sphere of influence of the target body, and (4) a radius relative to a target body. Termination at final time is straight forward. For the other conditions, termination criteria are tested after each integration step. Once a cutoff condition is sensed, a step-size is computed so that TRAJ can propagate to an interpolated time. TRAJ takes the three previous planet relative position magnitudes plus the present relative position magnitude, and the corresponding trajectory times, and fits a third order polynomial through the four data points, using Newton's 3rd order divided difference interpolation polynomial (Appendix 3). The independent variables for sphere of influence and stopping radius termination are the position magnitudes and the corresponding trajectory times are the dependent variables. Since the radius of the sphere of influence or the stopping radius is known before hand, the information that is needed is the time at these position magnitudes. For

closest approach, the same information is stored as before, but now the independent variables are the trajectory times and the position magnitudes are the dependent variables. Instead of knowing a time for which the position magnitude is a minimum, a value is computed corresponding to the minimum of a 3rd order polynomial.

Of particular note is stopping on the sphere of influence or radius of closest approach (Figure 4-3). These stopping conditions are used to evaluate B- plane (impact plane) parameters, $\underline{B} \cdot \underline{T}$ and $\underline{B} \cdot \underline{R}$ (Figure 4-4). These parameters form a convenient set of variables for the description of the approach geometry for interplanetary missions. Let \underline{V}_{HE} denote the hyperbolic excess velocity of the spacecraft. Then

$$\underline{S} = \frac{\underline{V}_{HE}}{|\underline{V}_{HE}|}$$

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

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

where \underline{k} is a unit vector normal to the reference plane, usually the planetocentric ecliptic plane.

4.3 Trajectory Accuracy

As with all problems that require numerical integration, some criteria must be used in determining the nominal integration stepsize. The stepsize algorithm used in TRAJ is empirical, and it meets the

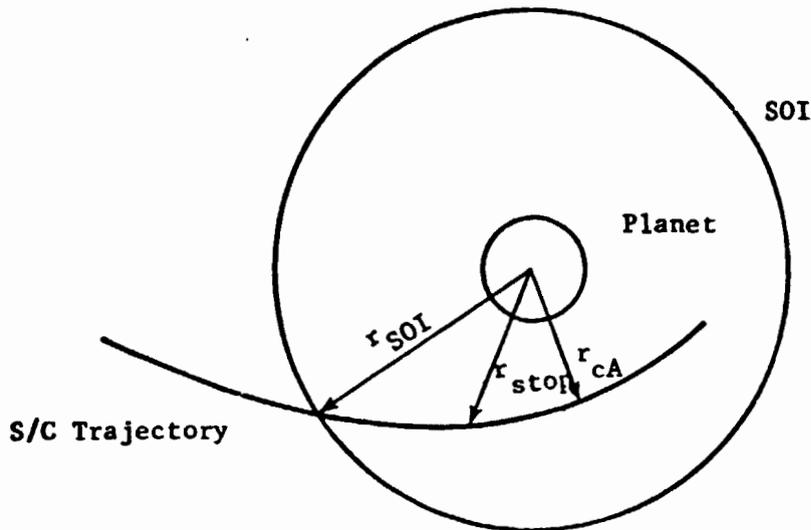
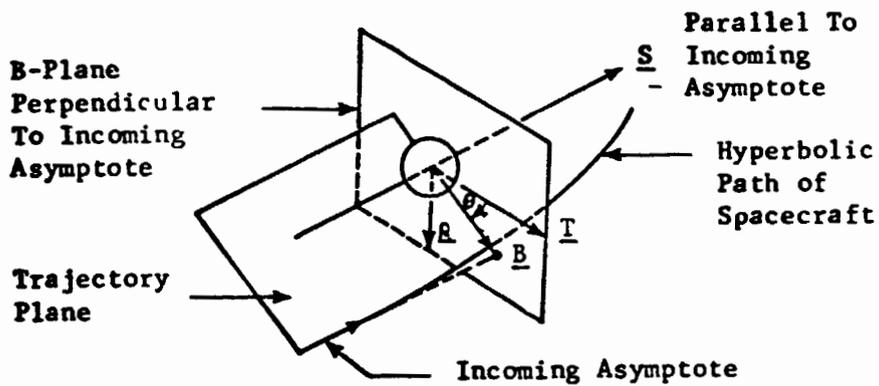


Figure (4-3). Radius Stopping Conditions



- B** = Impact Parameter (Vector from Planet Center to Aiming Point)
- S** = Orientation of **B** Relative to **T**
- T** = Parallel to Incoming Asymptote
- R** = Parallel to Reference Plane (Ecliptic Unless Otherwise Specified)
- R** = $\underline{S} \times \underline{T}$

Figure (4-4). B-Plane

requirements of reasonable numerical results and computer run time.

The algorithm is

$$h = \epsilon \cdot |f|^{\frac{1}{2}}$$

where h is the integration stepsize, f is the gravity gradient (See Appendix 4) and ϵ is a user input scale factor. There are constraints on h such that

$$h \leq 5 \text{ days}$$

for heliocentric trajectories and

$$h \leq 1 \text{ day}$$

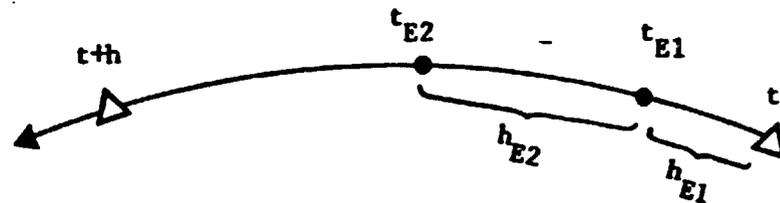
for planetary trajectories.

The effects of h on the state transition matrix are small compared to the spacecraft state. Mass and mass variation are also not strongly affected by h because they are affected primarily by the spacecraft heliocentric position. Therefore, a good choice of h ensures a satisfactory trajectory.

4.4 Trajectory Repeatability

A major goal in building TRAJ was the ability to reproduce the same trajectory in all modes, given the same initial and spacecraft related data. To accomplish this, all propagation that resulted in altering the nominal integration stepsize was decoupled from the event stepsize logic. That is, event times and print times do not alter the nominal stepsize; instead, the information at the previous

nominal stepsize, t , is saved and a separate stepsize is computed for the event or print time, $t + h_E$ (Figure 4-5). If there are many events or prints between t and $t + h$ a stepsize h_E is computed for each. Afterward, TRAJ returns to the nominal trajectory and continues the propagation. This suggests that, if the trajectory starts at any nominal integration step the trajectory will be duplicated, especially with respect to terminal conditions, for any run with different print and event times. For trajectories that do not start at a nominal integration step, there will be slight deviations in the terminal conditions from the nominal, but they will be close.



- △ - nominal integration time
- - event or print time
- h_E - event or print integration step
- h - nominal integration step

Figure 4-5. Trajectory Preservation

4.5 State Transition Matrix Generation

In a linear analysis, the state transition matrix, $\Phi_{k+1, k}$, is used to map perturbations about the reference trajectory from one epoch to another according to the equation,

$$\underline{\delta x}_{k+1} = \Phi_{k+1, k} \underline{\delta x}_k .$$

Because this mathematical operation is repeated many times, it is important to use the most efficient and accurate method of computing Φ available. This is best done by simultaneously integrating the variational differential equations which generate Φ and the S/C equations of motion. In MAPSEP these variational equations have been implemented and Φ is augmented to the state variables in the integrator.

The origin and composition of the variational equations are best understood by first considering the equations of S/C motion written as a system of first order, coupled differential equations; namely,

$$\dot{\underline{x}} = \underline{f}(\underline{x}, t) \quad (4-1)$$

In 4-1, \underline{x} is a six-dimensional state vector of position and velocity components, and \underline{f} is a vector function giving the time derivative of each state component. The most general form for \underline{f} may be written as

$$\underline{f} = \left[\begin{array}{c} \underline{v} \\ \underline{g} + \underline{g}' + \underline{a}_T + \underline{q} \end{array} \right] \quad (6 \times 1)$$

where \underline{g} is the gravitational acceleration due to the primary body,

\underline{g}' is the gravitational acceleration contribution from secondary bodies, and \underline{a}_T is the thrust caused acceleration term discussed in Section 4.1. \underline{g} is a term of miscellaneous accelerations due to radiation pressure, planetary oblateness, etc., and is normally neglected for interplanetary missions as a low order effect. It should be noted that the three primary terms, \underline{g} , \underline{g}' and \underline{a}_T , are all dependent on the S/C position vector. For example, \underline{g} and \underline{g}' depend on position through the law of gravity; \underline{a}_T , through the electric power function and the transformation which relates the body axis system to the inertial representation.

To derive the variational equations, the vector function, \underline{f} , is expanded in a Taylor series about some reference solution to equation 4-1. Hence, the right hand side of 4-1 becomes

$$\underline{f}(\underline{x} + \delta \underline{x}, t) = \underline{f}(\underline{x}, t) + \frac{\partial \underline{f}}{\partial \underline{x}} \delta \underline{x} + \mathcal{O}(\delta \underline{x}^2)$$

where $\delta \underline{x}$ is relative to the reference trajectory state at time t .

By neglecting second and higher order terms, this expression reduces to

$$\delta \dot{\underline{x}} = F \delta \underline{x} \quad (4-2)$$

where $\delta \dot{\underline{x}}$ is defined by

$$\delta \dot{\underline{x}} = \underline{f}(\underline{x} + \delta \underline{x}, t) - \underline{f}(\underline{x}, t)$$

and F is a matrix of first order partials of the vector function \underline{f} with respect to state components. Explicitly writing the F matrix, it is seen that it has only two non-zero partitions.

$$F = \begin{bmatrix} 0_{33} & - & I_{33} \\ f_{33} & - & 0_{33} \end{bmatrix} \quad (6 \times 6)$$

where I_{33} is a 3x3 identity matrix and f_{33} is a 3x3 matrix of time varying expressions evaluated along the reference trajectory. The components of f_{33} , in terms of state variables, are obtained by analytically differentiating \underline{g} , \underline{g}' and \underline{a}_T with respect to \underline{r} .

The solution of 4-2 is known from the theory of linear differential equations to be of the form

$$\delta \underline{x} = \Phi \delta \underline{x}_0 \quad (4-3)$$

where Φ is identified as the state transition matrix and is representable as

$$\Phi = \frac{\partial (x, y, z, v_x, v_y, v_z)}{\partial (x_0, y_0, z_0, v_{x0}, v_{y0}, v_{z0})} \quad (6x6)$$

As noted before, $\delta \underline{x}_0$ is a perturbation, or deviation, from the reference trajectory at the initial epoch and, as such, it is arbitrary but constant. Differentiating 4-3 with respect to time and substituting the resulting expression into 4-2 yields

$$\dot{\Phi} = F \Phi \quad (4-4)$$

This equation is the variational differential equation for the state transition matrix and is numerically integrated to obtain Φ over an arbitrary interval, t_0 to t_f , with initial conditions at t_0 being given by

$$\Phi = I_{66}.$$

From the more general point of view, equation 4-1 can be

considered to be dependent on other parameters as well as the usual state variables. For example, the trajectory generated as a solution to 4-1 is dependent on the thrust controls, the inertial states and gravitational constants of planetary bodies, and the solar gravitational constant. When some of these parameters are of interest in a linear analysis, the state vector of dynamic parameters is said to be augmented, thus increasing its dimension to as great as seventeen. This occurs when there are three thrust controls, six ephemeris elements, and two gravitational constants in addition to the vehicle state.

For the sake of modeling simplicity, MAPSEP allows ephemeris elements for only one planet to be augmented as dynamic parameters. This body is referred to as the "ephemeris body" and is usually selected to be the planet that most strongly influences the S/C trajectory (other than the earth). In many cases, the ephemeris body is the same as the target planet.

For the problem where the state vector is augmented with parameters, the equations of motion are more suitably written as

$$\dot{\underline{x}}_A = \underline{f}_A (\underline{x}, \underline{u}, \underline{x}_p, \mu_p, \mu_s ; t) \quad (4-5)$$

where \underline{x}_A is the augmented state vector, i.e.,

$$\underline{x}_A = \begin{bmatrix} \underline{x} \\ \underline{u} \\ \underline{x}_p \\ \mu_p \\ \mu_s \end{bmatrix}$$

In the above expression, \underline{x} corresponds, as before, to the S/C state vector; \underline{u} , to the thrust control vector; \underline{x}_p , to the ephemeris planet state vector; and the μ 's refer to gravitational constants. Following the previous analysis, it is possible to expand 4-5 to obtain variational equations for the augmented state transition matrix. This differential equation is of the form

$$\dot{\Phi}_A = F_A \Phi_A \quad (4-6)$$

where Φ_A is partitioned as

$$\Phi_A = \begin{bmatrix} \bar{I} & \theta_\mu & \theta_p & M_p & M_s \\ 0_{36} & I_{33} & 0_{36} & 0_{31} & 0_{31} \\ 0_{66} & 0_{63} & \Omega_p & M'_p & M'_s \\ 0_{16} & 0_{13} & 0_{16} & 1 & 0 \\ 0_{16} & 0_{13} & 0_{16} & 0 & 1 \end{bmatrix} \quad (17 \times 17)$$

and where F_A is given as

$$F_A = \begin{bmatrix} \begin{bmatrix} 0 & I \\ f & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ g \end{bmatrix}_{63} & \begin{bmatrix} 0 \\ k \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{61} & \begin{bmatrix} 0 \\ d \end{bmatrix}_{61} & \begin{bmatrix} 0 \\ m \end{bmatrix}_{61} \\ \begin{bmatrix} 0 & 0 \end{bmatrix}_{36} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{33} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{36} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{31} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{31} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{31} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{63} & \begin{bmatrix} 0 \\ p \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{61} & \begin{bmatrix} 0 \\ q \end{bmatrix}_{61} & \begin{bmatrix} 0 \\ s \end{bmatrix}_{61} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{26} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{23} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{26} \end{bmatrix} \quad (17 \times 17)$$

Explicit definitions for the terms in F_A are given in Appendix 4. Terms appearing in $\bar{\Phi}_A$ are explicitly defined as follows:

$$\bar{\Phi} = (\text{partials of state component w.r.t. state components}) \frac{\partial \bar{x}}{\partial \underline{x}_0} \quad 66$$

$$\Theta_u = (\text{partials of state components w.r.t. thrust controls}) \frac{\partial \bar{x}}{\partial \underline{u}} \quad 63$$

$$\Theta_p = (\text{partials of state components w.r.t. ephemeris planet state components}) =$$

$$\frac{\partial \bar{x}}{\partial \underline{x}_p} \quad , \quad 66$$

$$M_p = (\text{partials of state components w.r.t. ephemeris planet gravitational constant}) =$$

$$\frac{\partial \bar{x}}{\partial \mu_p} \quad , \quad 61$$

$$M_s = (\text{partials of state components w.r.t. solar gravitational constant}) =$$

$$\frac{\partial \bar{x}}{\partial \mu_s} \quad , \quad 61$$

$$\underline{\Omega}_p = (\text{partials of ephemeris planet state components at the final epoch w.r.t. ephemeris planet state components at the initial epoch}) =$$

$$\frac{\partial \underline{x}_p(t_f)}{\partial \underline{x}_p(t_0)} \quad , \quad 66$$

M'_p = (partials of ephemeris planet state components w.r.t. ephemeris planet gravitational constant) =

$$\frac{\partial x}{\partial \mu_p}$$

61

and M'_s = (partials of ephemeris planet state components w.r.t. solar gravitational constant) =

$$\frac{\partial x}{\partial \mu_s}$$

61

Before concluding this section, it should be noted that MAPSEP has program logic which allows arbitrary augmentation of dynamic parameters. That is, the program organizes and integrates matrices in equation 4-6 dimensioned to accommodate only those parameters requested during input. In this way, there is no time wasted in unnecessary calculations.

4.6 Covariance Integration

In any linear error analysis, a major problem is the propagation of state error covariances (P) from one event to the next event. Two methods are generally used: propagation with state transition matrices and numerical integration of the covariance matrix differential equations; both of which are options in TRAJ. In the latter case, the covariance is integrated to an event, where operations are performed on P, and the updated P is integrated to the next event.

Given the nonlinear equations of motion

$$\dot{\underline{x}} = \underline{\dot{x}}(\underline{x}, \underline{u}, \underline{\omega}) \quad (4-7)$$

where \underline{x} is the spacecraft position and velocity, \underline{u} are constant spacecraft controls and $\underline{\omega}$ are time-varying thrust parameters (nominally zero), these equations can be linearized about a reference trajectory such that

$$\delta \dot{\underline{x}} = \frac{\partial \underline{\dot{x}}}{\partial \underline{x}} \delta \underline{x} + \frac{\partial \underline{\dot{x}}}{\partial \underline{u}} \delta \underline{u} + \frac{\partial \underline{\dot{x}}}{\partial \underline{\omega}} \delta \underline{\omega} \quad (4-8)$$

where $\delta \dot{x}$, δx , δu and $\delta \omega$ are errors in the respective dynamic parameters. Both δu and $\delta \omega$ are described in terms of the 3x1 parameter set: u, cone and clock. The 6x1 $\delta \omega$ actually models two distinct processes for each parameter set (See also Page 62-A). Whereas Equation 4-7 describes motion of the deterministic reference trajectory, Equation 4-8 describes the linearized propagation of trajectory deviations resulting from dynamic and a priori uncertainties. The covariance integrated by TRAJ not only maps dynamic errors but also measurement related errors, specifically, uncertainties in three station locations. The augmented state covariance is defined as

$$P = E \left[\begin{array}{cc} \delta x_A & \delta x_A^T \end{array} \right]$$

$$\text{where } \delta x_A = \begin{bmatrix} \delta x \\ \delta y \\ \delta u \\ \delta \omega \\ \delta \epsilon_1 \\ \delta \epsilon_2 \\ \delta \epsilon_3 \end{bmatrix}$$

so that $\dot{P} = FP + PF^T + Q$

where F is similar to that used in the definition of the state transition matrix, and is evaluated along the reference trajectory, and Q is a process noise matrix. The augmented F matrix is defined as

$$F = \begin{bmatrix} 0 & I & 0 & 0 & 0 & 0 & 0 \\ f & 0 & g & n & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c \\ 0 & 0 & 0 & h & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where I is a 3 x 3 identity matrix,

$$f = \frac{\partial \dot{v}}{\partial \dot{x}}$$

$$g = \frac{\partial \dot{v}}{\partial \dot{u}}$$

$$n = [g \quad g]$$

and h is the matrix of process noise correlation times

$$h = \begin{bmatrix} -\frac{1}{T_1} & 0 & \dots & 0 \\ 0 & -\frac{1}{T_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -\frac{1}{T_s} \end{bmatrix}$$

Analytical equations for terms in the F matrix appear in Appendix 4.

The process noise, Q, is modeled as a stationary first order Gauss-Markov process and is defined as

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 \cdot h \cdot E[\delta \underline{\omega} \zeta \underline{\omega}^T] & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

For a discussion of Q, see Chapter 6 (GODSEP).

Propagating P by integrating \dot{P} is more mathematically accurate than the use of effective process noise as in the \bar{I} method, and it lends itself to greater modeling flexibility for Q. The drawback to this method is the increased run time.

5.0 TRAJECTORY GENERATION - TOPSEP

The targeting and optimization mode, TOPSEP, generates a reference trajectory which is supplied as basic input to the error analysis and simulation modes. The primary purpose of TOPSEP is to incorporate in this trajectory all of the desired flight characteristics for a particular interplanetary or near-Earth mission while optimizing the final spacecraft mass. Injection conditions (See Appendix 5), a thrusting time history, and other control parameters are found which accomplish this optimization and yet lead to the required target conditions. The target constraints may be the final spacecraft state (cartesian or B-plane coordinates), final orbital elements, radius of closest approach, or other mission specifications which are listed in Table 5-1 and in the input section of the Users Manual.

Control Parameters	Target Parameters	Performance Parameter
Initial State and Mass	Impact (B) Plane	Final Mass (Payload)
Thrust Magnitude	Sphere-of-Influence Time	
Thrust Direction Thrust Times	Closest Approach Conditions (Radius, Inclination, Time)	
Base Power Level	Target Centered State	
Exhaust Velocity	Heliocentric State	

Table 5-1. Control, Target, and Performance Parameters

The manipulation of trajectories to satisfy mission requirements is managed in three submodes of TOPSEP which represent successive stages of trajectory development. These submodes are:

1. Nominal trajectory propagation
2. Grid generation
3. Targeting and Optimization
 - a. Trajectory targeting only
 - b. A combination of trajectory targeting and optimization
 - c. Trajectory optimization only

Generally, these submodes are employed in order as listed above. However, any submode may be skipped or used individually if the proper control profile is available. Due to the simplicity of the first two submodes a brief discussion of their operational procedures is all that is necessary to understand their analytical basis in TOPSEP. The targeting and optimization submode will be reviewed in greater depth.

5.1 NOMINAL TRAJECTORY PROPAGATION

The simplest TOPSEP application is propagation of a single trajectory for spacecraft ephemeris information. After all the trajectory parameters are initialized, the trajectory is propagated from the initial state to the termination condition. TOPSEP performs no additional analysis of the trajectory when operating in this submode. This submode is also used for manual manipulation of the control profile.

5.2 GRID GENERATION

The grid generation submode is available to produce a number of trajectories which do not necessarily satisfy mission requirements but provide a range of trajectory solutions. Thus, the main purpose of the grid submode is to locate desirable control regions for further examination.

In turn, each control is incremented a fixed amount while the remaining controls maintain their nominal values. A single low thrust trajectory is generated for each control change and the associated target error index is calculated. Subsequently, contours of constant target error may be plotted in the control space so that some control regions can be eliminated from further consideration. Upon completion of the grid, the trajectory generation mode is terminated and the program user must choose the best control profile to initialize targeting and optimization or to employ another grid approach.

5.3 TARGETING AND OPTIMIZATION

The trajectory targeting and optimization submode features a discrete parameter iteration algorithm which accommodates the non-linear aspects of the low thrust problem. The algorithm is a modification of Rosen's projected gradient method (PGM) for non-linear programming (Refs. 5 and 6). The parameters which have been chosen to shape the trajectory (Table 5-1) constitute the control profile and are subject to modification by the PGM algorithm. Based upon the sensitivities of the final S/C mass and state to control variations, corrections to the profile are computed which maximize performance while minimizing target errors. The performance is measured simply by the value of the final spacecraft mass while the target errors are measured according to the constraint violations. The method chosen to represent the target errors in terms of a scalar measure is the quadratic error index which is the weighted sum of the squares of the target errors.

When the targeting and optimization submode is entered, a nominal trajectory is propagated directly from the input parameters. A series of

tests is performed to determine which submode (targeting, optimization or both) is to be executed. If the target error index is large, the submode will be exclusively targeting. However, a target error index smaller than some value (TUP in namelist \$TOPSEP) will result in simultaneous targeting and optimization. Whenever the index is below a specified lower bound (TLOW in namelist \$TOPSEP), the optimization algorithm will be executed.

Prior to the application of the projected gradient algorithm, the targeting sensitivity matrix S and the performance gradient g , are computed. Elements of the S matrix represent the sensitivities of individual target parameters to changes in controls and are used for both targeting and optimization. Similarly, the elements of the g vector represent the sensitivity of the performance index to changes in controls although these elements are used only for optimization. For purposes of targeting only, S is computed from the integrated state transition matrix (STM) and g is ignored. Appendix 7 discusses the formulation of S from the integrated STM. Whenever optimization is to occur both S and g are constructed by finite differencing techniques. Following the determination of S and g a weighting matrix which amplifies or diminishes the effects of the chosen controls is calculated. Applying the projected gradient algorithm a control correction is established. The magnitude of the control change is determined by computing trial trajectories. The new control profile is simply the old control profile plus a scalar multiple of the control correction such that the targeting error index is minimized and/or the performance index is maximized. If the optimization is complete (the values of the performance index have converged to a maximum), TOPSEP is terminated. Otherwise, the submode decision is made again and the cycle is repeated.

Of primary importance in the targeting and optimization submode is the selection of the control correction. In the following sections this selection process will be discussed.

THE PROJECTED GRADIENT METHOD

The projected gradient method has been devised to maximize a performance index while simultaneously minimizing an error index. Since maximizing performance is equivalent to minimizing cost in an optimization sense, PGM's purpose relative to the trajectory problem may be restated as minimizing fuel expended as well as minimizing target error. The concept of net cost, which is simply a more realistic assessment of fuel expended, will be discussed later in this section.

The projected gradient algorithm employs cost-function and constraint gradient information to replace the multi-dimensional targeting and optimization problem by an equivalent sequence of one-dimensional searches (Ref. 7). In this manner, it solves a difficult multi-dimensional problem by solving a sequence of simpler problems. In general, at the initiation of the iteration sequence, PGM primarily satisfies the constraint requirements. As the iteration process proceeds, the emphasis changes from constraint satisfaction to cost-function reduction.

Since numerous analytical developments of this technique are available (Refs. 5 and 6), this presentation will primarily emphasize the geometrical aspects of the algorithm. Clearly, the geometric interpretation of the algorithm is the motivation for the logic contained in TOPSEP, and a basic understanding of these concepts is usually sufficient to enable the user to efficiently manipulate TOPSEP input and to handle diverse mission problems.

PROBLEM FORMULATION

The projected gradient method solves the following non-linear programming problem:

Determine the values of the control variables, \underline{u} , that minimize the cost function (optimization variable)

$$F(\underline{u})$$

subject to the equality constraints

$$\underline{e}(\underline{x}(\underline{u})) = \underline{T}(\underline{x}(\underline{u})) - \underline{T}_d = \underline{0},$$

$\underline{u} \in U$, an M-dimensional control space

$\underline{T} \in T$, an N-dimensional target space

$\underline{x} \in X$, a six-dimensional state space

$$M \geq N$$

where the elements of \underline{e} , \underline{T} , \underline{T}_d , and \underline{x} are referred to as the target error, the target values, the desired target values, and the final state respectively; and F is a scalar valued function measuring system cost.

In an attempt to solve the constrained optimization problem, iterative methods are employed. The following scheme briefly describes the process occurring in TOPSEP.

- Guess \underline{u}_0 (In general \underline{u}_0 will not satisfy the constraints nor

- Determine a correction $\Delta \underline{u}$ such that
 - $F(\underline{u}_0 + \Delta \underline{u}) < F(\underline{u}_0)$ and
 - $\| \underline{e}(\underline{u}_0 + \Delta \underline{u}) \| < \| \underline{e}(\underline{u}_0) \|$
- Iterate until
 - F is minimized and
 - $\| \underline{e} \| < \epsilon$, a pre-determined tolerance

NOMENCLATURE AND CONCEPTS

To facilitate the discussion of the projected gradient algorithm, the following nomenclature and basic concepts will be introduced. \underline{x} denotes a column vector whose elements are x_i , where $i = 1, 2, \dots, n$ and n is the dimension of the space containing \underline{x} . Y^T denotes the transpose of the real matrix Y . The feasible region defined in the M -dimensional control space within which PGM operates is the restricted space

$$u_{i\text{MIN}} \leq u_i \leq u_{i\text{MAX}}, \quad i = 1, 2, \dots, M.$$

The equality condition implies that the control is on a bound. The cost gradient \underline{g} is an M -vector of partial derivatives and is defined as

$$\underline{g} = \left(\frac{\partial F}{\partial \underline{u}} \right)^T.$$

The sensitivity matrix is that matrix whose rows are the gradients to the equality constraints, and is denoted by

$$S(\underline{u}) = \frac{\partial \underline{e}(\underline{u})}{\partial \underline{u}},$$

where \underline{e} is an N -dimensional vector. The target error function is defined to be

$$E(\underline{u}) = \underline{e}^T [w_e] \underline{e}$$

where W_e is a target weighting matrix which will be defined later.

Corresponding to each control vector \underline{u} in the control space U , there is an error vector \underline{e} . Let A_0 be the set of all \underline{u} such that

$$\underline{e}(\underline{u}) = \underline{0}.$$

A_0 then represents all the control vectors satisfying zero-target error.

It can be shown (Reference 6) that A_0 defines an $M-N$ dimensional non-linear hypersurface or manifold in U . Unfortunately, A_0 cannot be defined explicitly; hence, one cannot easily find a \underline{u} which is an element of A_0 .

However, A_0 can be estimated implicitly via the sensitivity matrix.

Let A_c be the set of all \underline{u} such that

$$\underline{e}(\underline{u}) = \underline{c},$$

where \underline{c} is a vector of constants. Thus, A_c represents a non-linear manifold containing those control vectors which provide constant target error.

It can also be shown (Reference 6) that any \underline{u} in the control space is contained in one and only one A_c . At a given \underline{u} , the corresponding non-linear manifold A_c may be approximated by a linear manifold $B(\underline{u})$ which is defined explicitly by the sensitivity matrix $S(\underline{u})$. The linear manifold $B(\underline{u})$ may be considered a tangent hyperplane to A_c at \underline{u} . The orientation of $B(\underline{u})$ in the control space allows one to define a search direction to A_0 which is orthogonal to $B(\underline{u})$. This search is in the direction of maximum decreasing target error.

Let $\tilde{B}(\underline{u})$ denote the orthogonal complement to $B(\underline{u})$. One can demonstrate (Reference 6) that $\tilde{B}(\underline{u})$ is the unique linear space that can be

translated to obtain the linear manifold $B(\underline{u})$. Furthermore, there exist unique orthogonal projection operators $\tilde{P}(\underline{u})$ and $P(\underline{u})$ that resolve any vector in the control space into its corresponding components in $\tilde{B}(\underline{u})$ and $B(\underline{u})$, respectively; that is

$$\underline{u} = \tilde{P}(\underline{u}) \underline{u} + P(\underline{u}) \underline{u}.$$

In particular,

$$\tilde{P}(\underline{u}) = S^T (SS^T)^{-1} S \quad (5-1)$$

and

$$P(\underline{u}) = I - \tilde{P}$$

where I is an identity matrix. The projection operators \tilde{P} and P thus provide a simple method for reconstructing a general vector $\Delta \underline{u}$ emanating from \underline{u} into its two components in \tilde{B} and B . In a discussion to follow later in this section, it will be explained how $\Delta \underline{u}$ may be defined such that $\tilde{P} \Delta \underline{u}$ represents the control correction to minimize the target error and $P \Delta \underline{u}$ represents the control correction to minimize cost.

The final key concept employed by PGM is the idea of problem scaling. The purpose of problem scaling is to increase the efficiency of the targeting and optimization algorithms by transforming the original problem into an equivalent problem that is numerically easier to solve.

To numerically scale a problem, two general types of scaling are required: 1) control variable scaling, and 2) target variable scaling. Control variable scaling is accomplished by defining a positive diagonal scaling matrix, W_u (UWATE in namelist \$TOPSEP), such that the weighted control variables are given by

$$\underline{u}' = W_u \underline{u}.$$

Similarly, target variable weighting is accomplished by defining a positive diagonal scaling matrix, W_e (TARTOL in namelist \$TOPSEP), such that the weighted target variables are

$$\underline{T}'(\underline{u}) = [W_e] \underline{T} (W_u^{-1} \underline{u}') .$$

The target error index is then

$$E(\underline{u}) = e'^T e' .$$

TOPSEP contains several options for computing the control variable weighting matrix. The option most often used is the normalization scaling matrix (See Appendix 6 for other options).

$$[W_u]_{ii} = \frac{1}{|u_i|} .$$

The target variable weighting matrix is always computed as the reciprocal of the constraint tolerances and is given by

$$[W_e]_{ii} = \frac{1}{\epsilon_i} .$$

where ϵ_i is the tolerance for the i^{th} target error.

For simplicity, the following discussion of the algorithm assumes an appropriately scaled problem. However, the scaled equations can be obtained by making the following simple substitutions.

\underline{u} replaced by \underline{u}'

\underline{e} replaced by \underline{e}'

S replaced by $[W_e][S][W_u]^{-1}$

\underline{g} replaced by $[W_u]^{-1} \underline{g}$

DIRECTION OF SEARCH

The concept of the direction of search in control space needs slightly more elaboration. The direction of search is nothing more than a particular line in the control space along which the target error is reduced or along which the cost function is decreased. In a more precise sense, the direction of search at $\hat{\underline{u}}$ is a half-ray emanating from $\hat{\underline{u}}$. Thus, for any positive scalar, γ , the equation

$$\underline{u} = \hat{\underline{u}} + \gamma \Delta \hat{\underline{u}}$$

sets the limits of this half-ray and represents a "step" in the direction $\Delta \hat{\underline{u}}$ from $\hat{\underline{u}}$. This is illustrated in Figure 5-1.

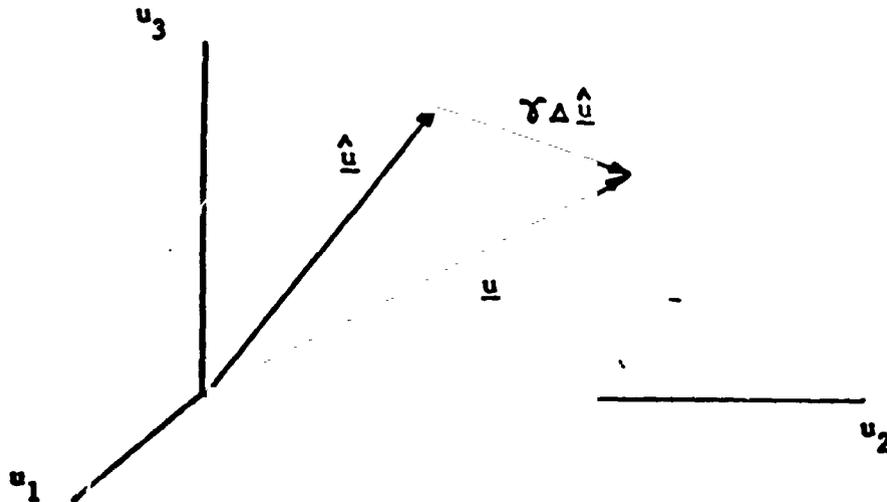


Figure 5-1. Direction of Search

This concept of direction-of-search is particularly important since it enables the M-dimensional non-linear programming problem to be replaced by a sequence of one-dimensional minimizations. What remains to be explained is: 1) how to select the direction of search, and 2) how to determine the step size in that direction.

The projected gradient method uses two basic search directions. For this discussion they will be termed constraint and optimization directions. PGM proceeds by taking successive steps in one or the other of these two directions. The computation of each of these search directions is described below at a particular point \underline{u} in the M-dimensional control space where N constraints (target conditions) are enforced.

CONSTRAINT DIRECTION

The constraint direction depends critically on the number of targets. Two cases are distinguished below:

1. If $N < M$, then that unique control correction $\Delta \hat{\underline{u}}$ is sought which solves the linearized constraint equation

$$[S] \Delta \hat{\underline{u}} + \underline{e}(\hat{\underline{u}}) = \underline{0} \quad (5-2)$$

and minimizes the norm of $\Delta \hat{\underline{u}}$. The solutions to the preceding vector equation define the M-N dimensional linear manifold $B_0(\underline{u})$ which is an estimate of the non-linear manifold A_0 (zero-target error). The desired minimum norm correction $\Delta \hat{\underline{u}}$ is then the vector of minimum length in the control space from $\hat{\underline{u}}$ to the linear manifold $B_0(\underline{u})$, thus requiring $\Delta \hat{\underline{u}}$ to be orthogonal to $B_0(\underline{u})$.

Application of the linear operators \tilde{P} and P allow one to represent $\Delta \hat{\underline{u}}$ as the sum of two orthogonal vectors relative to $B_0(\underline{u})$ or

$$\Delta \hat{\underline{u}} = \tilde{P} \Delta \hat{\underline{u}} + P \Delta \hat{\underline{u}};$$

however,

$$P \Delta \hat{\underline{u}} = \underline{0}$$

since there are no components of $\Delta \hat{\underline{u}}$ in B_0 . From equation (5-1)

$\Delta \hat{\underline{u}}$ may be expressed as

$$\Delta \hat{\underline{u}} = \mathbf{S}^T (\mathbf{S}\mathbf{S}^T)^{-1} \mathbf{S} \Delta \underline{u}$$

which may be reduced to

$$\Delta \underline{u} = -\mathbf{S}^T (\mathbf{S}\mathbf{S}^T)^{-1} \underline{e}(\underline{u})$$

using the constraint equation (5-2). This correction is illustrated in Figure 5-2.

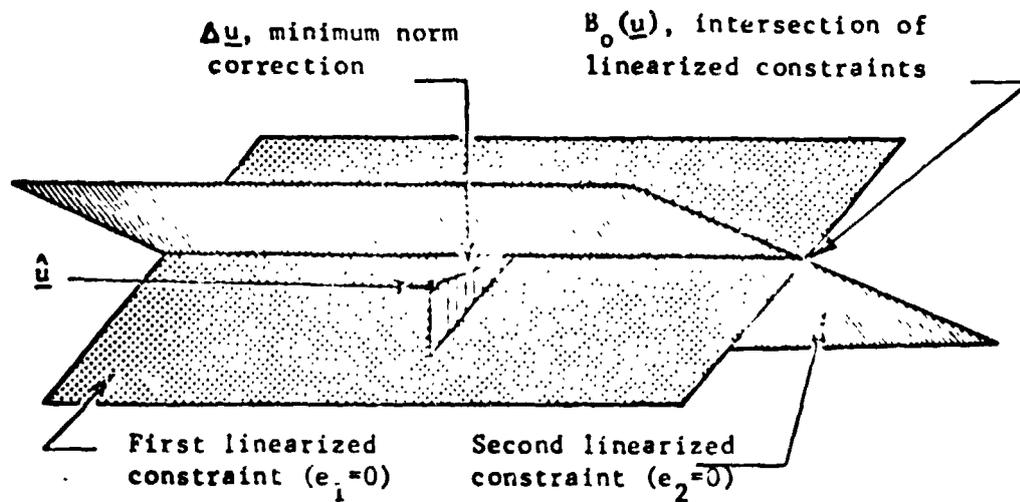


Figure 5-2. Illustration of Minimum Norm Constraint Direction for $N = 2$, $M = 3$.

The direction of search then is simply taken to be this minimum norm correction to the linearized constraints,

2. If $N = M$ there is unique solution to the linearized constraint equations without the additional requirement that the norm of the control correction be minimized. The solution for $\Delta \hat{\underline{u}}$ reduces to the familiar Newton-Raphson formula for solving M equations with M unknowns; namely

$$\Delta \hat{\underline{u}} = -S^{-1} \underline{e}(\underline{u})$$

The Newton-Raphson correction is illustrated geometrically in Figure 5-3.

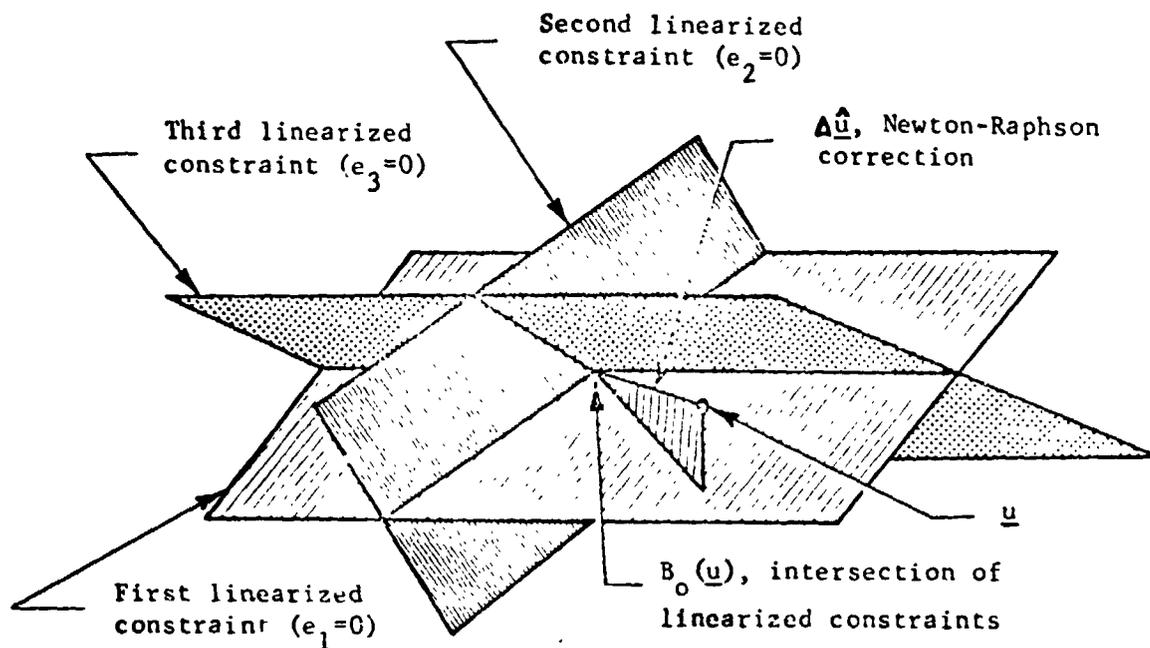


Figure 5-3. Illustration of Newton-Raphson constraint direction for $N = M = 3$

OPTIMIZATION DIRECTION

When the number of targets is less than the number of controls, it is then possible to minimize the cost function $F(\hat{\underline{u}})$ assuming, of course, that $\hat{\underline{u}}$ is some non-optimal control profile. Obviously, the steepest descent direction, $-\underline{g}(\hat{\underline{u}})$, would be the best local search direction for reducing the cost function. Such a direction, however, could produce unacceptable constraint violations. To avoid this difficulty PGM orthogonally projects the unconstrained negative gradient, $-\underline{g}$, onto the local

linearized constraint manifold $B_c(\hat{u})$. By searching in the direction of this negative projected gradient the algorithm can guarantee in a linear sense that there is no further constraint violation than that of $\underline{e}(\hat{u})$. To calculate this direction, it is only necessary to apply to $-\underline{g}$ the projection operator $P(\hat{u})$ which will map the vector into its component on the linear manifold $B_c(\hat{u})$. Thus,

$$\begin{aligned}\Delta \underline{u} &= -P \underline{g}(\hat{u}) \\ &= -[I - \tilde{P}] \underline{g}(\hat{u}) \\ &= -[I - S^T (SS^T)^{-1} S] \underline{g}(\hat{u})\end{aligned}$$

COMBINED TARGETING AND OPTIMIZATION DIRECTION

When it is desirable to minimize the cost function as well as reducing the target error the constraint direction and optimization direction may be combined such that the resulting control correction is of the form

$$\Delta \underline{u} = \Delta \underline{u}_1 + \Delta \underline{u}_2$$

where $\Delta \underline{u}_1$ is the optimization correction and $\Delta \underline{u}_2$ is the constraint correction. Note that $\Delta \underline{u}_1$ and $\Delta \underline{u}_2$ are orthogonal components of $\Delta \underline{u}$. Depending upon the magnitude of the target error, one may want to emphasize either optimization or targeting. Since this decision is rather subjective and linked directly to the degree of problem non-linearity, an option is provided to weight $\Delta \underline{u}$ (See Page 50) in one of its component directions. Figure 5-4 and Figure 5-5 illustrate the geometric interpretation of the resulting control correction.

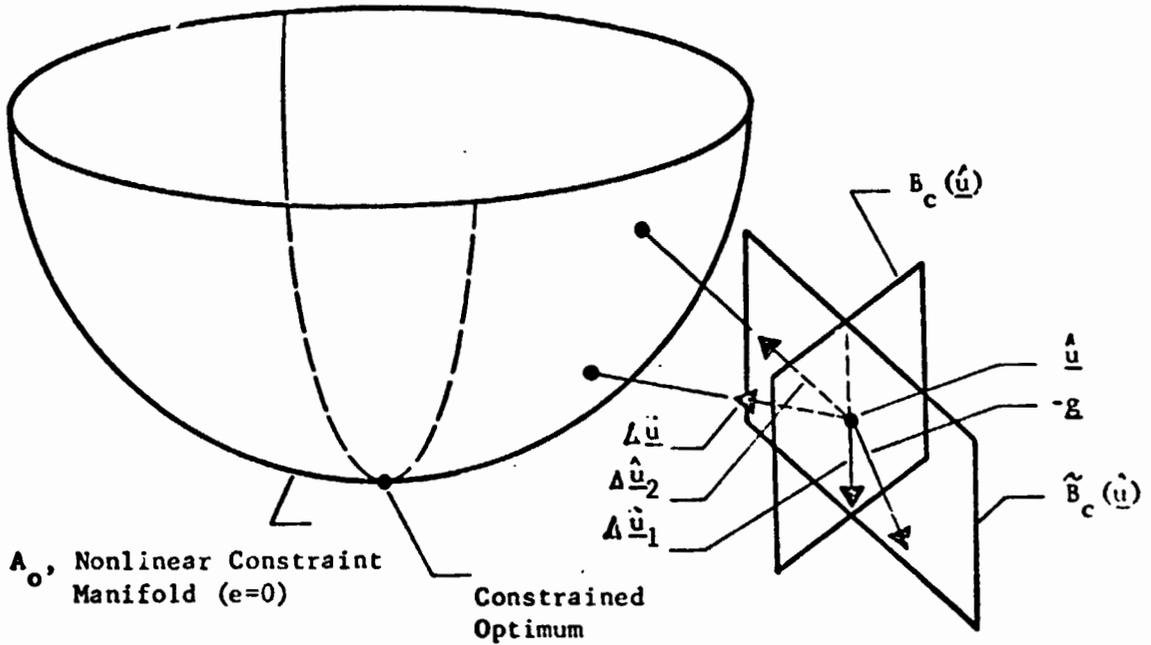


Figure 5-4. Geometric Interpretation of a Combined Targeting and Optimization Control Correction, $N=1$, $M=3$

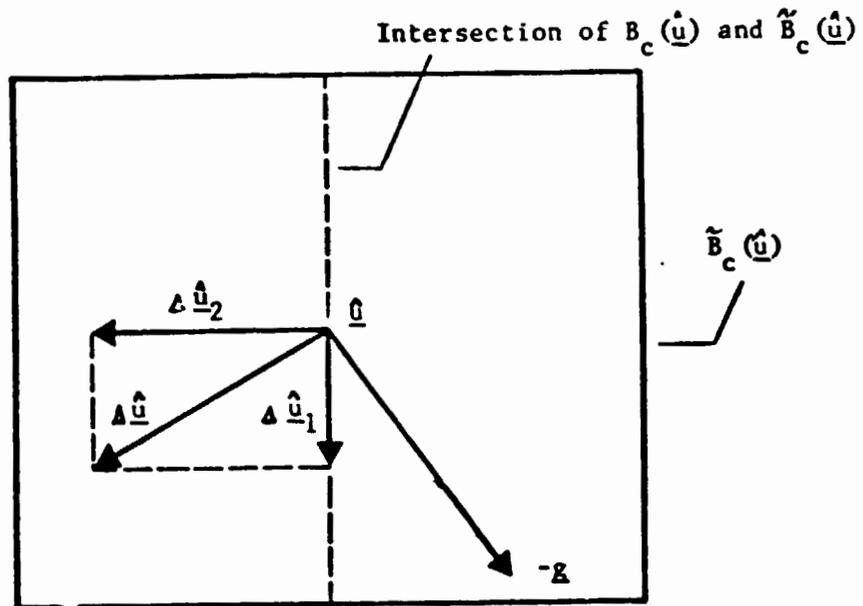


Figure 5-5. Illustration of Combined Targeting and Optimization Control Correction As Seen In $\tilde{B}_c(\hat{u})$, (Enlargement of $\tilde{B}_c(\hat{u})$ from Figure 5-4).

The total control correction is constructed as follows, where d is an input scalar,

$$\Delta \underline{u} = \frac{\|\Delta \underline{u}_2\|}{\|\Delta \underline{u}_1\|} * d * \Delta \underline{u}_1 + \Delta \underline{u}_2$$

(DP2 in namelist \$TOPSEP). Thus, one has the flexibility of determining the magnitude of the optimization correction relative to the magnitude of the constraint correction. The optimization correction can then be written as

$$\Delta \underline{u}_1 = \frac{\sqrt{\underline{e}^T (SS^T)^{-1} \underline{e} * (1 + d^2)} (I - S^T (SS^T)^{-1} S) \underline{g}}{\|(I - S^T (SS^T)^{-1} S) \underline{g}\|}$$

The norm of the control correction $\Delta \underline{u}$, which is obtained by summing $\Delta \underline{u}_1$ and $\Delta \underline{u}_2$, is not as important as the direction. The resulting half-ray provides the basic search direction with which to calculate the trial step.

TRIAL STEP-SIZE CALCULATION

At any particular point $\hat{\underline{u}}$ in the control space, the PGM algorithm proceeds by reducing the multi-dimensional problem to a one-dimensional search in the direction prescribed by the constraint or optimization control change vector. Once the initial point $\hat{\underline{u}}$ and the direction of search $\Delta \hat{\underline{u}}$ are specified, the problem reduces to the numerical minimization of a function of a single variable, namely, the step scale factor δ . PGM performs this numerical minimization by polynomial interpolation based on function values along the search ray and the function's value and slope at the starting point.

CONSTRAINT DIRECTION

The function to be minimized along the constraint direction $\Delta \hat{\underline{u}}_2$ is

$E(\delta)$, the sum of the squares of the target errors.

$$E(\delta) = \underline{e}^T (\hat{\underline{u}} + \Delta \hat{\underline{u}}_2) [W_e] \underline{e} (\hat{\underline{u}} + \Delta \hat{\underline{u}}_2)$$

Evaluation of the function at $\delta = 0$ results in

$$E(0) = \underline{e}^T (\hat{\underline{u}}) [W_e] \underline{e} (\hat{\underline{u}})$$

Differentiation via the chain rule yields

$$\left. \frac{\partial E(\delta)}{\partial \delta} \right|_{\delta=0} = 2 \underline{e}^T (\hat{\underline{u}}) S \Delta \hat{\underline{u}}_2$$

If constraints are reasonably linear, a good initial estimate for the minimizing δ is one.

OPTIMIZATION DIRECTION

The function to be minimized along the optimization direction $\Delta \hat{\underline{u}}_1$ is the estimated net cost function $C(\delta)$, where

$$C(\delta) = \underbrace{F(\hat{\underline{u}} + \delta \Delta \hat{\underline{u}}_1) - F(\hat{\underline{u}})}_{\text{Change in cost produced by a step of length } \delta \|\Delta \hat{\underline{u}}_1\| \text{ along } \Delta \hat{\underline{u}}_1} + \underbrace{\underline{g} \left[-S^T (SS^T)^{-1} \underline{e} (\hat{\underline{u}} + \delta \Delta \hat{\underline{u}}_1) \right]}_{\text{Linearized approximation to change in cost required to perform a minimum norm correction in order to retarget}}$$

Change in cost produced by a step of length $\delta \|\Delta \hat{\underline{u}}_1\|$ along $\Delta \hat{\underline{u}}_1$

Linearized approximation to change in cost required to perform a minimum norm correction in order to retarget

Clearly,

$$C(0) = - \underline{g} S^T (SS^T)^{-1} \underline{e} (\hat{\underline{u}})$$

By expanding $C(0)$ in a Taylor series in δ about $\delta = 0$, and by making use of the fact that $\tilde{P} \Delta \hat{\underline{u}}_1 = \underline{0}$, it can be shown that

$$\left. \frac{\partial C(\delta)}{\partial \delta} \right|_{\delta=0} = \underline{g} \Delta \hat{\underline{u}}_1$$

These properties are illustrated in Figure 5-6.

Both the constraint and optimization directions are based on a sensitivity matrix assuming a linear space. Due to nonlinearities in the problem, it is often necessary to restrict the trial step such that unexpected increases in cost or target error are reduced. In addition, the control correction $\Delta \underline{u}$ does not reflect the "nearness" of control bounds as long as \underline{u} is not on any bound. Thus, the trial step must also be restricted so that the new control vector remains within the bounded control space. For these reasons a maximum limit is placed on γ . After

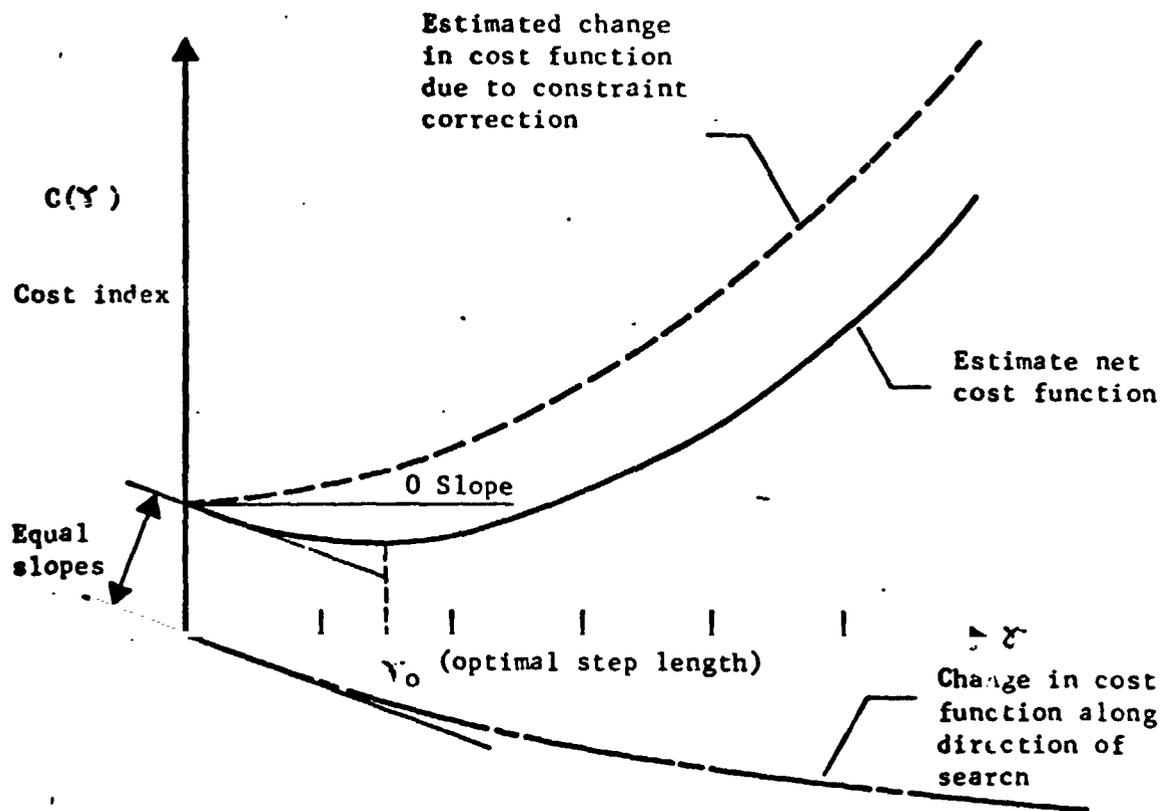


Figure 5-6. Properties of net-cost function along the direction of search.

$\Delta \underline{u}$ is computed, a calculation is made to find the δ along the search direction which will size a trial step that intersects the boundary of the feasible region. This value is compared to the maximum step allowed by the user to counter the nonlinearity problem. The smaller value is specified as the maximum δ allowed during the search. A method has been devised to alleviate the problem of a control which is very near a boundary. A tolerance region is defined in the neighborhood of the boundary surface such that if the control is within this region and $\Delta \underline{u}$ intersects the boundary the search to minimize the net cost or target error can continue along the boundary without calculating a new sensitivity matrix. Once a control element reaches one of its bounds it becomes inactive. Unless a subsequent correction for this control element is back into the feasible region it remains inactive.

ONE DIMENSIONAL MINIMIZATION

Nonvariant minimization in PGM is performed exclusively by polynomial interpolation. The actual function to be minimized is fitted with one or more quadratic or cubic polynomials until a sufficiently accurate curve fit is obtained. The minimum of this curve and the corresponding scale factor can easily be found analytically.

The one-dimensional search proceeds by taking trial steps in the $\Delta \underline{u}$ direction to obtain information about the function to be minimized. If $\nabla \Delta \underline{u}$ is a constraint correction, the quadratic error function is evaluated; if $\nabla \Delta \underline{u}$ is an optimization correction,

the net-cost function is evaluated; and if $\nabla\Delta u$ is a combined correction, a function which is a weighted combination of the error function and net-cost function is evaluated.

The minimization routine makes ingenious use of all the information it accumulates. The following curve-fitting techniques are applied in order.

1. Quadratic polynomial fit: two points-one slope;
2. Cubic polynomial fit: three points-one slope;
3. Quadratic polynomial fit: three points;
4. Cubic polynomial fit: four points.

Each time a trial step is taken, the function which is evaluated is used as a trial point to analytically determine the next trial step. The analytical formulation of the above curve fits may be found in the subroutine description of MINMUM in the Program Manual.

6. LINEAR ERROR ANALYSIS - GODSEP

GODSEP analyzes spacecraft and trajectory related dispersions as a function of expected uncertainties in dynamic and navigation parameters. The ensemble of expected errors is studied without actually simulating individual trajectories by applying linear techniques. That is, small deviations about a reference trajectory are linearly related to other deviations by a transformation matrix. For example, the state transition matrix relates position and velocity deviations about the reference trajectory from one time point to another. The ensemble of errors, or covariance, is assumed to have a zero-mean Gaussian distribution, except for special processes.

Probabilistic a-priori errors in the environment, spacecraft and tracking systems are propagated in time along the reference trajectory through sequential events such as orbit determination (OD) and guidance corrections. Two types of ensemble error or covariances are distinguished - knowledge, which reflects the ability of the OD algorithm to estimate the spacecraft state and other parameters; and control, which represents the dispersions of the actual spacecraft trajectory about the reference. Covariance propagation is done by either integration of covariance variational equations, or by the state transition matrix method.

The error analysis proceeds sequentially from start time through each specified trajectory event to final time. Event types available are measurement, propagation, eigenvector, prediction, thrust switching, and guidance. A measurement event processes tracking data at a time point by applying the user specified OD algorithm. Available to the user are both Kalman-Schmidt (K-S) and sequential weighted least squares (WLS) filters.

GODSEP modularit, also allows the user to insert his own filter algorithm quite easily. The filters are distinguished by their methods of gain matrix calculation and subsequent update of the knowledge covariance.

A propagation event merely updates the knowledge (and control) covariance at the event time. Its primary value is in maintaining accurate covariance values during long propagations by forcing computation of the effective process noise over predetermined, user-specified intervals.

An eigenvector event is used for information display and behaves similar to a propagation event. Covariance matrix sub-blocks are converted to standard deviations and correlation coefficients. It also computes eigenvalues, their square roots, and eigenvectors for the position and velocity 3x3 sub-blocks of the state covariance matrix. Thrust switching events are simply eigenvector events at the time where a change in the number of thrusters or thrust policy has occurred.

A guidance event is an update of the control covariance to reflect implementation of a trajectory correction. A correction is not performed deterministically, but only in a probabilistic sense. The guidance event computes expected correction covariances (Δv or thrust control), target error covariances before and after the guidance event, and the updated state control covariance.

The following sections will describe in more detail the analytical foundations of GODSEP.

6.1 AUGMENTED STATE

The augmented state discussed previously in TRAJ (Section 4.5) includes dynamic parameters besides the basic spacecraft position and velocity vectors. In GODSEP, the augmentation not only adds measurement

related parameters to this list, but also distinguishes between solve-for and consider. Solve-for parameters are directly estimated by the OD process. Consider parameters are system uncertainties which are recognized and accounted for in the estimation algorithm but are not estimated, usually because the process cannot be adequately modeled or there is a high correlation between two (or more) parameters which might cause numerical difficulties if both were solved-for.

The possible augmented parameters that can be either solved-for or considered are

- | | | |
|-------------|---|--|
| dynamic | { | <ul style="list-style-type: none"> • thrust bias (magnitude and pointing) • position and velocity of a selected planetary (ephemeris) body • gravitational constants of ephemeris body and/or sun |
| measurement | { | <ul style="list-style-type: none"> • tracking station locations • sensor bias (range, range-rate, etc.) |

Time varying thrust noise (magnitude and pointing) can only be considered in the standard CODSEP analysis, but can be solved-for (or considered) in the covariance integration option (PDOT in Section 6.2). A third possible category, in addition to solve-for and consider, is the ignore parameter used in generalized covariance analysis (Section 6.5).

The total ensemble of state uncertainties, or error covariance, including all augmented parameters, is formed by applying the expected value operation on state deviations from their reference values.

$$P = E [\delta \underline{x} \delta \underline{x}^T]$$

The covariance P contains uncertainties and their respective correlations for all parameters in the augmented state. There are two covariances,

corresponding to control and knowledge, which are computed in parallel through a GODSEP analysis. Starting with a-priori values, each P is modified between events by trajectory propagation effects, and at events by either OD (knowledge) or by guidance corrections (control).

6.2 COVARIANCE PROPAGATION

There are two methods available in GODSEP for propagating covariances between events: transition matrices (ϕ) and explicit covariance integration (PDOT). Although these two techniques were discussed in TRAJ, Sections 4.5 and 4.6, respectively, their importance in GODSEP requires additional explanation.

The most common form of covariance propagation, both in GODSEP and in other linear error analyses, makes use of transition matrices. This is because the ϕ 's are a characteristic of the trajectory, not of the covariance. A covariance $P(t_1)$ is propagated from time t_1 to t_2 by

$$P(t_2) = \phi_{21} P(t_1) \phi_{21}^T + \bar{Q}_{21} \quad (6-1)$$

where $\phi_{21} = \phi(t_2, t_1)$ and $\bar{Q}_{21} = \bar{Q}(t_2, t_1)$ is an effective process noise covariance.

Transition matrices can be stored, for example on magnetic tape, to be used for analyses of different error source levels and navigation strategies. Obviously, if ϕ 's have been computed between the intervals $t_0, t_1, t_2, \dots, t_N$, they can be used to propagate any P as long as the set of propagation times is a subset or equal to the original set. Transition matrices can always be chained to cover desired propagation intervals, for example, letting

$$\phi_{31} = \phi(t_3, t_1) = \phi(t_3, t_2) \phi(t_2, t_1)$$

$$\text{then } P(t_3) = \phi_{31} P(t_1) \phi_{31}^T + \bar{Q}_{31}$$

The method of computing and storing ϕ 's (on the STM file) over a grid of time points is used in GODSEP to facilitate parametric error analyses.

Since covariance propagation accounts for uncertainties in all dynamic parameters which have been augmented to the basic spacecraft state, the transition matrix must have the same augmentation. In actual operation, TRAJ provides a transition matrix containing only dynamic variations which GODSEP must augment with appropriate rows and columns of zeros (for measurement parameters) such that the total augmented ϕ is consistent with P.

An additional requirement for GODSEP is the modification of the thrust sensitivity matrix θ computed by TRAJ as part of the augmented transition matrix.

$$\theta(t_2, t_1) = \frac{\partial \underline{x}(t_2)}{\partial \underline{u}}$$

where \underline{u} are constant thrust controls (proportionality, cone and clock) over the interval (t_1, t_2) . The θ matrix is used in GODSEP to map thrust biases into spacecraft state uncertainties. However, GODSEP thrust biases refer to a single thruster. If more thrusters are operating, and if each operating thruster is assumed to be independent of all others in terms of bias, then the total effective bias is simply the single thruster bias divided by \sqrt{N} where N is the number of thrusters, or

$$\theta_1 = \sqrt{N} \theta_N$$

The effective process noise, \bar{Q} , is a very important conditioning term on the covariance propagation. Because a rigorous mathematical computation of Q involves (1) modeling of a process or processes which are ill-defined and (2) evaluation of complex double integrals, GODSEP uses a

simple analytic approximation. The effective process noise assumes that time-varying thrust errors appear as stationary first-order Gauss Markov processes. The more rigorous modeling is performed in the PDOT option to be discussed shortly. The relationship between P and \bar{Q} precludes the augmented state from containing time-varying thrust terms so that process noise takes on the appearance of constant parameters.

The effective noise over a time interval t_1 to t_2 directly affects only the spacecraft position and velocity uncertainties at t_2 .

$$\bar{Q}_{21} = 1/2 \Delta t \left[\begin{pmatrix} 0 & 0 \\ 0 & H_2 \end{pmatrix} + \bar{\Phi}_{21} \begin{pmatrix} 0 & 0 \\ 0 & H_1 \end{pmatrix} \bar{\Phi}_{21}^T \right]$$

where $\bar{\Phi}$ is the 6x6 state transition sub-matrix of the augmented transition matrix (ϕ),

$$\begin{aligned} H_1 &= g(t_1) P_\omega g^T(t_1) \\ H_2 &= g(t_2) \alpha P_\omega g^T(t_2) \\ P_\omega &= \begin{bmatrix} T_1 \sigma_1^2/N & 0 & 0 \\ 0 & T_2 \sigma_2^2/N & 0 \\ 0 & 0 & T_3 \sigma_3^2/N \end{bmatrix} \\ \alpha &= \begin{bmatrix} \alpha_1 & 0 & 0 \\ 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_3 \end{bmatrix} \quad \alpha_i = \begin{cases} 2, & \text{if } \Delta t > T_i \\ 0, & \text{if } \Delta t \leq T_i \end{cases} \end{aligned}$$

$$\Delta t = t_2 - t_1$$

$\sigma_1^2, \sigma_2^2, \sigma_3^2$ = variances in thrust proportionality, cone, clock, respectively.

N = number of operating thrusters

$g = \frac{\partial \dot{v}}{\partial \omega}$ (see Section 4.1 and Appendix 4)

T_1, T_2, T_3 = process noise correlation times

Thrust proportionality error is scaled by the number of thrusters because it is assumed that time-varying noise is independent for each thruster, just as bias is. Thrust pointing noise is also scaled because it is assumed to be caused by the thrust vector control system which currently consists of gimbaling each thruster independently.

This empirical model for \bar{Q} is generally effective over propagation intervals on the order of 50 days or less. Propagation events can be employed in GODSEP to break up longer intervals and to ensure the accuracy of \bar{Q} .

The second method of covariance propagation, PDOT, is used primarily to examine thrust noise effects. Process modeling is mathematically rigorous and includes augmentation of thrust noise parameters to the basic state. Recalling from Section 4.1, the linearized equations of motion,

$$\delta \dot{\underline{x}} = F \delta \underline{x}$$

and from Section 4.6, the corresponding covariance matrix differential equations

$$\dot{P} = FP + PF^T + Q$$

where \underline{x} is the augmented state which, for PDOT, includes at most spacecraft position and velocity, thrust biases, thrust noise and tracking station locations, F is the variation matrix, and Q is a white noise term which affects only the thrust noise directly. If thrust noise is omitted, then the integrated covariance would in theory be identical to a similarly augmented covariance propagated by transition matrices.

In PDOT, the time-varying noise is modeled as a stationary Gauss-

Markov process, as in \bar{Q} ,

$$\dot{\underline{\omega}} = \begin{bmatrix} -\frac{1}{T_1} & & & 0 \\ & \dots & & \\ 0 & & & -\frac{1}{T_6} \end{bmatrix} \underline{\omega} + \underline{W}$$

where $\underline{\omega}$ is a 6x1 vector of independent noise parameters corresponding to thrust proportionality, cone and clock, each of which is described by two processes having their own distinct correlation times (T). This permits the study of superimposed and multi-process effects. \underline{W} is a white noise component which drives the time varying noise (and defines the only non-zero term of Q which is $E [\underline{W} \underline{W}^T]$).

Since $\underline{\omega}$ is in the desired form of the linearized equations of motion, it can easily be augmented to the state vector (and covariance). Thus, $\underline{\omega}$ can be solved-for in the PDOT mode although in reality this practice is questionable because of the $\underline{\omega}$ modeling assumptions - who knows how thrust noise really behaves?

One of the more useful applications of PDOT is in refining the form of effective noise, \bar{Q} , for a particular mission and in verifying the explicit assumption in \bar{Q} of zero correlation between noise and state parameters.

Whether state transition matrices or PDOT is used for covariance propagation, an auxiliary computation is the vehicle mass uncertainty. Since mass and thrust magnitude uncertainties are indistinguishable in their trajectory effects, that is, they are correlated one to one, GODSEP has chosen to model thrust (acceleration proportionality) magnitude explicitly, and provide the approximate equivalent mass uncertainty as supplementary information.

Two types of mass uncertainty are distinguished: knowledge (estimated) and control (actual). Estimated mass uncertainty is the instantaneous knowledge error in thrust magnitude, bias and noise.

$$\text{estimated } \sigma_m^2 = (\sigma_{ab}^2 + \sigma_{an}^2) m$$

where σ_m^2 , σ_{ab}^2 , σ_{an}^2 are the variances in mass, thrust bias proportionality (from P_k) and thrust noise proportionality (from P_k or Q), respectively.

Actual mass uncertainty is the cumulative mass variation reflected by the control error covariance. The actual mass deviation from the reference at time $t + \Delta t$ based upon uncertainties from time t is

$$\begin{aligned} \text{actual } \sigma_m^2(t + \Delta t) &= \left[\sigma_m(t) + \frac{m}{c} \sigma_{ab} \Delta t \right]^2 + 2 \sigma_{an}^2 \tau^2 \times \\ &\times \left[1 - e^{-\frac{\Delta t}{\tau}} \right] \left[\frac{m}{c} \right]^2 \end{aligned}$$

where m , σ_{ab} and σ_{an} are averaged over the interval Δt , c is the exhaust velocity and τ the correlation time. σ_{ab} (and σ_{an} for PDOT) are obtained from the augmented control covariance. Accuracy of the mass variance computation depends upon the event schedule because GODSEP evaluates σ_m^2 from one event to the next.

6.3 MEASUREMENT TYPES

In a linear error analysis, the reference trajectory deterministically characterizes the motion of the S/C, and no real state vector estimation is explicitly performed in GODSEP. Rather, an orbit determination analysis estimates how well the state vector can be determined if the S/C were to move along the reference trajectory and were to be observed as directed by the analyst. In this sense, the term "orbit determination" refers to the calculation of a knowledge covariance based upon the processing of modeled observational data. This section of the Analytic Manual describes the data types and mathematical models that have been implemented in GODSEP. The next section will treat the problem of filter formulation and the process of updating the knowledge covariance.

When an observation is to be simulated in GODSEP, the knowledge covariance is propagated to the scheduled measurement point and is made available to be updated by the filter. Before this can happen, it is necessary to evaluate the observation matrix which relates the observables to the state vector. Given an arbitrary vector (or scalar) measurement $y = y(\underline{X})$ where \underline{X} is the total augmented state consisting of

$$\underline{X} = \begin{bmatrix} \underline{x} \\ \underline{s} \\ \underline{u} \\ \underline{v} \\ \underline{w} \end{bmatrix} = \begin{bmatrix} \text{spacecraft position and velocity} \\ \text{solve-for parameters} \\ \text{dynamic consider parameters} \\ \text{measurement consider parameters} \\ \text{ignore parameters} \end{bmatrix}$$

then the linearized measurement, which assumes small deviations from the nominal, is

$$\delta y = H_x \delta x + H_s \delta s + H_u \delta u + H_v \delta v + H_w \delta w$$

where $H_x = \frac{\partial y}{\partial x}$, ..., $H_w = \frac{\partial y}{\partial w}$ are the observation matrices, all of which are computed analytically in GODSEP.

GODSEP has the capability of processing the following measurement types:

- | | | |
|------------------|---|---|
| earth-based | } | <ul style="list-style-type: none"> o 2-way range o 2-way doppler o 3-way range o 3-way doppler o simultaneous 2-way and 3-way range o simultaneous 2-way and 3-way doppler o differenced 2-way and 3-way range o differenced 2-way and 3-way doppler o azimuth and elevation angles o right ascension and declination angles (of target body) |
| spacecraft based | } | <ul style="list-style-type: none"> o star-planet/target body angles o planet limb angles (apparent planet diameter) |

All earthbased data types which make observations of the S/C are applicable to both near earth and deep space missions; however as a practical matter, azimuth and elevation angles are normally used for near earth analysis only. Astronomical observations of the apparent right ascension and declination of the target body are used to determine ephemeris errors and can be made concurrently with earthbased tracking of the S/C. (Astronomical observations provide information about the state of the target body and indirectly imply information about the S/C motion if there are dynamic and/or measurement correlations between the

S/C and the target body). Spacecraft based observations have been formulated to model measurements of the target body with an onboard optical system. Hence, GODSEP permits an integrated navigational analysis for interplanetary missions with astronomical, radio, and onboard optical measurements all in one computer run.

For earthbased radio observations of the S/C, the program normally uses the standard DSN tracking stations located at Goldstone, Madrid, and Canberra. However, the locations of these stations may be changed by input and as many as six others added. For astronomical observations, the nominal observatory is assumed to be Kitt Peak in Arizona. As with the DSN stations, the location of the observatory may also be changed to model whatever real observatory the analyst chooses.

In order to display the analytic partials contained in the observation matrices, earthbased data types are separated into two categories according to their normal application for deep space or near earth missions. Thus, all mathematical models for range and range-rate data will be described as deep space data, although they are directly applicable, without reformulation, to the near earth problem. The right ascension and declination observations are also grouped in the deep space category although visual observations of near earth satellites could be made with a few minor changes to the model. Only azimuth and elevation angle measurements are specifically treated as near earth data, and a discussion of their observation partials will follow the development of the deep space data types. Finally, a description is given for the S/C based data types and the observation partials.

The following definitions of position and velocity vectors are necessary to relate the mission geometry to the observable quantities. All vectors are assumed to be column vectors and are expressed relative to an inertial, ecliptic coordinate frame, unless otherwise noted.

$\underline{x}, \dot{\underline{x}}$ = S/C heliocentric cartesian position and velocity

$\underline{x}_E, \dot{\underline{x}}_E$ = Earth heliocentric cartesian position and velocity

$\underline{x}_T, \dot{\underline{x}}_T$ = Target body heliocentric cartesian position and velocity

$\underline{x}_1, \dot{\underline{x}}_1$ = Station 1 geocentric cartesian position and velocity

$\underline{x}_2, \dot{\underline{x}}_2$ = Station 2 geocentric cartesian position and velocity

$\underline{\rho}_{-1}, \dot{\underline{\rho}}_{-1}$ = S/C position and velocity relative to Station 1

$\underline{\rho}_{-2}, \dot{\underline{\rho}}_{-2}$ = S/C position and velocity relative to Station 2

$\hat{\rho}_1, \hat{\rho}_2$ = Unit vectors defining direction of S/C from Stations 1 and 2 respectively

- $\rho_1, \dot{\rho}_1$ = S/C range and range-rate from Station 1
 $\rho_2, \dot{\rho}_2$ = S/C range and range-rate from Station 2
 $\rho_3, \dot{\rho}_3$ = 3-way range and range-rate
 $\Delta\rho, \Delta\dot{\rho}$ = Differenced 2-way and 3-way range and range-rate
 $\underline{s}_1, \underline{s}_2$ = Geocentric cylindrical coordinates of Stations 1 and 2, $\underline{s} = (r_s, \lambda, z)^T$
 \underline{z} = Zero vector, 3 x 1
 \underline{I} = Identity matrix, 3 x 3 unless noted otherwise

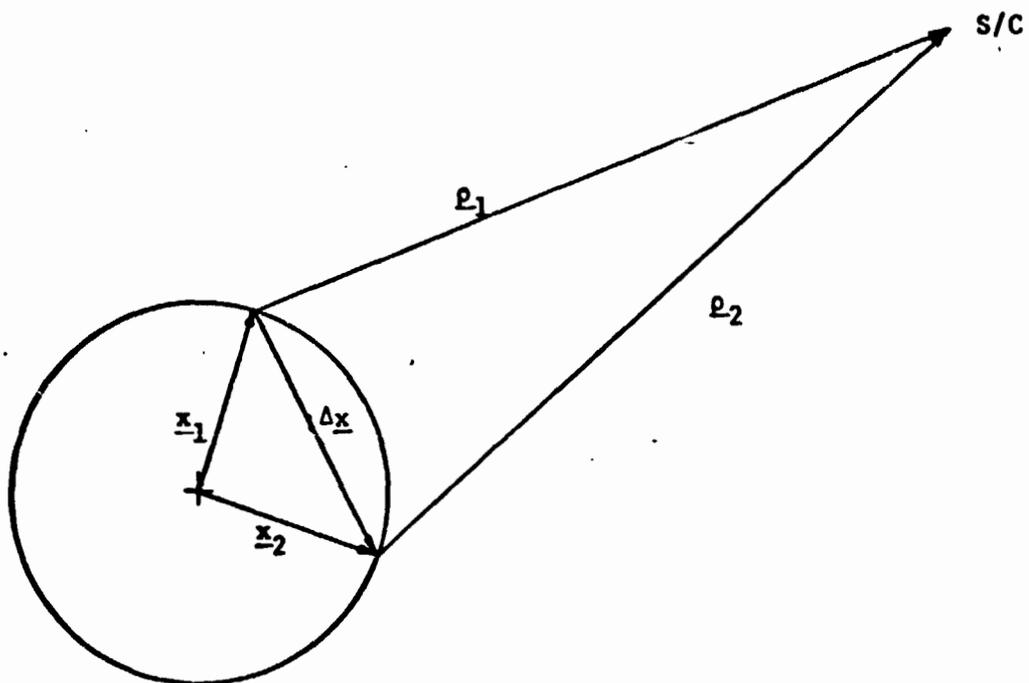


Figure 6-1. Tracking Geometry for Range and Range-Rate

We first note one identity which is used numerous times in the following derivations. Given the vector $\underline{a} = \underline{b} - \underline{c}$ and its corresponding unit vector $\hat{a} = \underline{a}/|\underline{a}|$,

$$\frac{\partial \underline{a}}{\partial \underline{b}} = \frac{1}{|\underline{a}|} [\mathbf{I} - \hat{a} \hat{a}^T] \quad (6-2.1)$$

Two-way range and range rate from Station 1 are modeled

$$\rho_1 = \underline{e}_1^T \hat{\rho}_1$$

$$\dot{\rho}_1 = \dot{\underline{e}}_1^T \hat{\rho}_1$$

where

$$\underline{e}_1 = \underline{x} - \underline{x}_E - \underline{x}_1$$

$$\dot{\underline{e}}_1 = \dot{\underline{x}} - \dot{\underline{x}}_E - \dot{\underline{x}}_1$$

Differentiation yields the following results.

$$\begin{aligned} \frac{\partial \rho_1}{\partial \underline{x}} &= \underline{e}_1^T \frac{\partial \hat{\rho}_1}{\partial \underline{x}} + \hat{\rho}_1^T \frac{\partial \underline{e}_1}{\partial \underline{x}} \\ &= \underline{e}_1^T \cdot \frac{1}{e_1} [\mathbf{I} - \hat{\rho}_1 \hat{\rho}_1^T] + \hat{\rho}_1^T \cdot \mathbf{I} \\ &= \hat{\rho}_1^T [\mathbf{I} - \hat{\rho}_1 \hat{\rho}_1^T] + \hat{\rho}_1^T \end{aligned}$$

$$\partial \rho_1 / \partial \underline{x} = \hat{e}_1^T$$

$$\partial \rho_1 / \partial \dot{\underline{x}} = z^T$$

or

$$\partial \rho_1 / \partial (\underline{x}, \dot{\underline{x}}) = [\hat{e}_1^T, z^T] \quad (6-2.2)$$

The remainder of the partials are produced in like manner but for brevity only the results will be printed

$$\partial \rho_1 / \partial s_1 = \frac{\partial \rho_1}{\partial (\underline{x}_1, \dot{\underline{x}}_1)} \cdot \frac{\partial (\underline{x}_1, \dot{\underline{x}}_1)}{\partial s_1}$$

$$\partial \rho_1 / \partial s_1 = - \frac{\partial \rho_1}{\partial (\underline{x}, \dot{\underline{x}})} \cdot \frac{\partial (\underline{x}_1, \dot{\underline{x}}_1)}{\partial s_1} \quad (6-2.3)$$

$$\partial \dot{\rho}_1 / \partial (\underline{x}, \dot{\underline{x}}) = \left[\frac{1}{e_1} (\dot{\rho}_1 - \dot{\rho} \hat{e}) \right]^T \quad (6-2.4)$$

$$\partial \dot{\rho}_1 / \partial s_1 = - \frac{\partial \dot{\rho}_1}{\partial (\underline{x}, \dot{\underline{x}})} \cdot \frac{\partial (\underline{x}_1, \dot{\underline{x}}_1)}{\partial s_1} \quad (6-2.5)$$

For use in Equations 6-2.3 and 6-2.5 above, we need the derivative of the instantaneous geocentric ecliptic cartesian state of the tracking station with respect to its cylindrical equatorial coordinates of spin

radius (r_s), longitude (λ) and z-height (z). If G represents the Greenwich hour angle at launch, t_0 the universal time (U.T.) at the launch epoch, t the U.T. at the current epoch, and ω the Earth's sidereal rotation rate, we have

$$\underline{x}_1 = E \begin{bmatrix} r_s \cos [\lambda + G + \omega (t-t_0)] \\ r_s \sin [\lambda + G + \omega (t-t_0)] \\ z \end{bmatrix}$$

$$\underline{x}_1 = E \begin{bmatrix} -\omega r_s \sin [\lambda + G + \omega (t-t_0)] \\ \omega r_s \cos [\lambda + G + \omega (t-t_0)] \\ 0 \end{bmatrix}$$

where E represents the 3 x 3 transformation from geocentric equatorial to geocentric ecliptic cartesian coordinates. This transformation is assumed constant. Even though the Earth's obliquity to the ecliptic does vary slightly, its effect is negligible over the duration of the missions for which these programs are used. Letting

$$\theta = \lambda + G + \omega (t-t_0)$$

$$\frac{\partial x_i}{\partial s_1} = \frac{\partial x}{\partial (r_s, \lambda, z)} = E \begin{bmatrix} \cos \theta & -r_s \sin \theta & 0 \\ \sin \theta & r_s \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6-2.6)$$

$$\frac{\dot{\partial x_i}}{\partial s_1} = E \begin{bmatrix} -\omega \sin \theta & -\omega r_s \cos \theta & 0 \\ \omega \cos \theta & -\omega r_s \sin \theta & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (6-2.7)$$

Three-way range and range-rate are measured with one station on the DSN uplink and another station on the downlink. Three-way data may be processed by itself, simultaneously with conventional two-way data, or as differenced two-way minus three-way data, also known as QVLBI (quasi-very long baseline interferometry). Three-way data types are modeled as the sum of the two-way types plus a timing error term for ranging and a frequency bias term for range-rate.

$$p_3 = p_1 + p_2 + c \Delta t$$

$$\dot{p}_3 = \dot{p}_1 + \dot{p}_2 + c \frac{\Delta f}{f}$$

where Δt is the timing error, c the speed of light, and $\Delta f/f$ the frequency bias term which results from drift error between the frequency

standards at the two separate tracking stations. The sensitivity partials for the three-way data types are formed by adding the partials computed for each station individually. The $c \Delta t$ and $c \Delta f/f$ terms are treated either as biases or part of the white noise term. The differenced data types are modeled:

$$\Delta \rho = \rho_1 - \rho_2 - c \Delta t$$

$$\Delta \dot{\rho} = \dot{\rho}_1 - \dot{\rho}_2 - c \Delta f/f$$

The partials for the differenced data types are formed by differencing the individual partials, with the following exception. Since

$$\frac{\partial \Delta \rho}{\partial \underline{x}} = \frac{\partial \Delta \dot{\rho}}{\partial \dot{\underline{x}}} = [\hat{e}_1 - \hat{e}_2]^T$$

and \hat{e}_1 and \hat{e}_2 are very nearly equal (as are e_1 and e_2) for interplanetary missions, we use the following substitutions

$$\Delta \underline{x} = \underline{x}_2 - \underline{x}_1$$

$$\Delta \rho = \frac{[\hat{e}_1 + \hat{e}_2]^T \Delta \underline{x}}{1 + \hat{e}_1^T \hat{e}_2}$$

$$\hat{e}_1 - \hat{e}_2 = [\Delta \underline{x} - \Delta \rho \hat{e}_2]/e_1$$

$$\frac{\partial \Delta \rho}{\partial \underline{x}} = \frac{\partial \Delta \hat{\rho}}{\partial \underline{x}} = [\Delta \underline{x} - \Delta \rho \hat{e}_2] / \rho_1 \quad (6-2.8)$$

For the astronomical observations, it is necessary to re-define, slightly, certain vectors in the table of vector definitions on Page 64-C. Namely, the vector $\underline{\rho}_1$ is computed as

$$\underline{\rho}_1 = \underline{x}_p - \underline{x}_F - \underline{x}_1$$

and is the position vector of the target body relative to tracking Station 1. Station 1 will now be taken to be the astronomical observatory from which the right ascension (α) and declination (δ) measurements are made. In order to compute the apparent α and δ , $\underline{\rho}_1$ must be rotated from its ecliptic representation into the geoequatorial system according to the transformation

$$\underline{\rho}'_1 = E^T \underline{\rho}_1$$

From the components of $\underline{\rho}'_1$, the right ascension of the target body is computed as

$$\alpha = \tan^{-1} (\rho'_{1y} / \rho'_{1x})$$

and the declination as

$$\delta = \sin^{-1} (\rho'_{1z} / \rho_1)$$

where $\rho_1 = | \underline{\rho}_1 |$.

The observation partials for the astronomical observations can be written as

$$H_s = \left[\frac{\partial (\alpha, \delta)}{\partial (\underline{x}_p, \underline{\dot{x}}_p)} \right]_{(2 \times 6)} = \left[\frac{\partial (\alpha, \delta)}{\partial (\underline{x}'_p, \underline{\dot{x}}'_p)} \right] \left[\frac{\partial (\underline{x}'_p, \underline{\dot{x}}'_p)}{\partial (\underline{x}_p, \underline{\dot{x}}_p)} \right]$$

where the second term on the right hand side of this equation is identified as

$$\left[\frac{\partial (\underline{x}'_p, \underline{\dot{x}}'_p)}{\partial (\underline{x}_p, \underline{\dot{x}}_p)} \right]_{66} = \begin{bmatrix} E^T & 0_{33} \\ 0_{33} & E^T \end{bmatrix}$$

Elements in the first matrix on the right hand side above are given by

$$\frac{\partial \alpha}{\partial x'_p} = -\sin \alpha / (e_1 \cos \delta)$$

$$\frac{\partial \alpha}{\partial y'_p} = +\cos \alpha / (e_1 \cos \delta)$$

$$\frac{\partial \alpha}{\partial z'_p} = \frac{\partial \alpha}{\partial x'_p} = \frac{\partial \alpha}{\partial y'_p} = \frac{\partial \alpha}{\partial z'_p} = 0$$

$$\frac{\partial \delta}{\partial x'_p} = -\cos \alpha \sin \delta / \rho_1$$

$$\frac{\partial \delta}{\partial y'_p} = -\sin \alpha \sin \delta / \rho_1$$

$$\frac{\partial \delta}{\partial z'_p} = \cos \delta / \rho_1$$

$$\frac{\partial \delta}{\partial x'_p} = \frac{\partial \delta}{\partial y'_p} = \frac{\partial \delta}{\partial z'_p} = 0$$

These partials are handled differently than the observation partials for other deep space data types. Since the astronomical observations are made on the target body whose state must be augmented to the usual S/C state vector as solve-for or dynamic consider parameters, the partials correspond to partitions H_s or H_u , respectively, in the augmented observation matrix. Furthermore, only when there are onboard optical observations or significant dynamical interactions between the S/C and the body is the S/C knowledge covariance affected by astronomical

observations. One final note about astronomical observations: GODSEP's mathematical model has been designed to handle both measurement noises and biases separately for each observable (α and/or δ).

The following definitions are used in the azimuth and elevation angle partials.

- α = S/C azimuth, measured positive from north toward east
(See Figure 6-2)
- β = S/C elevation
- $\underline{\rho}$ = S/C range vector from station
- $\hat{\rho}$ = Unit vector in $\underline{\rho}$ direction
- $\underline{\rho}_p$ = Projection of $\underline{\rho}$ onto plane normal to \underline{x}_s
- \underline{x} = Geocentric equatorial S/C position
- \underline{x}_c = Heliocentric ecliptic S/C position
- \underline{x}_s = Geocentric equatorial station position
- \hat{x}_s = Unit vector in \underline{x}_s direction
- \hat{w} = Unit vector orthogonal to \underline{x}_s and pole (local east from station)
- \hat{u} = Unit vector orthogonal to \underline{x}_s and w (local north from station)
- L = Transformation from equatorial to ecliptic coordinates.

For simplicity, all azimuth and elevation partials are derived in geocentric equatorial cartesian coordinates and then transformed to ecliptic.

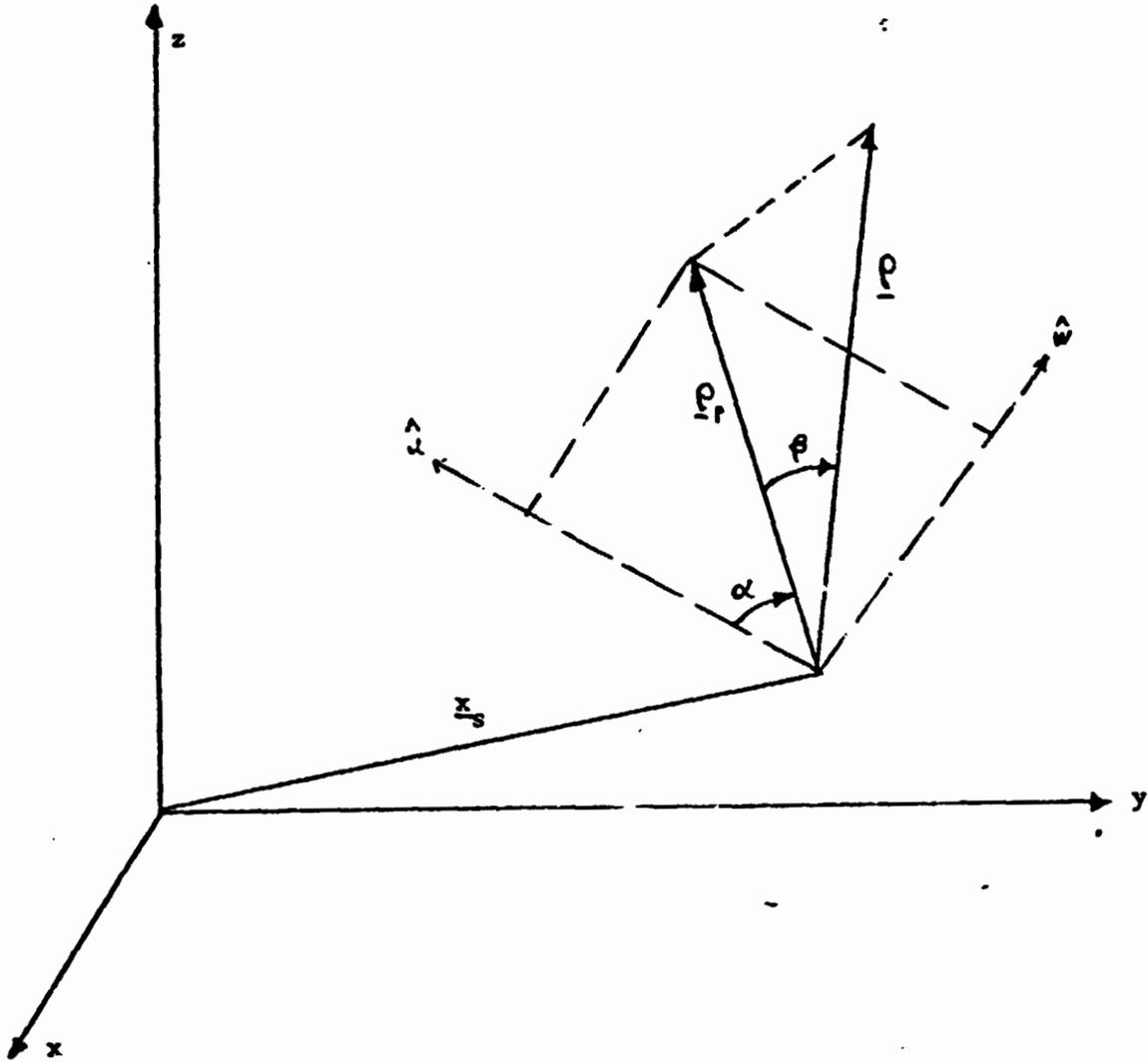


Figure 6-2. Tracking Geometry for Azimuth and Elevation Angles

Referring to Figure 6-2, we see that the projection of \hat{e} onto the x_p direction will have magnitude $\sin \beta$, or

$$\sin \beta = \hat{x}_s^T \hat{e}$$

$$\frac{\partial \beta}{\partial \underline{x}} = \frac{1}{\rho \cos \beta} \left[\hat{x}_s \cdot \sin \beta \hat{e} \right]^T \quad (6-2.9)$$

$$\frac{\partial \beta}{\partial x_s} = \frac{1}{|\underline{x}_s| \cos \beta} \left[\hat{e} - \sin \beta \hat{x}_s \right] \cdot \frac{\partial \beta}{\partial \underline{x}} \quad (6-2.10)$$

$$\frac{\partial \beta}{\partial s} = \frac{\partial \beta}{\partial x_s} \frac{\partial x_s}{\partial s} \quad (6-2.11)$$

Again referring to Figure 6-2, the projection of \hat{e} onto \hat{w} will have magnitude $\cos \beta \sin \alpha$, or

$$\sin \alpha = \sec \beta \left[\hat{w} \right]^T \hat{e}$$

$$\frac{\partial \alpha}{\partial \underline{x}} = \tan \alpha \tan \beta \frac{\partial \beta}{\partial \underline{x}} + \underline{a}^{-T}$$

$$\underline{a} = \left[\sec \alpha \sec \beta \hat{w} - \tan \alpha \hat{e} \right] / \rho$$

(6-2.12)

$$\frac{\partial \alpha}{\partial x_s} = \tan \alpha \tan \beta \frac{\partial \beta}{\partial x_s} + \left[\underline{b} - \underline{a} \right]^T$$

where

$$\underline{b} = \frac{1}{r_s} \left[\sec \alpha \sec \beta (\hat{w}_x \hat{e}_y - \hat{w}_y \hat{e}_x) \right] \hat{w}$$

(6-2.13)

$\hat{e}_x, \hat{e}_y, \hat{w}_x, \hat{w}_y$ are components are w and

$$\frac{\partial \alpha}{\partial \underline{x}_s} = \frac{\partial \alpha}{\partial \underline{x}_s} \frac{\partial \underline{x}_s}{\partial \underline{s}} \quad (6-2.14)$$

For use in Equations 6-2.11 and 6-2.13 above

$$\frac{\partial \underline{x}_s}{\partial \underline{s}} = \frac{\partial \underline{x}_s}{\partial (r_s, \lambda, z)} = \begin{bmatrix} x_{1s}/r_s & -x_{2s} & 0 \\ x_{2s}/r_s & x_{1s} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6-2.15)$$

where x_{1s} and x_{2s} are components of \underline{x}_s . Finally, the partials computed in equatorial coordinates must be transformed into ecliptic

$$\frac{\partial (\alpha, \beta)}{\partial \underline{x}_c} = \frac{\partial (\alpha, \beta)}{\partial \underline{x}} \cdot \frac{\partial \underline{x}}{\partial \underline{x}_c}$$

$$\frac{\partial (\alpha, \beta)}{\partial \underline{x}_c} = \frac{\partial (\alpha, \beta)}{\partial \underline{x}} \cdot E^T \quad (6-12.16)$$

For vehicle based optical measurements, we use the following definitions.

- \underline{x} = spacecraft heliocentric cartesian position
- \underline{x}_p = planet/target body heliocentric cartesian position
- $\underline{\rho}$ = $\underline{x}_p - \underline{x}$ = planet range vector
- \hat{d} = vector of star direction cosines

- \underline{p} = vector of target planet orbital elements (Keplerian)
 R_p = target planet radius
 γ = star-planet angle
 δ = apparent planet diameter measurement - angle subtended by planet disc at the spacecraft
 \underline{z} = zero vector, 3 x 1.

Star-planet angle partials:

$$\cos \gamma = \hat{d}^T \hat{p}$$

$$\frac{\partial \gamma}{\partial \underline{x}} = \left[\frac{1}{\rho \sin \gamma} \left(\hat{d} - \cos \gamma \hat{p} \right) \right]^T \quad (6-12.17)$$

$$\frac{\partial \gamma}{\partial \underline{x}_s} = - \frac{\partial \gamma}{\partial \underline{x}} \quad (6-12.18)$$

and if the ephemeris body elements are Keplerian rather than Cartesian

$$\frac{\partial \gamma}{\partial \underline{p}} = \frac{\partial \gamma}{\partial \underline{x}_p} \cdot \frac{\partial \underline{x}_p}{\partial \underline{p}} \quad (6-12.19)$$

where the partials of cartesian to Keplerian elements are computed numerically.

Apparent planet diameter partials:

$$\sin \frac{\delta}{2} = \frac{R_p}{\rho}$$

$$\frac{\partial \delta}{\partial \underline{x}} = - \frac{2}{\rho} \tan \frac{\epsilon}{2} \hat{\rho}^T \quad (6-12.20)$$

$$\frac{\partial \delta}{\partial \underline{x}_p} = - \frac{\partial \delta}{\partial \underline{x}} \quad (6-12.21)$$

$$\frac{\partial \delta}{\partial \underline{p}} = \frac{\partial \delta}{\partial \underline{x}_p} \cdot \frac{\partial \underline{x}_p}{\partial \underline{p}} \quad (6-12.22)$$

For any data type which has a bias,

$$y = H \delta X + b$$

$$\frac{\partial y}{\partial b} = 1.0 \quad (6-12.23)$$

6.4 FILTER

After the knowledge covariance has been propagated to a measurement time, and the observation matrices are computed, the OD filter can perform its function of estimating the set of solve-for parameters (non-deterministically) and updating the knowledge covariance accordingly.

There are two types of filters available, Kalman-Schmidt (K-S) and weighted least squares (WLS), plus capability for a third filter, to be established by the user. K-S is the most commonly used filter because it treats consider parameters in a realistic fashion.

The filter updating process requires computation of several matrices. First, the propagated estimation error (knowledge) covariance P at the measurement event time is

$$P = \begin{bmatrix} P_x & C_{xs} & \dots & C_{xw} \\ C_{xs}^T & P_s & & \vdots \\ \vdots & & P_u & \\ C_{xw}^T & \dots & & P_v & C_{vw} \\ & & & C_{vw}^T & P_w \end{bmatrix}$$

where P_x, \dots, P_w are the covariances of the S/C state, \dots , ignore parameters, respectively, and C_{xs}, \dots, C_{vw} are the cross-covariances between appropriate augmented parameters. The observation matrix H defined in the previous section is

$$H = \begin{bmatrix} H_x \\ H_s \\ H_u \\ H_v \\ H_w \end{bmatrix}$$

The measurement residual matrix J is defined as

$$J = HPH^T + R$$

where R is a diagonal matrix containing variances of the measurement white noise. For example, a simultaneous 2-way/3-way range measurement would look like

$$R = \begin{bmatrix} \sigma_{2R}^2 & 0 \\ 0 & \sigma_{2R}^2 + \sigma_{3R}^2 \end{bmatrix}$$

where σ_{2R}^2 is the 2-way range noise variance and σ_{3R}^2 is the additional 3-way range noise variance due to timing synchronization. For a single star-planet angle measurement, R would be a scalar

$$R = \sigma_y^2 + \sigma_\theta^2 / r^2$$

where σ_r^2 and σ_b^2 are the optical resolution and planet or body center finding noise variances, respectively, and r is the spacecraft range to the planet.

The updated covariance (P^+) after the measurement has been processed (and in theory after the state estimate has been updated) is in general

$$P^+ = (I - KH) P (I - KH)^T + KRK^T \quad (6-3)$$

where K is the filter gain.

KALMAN-SCHMIDT

The filter gain for K-S is straightforward

$$K = PH^T J^{-1}$$

Since only estimated parameters can be updated by the OD process, the entries of K corresponding to consider and ignore parameters are zeroed out, that is, $K = [K_x K_s 0 0 0]^T$. The updated covariance is then formed by Equation 6-3.

WEIGHTED LEAST SQUARES

The sequential, or recursive weighted least squares (WLS) algorithm implemented in GODSEP is equivalent to a batch WLS filter if there is no

process noise. Since process noise is a significant part of low thrust analysis, the WLS filter must be used recursively, because it has no batch equivalent. The sequential WLS consider filter acknowledges consider parameters only in the covariance update (Eqn. 6-3) and not in the gain matrix calculation. Therefore a set of reference covariances for the state and solve-for parameters must be maintained at all times. This set also represents the filter analysis as it would be in non-consider form.

Thus, the WLS filter computation requires three operations: (1) propagation of the reference covariance to the measurement event, (2) computation of the filter gain and (3) updating both the reference and knowledge covariances.

- (1) The reference covariance (\hat{P}) consists of

$$\hat{P} = \begin{bmatrix} P_x & C_{xs}^T \\ C_{xs} & P_s \end{bmatrix}$$

and is initialized at the a-priori values. Thereafter, it is propagated from one measurement to the next by

$$\hat{P}_{k+1} = \phi \hat{P}_k \phi^T$$

where ϕ is the augmented transition matrix corresponding to the \underline{x} and \underline{s} parameters. \hat{P} is computed in parallel with the actual knowledge covariance P .

- (2) Given \hat{P} at the measurement event, the WLS filter gain is

$$K = \hat{P} \hat{H}^T (\hat{H} \hat{P} \hat{H}^T + R)^{-1}$$

where $\hat{H} = \begin{bmatrix} H_x \\ H_s \end{bmatrix}$.

- (3) The reference covariance is updated, after measurement processing, by

$$\hat{P}^+ = (I - KH) \hat{P}$$

and the knowledge covariance P is updated by Eqn. 6-3.

Thus, at measurement events, the OD filter updates the knowledge covariance to simulate taking a tracking measurement, processing the measurement in an orbit determination algorithm, estimating desired parameters and reducing (or updating) the knowledge uncertainty to reflect this new information about the trajectory.

6.5 GENERALIZED COVARIANCE

The function of any filtering algorithm is to process available measurement information and produce a best estimate of the spacecraft state and any parameters that are being solved-for. Best is usually defined in a statistical sense, such as the minimum variance processes used in differing forms in the weighted least squares and Kalman-Schmidt filters. But in practice, filter performance is dependent on how well the assumptions used in the filter definition approximate real-world processes, because all error sources cannot be modeled, nor can those that are modeled ever be modeled exactly. Therefore, each filter must be evaluated not only on its ability to produce small error covariances in the resulting estimated state, but also be as insensitive as possible to errors in its model assumptions.

Generalized covariance error analysis is a useful tool for studying filter sensitivity. For generalized covariance studies, two sets of knowledge errors are carried during the orbit-determination process. Assumed knowledge uncertainties are those generated by the filtering algorithm

according to the mathematical model and all the assumed errors input to it. True knowledge uncertainties represent the effect the filtering algorithm has on actual state estimation when the real-world error sources are not the same as those assumed by the filter. Evaluating filter sensitivity to a model assumption involves comparing the resultant effect on assumed and true uncertainties of a modeling mismatch between the filter and real-world uncertainties. This modeling mismatch is accomplished in one of two ways; true a priori uncertainties may be set at levels other than assumed levels or the true state may be augmented by a vector of ignore parameters--parameters whose uncertainties are recognized by the true covariance analysis, but which are completely ignored by the assumed filter analysis.

The filter that is least sensitive to a model mismatch is determined on the basis of two criteria. First, which filter yields the smallest true estimation errors. Second, for which filter are the true errors most closely approximated by the errors predicted by the filter covariance analysis. Thus, given a mission with a specific set of model mismatches, if two different filters produce equivalent true errors, then the superior filter is the one whose assumed errors are closest to the true ones. Similarly, if the resultant assumed errors of the two filters are equivalent, the superior filter is the one with the least true estimation uncertainty. Generally, qualitative judgments are required because several sets of mismatches must be studied, and the relative performance of the filters may vary.

In error analysis, generalized covariance is a filter sensitivity study tool that is normally available only in a simulation program. It is

accomplished in GODSEP with a minor increase in core and computational time compared to a full simulation and has the additional advantage of generating ensemble true state statistics rather than a single sample as in a simulation. The only disadvantage of generalized covariance is that it uses the same linearized dynamic and observation models as the assumed filter analysis, and can therefore not study problems that arise from nonlinearities.

The actual operation of generalized covariance in GODSEP requires that a standard error analysis be run first. The filter gains, associated with the assumed knowledge uncertainties, are stored on disc or tape. Now the error analysis with all the same measurement events is repeated. Only this time, a-priori uncertainty levels and measurement noise are modified, and ignore parameters are added, to the extent of desired mismodeling. At each measurement event, Eqn. 6-3 is applied to what is now the true knowledge covariance using the appropriate stored filter gain. The true covariance analysis thus proceeds in analogous fashion to the previous assumed covariance analysis. Obviously, many mismodeling conditions can be studied with the same filter by repeating the generalized covariance analysis.

6.6 GUIDANCE

Although the knowledge covariance is modified by measurement events, the control covariance, which represents the ensemble of actual deviations from the desired or reference trajectory, will grow without bound. The only process which will reduce the control covariance is a guidance event that is, the design and execution of a trajectory correction, either

impulsive Δv or low thrust.

Low thrust guidance represents an update of the nominal thrust controls (magnitude, direction and cut-off time). In terms of system cost and efficiency, it is better to use the existing low thrust propulsion system for guidance than to add auxiliary means, for example high thrust chemical engines to produce impulsive Δv . Of course, certain problems inherent in low thrust propulsion, in particular terminal controllability, may force the addition and use of an auxiliary chemical propulsion system.

In mathematical terms, given a trajectory state deviation, $\delta \underline{x}_0 = \delta \underline{x}(t_0)$, where t_0 is the guidance epoch, we wish to null out the effects of $\delta \underline{x}_0$ by making a bias type correction $\delta \underline{u}$ to the nominal thrust controls. To be efficient, the correction is applied over some finite interval $[t_0, t_c]$ such that the target error $\delta \underline{T}$, caused by $\delta \underline{x}_0$, at some final time (t_f) is removed. For linear analysis, we seek the guidance matrix Γ , such that

$$\delta \underline{u} = \Gamma \delta \underline{x}_0$$

The linear ensemble of thrust control corrections is then

$$\begin{aligned} U &= E [\delta \underline{u} \delta \underline{u}^T] \\ U &= \Gamma E [\delta \underline{x}_0 \delta \underline{x}_0^T] \Gamma^T \end{aligned} \quad (6-4)$$

In GØDSEP, the trajectory error ensemble $E [\delta \underline{x}_0 \delta \underline{x}_0^T]$ is the control covariance $P_c^-(t_0)$. Using P_c^- represents a pessimistic sizing of the thrust corrections because only the known trajectory error (not to be confused with the knowledge error) can be removed. The known error generally corresponds to the control error as long as $P_c^-(t_0) \gg P_k(t_0)$ where P_k is the knowledge covariance.

C-2

To compute the guidance matrix Γ we first compute the sensitivity matrices

$$S(t_f, t_o) = \frac{\partial \underline{I}(t_f)}{\partial \underline{u}}$$

$$V(t_f, t_o) = \frac{\partial \underline{I}(t_f)}{\partial \underline{x}(t_o)}$$

or,

$$S = \left[\frac{\partial \underline{I}(t_f)}{\partial \underline{x}(t_f)} \right] \phi(t_f, t_o) \theta(t_c, t_o)$$

$$V = \left[\frac{\partial \underline{I}(t_f)}{\partial \underline{x}(t_f)} \right] \phi(t_f, t_c) \phi(t_c, t_o)$$

where the first matrix in S and V is formed by numerical differencing and the second two matrices (ϕ and θ) are obtained from transition matrices generated by the trajectory propagation routine (Section 6.2). If variable time of arrival is desired, the control array \underline{u} is augmented with the arrival time and the S matrix is augmented by $\dot{\underline{x}}(t_f)$, relative to the target.

The guidance matrix can now be defined by

$$\Gamma = -W_u^2 S^T [S W_u^2 S^T]^{-1} V \quad \text{If } N_u \geq N_T \quad (6-5.1)$$

$$\Gamma = -[S^T W_T^2 S]^{-1} S^T W_T^2 V \quad \text{If } N_T > N_u \quad (6-5.2)$$

where N_u and N_T are the number of parameters in the control and target set, respectively, and W_u and W_T are diagonal weighting matrices for the control and target parameters, respectively. The first form of Γ , 6-5.1, reflects a minimum quadratic control correction and the second, 6-5.2, corresponds to minimum quadratic target error.

Generally, there are more control parameters than targets with the exception of two cases: (1) terminal approach to the target where controllability drops off rapidly; and (2) the application of control constraints which effectively reduces the set of available controls.

If constraints on the control corrections are imposed, then Γ must be suitably modified. First, the unconstrained Γ is computed along with the ensemble unconstrained control corrections,

$$U = \Gamma P_c^{-1} \Gamma^T \quad (6-4) \text{ again}$$

Each diagonal component of U , σ_u^2 , is compared against its constraint value $(\delta u_{MAX})^2$. If σ_u is greater than δu_{MAX} , then the appropriate row of Γ is scaled by $\delta u_{MAX}/\sigma_u$. The total control set (and guidance matrix) is then separated into two subsets: unconstrained controls, δu_1 and Γ_1 , and constrained controls, δu_2 and Γ_2 .

$$\delta \underline{u} = \begin{bmatrix} \delta u_1 \\ \delta u_2 \end{bmatrix} \quad \Gamma = \begin{bmatrix} \Gamma_1 \\ \frac{\delta u_{MAX}}{\sigma_u} \Gamma_2 \end{bmatrix}$$

The new control corrections are computed with (6-4), (actually only the remaining unconstrained controls are computed) the test for constraints are made, Γ is modified again, and the entire process is repeated until all constraints are met, or there are no more controls left. The guidance corrections are executed (figuratively) at time t_0 , that is, are uplinked to the spacecraft, but apply over the entire guidance interval $[t_0, t_c]$.

Execution of the guidance updates causes the control covariance to diminish from t_0 to t_c whereupon it begins to grow again. Guidance accuracy is measured by how much the control covariance can be reduced at t_c which depends upon how well the thrust corrections were designed. The limiting

accuracy of maneuver design is the knowledge error or covariance at t_0 . Thus, the post maneuver control covariance at t_c is the propagated knowledge covariance (as per Eqn. 6-1) from t_0 to t_c ,

$$P_c^+(t_c) = \phi(t_c, t_0) P_k(t_0) \phi^T(t_c, t_0) + \bar{Q}(t_c, t_0) \quad (6-6)$$

In GDDSEP, to denote guidance execution, $P_c^+(t_0)$ is set equal to $P_k(t_0)$. This is equivalent in effect at t_c to applying Eqn. 6-6. However, it means the value of P_c^+ in the guidance interval is not valid. This is a relatively minor problem compared to the reduced burden on computational storage and logic.

One exception to setting $P_c^+ = P_k$ occurs when there are more targets than available controls, which often happens when control constraints have been activated. In this case, there will be some non-zero target error that was not removed by the guidance corrections. This implies that not all of P_c^- was removed. Hence, the post maneuver control covariance must include the residual state error.

$$P_c^+ = P_k + \left\{ V^T [V V^T]^{-1} [V + S r] \right\} P_c^- \left\{ V^T [V V^T]^{-1} [V + S r] \right\}^T$$

As long as no more guidance events are executed between $[t_0, t_c]$, updating the control covariance at t_0 is theoretically no different than using Eqn. 6-6 at t_c . However, if another guidance event is scheduled in the burn interval, say at t_1 , then a somewhat different logic is applied to size the correction. It is assumed that the first guidance event between t_0 and t_c is a primary maneuver. Subsequent guidance events in this interval are considered to be vernier maneuvers, representing refinements of the thrust corrections computed in the primary maneuvers.

For vernier maneuvers, θ and ϕ are redefined for the vernier burn interval and Γ is computed as in Eqn. 6-5. The guidance updates are computed using Eqn. 6-4 with the knowledge gained since the primary (or previous vernier), that is,

$$E \left[\delta \underline{x}_1 \delta \underline{x}_1^T \right] = P_c^+(t_1) - P_k(t_1)$$

Recall from the previous discussion that $P_c^+(t_1)$ is usually the propagated knowledge from the previous guidance event.

A measure of guidance effectiveness is the estimated target error before and after the maneuvers.

$$\text{before guidance correction, } E \left[\delta \underline{T} \delta \underline{T}^T \right] = V P_c^-(t_0) V^T$$

$$\text{after guidance correction, } E \left[\delta \underline{T} \delta \underline{T}^T \right] = V P_c^+(t_0) V^T.$$

This simple measure assumes, of course, that no further dynamic error will occur from t_0 to the target time t_f .

An important part of the guidance and navigation process is the time interval from the last navigation measurement used for guidance design to the actual time of guidance implementation (t_0). The time interval (or delay) is necessary for ground processing of all previous measurements, estimation of the S/C state, designing the thrust updates to correct the trajectory, and execution of the updates. Typical intervals are 3 to 15 hours, and are usually critical only in the terminal mission phase where trajectory controllability (with respect to thrust controls) diminishes rapidly. This time delay is user specified for each guidance event.

Impulsive ΔV guidance is very similar to low thrust except for a zero burn interval ($t_0 = t_c$). The delta-velocity is treated as if it were a control correction $\delta \underline{u}$, that is,

$$\Delta \underline{y} = \Gamma \delta \underline{x}_0$$

To compute Γ , the sensitivity matrix S is first partitioned into position and velocity submatrices,

$$S = [A \vdots B]$$

where $A = \frac{\partial \underline{I}(t_f)}{\partial \underline{x}_0}$ and $B = \frac{\partial \underline{I}(t_f)}{\partial \underline{v}_0}$

If the target vector \underline{T} has only two components, e.g., B·T and B·R, then $|\Delta Y|$ is minimized and

$$\Gamma = \begin{bmatrix} -B^T (BB^T)^{-1} A & \vdots & -B^T (BB^T)^{-1} B \end{bmatrix}$$

If \underline{T} has three components, e.g., the position vector at t_f , then

$$\Gamma = \begin{bmatrix} -B^{-1}A & \vdots & -I \end{bmatrix}$$

The computation of ensemble velocity correction, U in Equation 6-4, follows directly. As in low thrust guidance, the pre-maneuver control covariance $P_c^-(t_0)$ is used to size U .

$$U = E[\Delta Y \Delta Y^T] = \Gamma P_c^-(t_0) \Gamma^T$$

Execution errors related to low thrust control updates are neglected because they are second order effects compared to thrust error associated with the nominal thrust profile. However, ΔY execution errors are taken into account because impulsive maneuvers often occur during ballistic or coasting portions of the mission and can represent a significant contribution to trajectory error. In order to compute ΔY execution errors, the most probable ΔY is first determined by the Hoffman-Young approximation (Reference 8). Let $\lambda_1, \lambda_2, \lambda_3$ be the eigenvalues of the ΔY covariance, U , and \hat{a} be the largest eigenvector of U , define

$$A = \lambda_1 + \lambda_2 + \lambda_3$$

$$B = \lambda_1 \lambda_2 + \lambda_1 \lambda_3 + \lambda_2 \lambda_3$$

$$P = \sqrt{\frac{2A}{\pi}} \left[1 + \frac{B(\pi-2)}{A^2 \sqrt{5.4}} \right]$$

then the probable ΔY is $E[\Delta Y] = \rho \hat{z} = \begin{bmatrix} \Delta V_1 \\ \Delta V_2 \\ \Delta V_3 \end{bmatrix}$

Now the 3 x 3 ΔY execution error covariance \tilde{Q} is composed of

$$\tilde{Q}_{11} = \Delta V_1^2 \sigma_m^2 + \frac{\Delta V_2^2 \rho^2 \sigma_u^2 + \Delta V_1^2 \Delta V_3^2 \sigma_B^2}{\rho_{xy}^2} \quad (6-7)$$

$$\tilde{Q}_{12} = \tilde{Q}_{21} = \Delta V_1 \Delta V_2 \left[\sigma_m^2 - \frac{\rho^2 \sigma_u^2 + \Delta V_3^2 \sigma_B^2}{\rho_{xy}^2} \right]$$

$$\tilde{Q}_{13} = \tilde{Q}_{31} = \Delta V_1 \Delta V_3 \left[\sigma_m^2 - \sigma_B^2 \right]$$

$$\tilde{Q}_{22} = \Delta V_2^2 \sigma_m^2 + \frac{\Delta V_1^2 \rho^2 \sigma_u^2 + \Delta V_2^2 \Delta V_3^2 \sigma_B^2}{\rho_{xy}^2}$$

$$\tilde{Q}_{23} = \tilde{Q}_{32} = \Delta V_2 \Delta V_3 \left[\sigma_m^2 - \sigma_B^2 \right]$$

$$\tilde{Q}_{33} = \Delta V_3^2 \sigma_m^2 + \rho_{xy}^2 \sigma_B^2$$

$$\text{where } \rho_{xy}^2 = \Delta V_1^2 + \Delta V_2^2$$

$$\sigma_m^2 = \sigma_p^2 + \frac{\sigma_r^2}{2}$$

$$\sigma_r^2, \sigma_p^2 = \Delta V \text{ resolution and proportionality variances}$$

$$\sigma_\alpha^2 = \text{ecliptic (X-Y) pointing variance}$$

$$\sigma_\beta^2 = \text{out of ecliptic (z) pointing variance.}$$

As in low thrust guidance, the post-maneuver control covariance P_c^+ (t_o) is set equal to the knowledge covariance P_k (t_o) corrupted by the ΔV execution errors,

$$P_c^+ (t_o) = P_k (t_o) + \begin{bmatrix} 0 & 0 \\ 0 & \tilde{Q} \end{bmatrix}$$

Pre and post maneuver target uncertainties are computed in equivalent fashion to low thrust guidance.

7.0 SIMSEP Analysis

The trajectory simulation mode SIMSEP has been designed to provide deterministic analysis of ballistic and low thrust missions. Computationally, SIMSEP imitates "real" trajectories in the presence of a wide variety of environmental and system uncertainties. A primary objective is to deduce expected or probabalistic behavior of the real mission by studying a relatively small subset of simulated missions.

The purpose of this section is to discuss the key analytic concepts in SIMSEP. This will be done in two parts: 1) by discussing the principal algorithms, and 2) by outlining the basic computational structure. Although many algorithms used in SIMSEP are similar in function to algorithms used in TØPSEP and GØDSEP, their specific applications here warrant an extended discussion of their underlying theories.

7.1 Program Scope and Methods

Before proceeding with a step-by-step description of the algorithms and computational structure, it is worthwhile to compare the essential similarities and differences between GØDSEP and SIMSEP. Unlike the error analysis mode which works exclusively with error ensembles and a reference trajectory, the simulation mode actually formulates many discrete examples of the "real world" or "actual" trajectory. Each of these is propagated, in a deterministic sense, by the same trajectory integrator, integrating the same equations of motion. However, many variables and parameters appearing in these

equations are subjected to random alterations, corresponding to discrete uncertainties. Hence, each deterministic simulation of a mission is different according to the effects of the sampled errors.

Operationally, SIMSEP does not sample each error in succession and propagate trajectories with just one error source active at a time. Rather, all are initialized by differing amounts for each actual trajectory. Thus the averaged effect of all error sources acting in concert can be estimated by repeating the mission simulation process a sufficient number of times. This is the essence of the Monte Carlo method and is the basis of the simulation approach to determining how trajectory nonlinearities and uncertainties can affect the G&N process.

Perhaps the best example to illustrate the fundamental differences between the methods used in GODSEP and SIMSEP is the problem of propagating, or mapping, an error covariance from one point to another along the reference trajectory. It will be recalled that the principal method for propagating a covariance in GODSEP is by the state transition matrix mapping, namely,

$$P_{k+1} = \Phi_{k+1, k} P_k \Phi_{k+1, k}^T$$

where $\Phi_{k+1, k}$ represents the state transition matrix and P_k and P_{k+1} are covariances at t_k and t_{k+1} , respectively. When there is dynamic process noise, an effective process noise matrix, $\bar{Q}_{k+1, k}$, is also added, (See Section 6.2). The state transition matrix is generally computed simultaneously with the trajectory by integrating

the variational equations. However, the variational equations are based on linearized expansions of the differential equations of motion and neglect all second and higher order terms. Hence, a state transition matrix mapping of covariances must theoretically be limited to mapping covariances within the envelope of linearity surrounding the reference trajectory. Rarely is this assumption true at all points along an interplanetary trajectory, especially a low thrust trajectory. Covariances propagated by this means are subject to error whenever a region of significant trajectory nonlinearities is encountered.

On the other hand, the method for propagating a control covariance in SIMSEP is not plagued by these effects, although it has its own peculiar shortcomings. The SIMSEP approach to this problem relies on the Monte Carlo method where a multitude of sample trajectories are propagated between the two time points in question. The trajectory state vector data at t_{k+1} are processed and accumulated in such a way that the covariance can be reconstructed by standard statistical calculations. Hence, SIMSEP maps a covariance as a statistical ensemble calculated from many data points and not as a simple mathematical entity like GODSEP.

Therein lies the primary drawback to the Monte Carlo method and to the use of SIMSEP for general G&N analysis. Although the Monte Carlo method will, in theory, converge to the exact covariance, the rate of convergence tends to be extremely slow. For accurate statistics, inordinately large amounts of computer time are often

necessary to perform the many trajectory propagations.

Although both GØDSEP and SIMSEP are preflight mission analysis programs used to identify the general G&N subsystem characteristics, they are usually used at differing phases in the development of an overall systems analysis. For the purposes of a preliminary systems design, a GØDSEP analysis is the most cost effective means of evaluating the basic G&N subsystem requirements. As such, SIMSEP is generally relegated to verifying the linear analysis results, and only in the advent of serious nonlinearities is the simulation mode called upon for more extensive studies.

7.2 Definitions and Concepts

The first important concept in SIMSEP and common to all MAPSEP modes is the reference trajectory, denoted by $\underline{X}_R = \underline{X}_R(t)$. The reference trajectory is computed under some set of "reference integrating conditions" to satisfy desired targets at mission's end. Moreover, \underline{X}_R represents a deterministic solution to the equations of motion for the assumed dynamic and systems models. For SIMSEP, the initial state and reference integrating conditions, i.e., ephemeris parameters, thruster characteristics, etc., are read as input since it is assumed that they have already been computed as output from a TOPSEP analysis.

A second quantity important in SIMSEP and common to GØDSEP is the control error covariance, P_C . Generally, an a priori control covariance is defined at injection (or at the starting point of the mission being studied). This matrix mathematically describes the

distribution of real state errors relative to the initial reference trajectory state. In SIMSEP, it is implicitly assumed that the probability distribution of these errors is Gaussian with zero mean. Once an a priori control has been given, it is randomly sampled to form an error vector, $\delta \underline{X}_A$, which corresponds to a deviation of the actual trajectory state relative to the reference. At the same time, error sources associated with the host of other dynamical and systems uncertainties are also sampled to create the so-called "real world integrating conditions". For the actual trajectory state vector, this procedure may be written as

$$\underline{X}_A = \underline{X}_R + \delta \underline{X}_A$$

where $\delta \underline{X}_A$ is a deviation obtained by sampling P_G . Utilizing these integrating conditions, the actual trajectory state, \underline{X}_A , is propagated from point to point as a discrete example of an actual trajectory.

The third critical variable used in both GODSEP and SIMSEP is the knowledge error covariance, P_k . This matrix is propagated in the error analysis from measurement to measurement where it is systematically updated according to the filtering algorithm. In SIMSEP, instantaneous evaluations of P_k are input at each guidance event and are left unchanged throughout a given run since there is no explicit orbit determination process modeled. However, the knowledge covariance, like the control covariance, is sampled to formulate an error vector. This error vector determines the error

in the estimated state relative to the actual trajectory state, and is used to compute an estimated state vector by

$$\underline{x}_E = \underline{x}_A + \underline{e}$$

where \underline{e} is the sampled error vector from P_k . If other parameters are estimated during the orbit determination calculations, they are included as augmentation parameters. These too are sampled in order to formulate a set of "estimated world integrating conditions".

With each of these key quantities having been defined, it is worthwhile to mention why each is important in a simulation run. The reference trajectory, for example, serves to define the mean for all actual trajectories, as well as defining the reference target conditions used during guidance. The actual trajectory is, of course, the mathematical representation of the real motion and is carried from event to event until the final target is reached. On the other hand, the estimated trajectory is used exclusively for re-targeting the actual trajectory back to the desired targets and is computed only during guidance.

7.3 Guidance

One of the principal purposes of SIMSEP is the detailed examination of nonlinear trajectory effects, especially as they bear upon the guidance problem. In this section, the fundamental concepts underlying linear and nonlinear guidance will be presented, and the implementation of these concepts into algorithms will be discussed. Beforehand, a careful distinction between targeting and guidance

must be drawn since MAPSEP has algorithms for both and since many of the basic steps and operations are similar. Whereas a targeting problem is solved by formulating an entire control strategy for a complete mission, a guidance problem assumes that a solution to the targeting problem has already been found. Furthermore, the current trajectory which is to be corrected is assumed to be in a "close neighborhood" to the original reference solution. Hence, the control changes computed by a guidance law are expected to be small refinements to the original controls, even in the presence of nonlinearities.

7.3.1 Linear Guidance

The linear guidance option in SIMSEP is analogous to the guidance used in GØDSEP except that it applies to a discrete trajectory error as opposed to an ensemble of errors. For both modes, the linear guidance matrix is the same. To compute a guidance matrix, a sensitivity matrix is evaluated between the point of the guidance event and the target about the reference trajectory. This matrix of linear partials relates control changes to target deviations and is used to map estimated trajectory errors, δx_E , into control updates, according to the guidance laws:

$$1) \quad \underline{\Delta v} = \Gamma \underline{\delta x_E}, \text{ for impulsive corrections, and}$$

$$2) \quad \underline{\Delta u} = \Gamma \underline{\delta x_E}, \text{ for low thrust.}$$

In spite of its overall simplicity and ease of implementation,

the advantages of linear guidance are off-set somewhat by its drawbacks. First, the trajectory error, $\underline{\delta X}_2$, must lie within the envelope of linearity for this to be a valid method. Whenever this is violated, the resulting guidance correction can be invalidated. Furthermore, a linear guidance correction is executed without iterations. In fact, with this guidance there is no direct evaluation of a control correction's effectiveness in reducing target error. Only if the updated trajectory is propagated to the target can the resulting target error be determined, and even then there is no recourse for making further corrections if the original correction is ineffective.

7.3.2 Linear Impulsive Guidance

The essence of impulsive guidance is founded on the mapping relations which propagate arbitrary linear deviations relative to some known trajectory into new deviations at some later time. Clearly, this is a property of the state transition matrix which maps a six component state deviation, $\underline{\delta X}_k$, evaluated at t_k into a new deviation, $\underline{\delta X}_{k+1}$, at t_{k+1} , by the equation,

$$\underline{\delta X}_{k+1} = \Phi_{k+1,k} \underline{\delta X}_k \quad (7-1)$$

If t_{k+1} is the target time and t_k is the time of the guidance event, then $\Phi_{k+1,k}$ can also map state vector changes, like an impulsive velocity correction, into state changes at the target.

However, in most analysis the actual target conditions are

specified in terms of target variables such as B-plane parameters, Keplerian elements, etc., instead of X, Y, \dots, \dot{Z} state coordinates. Fortunately, the target variables are functions of the final trajectory state and it is possible to generate a differential transformation of the form

$$\underline{\delta T} = \eta \underline{\delta X}_{k+1} \quad (7-2)$$

which transforms differential coordinate changes into target variable variations. In the above equation, η represents a matrix of linear partials of the form

$$\eta = \frac{\partial (T_1, T_2, \dots, T_n)}{\partial (X, Y, Z, \dot{X}, \dot{Y}, \dot{Z})_{k+1}} \quad (7-3)$$

where there are n -target variables, T_1, T_2, \dots, T_n and six state components. By substituting Eq. 7-1 into 7-2, a relation for mapping state changes at t_k into target changes at t_{k+1} is obtained, that is,

$$\underline{\delta T} = \eta \Phi_{k+1,k} \underline{\delta X}_k$$

Performing the indicated matrix multiplication and replacing $\eta \Phi_{k+1,k}$ with N , the equation becomes

$$\underline{\delta T} = N \underline{\delta X}_k \quad (7-4)$$

where N has dimensions $(n \times 6)$.

With this background, the impulsive guidance problem can be stated most simply as a determination of a velocity change, $\underline{\Delta V}$,

which, when added to $\underline{\delta x}_k$, nulls the target error, $\underline{\delta T}$. Again writing Eq. 7-4 but in a partitioned format,

$$\underline{\delta T} = N(1) \underline{\delta r}_k + N(2) \underline{\delta v}_k,$$

it is recognized that $\underline{\delta T}$ can be made zero by adding some appropriate $\underline{\Delta V}$ to $\underline{\delta v}_k$, i.e.

$$N(1) \underline{\delta r}_k + N(2) (\underline{\delta v}_k + \underline{\Delta V}) = 0.$$

For the case of three-variable impulsive guidance, i.e., three unique targets, the solution for $\underline{\Delta V}$ is given as

$$\underline{\Delta V} = -N(2)^{-1} N(1) \underline{\delta r}_k - \underline{\delta v}_k,$$

provided $N(2)$ is nonsingular. Note that this can be re-written as

$$\underline{\Delta V} = \begin{bmatrix} -N(2)^{-1} N(1) & -I \end{bmatrix} \begin{pmatrix} \underline{\delta r}_k \\ \underline{\delta v}_k \end{pmatrix} \quad (7-5a)$$

or
$$\underline{\Delta V} = \Gamma \underline{\delta x}_k, \quad (7-5b)$$

the desired guidance law.

For the case where there are two target variables instead of three, the problem has more controls (3-velocity components) than end conditions and a generalized inverse which minimizes the magnitude of the velocity correction is used according to

$$\underline{\Delta V} = -N(2)^T \left[N(2) N(2)^T \right]^{-1} N(1) \underline{\delta r}_k \\ - N(2)^T \left[N(2) N(2)^T \right]^{-1} \underline{\delta v}_k$$

where $N(1)$ and $N(2)$ are non-square matrices with dimensions $(n \times 3)$.

Again this relation can be re-written as

$$\underline{\Delta V} = \begin{Bmatrix} -N(2)^T \left[N(2) N(2)^T \right]^{-1} N(1) \\ \left[-N(2) N(2)^T \right]^{-1} \end{Bmatrix} \begin{pmatrix} \underline{\delta r}_k \\ \underline{\delta v}_k \end{pmatrix} \quad (7-6a)$$

or $\underline{\Delta V} = \Gamma \underline{\delta x}_k. \quad (7-6b)$

Algorithms based on Eqs. 7-5a and 7-6a are the basis of the linear impulsive guidance contained in subroutine LGUID. The guidance matrix, Γ , for either the two or three variable cases, are computed as outlined above and the state vector deviation, $\underline{\delta x}_k$, is calculated as the error in the estimated trajectory state relative to the reference, namely,

$$\underline{\delta x}_k = \underline{\delta x}_E = \underline{x}_E - \underline{x}_R,$$

evaluated at the guidance event.

7.3.3 Low Thrust Linear Guidance

The low thrust linear guidance law has the same format as the impulsive law except control changes are made to the vector of low thrust control variables, \underline{u} . Another difference is that the low

thrust acceleration acts slowly to bring about state changes; hence, the low thrust linear guidance matrix operates in an integrated fashion over a fixed trajectory segment to redirect the motion. Otherwise, low thrust guidance is simply an extension of the basic methods discussed above.

The most complex part of computing low thrust corrections is the determination of the guidance matrix, Γ . This matrix depends not only on the trajectory dynamics between the maneuver point and the target, but it also depends on the trajectory response to control changes, i.e., controllability. As before, the first step is to integrate the reference trajectory from the guidance point to the target, evaluating the augmented state transition matrix. In SIMSEP, the transition matrix is computed by integrating the variational equations as was discussed in Section 4-5. By selectively partitioning the transition matrix, the requisite sensitivity matrices relating state and control variable deviations to future state deviations are obtained.

In terms of partitions in the augmented state transition matrix, state deviations at the target time, t_{k+1} , are given by

$$\underline{\delta X}_{k+1} = \mathbf{I}_{k+1,k} \underline{\delta X}_k + \mathbf{C} \underline{\delta u} \quad (7-7)$$

where $\mathbf{I}_{k+1,k}$ is a state transition matrix as defined in Eq. 7-1 and \mathbf{C} is a matrix which maps control variable deviations into state changes at t_{k+1} . $\underline{\delta u}$ in Eq. 7-7 corresponds to a set of thrust control biases. \mathbf{C} can also be written as

$$\Theta = \frac{\partial (x, y, z, \dot{x}, \dot{y}, \dot{z})_{k+1}}{\partial (u_1, u_2, \dots, u_m)} \quad (7-7.5)$$

and is seen to be $(6 \times m)$ where m is the number of controls. Following the same line of reasoning as was given in Section 7.3.2, it is recognized that partials of target variable variations with respect to control variable changes are needed. Hence, Eq. 7-7 is multiplied by the transformation matrix η (See Eq. 7-3) to obtain,

$$\delta T = \eta \bar{f}_{k+1,k} \frac{\delta x_k}{\delta x_k} + \eta \Theta \delta u. \quad (7-8)$$

Therefore, the guidance problem is reduced to finding a Δu which when added to δu will make $\delta T = 0$. For convenience, it is assumed that δu is either zero or that it can be solved-for during the orbit determination; thus permitting Eq. 7-8 to be re-written as

$$\eta \Theta \Delta u + \eta \bar{f}_{k+1,k} \frac{\delta x_k}{\delta x_k} = 0. \quad (7-9)$$

For the problem where the number of controls (m) equals the number of targets (n) and the matrix η has an inverse, the solution for Δu in Eq. 7-9 is

$$\Delta u = -\Theta^{-1} \bar{f}_{k+1,k} \frac{\delta x_k}{\delta x_k} \quad (7-10)$$

where $\frac{\delta x_k}{\delta x_k}$ is the deviation of the estimated state vector relative to the reference at the guidance event. From Eq. 7-10 it is easy to see that the desired guidance matrix for this particular case is given as

$$\Gamma = \Theta^{-1} \Phi_{k+1,k}$$

and is a (6x6) matrix.

The problem is complicated somewhat when the number of targets is less than the number of controls. In this case, a generalized, or pseudo-, inverse matrix operation is used, and the transformation matrix η does not drop out. Nevertheless, a solution is obtained by determining the Δu that makes $\delta T = 0$. Letting $A = \eta \Theta$ and $B = \eta \Phi_{k+1,k}$ in Eq. 7-9, a particular solution (out of the infinity of solutions) is given as

$$\Delta u = -A^T [AA^T]^{-1} B \delta x_E. \quad (7-11)$$

This particular choice of Δu also minimizes the magnitude of the control change (See Section 5.3). Therefore, the desired guidance law can be written as

$$\Delta u = \Gamma \delta x_E$$

where $\Gamma = -A^T [AA^T]^{-1} B$.

As before, the computational steps described here are implemented in LGUID.

7.3.4 Nonlinear Guidance

Nonlinear guidance parallels in many respects a targeting problem where an iterative, linear algorithm is used to determine control changes. The primary difference is that the trajectory is

assumed to be reasonably close to the reference. By reasonable, it is meant to imply that the algorithm should be able to redirect the s/c motion to the designated target by making a few iterations (three or four). In practice, a real trajectory can deviate widely from the reference and thereby require as many as eight to ten iterations before the guidance algorithm is able to compensate. For situations where convergence is not achieved after many iterations, the guidance is said to be divergent. The real criterion for qualifying a maneuver as divergent is somewhat subjective and established by the analyst. In some cases, an extremely slow rate of convergence on the part of the linear correction scheme can be attributed to non-adaptive iteration logic.

Mathematically, divergence implies that the real world integrating conditions acted in such a way that the guidance algorithm was unable to rectify the motion. Physically, divergence suggests that something in the dynamics or s/c system has been mismodeled, or under-designed, and that it has interactions with the other systems to cause wide, usually nonlinear, deviations from the reference mission. From the G&N point of view, it is these missions that are often of the greatest interest. They identify potential problems either in the baseline configuration or with the navigation and operational procedures. In many instances, divergence in the guidance can be traced to some well-understood phenomena, e.g., controllability, trajectory non-linearities, suboptimal schedule of guidance events, etc., but a clear identification and resolution of these problems in terms of

changes to the system design and/or operational procedures may not be so straightforward.

The basic computational steps taken in a single iteration are as follows: First, an estimate of the actual state vector and the corresponding estimated integrating conditions are obtained from the simulated orbit determination logic. The estimated state is integrated to generate the estimated trajectory between the guidance point and the target. At the targeted stopping conditions, an estimated target error is computed by

$$\Delta \underline{T} = \underline{T}_R - \underline{T}_E$$

where \underline{T}_E and \underline{T}_R are the target variables on the estimated and reference trajectories, respectively. Next, a sensitivity matrix, S , of target variations with respect to control variations is computed about the estimated trajectory according to the matrix relation,

$$S = \eta \theta$$

where η is the state to target transformation defined in Eq. 7-3 and where θ is that partition in the augmented state transition matrix which maps control changes into state variations (Eq. 7-25). The matrix S has the format,

$$S = \frac{\partial (T_1, T_2, \dots, T_n)}{\partial (u_1, u_2, \dots, u_m)}$$

and has dimensions $(n \times m)$, where n is the number of targets and m

the number of controls. With S it is possible to relate control changes at the maneuver to target variable changes according to

$$\Delta \underline{T} = S \Delta \underline{u}. \quad (7-12)$$

If the matrix S is square, i.e., the number of controls and targets are equal, then the solution to (7-12) requires only a single matrix inversion, namely,

$$\Delta \underline{u} = S^{-1} \Delta \underline{T}. \quad (7-13)$$

However, in most practical cases, S is a nonsquare matrix ($m > n$) and a generalized inversion must be used, i.e.

$$\Delta \underline{u} = S^T [SS^T]^{-1} \Delta \underline{T}. \quad (7-14)$$

Note that Eqs. (7-13) and (7-14) again assume the form of a linear guidance relation $\Delta \underline{u} = \Gamma \Delta \underline{T}$.

Once $\Delta \underline{u}$ has been determined, the updated controls are used to generate a new estimated trajectory to see if the target errors have decreased. This overall process is repeated until the target errors are made less than some specified tolerances, or until a maximum number of allowable iterations has occurred. In SIMSEP, the principal measure of target error is given by a so-called quadratic error function defined by

$$Q = \sum_{j=1}^n \left[\frac{\Delta T(j)}{T_{TOL}(j)} \right]^2$$

where $\Delta T(j)$ is the j^{th} component of the target error vector and

$T_{TOL}(j)$ is the j^{th} component of a vector of target error tolerances. Convergence occurs in the nonlinear guidance whenever Q is made less than one.

Guidance divergence is said to have occurred if the quadratic error function is greater than the backup convergence criterion (AOK) after iterating a maximum number of times ($NMAX$). Divergence also occurs if the quadratic error function increases on three successive predicted corrections. As far as the Monte Carlo mission currently being executed, divergence is considered to be catastrophic, and the mission is ended. As a backup, weak convergence occurs if strong convergence fails to be satisfied but Q is less than AOK . In this way, the nonlinear guidance algorithm can be tolerant of "near misses" without bringing the mission to a halt.

Another feature included in the nonlinear guidance logic is the ability to weight certain low thrust controls more than others. This permits the user an added flexibility whereby he can effect a normalization of disparities in the units associated with controls. In addition, the user has the option to arbitrarily weight some controls more heavily, based upon knowledge and experience gained during the trajectory targeting process.

Algorithms based on the computational steps outlined above are implemented in NLGUID. Both delta-velocity maneuvers and low thrust corrections are handled by essentially the same logic with Δu in

Eqs. (7-13) and (7-14) being a velocity update for impulsive guidance.

7.4 Simulated Orbit Determination

SIMSEP, in the strictest sense of the word, is not a complete "simulation" in that an explicit orbit determination process is not included in the computational algorithms. The problem of estimating a state vector is done by sampling a knowledge covariance in much the same way as it samples a control covariance or an ephemeris error covariance. Simply stated, an augmented knowledge covariance is sampled to obtain an error in the estimated state vector, \underline{x} , relative to the actual trajectory state, \underline{x}_A . Therefore, the estimated state vector is given by

$$\underline{x}_e = \underline{x}_A + \underline{e}$$

Likewise, parameters which have been augmented to the state and estimated during the orbit determination process are also computed.

Typically, the knowledge error covariances which are read as input to SIMSEP have been computed in an equivalent GØDSEP run, they are equivalent in the sense that the same trajectory and sequence of guidance events are evaluated. With this procedure, there is an added advantage of permitting a direct comparison of guidance results computed in GØDSEP and SIMSEP and their dependencies on the same state estimation results.

There are several reasons why an explicit orbit determination capability has not been included in SIMSEP. Primarily, the

estimation process is not as subject to trajectory nonlinearities as is the guidance process. This is because the estimation errors are generally small and well within the envelope of linearity. In addition, this method of simulating orbit determination minimizes the computational complexity of the program, while at the same time representing a cost effective means of performing an effective state estimation.

7.5 Thrust Process Noise

Digressing briefly to discuss the actual trajectory again, there is one very important process related to the generation of a real trajectory that is either ignored or modeled by an effective process in the other modes. This is the time-correlated thrust noise. These independent stochastic processes corrupt the commanded thrust controls, i.e., thrust magnitude, cone and clock angles, as small time-correlated perturbations. Each stochastic parameter is modeled in SIMSEP as a Gauss-Markov sequence which is computed during the actual trajectory integration. At time point t_{k+1} , the vector \underline{u}_{k+1} of stochastic parameters is given by

$$\underline{u}_{k+1} = A \underline{u}_k + \underline{w}_{k+1}$$

where \underline{u}_k has been evaluated at t_k . \underline{u}_k is assumed to remain constant over the interval $\Delta t = t_{k+1} - t_k$, with its effect being determined by the coefficient matrix, A.

$$A = \begin{bmatrix} e^{-\Delta t/\tau_1} & & & 0 \\ & e^{-\Delta t/\tau_2} & & \\ & & \ddots & \\ 0 & & & e^{-\Delta t/\tau_N} \end{bmatrix}$$

where $\tau_1, \tau_2, \dots, \tau_n$ are correlation times associated with each corresponding stochastic parameter. \underline{w}_{k+1} is a vector of white noise terms which have statistics dictated by the requirement that the process remain stationary; namely

$$\sigma_{w_j}^2 = (1 - e^{-2\Delta t/\tau_j}) \sigma_{u_j}^2$$

where $\sigma_{u_j}^2$ is the variance associated with the j^{th} component of the \underline{u}_{k+1} vector. During the integration, \underline{u}_{k+1} is evaluated at the start, the half-interval, and the end of a normal integration step.

7.6 Guidance Execution Errors

Once that a guidance correction has been formulated, the execution of that correction must be performed to affect the actual trajectory. The commanded correction computed by the guidance is an idealized set of control changes which are invariably corrupted by execution errors. For a low thrust control change, the execution errors are actually built into the thrust process through the thrust

biases and the dynamic process noise. However, for an impulsive maneuver, an explicit set of logic to corrupt the commanded delta-velocity change must be implemented.

In general there are three basic execution errors which are modeled for impulsive maneuvers: 1) pointing, 2) resolution, and 3) proportionality. Given the commanded delta-velocity vector, $\underline{\Delta v}_c$, in its heliocentric representation the in-and-out of the ecliptic plane angles are determined as,

$$\alpha = \tan^{-1} (\Delta v_c(2) / \Delta v_c(3))$$

$$\beta = \sin^{-1} (\Delta v_c(3) / |\underline{\Delta v}_c|).$$

Specified pointing angle errors are sampled to formulate changes in the commanded angles ($\delta\alpha$ and $\delta\beta$) to simulate orientation errors for the actual delta-velocity, $\underline{\Delta v}_A$. Likewise, errors specified as proportional to the maneuver magnitude, δP , and as a minimum measurable resolution of a maneuver magnitude, δr , are also sampled and added to the commanded correction. The actual velocity change is given as

$$|\underline{\Delta v}_A| = |\underline{\Delta v}_c| + \delta P + \delta r$$

for the actual velocity magnitude. The actual velocity vector is given as

$$\Delta v_A(1) = |\underline{\Delta v}_A| \cos(\alpha + \delta\alpha) \cos(\beta + \delta\beta)$$

$$\Delta v_A(2) = |\underline{\Delta v}_A| \sin(\alpha + \delta\alpha) \cos(\beta + \delta\beta)$$

$$\Delta v_A(3) = |\underline{\Delta v}_A| \sin(\beta + \delta\beta)$$

8.0 REFERENCES

1. "Final Report for NAS1-11686, Low Thrust Orbit Determination Program," P. Hong, et al, NASA CR-112256, December, 1972.
2. "Guidance and Navigation Analysis for Solar Electric 1979 Encke Flyby and 1981 Encke Rendezvous Mission (Final Report for Rockwell Subcontract, M3-201008)," G. Shults and P. Hong, MCR-73-182, July, 1973.
3. "System Design Impact of Guidance and Navigation Analysis for a SEP 1979 Encke Flyby," P. Hong, G. Shults, R. Loain, Presented at AIAA 10th Electric Propulsion Conference, October, 1973.
4. "Astronautical Guidance," R. Battin, McGraw-Hill, 1964.
5. "The Gradient Projection Method for Nonlinear Programming, Part I - Linear Constraints", J. B. Rosen, J. Soc. Ind. Appl. Math., No. 8, 1967, pp. 181-217.
6. "The Gradient Projection Method for Nonlinear Programming, Part II - Nonlinear Constraints", J. B. Rosen, J. Soc. Ind. Appl. Math., No. 8, 1961, pp. 514-532.
7. Program to Optimize Simulated Trajectories (POST), Final Report. G. L. Brauer, D. E. Cornick, R. T. Steinhoff, and R. Stevenson; National Aeronautics and Space Administration, Contract No. NAS1-12165; October, 1973.
8. "Approximation to the Statistics of Midcourse Velocity Corrections", L. Hoffman and G. Young, NASA TND-5381.

APPENDIX 1.

9.1 Conic Equations For Position And Velocity In Elliptical
And Hyperbolic Orbits

Given: \underline{r}_0 , \underline{v}_0 , t_0 and μ

Find: \underline{r} and \underline{v} at time t .

From the initial conditions we can find the inverse semi-major axis

$$\frac{1}{a} = \frac{2}{r_0} - \frac{v_0^2}{\mu}$$

The mean angular motion

$$n = \sqrt{\frac{\mu}{|a|^3}}$$

and relationships for the eccentricity and eccentric anomaly for elliptical orbits ($a > 0$)

$$e \cdot \sin E_0 = \frac{\underline{r}_0 \cdot \underline{v}_0}{\sqrt{\mu a}}$$

$$e \cdot \cos E_0 = \frac{r_0 v_0^2}{\mu} - 1$$

or for hyperbolic orbits ($a < 0$)

$$e \cdot \sinh H_0 = \frac{\underline{r}_0 \cdot \underline{v}_0}{\sqrt{-\mu a}}$$

$$e \cdot \cosh H_0 = \frac{r_0 v_0^2}{\mu} - 1$$

To find the position and velocity along the conic orbit. Keplers equation must be solved for the change in eccentric anomaly.

Newton's method is used to solve the equation iteratively. Let

Newton's method be given in the form

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

Then for elliptical orbits

$$x = E - E_0$$

$$f(x) = x + e \cdot \sin E_0 (1 - \cos x) - e \cdot \cos E_0 \sin x - n(t - t_0)$$

and for hyperbolic orbits

$$x = \exp(H - H_0)$$

$$f(x) = \frac{1}{2} e \cdot \exp(H_0) \cdot x + \frac{1}{2} e \cdot \exp(-H_0) \frac{x}{x+1} - \ln(x+1) - n(t - t_0).$$

The position and velocity at time t in elliptical orbits is given by

$$\underline{r} = \left\{ 1 - \frac{a}{r_0} [1 - \cos(E - E_0)] \right\} \underline{r}_0 + \frac{1}{n} \left\{ \sin(E - E_0) - e \cdot (\sin E - \sin E_0) \right\} \underline{v}_0$$

$$\underline{v}_0 = -\frac{\sqrt{\mu \cdot a}}{r r_0} \sin(E - E_0) \cdot \underline{r}_0 + \left\{ 1 - \frac{a}{r} [1 - \cos(E - E_0)] \right\} \cdot \underline{v}_0$$

and for a hyperbolic orbit

$$\underline{r} = \left\{ 1 - \frac{a}{r_0} [\cosh(H - H_0) - 1] \right\} \underline{r}_0 + \frac{1}{n} \left\{ e \cdot (\sinh H - \sinh H_0) - \sinh(H - H_0) \right\} \cdot \underline{v}_0$$

$$\underline{v} = -\frac{\sqrt{-\mu a}}{r r_0} \sinh(H - H_0) \underline{r}_0 + \left\{ 1 - \frac{a}{r} [\cosh(H - H_0) - 1] \right\} \underline{v}_0$$

APPENDIX 2

9.2 A Generalized 4th Order Runge-Kutta Algorithm With Runge's Coefficients For A Matrix System Of First Order Differential Equations.

The 4th Order Runge-Kutta formula for numerically integrating first order Differential Equations of the form

$$y' = f(x, y)$$

is

$$y_{k+1} = y_k + \frac{h_k}{6} (f_1 + 2 f_2 + 2 f_3 + f_4) \quad (1)$$

where h is the stepsize, x is the independent variable, y is the dependent variable and

$$f_1 = f'(x_k, y_k)$$

$$f_2 = f' \left(x_k + \frac{h_k}{2}, y_k + \frac{h_k \cdot f_1}{2} \right)$$

$$f_3 = f' \left(x_k + \frac{h_k}{2}, y_k + \frac{h_k \cdot f_2}{2} \right)$$

$$f_4 = f'(x_k + h_k, y_k + h_k \cdot f_3)$$

To generalize these equations for an $m \times n$ Matrix system of first order differential equations, write equation (1) as

$$y_{k+1}(i, j) = y_k(i, j) + \frac{h_k}{6} \left[f_1(i, j) + 2 \cdot f_2(i, j) + 2 \cdot f_3(i, j) + f_4(i, j) \right]$$

and this can be written as

$$Y_{k+1} = Y_k + \frac{h_k}{6} (F_1 + 2 \cdot F_2 + 2 \cdot F_3 + F_4)$$

where

$$F_1 = F'(x_k, Y_k)$$

$$F_2 = F'(x_k + \frac{h_k}{2}, Y_k + \frac{h_k}{2} \cdot F_1)$$

$$F_3 = F'(x_k + \frac{h_k}{2}, Y_k + \frac{h_k}{2} \cdot F_2)$$

$$F_4 = F'(x_k + h_k, Y_k + h_k \cdot F_3)$$

APPENDIX 3

9.3 Newton's 3rd Order Divided Difference InterpolationPolynomial.

Given: $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ and (x_4, y_4)

Find: A third Order Polynomial that fits the given points

Construct the following table

x_1	y_1			
		$[x_2, x_1]$		
x_2	y_2	$[x_3, x_2]$	$[x_3, x_2, x_1]$	
		$[x_4, x_3]$	$[x_4, x_3, x_2]$	$[x_4, x_3, x_2, x_1]$
x_3	y_3	$[x_4, x_3]$	$[x_4, x_3, x_2]$	
x_4	y_4			

where

$$[x_{i+1}, x_i] = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \quad , i = 1, 2, 3$$

$$[x_{j+2}, x_{j+1}, x_j] = \frac{[x_{j+2}, x_{j+1}] - [x_{j+1}, x_j]}{x_{j+2} - x_j} \quad , j = 1, 2$$

$$[x_{k+3}, x_{k+2}, x_{k+1}, x_k] = \frac{[x_{k+3}, x_{k+2}, x_{k+1}] - [x_{k+2}, x_{k+1}, x_k]}{x_{k+3} - x_k} \quad , k = 1$$

Newton's 3rd Order Divided Difference Interpolation Polynomial

has the following form,

$$y(x) = y_1 + (x-x_1) \cdot [x_2, x_1] + (x-x_1) \cdot (x-x_2) \cdot [x_3, x_2, x_1] + \\ (x-x_1) \cdot (x-x_2) \cdot (x-x_3) \cdot [x_4, x_3, x_2, x_1]$$

while a 3rd Order Polynomial can be written

$$y(x) = ax^3 + bx^2 + cx + d$$

To find a, b, c and d, the coefficients of the x terms in the Newton's Divided Difference Polynomial can be equated to a, b, c and d, such that

$$a = [x_4, x_3, x_2, x_1]$$

$$b = - (x_3, x_2, x_1) \cdot a + [x_3, x_2, x_1]$$

$$c = (x_3, x_2 + x_3, x_2, x_1) \cdot a - (x_2 + x_1) \cdot [x_3, x_2, x_1] + [x_2, x_1]$$

$$d = -(x_1, x_2, x_3) \cdot a + x_1 x_2 [x_3, x_2, x_1] - x_1 [x_2, x_1] + y$$

Now a third order polynomial can be fitted to the four points and y can be determined from a given x or a maximum or minimum can be found from the following values of x,

$$x = \frac{-b + \sqrt{b^2 - 3ac}}{3a}$$

is minimum

and

$$x = \frac{-b - \sqrt{b^2 - 3ac}}{3a}$$

is maximum

APPENDIX 4

9.4 Analytic Expressions for Terms in the F_A Matrix

In Section 4.5 it was shown that the augmented state transition matrix, ϕ_A , is computed by integrating the matrix differential equation,

$$\dot{\phi}_A = F_A \phi_A$$

In order to efficiently integrate this expression, it is necessary to have analytic representations for the individual elements of F_A .

F_A has been identified in equation 4-6 as a matrix of first order partial derivatives obtained by expanding the equations of motion. In concise symbolism, F_A may be written as,

$$F_A = \frac{\partial \underline{f}_A}{\partial \underline{x}_A}$$

where \underline{f}_A was defined in equation 4-5 and \underline{x}_A is the augmented dynamic state. For an analysis where covariances are propagated by the state transition matrix, i.e., the STM mode, the (maximum) component vectors of \underline{x}_A are defined as:

$$\underline{x}_A = \begin{bmatrix} \underline{r} \\ \underline{v} \\ \underline{u} \\ \underline{r}_p \\ \underline{v}_p \\ \mu_p \\ \mu_s \end{bmatrix} = \begin{bmatrix} \text{s/c position vector} \\ \text{s/c velocity vector} \\ \text{thrust control vector} \\ \text{ephemeris body position vector} \\ \text{ephemeris body velocity vector} \\ \text{ephemeris body gravitational constant} \\ \text{solar gravitational constant} \end{bmatrix}$$

and the corresponding F_A matrix, in partitioned format, is

$$F_A = \begin{bmatrix} \begin{bmatrix} 0 & I_{33} \\ f_{33} & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ g_{33} \end{bmatrix}_{63} & \begin{bmatrix} 0 & 0 \\ k_{33} & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ d_{31} \end{bmatrix}_{61} & \begin{bmatrix} 0 \\ m_{31} \end{bmatrix}_{61} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{36} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{33} & \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{36} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{31} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{31} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{63} & \begin{bmatrix} 0 & I_{33} \\ p_{33} & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ q_{31} \end{bmatrix}_{61} & \begin{bmatrix} 0 \\ s_{31} \end{bmatrix}_{61} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{26} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{23} & \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{26} & 0 & 0 \end{bmatrix} \quad (17 \times 17)$$

where the subscripts refer to the dimensions of the appropriate partitions.

When the covariances are propagated by integrating the covariance differential equation (see Section 4.6) in the PDOT mode, the augmented dynamic state vector is defined as

$$\underline{x}_A = \begin{bmatrix} \underline{r} \\ \underline{v} \\ \underline{u} \\ \underline{w} \end{bmatrix} = \begin{bmatrix} \text{s/c position vector} \\ \text{s/c velocity vector} \\ \text{constant thrust controls} \\ \text{time-varying thrust parameters (6x1)} \end{bmatrix} ,$$

and F_A in this case is given as

$$F_A = \begin{bmatrix} \begin{bmatrix} 0 & I_{33} \\ f_{33} & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ g \end{bmatrix}_{63} & \begin{bmatrix} 0 \\ n \end{bmatrix}_{66} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{36} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{33} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{36} \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}_{66} & \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{63} & \begin{bmatrix} 0 \\ h \end{bmatrix}_{66} \end{bmatrix} \quad (15 \times 15)$$

In either STM or PDOT mode, the fully augmented state, as used by GODSEP, may also include measurement parameters. However, the terms appearing in F_A corresponding to measurement parameters are zero because they do not affect the dynamic process.

Specific matrices in F_A are defined as follows:

- f_{33} = partials of the s/c acceleration vector w.r.t. position components = $\partial \underline{a} / \partial \underline{r}$,
- g_{33} = partials of the s/c acceleration vector w.r.t. the thrust controls = $\partial \underline{a} / \partial \underline{u}$,
- k_{33} = partials of the s/c acceleration vector w.r.t. the position of the ephemeris body = $\partial \underline{a} / \partial \underline{r}_p$,
- d_{31} = partials of the s/c acceleration vector w.r.t. the gravitational constant of the ephemeris body = $\partial \underline{a} / \partial \mu_p$,
- m_{31} = partials of the s/c acceleration vector w.r.t. the solar gravitational constant = $\partial \underline{a} / \partial \mu_s$,
- p_{33} = partials of the ephemeris body acceleration vector w.r.t. the position of the ephemeris body = $\partial \underline{a}_p / \partial \underline{r}_p$,
- q_{31} = partials of the ephemeris body acceleration vector w.r.t. the ephemeris body's gravitational constant = $\partial \underline{a}_p / \partial \mu_p$,
- s_{31} = partials of the ephemeris body acceleration vector w.r.t. the solar gravitational constant = $\partial \underline{a}_p / \partial \mu_s$,
- n_{36} = partials of the s/c acceleration vector w.r.t. the time-varying thrust parameters = $\partial \underline{a} / \partial \underline{w}$, and
- h_{66} = partials of the time derivative of the time-varying thrust parameters w.r.t. the time-varying parameters = $\partial \dot{\underline{w}} / \partial \underline{w}$.

As noted from its definition, the elements of the f_{33} matrix are evaluated by differentiating components of the s/c acceleration vector, \underline{a} , with respect to s/c coordinates, i.e.:

$$f_{33} = \frac{\partial \underline{a}}{\partial \underline{r}} ,$$

where \underline{a} is the sum of two contributing terms: the gravitational acceleration, \underline{g} , and the thrust acceleration, \underline{a}_T . The partials of \underline{g} with respect to \underline{r} are the components of the so-called gravity gradient matrix and are well-known, e.g. see Battin (Reference 4). In terms of s/c position vectors relative to the gravitating bodies, \underline{r}_i , the gravity gradient matrix is

$$\partial \underline{g} / \partial \underline{r} = \sum_i \left[3 \underline{r}_i \underline{r}_i^T - r_i^2 \underline{I}_{33} \right] \mu_i / r_i^5 ,$$

with the summation being performed for all bodies. In this equation, μ_i is the gravitational constant of the i^{th} body, and \underline{I}_{33} is a (3x3) identity matrix.

The second matrix comprising f_{33} is obtained by differentiating the thrust acceleration vector, as seen in the inertial frame, with respect to the heliocentric position of the s/c. Since the rotation matrix relating the body axis thrust vector to the inertial frame is dependent on the heliocentric position, it is necessary to differentiate components of the orthogonal rotation matrix as well as the components of the body axis thrust vector. Recalling from Section 4.1 (page 20) that the thrust acceleration is given by

$$\underline{a}_T = A \underline{a}'_T ,$$

this becomes upon differentiation,

$$\frac{\partial \mathbf{a}_T}{\partial \underline{r}} = \left[\frac{\partial \hat{\mathbf{i}}_B}{\partial \underline{r}} \right] \mathbf{a}'_{Tx} + \left[\frac{\partial \hat{\mathbf{j}}_B}{\partial \underline{r}} \right] \mathbf{a}'_{Ty} + \left[\frac{\partial \hat{\mathbf{k}}_B}{\partial \underline{r}} \right] \mathbf{a}'_{Tz} + A \left[\frac{\partial \mathbf{a}'_T}{\partial \underline{r}} \right] \quad (\text{A4-1})$$

where $\hat{\mathbf{i}}_B$, $\hat{\mathbf{j}}_B$ and $\hat{\mathbf{k}}_B$ are unit vectors defined by

$$\hat{\mathbf{k}}_B = \underline{r} / |\underline{r}| ,$$

$$\hat{\mathbf{j}}_B = \underline{r} \times \underline{z}_s / |\underline{r} \times \underline{z}_s| ,$$

and
$$\hat{\mathbf{i}}_B = \hat{\mathbf{j}}_B \times \hat{\mathbf{k}}_B .$$

\underline{r} is, of course, the heliocentric position vector of the s/c, and \underline{z}_s is

the unit vector of ecliptic direction cosines pointing toward the reference star for the cone/clock system. In terms of the unit vectors defined above and the unit vector for the reference star, the first three terms of the right hand side of equation A4-1 are

$$\frac{\partial \hat{\mathbf{i}}_B}{\partial \underline{r}} = \frac{1}{r} \left[I_{33} - \hat{\mathbf{k}}_B \hat{\mathbf{k}}_B^T \right] ,$$

$$\frac{\partial \hat{\mathbf{j}}_B}{\partial \underline{r}} = \frac{1}{R} \left[\hat{\mathbf{j}}_B \hat{\mathbf{j}}_B^T - I_{33} \right] \underline{z}_s , \quad \text{and}$$

$$\frac{\partial \hat{\mathbf{k}}_B}{\partial \underline{r}} = J \left[\frac{\partial \hat{\mathbf{k}}_B}{\partial \underline{r}} \right] - K \left[\frac{\partial \hat{\mathbf{j}}_B}{\partial \underline{r}} \right]$$

R is the magnitude of the vector cross product of \underline{r} and \underline{z}_s . The matrices Z , J , and K are skew-symmetric matrices corresponding to the unit vectors \underline{z}_s , $\hat{\mathbf{j}}_B$ and $\hat{\mathbf{k}}_B$, respectively, and are defined as

$$\begin{aligned}
 \underline{Z} &= \begin{bmatrix} 0 & -z_s(3) & z_s(2) \\ z_s(3) & 0 & -z_s(1) \\ -z_s(2) & z_s(1) & 0 \end{bmatrix} \\
 \underline{J} &= \begin{bmatrix} 0 & -j_B(3) & j_B(2) \\ j_B(3) & 0 & j_B(1) \\ -j_B(2) & j_B(1) & 0 \end{bmatrix} \\
 \underline{K} &= \begin{bmatrix} 0 & -k_B(3) & k_B(2) \\ k_B(3) & 0 & -k_B(1) \\ -k_B(2) & k_B(1) & 0 \end{bmatrix}
 \end{aligned}$$

The last term in A4-1 reflects how the body axis components of the thrust acceleration vary with variations in the solar distance. Since the only quantity in each acceleration component which depends on the solar distance is the (scalar) power function, $P(r)$, (See Page 16), the partials of \underline{a}'_T with respect to \underline{r} become

$$\frac{\partial \underline{a}'_T}{\partial \underline{r}} = \frac{1}{P(r)} \underline{a}'_T \left[\frac{\partial P(r)}{\partial \underline{r}} \right]$$

where the product of \underline{a}'_T and $\partial P / \partial \underline{r}$ is a 3x3 matrix.

The 3x3 g matrix is

$$\underline{g} = \frac{\partial \underline{y}}{\partial \underline{u}} = \underline{A} \left[\frac{\partial \underline{y}'}{\partial \underline{u}} \right]$$

where A is the transformation matrix from spacecraft cartesian to inertial coordinates (Section 4.1) and $\frac{\partial \dot{v}'}{\partial \underline{u}}$ transforms thrust controls to spacecraft coordinates.

$$\frac{\partial \dot{v}'}{\partial \underline{u}} = \begin{bmatrix} a'_x & a' \cos(\text{clock}) \cos(\text{cone}) & -a'_y \\ a'_y & a' \sin(\text{clock}) \cos(\text{cone}) & a'_x \\ a'_z & -a' \sin(\text{cone}) & 0 \end{bmatrix}$$

for the cone/clock system, with $\underline{a}' =$ thrust acceleration in spacecraft coordinates, and

$$\frac{\partial \dot{v}'}{\partial \underline{u}} = \begin{bmatrix} a'_x & -a'_y & -a' \sin \delta & \cos \gamma \\ a'_y & a'_x & -a' \sin \delta & \sin \gamma \\ a'_z & 0 & a' \cos \delta & \end{bmatrix}$$

for the in/out of plane system.

The 3x3 k matrix is

$$k = \frac{\partial \dot{v}}{\partial \underline{r}_p} = \frac{\mu_p}{r_p^5} \left[3 \underline{r}_p \underline{r}_p^T - r_p^2 \underline{I} \right] - \frac{\mu_s}{R^5} * \left[3 \underline{R} \underline{R}^T - R^2 \underline{I} \right]$$

where $\underline{R} =$ spacecraft position WRT the ephemeris body.

The 3x1 d vector is

$$d = \frac{\partial \dot{v}}{\partial \mu_p} = -\frac{\underline{R}}{R^3}$$

The 3x1 m vector is

$$m = \frac{\partial \dot{v}}{\partial \mu_s} = -\frac{\underline{r}}{r^3}$$

The 3x3 p matrix is

$$p = \frac{\partial \dot{v}_p}{\partial \underline{r}_p} = -\frac{\mu_p}{r_p^5} \left[3\underline{r}_p \underline{r}_p^T - r_p^2 \mathbf{I} \right]$$

The 3x1 q vector is

$$q = \frac{\partial \dot{v}_p}{\partial \mu_p} = -\frac{r_p}{r_p^3}$$

The 3x1 s vector is

$$s = \frac{\partial \dot{v}_p}{\partial \mu_s} = -\frac{r_p}{r_p^3}$$

The 3x6 n matrix is

$$n = \frac{\partial \dot{v}}{\partial \underline{w}} = \left[\begin{array}{c|c} \mathbf{g} & \mathbf{g} \end{array} \right]$$

The 6x6 h matrix is

$$h = \frac{\partial \dot{w}}{\partial \underline{w}} = \begin{bmatrix} \frac{1}{T_1} & 0 & \dots & 0 \\ 0 & -\frac{1}{T_2} & \dots & 0 \\ \vdots & & \ddots & \\ 0 & \dots & 0 & -\frac{1}{T_6} \end{bmatrix}$$

where T_1, \dots, T_6 are process noise correlation times.

APPENDIX 5

9.5 TOPSEP Injection Modeling9.5.1 Injection Controls

One set of controls which may be used in the targeting and optimization submode of TOPSEP is the initial state \underline{X}_0 , which defines a hyperbolic escape orbit relative to the launch planet, where

$$\underline{X}_0 = \begin{bmatrix} \underline{r}_0 \\ \underline{v}_0 \end{bmatrix}$$

and

$$\underline{r}_0 = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \underline{v}_0 = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}$$

\underline{X}_0 is the state of the S/C in cartesian elements immediately following injection from a parking orbit about the launch planet. Instead of using \underline{X}_0 directly as control parameters, one may simulate the heliocentric injection process more realistically with a different set of injection parameters (See Figure 9-1, Page 128):

- 1) r_0 , the magnitude of the radius vector at the point of injection.
- 2) i , the inclination of the parking orbit.
- 3) Δv , the magnitude of the velocity change from the parking orbit to a hyperbolic escape orbit.
- 4) χ , the angle locating the projection of $\Delta \underline{v}$ vector in the parking orbit plane relative to \underline{r}_0 .

- 5) ψ , the angle locating the $\Delta \underline{v}$ vector out of the parking orbit plane
- 6) t_0 , the heliocentric injection time.

These injection controls are augmented to the control profile in TOPSEP by correctly setting the following elements of the H array in the \$TOPSEP namelist: H (9, 21), H (10, 21), H (1, 22), H (2, 22), H (3, 22), and H (4, 22). Refer to Page 15 in the User's Manual for additional information for implementation. The initial values of the injection parameters are not input in either the \$TRAJ namelist or the \$TOPSEP namelist but are determined analytically from the initial state \underline{X}_0 based on certain assumptions about the parking orbit.

The injection state $(\underline{r}_0, \underline{v}_0)$ of the reference trajectory for any given iteration in TOPSEP is assumed to define a circular coplanar parking orbit (the eccentricity is very small but non-zero to accommodate future program modifications for elliptical parking orbits). An in-plane $\Delta \underline{v}$ is applied at time t_0 which injects the S/C from the parking orbit to the hyperbolic orbit. Thus, the nominal or reference parking orbit and injection controls are uniquely defined.

$$r_0 = |\underline{r}_0|$$

$$i = \text{the inclination of the hyperbolic orbit}$$

$$\Delta v = |\underline{v}_0 - \underline{v}_{pk}|, \text{ where } \underline{v}_{pk} \text{ is the S/C's velocity in the parking orbit prior to injection}$$

$$\chi = \cos^{-1} \left[\frac{\underline{r}_0 \cdot \Delta \underline{v}}{r_0 \Delta v} \right]$$

$$\psi = 0$$

$$t_0 = 0$$

The velocity in the circular parking orbit prior to injection is

$$\underline{v}_{pk} = \sqrt{\frac{\mu_E}{r_o}} \left[\frac{(\underline{r}_o \times \underline{v}_o) \times \underline{r}_o}{|(\underline{r}_o \times \underline{v}_o) \times \underline{r}_o|} \right]$$

where μ_E is the mass of the Earth.

During the construction of the perturbed trajectories* and trial trajectories in each iteration of TOPSEP, the injection controls are changed by some pre-determined amount so that a new cartesian state $(\underline{r}'_o, \underline{v}'_o)$ may be defined. When the injection controls r_o , i , or t_o are modified, a new parking orbit must be computed. Changes to these injection controls affect both position \underline{r}'_o and velocity \underline{v}'_o ; however, modifications to Δv , χ , or Ψ affect only velocity \underline{v}'_o .

If r_o , i , and t_o are changed to r'_o , i' , and t'_o a new state $(\underline{r}'_o, \underline{v}'_o)$ may be found. First, the unitary angular momentum vector \hat{h} , the node vector \hat{n}_Ω , and the longitude of the ascending node Ω must be computed. If

$$\underline{h} = \underline{r}_o \times \underline{v}_{pk}$$

then

$$\hat{h} = \underline{h} / h$$

and

$$\underline{n}_\Omega = \begin{bmatrix} -hy \\ hx \\ 0 \end{bmatrix}, \quad \hat{n}_\Omega = \underline{n}_\Omega / n_\Omega, \quad \Omega = \tan^{-1}(\hat{h}_x / -\hat{h}_y).$$

For the case in which the parking orbit is in the ecliptic, the node is defaulted to be the x axis so

$$\hat{n}_\Omega = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Omega = 0 \text{ radians.}$$

* Finite differencing techniques are used to calculate injection control corrections for targeting purposes.

The angular position θ of the S/C in the orbital plane as measured from the node is

$$\theta = \left\{ \begin{array}{l} \Delta \text{ traveled by changing} \\ \text{injection time} \end{array} \right\} + \left\{ \begin{array}{l} \Delta \text{ of S/C from node in} \\ \text{nominal parking orbit} \end{array} \right\}$$

$$\theta = (t'_o - t_o) \sqrt{\frac{\mu_E}{(r'_o)^3}} + \cos^{-1} \left(\frac{\underline{r}_o \cdot \hat{n}}{r_o} \Omega \right)$$

Finally,

$$\underline{r}'_o = r'_o \begin{bmatrix} \cos \Omega \cos \theta & -\sin \Omega \sin \theta & \cos i' \\ \sin \Omega \cos \theta & +\cos \Omega \sin \theta & \cos i' \\ \sin \theta & \sin i' & \end{bmatrix}$$

and

$$\underline{v}'_{pk} = \sqrt{\frac{\mu_E}{r'_o}} \begin{bmatrix} \cos \Omega \sin \theta & +\sin \Omega \cos \theta \cos i' \\ \sin \Omega \sin \theta & -\cos \Omega \cos \theta \cos i' \\ -\cos \theta & \sin i' \end{bmatrix}$$

\underline{v}'_o is constructed by adding the delta-velocity vector $\Delta \underline{v}'$ to the S/C's parking orbit velocity \underline{v}'_{pk} . The vector $\Delta \underline{v}'$ is computed in terms of in-orbit plane and out-of-orbit plane components. Let $\Delta \underline{v}'$, χ' , and ψ' be modified injection controls. Then

$$\Delta \underline{v}' = \Delta \underline{v}'_1 + \Delta \underline{v}'_2 + \Delta \underline{v}'_3$$

$$\Delta \underline{v}'_1 = \Delta v' \cos(\psi') \cos(\chi') \frac{\underline{r}'_o}{r'_o}$$

$$\Delta \underline{v}'_2 = \Delta v' \cos(\psi') \sin(\chi') \frac{\underline{h}' \times \underline{r}'_o}{|\underline{h}' \times \underline{r}'_o|}$$

$$\Delta \underline{v}'_3 = \Delta v' \sin(\psi') \frac{\underline{h}'}{h'}$$

where

$$\underline{h}' = \underline{r}'_o \times \underline{v}'_{pk}$$

$\Delta \underline{v}'_1$ is the vector in the radial direction \underline{r}'_o , $\Delta \underline{v}'_2$ is the vector orthogonal to \underline{r}'_o in the orbit plane, and $\Delta \underline{v}'_3$ is the vector normal to the parking orbit plane and in the direction of the angular momentum vector. Thus $\underline{v}'_o = \underline{v}'_{pk} + \Delta \underline{v}'$ and a new initial state vector \underline{x}'_o has been established.

Figure 9-1 illustrates how the injection parameters determine the new initial state vectors \underline{r}'_0 and \underline{v}'_0 .

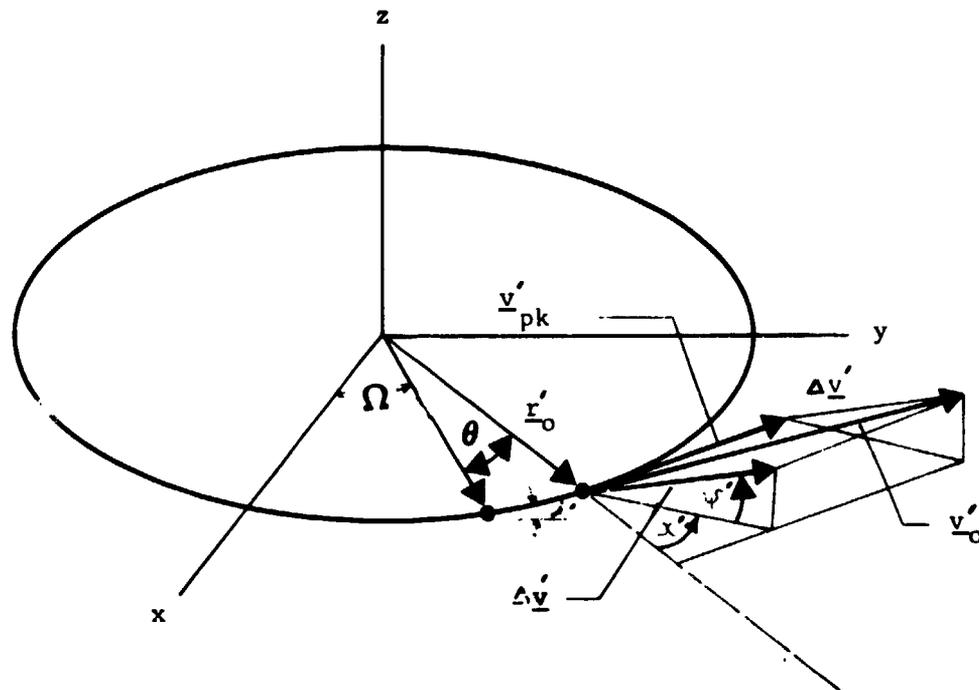


Figure 9-1. The new state vectors, \underline{r}'_0 and \underline{v}'_0 , as defined for perturbed and trial trajectories using injection controls.

9.5.2 Tug Multiple-Impulse Orbit Transfer

When the injection controls (r_0 , i , t_0 , Δv , χ , ψ) are applied in TOPSEP, the reference parking orbit is likely to change from iteration to iteration. In fact, it is possible that the parking orbit characteristic of the last iteration may not be attainable directly from an Earth based launch within realistic launch constraints. Therefore, it becomes necessary to consider the interface between the SEP S/C and the launch vehicle in order to predict the "cost" of achieving the reference parking orbit. The cost may be estimated indirectly by determining the launch vehicle fuel budget to transfer the SEP S/C from some nominal inner parking orbit to the outer reference parking orbit. If the inclination of the inner parking orbit is realistically constrained, an orbit plane change may be necessary to complete the transfer. As the angle of the plane change increases the estimated cost of the orbit transfer will increase dramatically. The estimated cost may then be used to distinguish between acceptable and unacceptable outer parking orbits while simultaneously sizing the fuel expenditure for presently conceived or operational launch vehicles (or intermediate stages). An additional efficiency check may be made by comparing the fuel expenditure of the multiple impulse orbit transfer with that of a single impulse injection from the inner parking orbit. The only requirement on the single impulse is that it provide the correct \underline{v}_∞ vector. The single impulse calculation is not intended to be used in conjunction with the thrust control profile for targeting but rather as a standard for determining the inefficiencies of the multiple impulse injection process.

The launch vehicle simplistically modeled in TOPSEP is the expendable space tug*. Initially, the tug is in a circular inner parking orbit whose equatorial inclination is constrained due to bounds on the booster launch azimuth. The tug then performs the transfer to the outer parking orbit, the characteristics of which are described in Section 9.5.1. The outer parking orbit is also circular; however, it is assumed to have an inclination and ascending node equal to those of the hyperbolic escape orbit. The tug's path from the inner parking orbit to the outer parking orbit will be a coplanar Hohmann transfer if the required equatorial inclination does not violate the launch azimuth constraints. Otherwise, the tug follows a modified Hohmann transfer (i.e., a regular Hohmann transfer in the inner orbit plane followed by a plane change and circularization at the line of intersection with the outer orbit plane). The equatorial inclination of this transfer orbit is either the maximum or minimum inclination bound such that the required plane change is a minimum. For the coplanar transfer the ascending node of the inner orbit is fixed; thus, the launch azimuth may be computed explicitly and tested for a constraint violation. For the plane change transfer the launch azimuth is fixed, and the inner parking orbit is uniquely determined when the minimum plane change condition is enforced.

Figure 9-2 illustrates the selection of the inner orbit normal vector which minimizes the plane change for the transfer to the outer parking orbit.

* Although the launch vehicle is referred to as the space tug, in the remaining discussion, the specific vehicle is inconsequential to the sizing of the fuel budget (i.e., the vehicle may be a trans-stage, Burner II, Centaur, etc.)

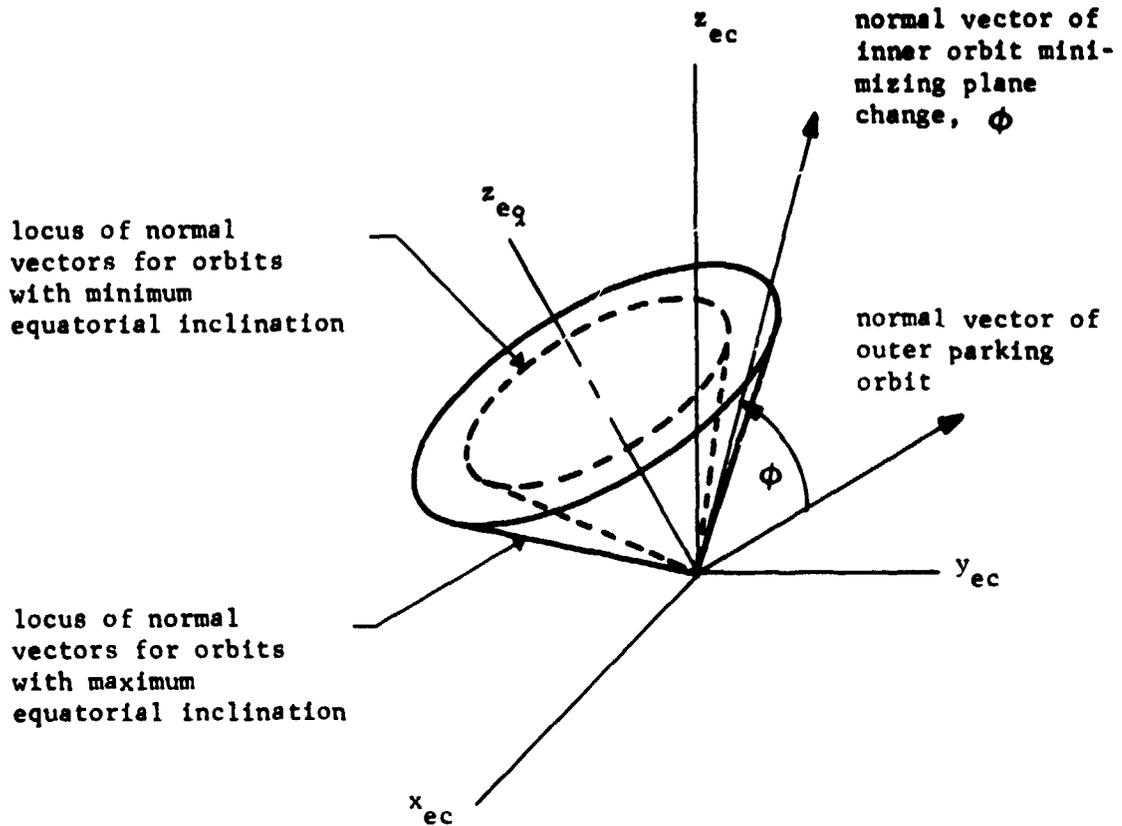


Figure 9-2 Selection of the Normal Vector for the Inner Orbit Which Minimizes the Plane Change

The larger cone in the figure represents the locus of normal vectors for all possible inner orbits having the maximum equatorial inclination allowed by the launch azimuth constraints (AZMIN and AZMAX in STOPSSEP). The smaller cone is the locus of normal vectors for orbits having the minimum equatorial inclination, which in general will be equal to the latitude of the Kennedy Space Center (28.608 deg). If the normal vector of the outer parking orbit falls between the two cones, the parking orbits are assumed coplanar. If the normal vector falls outside of this region, the inner parking orbit is characterized as follows:

- (1) the inner orbit is inclined to the outer orbit by the angle ϕ where ϕ is the minimum angle between the normal vector of the outer orbit and the nearest cone.
- (2) the normal vector of the inner orbit becomes the projection of the outer orbit normal vector on the nearest cone.

For both the Hohmann and modified Hohmann transfers three distinguishable velocity increments or Δv 's will occur. Figure 9-3 illustrates the relative positions where these maneuvers are executed for the general case (r_a specified by RP1 in \$STOPSEP).

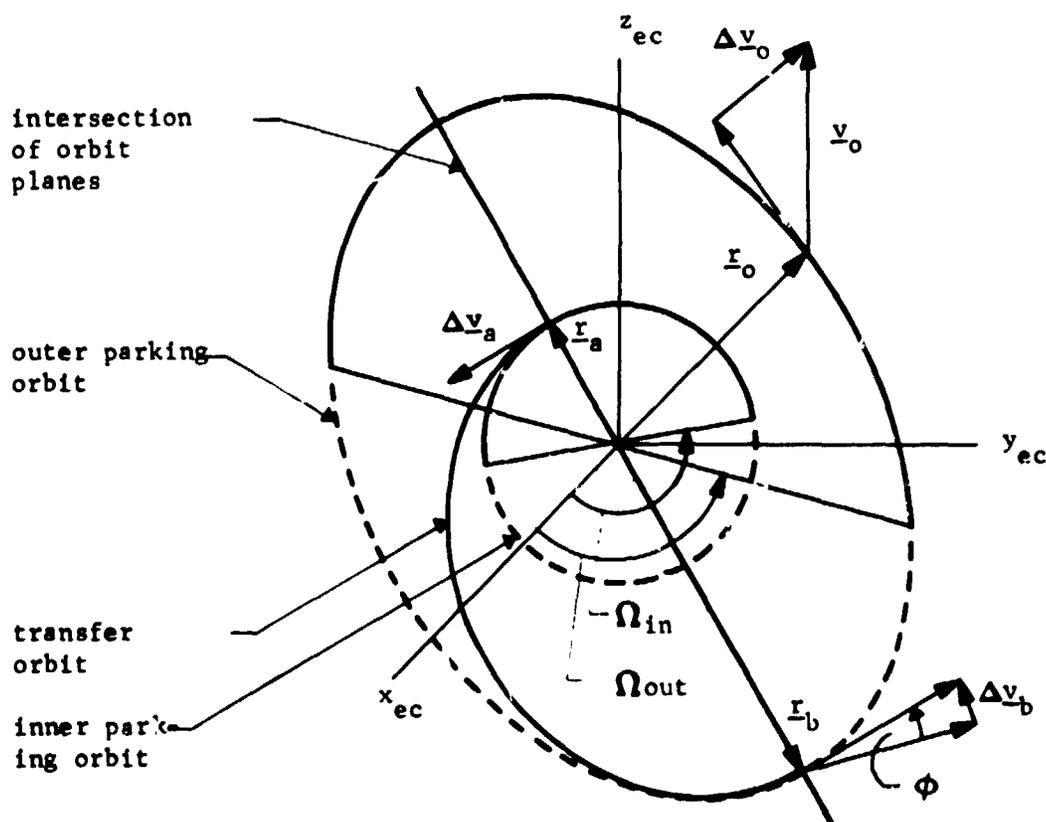


Figure 9-3 The Tug 3-Impulse Orbit Transfer

The first impulse Δv_a occurs at periapsis of the Hohmann transfer at the line of intersection of the parking orbit planes. The second impulse Δv_b occurs at apoapsis of the transfer orbit and provides a velocity increment to circularize the orbit and to change orbit planes if necessary. At a point on the outer parking orbit prescribed by the injection position vector \underline{r}_0 , a third impulse Δv_0 is executed which places the tug on a hyperbolic escape trajectory from Earth (Note that the injection impulse Δv_0 is exactly the same as the Δv vector discussed in Section 9.5.1). If both parking orbits are coplanar the line of intersection becomes ambiguous so the impulse position vectors \underline{r}_a and \underline{r}_b are assumed oriented such that

$$\underline{r}_a = \frac{-r_a r_0}{r_0}$$

and

$$\underline{r}_b = \frac{r_b r_0}{r_0} .$$

The total impulse at \underline{r}_0 becomes

$$\Delta v_T = \Delta v_b + \Delta v_0;$$

however, the impulses will always be referred to individually as in the orbital plane change example.

The magnitudes of the velocity increments Δv_a and Δv_b for the general case are computed as follows.

$$(v_a)^- = \sqrt{\frac{\mu_E}{r_a}}, \quad (v_a)^+ = \sqrt{\frac{2 \mu_E r_b}{r_a(r_a + r_b)}}$$

$$(v_b)^- = \sqrt{\frac{2 \mu_E r_a}{r_b(r_a + r_b)}}, \quad (v_b)^+ = \sqrt{\frac{\mu_E}{r_b}}$$

Since the first impulse does not initiate a plane change and since the velocity vectors before and after the impulse are aligned

$$\Delta v_a = (v_a)^+ - (v_a)^-$$

$$\Delta v_a = \sqrt{\frac{\mu_E}{r_a}} \left(\sqrt{\frac{2 r_b}{r_a + r_b}} - 1 \right)$$

The second impulse may perform a plane change through the angle ϕ .

Thus, by applying the law of cosines

$$\Delta v_b = \left((v_b^+)^2 + (v_b^-)^2 - 2(v_b^+)(v_b^-) \cos \phi \right)^{\frac{1}{2}}$$

If $\phi = 0$ the above equation reduces to

$$\Delta v_b = (v_b)^+ - (v_b)^-$$

$$\Delta v_b = \sqrt{\frac{\mu_E}{r_b}} \left(1 - \sqrt{\frac{2 r_a}{r_a + r_b}} \right)$$

The magnitude for the injection velocity increment Δv_o is dependent upon the state vector X_o . The computations are discussed in detail in Section 9.5.1.

Once Δv_a , Δv_b , and Δv_o have been calculated the fuel budget for the space tug may be computed. The tug's dry weight (W_{tug}), the propellant weight (W_{fuel}^{MAX}), and the tug's specific impulse (I_{sp}) are specified in the input namelist (TUGWT, TGFUEL, and TUGISP in \$TOPSEP). The fuel required for the first maneuver (W_{fuel}^a) is then

$$(W_{fuel}^a) = \left(1 - \exp \left(\frac{-\Delta v_a}{g I_{sp}} \right) \right) W_{tot}$$

where

$$W_{tot} = W_{tug} + (W_{fuel}^{MAX}) + W_{sep}$$

W_{sep} is the weight of the SEP vehicle and g is the gravitational constant. It follows that the fuel for the second and third maneuvers are

$$(W_{fuel}^b) = \left(1 - \exp \left(\frac{-\Delta v_b}{g I_{sp}} \right) \right) \cdot \left(W_{tot} - (W_{fuel}^a) \right)$$

and

$$(W_{fuel}^o) = \left(1 - \exp \left(\frac{-\Delta v_o}{g I_{sp}} \right) \right) \cdot \left(W_{tot} - (W_{fuel}^a) - (W_{fuel}^b) \right)$$

The required fuel budget to perform the orbit transfers including injection is then

$$(W_{fuel}^{tot}) = (W_{fuel}^a) + (W_{fuel}^b) + (W_{fuel}^o)$$

If

$$(W_{\text{fuel}})_{\text{tot}} > (W_{\text{fuel}})_{\text{max}},$$

the outer reference parking orbit is undesirable and measures must be taken to make the injection state and outer parking orbit realistic.

Calculations for the single impulse injection from the inner orbit proceed as follows. The velocity at periapsis $(v_a)^+$ necessary to obtain the hyperbolic excess velocity v_{∞} is

$$(v_a)^+ = \sqrt{\frac{\mu_E}{|a|} \cdot \frac{(e+1)}{e-1}}$$

where a and e are the semi-major axis and eccentricity of the hyperbolic orbit. The parameters a and e are determined so that the vector v_{∞} resulting from this orbit is the same as that obtained from the multiple-impulse injection process.

The single velocity increment is then

$$\Delta v_a = \sqrt{\frac{\mu_E}{|a|} \frac{(e+1)}{e-1}} - \sqrt{\frac{\mu_E}{r_a}}$$

The required fuel expenditure is then

$$(W_{\text{fuel}})_a = \left(1 - \exp\left(\frac{-\Delta v_a}{g I_{\text{sp}}}\right) \right) W_{\text{tot}}$$

A comparison of $(W_{\text{fuel}})_a$ and $(W_{\text{fuel}})_{\text{tot}}$ will indicate the relative efficiency of the multiple impulse injection process.

APPENDIX 6

9.6 Control Weighting Schemes for TOPSEP

Various weighting schemes are provided to allow the MAPSEP user flexibility in scaling controls in the TOPSEP mode (Chapter 5). The purpose of these schemes is to alter the contours of constant cost and constant target error in the control space so that the projected gradient algorithm may converge more readily. Convergence problems occur most often when elements of the control vector differ in units. For example, there may be considerable difficulty in finding a converged solution when thrusting angles (cone or clock) are selected as controls in addition to thrust phase times. Whereas the internal units for the angles and times are radians and seconds respectively, the corresponding elements of the sensitivity matrix may vary by several orders of magnitude. That is, the sensitivity of the targets to a change of one radian in thrusting angle is many orders of magnitude greater than the sensitivity of the targets to a change of one second in thrust phase duration. In the example just described one would find that the PGM algorithm would compute a control correction which would try to eliminate the target errors by large changes in the thrusting angles and very small changes in thrust duration. Unfortunately, large control changes are often invalid in the very nonlinear control space associated with low thrust trajectories. To alleviate this problem, the normalized control weighting scheme has been devised. The diagonal elements of the control weighting matrix are defined as

$$[W_u]_{jj} = \frac{1}{|u_j|} .$$

Application of this weighting matrix to the controls determines a sensitivity matrix whose elements are roughly the same order of magnitude. Thus, the removal of the target error is spread evenly among the selected controls rather than among only a few. Hopefully all the control changes will be small enough to be valid in the nonlinear control space.

Another type of convergence problem may occur. Sometimes elements of the sensitivity matrix vary by orders of magnitude even though the controls are all of the same units. For example, target parameters are much more sensitive to changes in thrusting angles early in the trajectory than they are to changes in these angles late in a trajectory. If the controls are not scaled, the PGM algorithm computes a control correction where changes in thrusting angles during early phases are unacceptably large and changes to the angles during later phases are undetectable. The following weighting schemes have been devised to spread the removal of target error more evenly among the selected controls.

a) Sensitivity weighting

$$[W_u]_{jj} = \text{Max} \left\{ |S_{ij}|, i = 1, N \right\}$$

b) Combined sensitivity, target error, and control weighting

$$[W_u]_{jj} = \sum_{i=1}^N \frac{|S_{ij}| * e_i}{u_j}$$

c) Gradient weighting

$$G_j = 2 * \sum_{i=1}^N s_{ij} \epsilon_i$$

$$[w_u]_{jj} = \frac{|G_j|}{\|G\|}$$

d) Averaged gradient and control weighting

$$[w_u]_{jj} = \frac{(10 * u_j + \frac{0.1}{G_j})}{u_j^2 + \frac{1}{G_j^2}}$$

where G is defined in c.

9.7 Integrated State Transition Matrices for Computing the Targeting Sensitivity Matrix

Within the three basic modes of MAPSEP, trajectory guidance and/or retargeting represent one of the primary computational problems worked in the program. Whereas the logic controlling these calculations is, in general, straightforward and easily understood, the actual execution remains as one of the more costly computational operations to be performed. This is especially true in TOPSEP and SIMSEP where targeting over long trajectory arcs is done repeatedly. In order to minimize computational expenses, an algorithm which uses state transition matrices integrated with the trajectory has been implemented and is used exclusively in GODSEP and SIMSEP for computing the targeting sensitivity matrix. In TOPSEP, the user has the option to use either this or the more expensive, but equivalent, numerical differencing algorithm.

The targeting sensitivity matrix, S , is a matrix of linear partials relating variations in the control variables, $\Delta \underline{u}$, to variations in the targets, $\Delta \underline{T}$, according to

$$\Delta \underline{T} = S \Delta \underline{u} .$$

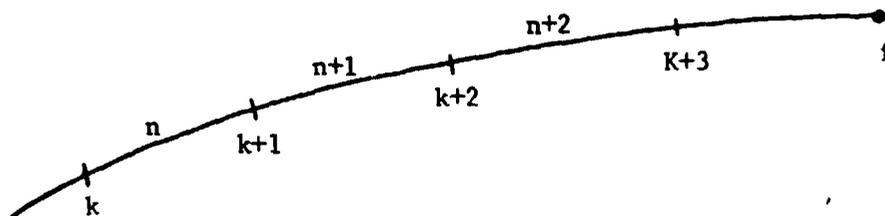
Looking at the targeting sensitivity matrix in more detail, it is seen to be of the form,

$$S = \frac{\partial (T_1, T_2, \dots, T_n)}{\partial (u_1, u_2, \dots, u_m)}$$

where the T 's are selected target variables and the u 's are controls.

Typical target variables include X_f , Y_f , Z_f , $B \cdot T$, $B \cdot R$, etc. which are all evaluated about the final trajectory state. Typical low thrust control variables are the thrust phase stop time, thruster throttling, cone angle, angle in the same or different thrust control phases.

To provide a conceptual understanding of how S is evaluated from trajectory information generated in the augmented state transition matrix, it is convenient to consider the trajectory segment depicted below where time points k ,



Reference Trajectory

$k + 1$, and f are shown bounding thrust control phases n , $n + 1$, etc. In the figure, time point f denotes the target time and represents the trajectory stopping condition. Considering a specific thrust control phase, say n , it is recalled from Section 4-5 that the augmented state transition matrix generates partials of state variations of time $k + 1$ with respect to state changes at k and control variable changes interior to thrust phase n . In particular, these partials are contained in the Φ and Θ_u partitions of the augmented matrix and may be symbolically written as,

$$\mathbb{I} = \frac{\partial (x, y, z, v_x, v_y, v_z)_{k+1}}{\partial (x, y, z, v_x, v_y, v_z)_k},$$

and

$$\theta_u = \frac{\partial (x, y, z, v_x, v_y, v_z)_{k+1}}{\partial (u_1, u_2, u_3, u_4)}.$$

The u 's correspond to the phase stop time, throttling, cone angle, and clock angle for the n^{th} phase. (Cone and clock angle rates are excluded from this treatment since their partials must be obtained by numerical differencing.)

If u_2 (for example) in thrust control phase n is specified as an active control for the targeting event being considered, then the action of u_2 on the state vector at f is computed by pre-multiplying the appropriate column from $\theta_u(n)$ for phase n by all intervening \mathbb{I} 's, i.e.

$$\frac{\partial X_f}{\partial u_2(n)} = \mathbb{I}_{f, k+3} \mathbb{I}_{k+3, k+2} \mathbb{I}_{k+2, k+1} \frac{\partial X_k}{\partial u_2(n)}.$$

Hence, an augmented θ_u , say $\hat{\theta}_u$, can be constructed by storing and pre-multiplying appropriate columns from the θ_u 's as they are computed during trajectory integration. The resulting $\hat{\theta}_u$ gives the partials of the final state with respect to the active controls occurring in the various thrust phases between k and f and may be written as

$$\hat{\theta}_u = \frac{\partial (x, y, z, v_x, v_y, v_z)_f}{\partial (u_1, u_2, \dots, u_m)}.$$

The final component necessary to compute the requisite S matrix is the evaluation of target variable partials with respect to the final state, X_f . This is most expediently done by a numerical differencing algorithm to generate the differential point transformation matrix, η . The η matrix can be written as

$$\eta = \frac{\partial (T_1, T_2, \dots, T_{N'})}{\partial (x, y, z, v_x, v_y, v_z)_f} .$$

Now S is seen to be

$$S = \eta \hat{\theta}_u .$$

In TOPSEP where initial state conditions are also permitted as control variables, it is necessary to extend the above procedure but not the computational method. For this special case, only the chained \mathbb{I} 's are needed to compute the partials of final state with respect to changes in the state at k . Clearly, selected columns from $\mathbb{I}_{f,k}$ which is defined by

$$\mathbb{I}_{f,k} = \mathbb{I}_{f,k+3} \quad \mathbb{I}_{k+3, k+2} \quad \mathbb{I}_{k+2, k+1} \quad \mathbb{I}_{k+1, k}$$

may be augmented to the previously defined $\hat{\theta}_u$ to give the requisite targeting sensitivity matrix to be used by TOPSEP.