

AD-A151 927

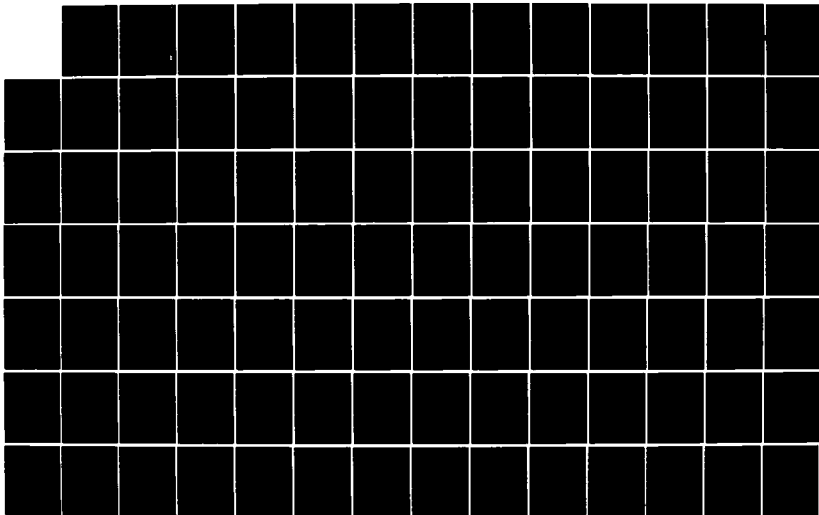
ESTIMATION OF LAUNCH VEHICLE PERFORMANCE PARAMETERS(U)  
AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL  
OF ENGINEERING D A VALLADO DEC 84 AFIT/GA/AA/84D-18

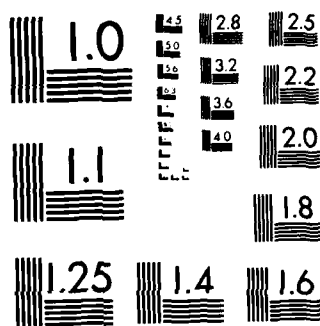
1/2

UNCLASSIFIED

F/G 12/1

NL

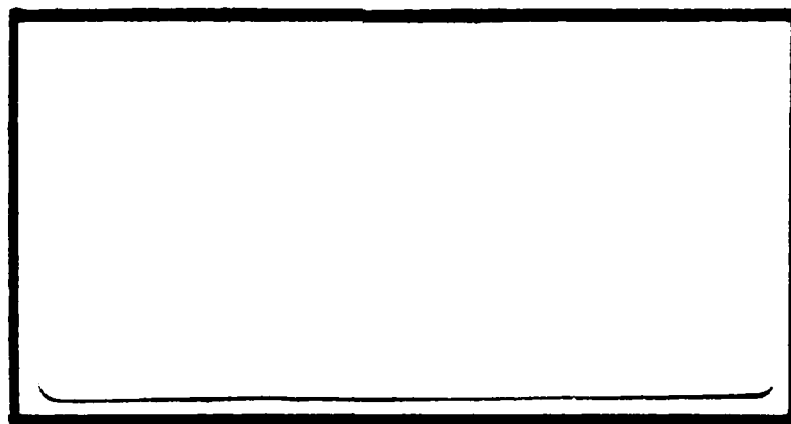




MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS 1963-A



AD-A151 927



DTIC FILE COPY

DTIC  
ELECTE  
APR 02 1985  
S E D

DEPARTMENT OF THE AIR FORCE  
AIR UNIVERSITY

**AIR FORCE INSTITUTE OF TECHNOLOGY**

Wright-Patterson Air Force Base, Ohio

This document has been approved  
for public release and sale; its  
distribution is unlimited.

85 03 13 073

AFIT/GA/AA/84D-10

**ESTIMATION OF LAUNCH VEHICLE  
PERFORMANCE PARAMETERS**

**THESIS**

**David A. Vallado  
Captain USAF**

**AFIT/GA/AA/84D-10**

**Approved for Public release: distribution unlimited**

AFIT/GA/AA/84D-10

**ESTIMATION OF LAUNCH VEHICLE  
PERFORMANCE PARAMETERS**

**THESIS**

**Presented to the Faculty of the School of Engineering  
of the Air Force Institute of Technology**

**Air University  
in partial fulfillment of the  
requirements for the degree of  
Master of Science**

**DAVID A. VALLADO, M.S., B.S.**

**Captain USAF**

**December 1984**

**Approved for Public release: distribution unlimited**

## ACKNOWLEDGEMENTS

I would like to express my thanks to my advisor Dr. Wiesel whose help was invaluable in completing this thesis. He provided tremendous help, motivation and insight during the course of study.

Next, the VAX-11/780 deserves it's due credit. Finally, I would like to thank AFIT for it was in this program that I met my wife Laura.

[illegible]

# TABLE OF CONTENTS

|                                                          | Page |
|----------------------------------------------------------|------|
| Acknowledgments . . . . .                                | ii   |
| List of Figures . . . . .                                | v    |
| List of Tables . . . . .                                 | vi   |
| Abstract . . . . .                                       | vii  |
| I INTRODUCTION . . . . .                                 | 1    |
| II DYNAMICS FORMULATION . . . . .                        | 6    |
| Equations of Motion . . . . .                            | 6    |
| Observation Relationships . . . . .                      | 7    |
| Truth Model Development . . . . .                        | 12   |
| III FILTER DEVELOPMENT . . . . .                         | 16   |
| Matrix Equations . . . . .                               | 16   |
| Least Squares Filter Development . . . . .               | 22   |
| Bayes Filter Development . . . . .                       | 25   |
| IV TESTING . . . . .                                     | 27   |
| Computer Program Development . . . . .                   | 27   |
| Least Squares Checkout . . . . .                         | 33   |
| Bayes Filter Checkout . . . . .                          | 38   |
| V RESULTS AND CONCLUSIONS . . . . .                      | 44   |
| ICBM Performance Parameters Estimation . . . . .         | 44   |
| Conclusions . . . . .                                    | 57   |
| BIBLIOGRAPHY . . . . .                                   | 59   |
| APPENDIX A - Truth Model Programs . . . . .              | 60   |
| tn.f, Truth Model for Satellite                          |      |
| Program Description . . . . .                            | 60   |
| Program Listing . . . . .                                | 63   |
| tlan.f, Truth Model for ICBM                             |      |
| Program Description . . . . .                            | 69   |
| Program Listing . . . . .                                | 70   |
| APPENDIX B - Differential Equations Integrator . . . . . | 73   |
| dhaming                                                  |      |
| Program Description . . . . .                            | 73   |
| Program Listing . . . . .                                | 74   |

## Table of Contents Continued

|                                                          |     |
|----------------------------------------------------------|-----|
| APPENDIX C - Equations of Motion and Variation . . . . . | 76  |
| rhs                                                      |     |
| Program Description . . . . .                            | 76  |
| Program Listing . . . . .                                | 77  |
| APPENDIX D - A Matrix . . . . .                          | 81  |
| tna.f                                                    |     |
| Program Description . . . . .                            | 81  |
| A Matrix Definition . . . . .                            | 82  |
| Program Listing . . . . .                                | 83  |
| APPENDIX E - $\phi$ Matrix . . . . .                     | 85  |
| tnph.f                                                   |     |
| Program Description . . . . .                            | 85  |
| Program Listing . . . . .                                | 86  |
| APPENDIX F - H Matrix . . . . .                          | 88  |
| tnh.f                                                    |     |
| Program Description . . . . .                            | 88  |
| H Matrix Definition . . . . .                            | 89  |
| Program Listing . . . . .                                | 90  |
| APPENDIX G - Observation Relationships . . . . .         | 92  |
| obser                                                    |     |
| Program Description . . . . .                            | 92  |
| Program Listing . . . . .                                | 93  |
| APPENDIX H - Least Squares Estimator . . . . .           | 96  |
| 1stsq.f                                                  |     |
| Program Description . . . . .                            | 96  |
| Program Algorithm . . . . .                              | 98  |
| Program Listing . . . . .                                | 99  |
| APPENDIX I - Bayes Filter Estimator . . . . .            | 105 |
| bayes.f                                                  |     |
| Program Description . . . . .                            | 105 |
| Program Algorithm . . . . .                              | 107 |
| Program Listing . . . . .                                | 108 |
| APPENDIX J - Test Results . . . . .                      | 113 |
| ICBM Test Results 100 data point case . . . . .          | 113 |
| Summary of all test cases . . . . .                      | 118 |
| Vita . . . . .                                           | 120 |

# LIST OF TABLES

| Table                                                                | Page |
|----------------------------------------------------------------------|------|
| 1. Launch Vehicle Data . . . . .                                     | 13   |
| 2. Launch Vehicle Performance Parameters . .                         | 13   |
| 3. Satellite Orbit, Numerical Integration Data                       | 28   |
| 4. ICBM Launch Trajectory,<br>Numerical Integration Data . . . . .   | 30   |
| 5. A $\phi$ H Matrix Data from Checkout . . . .                      | 31   |
| 6. Test Cases Input Data . . . . .                                   | 33   |
| 7. Truth Model Data for Satellite orbit . . .                        | 34   |
| 8. Filter Test Cases . . . . .                                       | 34   |
| 9. Least Squares Test Results,<br>Perfect Observation Data . . . . . | 36   |
| 10. Covariance data for Least Squares Tests .                        | 37   |
| 11. Least Squares Test Results,<br>Noisy Observation Data . . . . .  | 39   |
| 12. Bayes Filter Test Results,<br>Perfect Observation Data . . . . . | 40   |
| 13. Covariance data for Bayes Filter Tests . .                       | 41   |
| 14. Bayes Filter Test Results,<br>Noisy Observation Data . . . . .   | 42   |
| 15. Truth Model Data, ICBM Test Case . . . . .                       | 44   |
| 16. Numerical Integration of ICBM Test case .                        | 45   |
| 17. Summary of Data Points . . . . .                                 | 55   |

## LIST OF FIGURES

| FIGURE                                    | Page |
|-------------------------------------------|------|
| 1. Orbital Sensor Geometry . . . . .      | 2    |
| 2. Land Based Sensor Geometry . . . . .   | 3    |
| 3. Radar Site Coordinate System . . . . . | 9    |
| 4. Radar Site Geometry . . . . .          | 10   |
| 5. Observation Geometry . . . . .         | 11   |
| Exhaust Velocity Test Results             |      |
| 6. Test Cases: 64, 48, 32 . . . . .       | 47   |
| 7. Test Cases: 100, 66 . . . . .          | 48   |
| 8. Test Cases: 24, 66 . . . . .           | 49   |
| 9. Test Cases: 16, 50 . . . . .           | 50   |
| Mass Ratio Test Results                   |      |
| 10. Test Cases: 64, 48, 32 . . . . .      | 51   |
| 11. Test Cases: 100, 24 . . . . .         | 52   |
| 12. Test Cases: 66, 66 . . . . .          | 53   |
| 13. Test Cases: 16, 50 . . . . .          | 54   |



## I INTRODUCTION

The specific problem that will be examined in this paper is the estimation of launch vehicle performance parameters from either a ground based or a space based sensor system. We can easily understand the importance of this since, from a military point of view, it is a necessity that the United States be able to obtain the launch vehicle characteristics of the weapons of hostile countries. This is essential in the planning of our strategic policies and is needed to determine where the emphasis should be placed in trying to develop or improve existing U.S. systems. The individual parameters must be known to some tolerance so an adequate comparison can be made with respect to the U.S. systems. Since these countries do not publish any data relating to their launch vehicle systems, we must rely on estimation systems which use observation data to determine these parameters. The data can take many forms since it can be supplied from radar systems that are ground or space-based, or any other form of detection. However in all cases, the basic form of the estimator is essentially the same.

The problem scenario is developed as follows. Upon the launch of an ICBM from a hostile country, our sensor systems respond by first detecting that a launch has indeed occurred, and the subsequent tracking procedure is then carried out. The utility of the estimator, the development of which will

be examined in this paper, is to take this subsequent tracking data and estimate the vehicle performance parameters. Upon successful target recognition, data transmittal, and estimation calculation, some of the ICBM's performance characteristics will be known.

The following diagram shows the general geometry for an orbiting sensor.

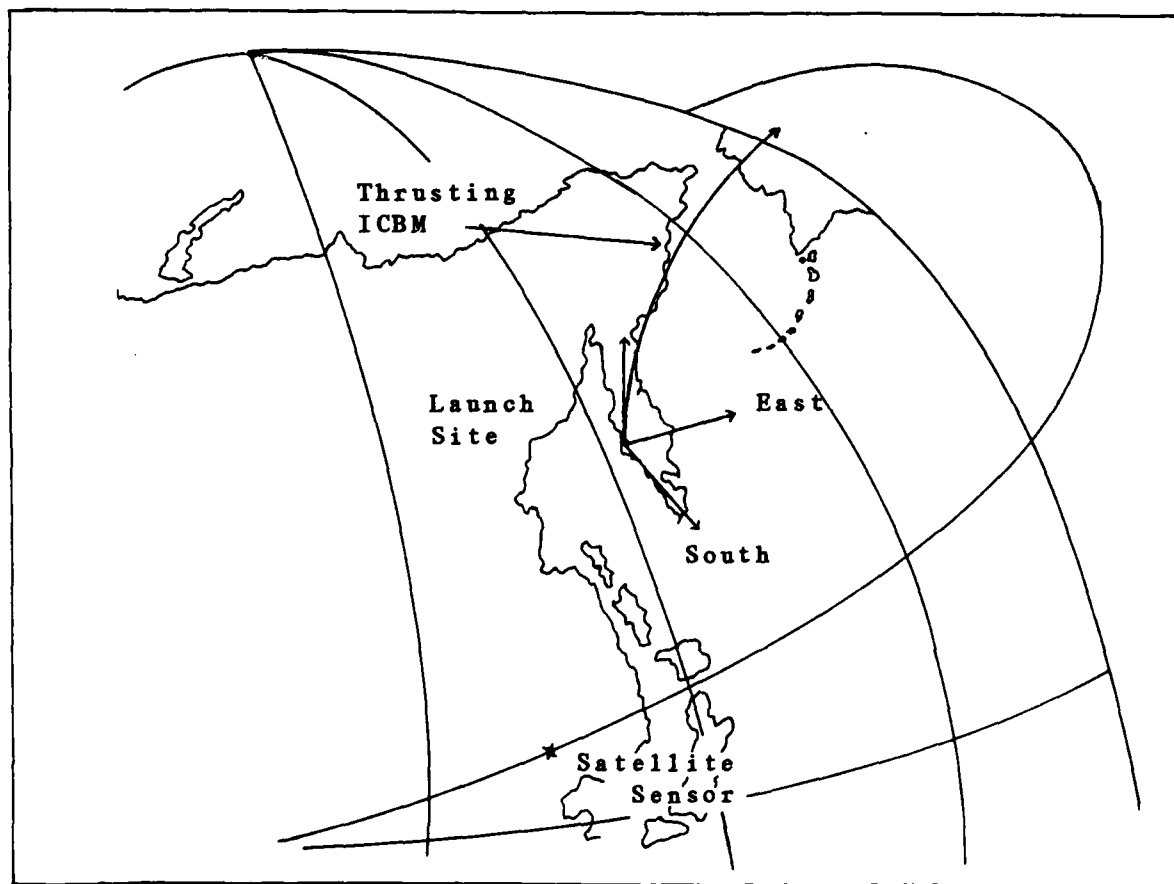


**Figure 1    Orbital Sensor Geometry**

Note that the initial conditions, to be discussed later, will be the launch site of the ICBM, either on the ground or

in the silo. Recognizing the physical limitations in acquiring the target, some time will elapse before the tracking procedure is fully functioning, and the trajectory portion that is actually observed will be only a subset of the entire flight.

The situation with a land based sensor is very similar, with a few distinctions. The basic situation is shown below.

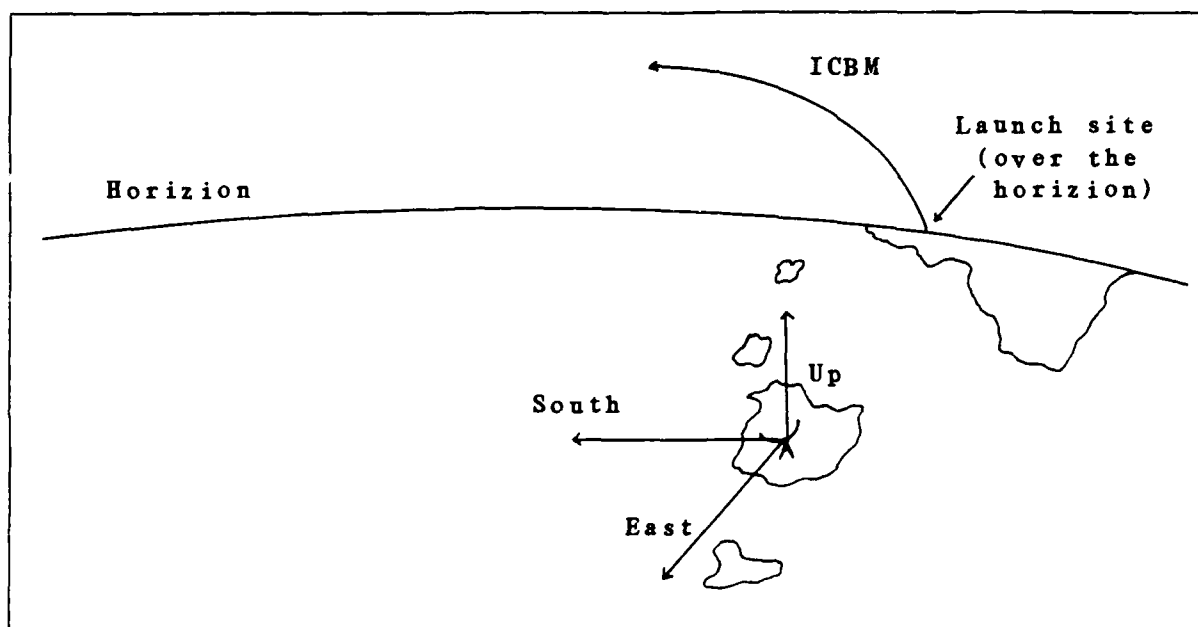


**Figure 2. Land Based Sensor Geometry**

Added difficulty appears in target acquisition since, although the site may be closer, instead of looking down on the target launch, the sensor is forced to discriminate as the vehicle rises over the horizon. In addition, depending on the launch site, the vehicle may not be in view until a significant part of its trajectory has been flown. This would only serve to complicate the tracking procedure, and

where

x, y, z are the 3 components of position  
 $\dot{x}$ ,  $\dot{y}$ ,  $\dot{z}$  are the 3 components of velocity  
 $V_e$  is the exhaust velocity  
 $M$  is the ratio  $\dot{m}$  over initial mass

The two-body equation is non-linear, so to move the state through time, it is differentiated and written in a general form of the equation of motion as :

$$\frac{d}{dt} (\bar{x}(t)) = F(\bar{x}(t), t) \quad (3-2)$$

where  $\bar{x}(t)$  is the state vector at each time.

Notice that this is simply a different expression for equation 2-5.

Using the equations of motion that were developed in the last section, the F vector is found as follows:

$$F = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \dot{V}_e \\ \dot{M} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ -x^{\mu}/r^3 + a \dot{x}/v \\ -y^{\mu}/r^3 + a \dot{y}/v \\ -z^{\mu}/r^3 + a \dot{z}/v \\ 0 \\ 0 \end{bmatrix} \quad (3-3)$$

where  $\bar{a} = V_e \frac{M}{1-Mt}$

Note here that the  $V_e$  and the  $M$  are assumed to be constant. In this program, variations in exhaust velocity and mass ratio are handled through the white Gaussian noise of the

### III FILTER DEVELOPMENT

#### Matrix Equations

The filter will be receiving sampled input data from the sensors. In order to process this data, a Bayes filter estimator was used. The Bayes filter was chosen since the problem dynamics are basically deterministic. Equations of motion can be written specifically for the vehicle, and since it is desired to evaluate each stage's performance, several data segments will need to be processed to obtain the results. The bayes filter uses sequential data segments and produces current estimates of the state and covariance, thereby lending itself to the specific problem of observing each of the stage's performance.

To start, define the state vector ( $\bar{x}$ ) describing the state of the vehicle. Recalling the launch vehicle equations of motion:

$$\bar{x} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ V_e \\ M \end{bmatrix} \quad (3-1)$$

The partial of range, azimuth, and elevation then becomes an identity matrix since the data matrix (G) contains only the variables (range, azimuth, elevation) and no functions of these quantities. The 'noisy' data is then formed as:

$$\begin{bmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{bmatrix}_{\text{noisy}} = \begin{bmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{bmatrix}_{\text{perfect}} + \delta \begin{pmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{pmatrix} \quad (2-23)$$

The next task is to develop the filter which will implement the dynamics formulated in this chapter.

A range of vehicles was chosen to ensure that the filter would operate, given a wide variety of possible input trajectories.

Runs made with the previous information produced actual results. Next the programs in Appendix A were run containing the option to calculate noisy data so that the program could simulate incorrect measurements from the sensors. The result here was to have the data files written with the random errors included in each measurement of range, azimuth and elevation. Basically the approach taken was to assume that the errors would occur randomly as per a Gaussian distribution. To obtain the difference between measurements with and without the noise included (designated  $d( )$ ), let:

$$d \begin{pmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{pmatrix} = G_{\text{au}} \sigma \begin{pmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{pmatrix} \quad (2-20)$$

where

$G_{\text{au}}$  represents a gaussian function whose mean = 0  
and standard deviation is  $\pm 1$   
 $\sigma(\text{range, azimuth, elevation})$  is the defined accuracy of  
each measurement

The  $\sigma(\text{range, azimuth, elevation})$  accuracies were input  
as follows:

$$\sigma \begin{pmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{pmatrix} = \begin{bmatrix} .00001 \text{ DU} \\ .001 \text{ deg} \\ .001 \text{ deg} \end{bmatrix} \cong 64 \text{m} \quad (2-21)$$

Then, using partial derivatives the deviations are:

$$\delta \begin{pmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{pmatrix} = G_{\text{au}} \begin{bmatrix} .00001 \\ .001 \\ .001 \end{bmatrix} \partial \begin{pmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{pmatrix} \quad (2-22)$$

Table 1. Launch Vehicle data

| Quantity            | Stg I  | Stg II | Stg III     | Stg IV |
|---------------------|--------|--------|-------------|--------|
| Titan IIIB          |        |        |             |        |
| I <sub>sp</sub> sec | 256    | 317    | 292 (Agena) |        |
| m <sub>o</sub> lb   | 305970 | 73816  | 14676       |        |
| F lb                | 434900 | 102300 | 16000       |        |
| Titan IIID          |        |        |             |        |
| I <sub>sp</sub> sec | 301    | 317    | 444         | 284    |
| m <sub>o</sub> lb   | 307500 | 73670  | 36122       | 2721   |
| F lb                | 523000 | 102300 | 30000       | 15000  |
| Thor LV-2F          |        |        |             |        |
| I <sub>sp</sub> sec | 251    | 290    |             |        |
| m <sub>o</sub> lb   | 106092 | 1743.7 |             |        |
| F lb                | 170000 | 10000  |             |        |

Using equation (2-19) the parameters are calculated which are needed to enable the correct calculation of the A matrix. The A matrix is simply the assemblage of the equations of variation. The values are as follows.

Table 2. Launch Vehicle Performance Parameters

| Stg | Quantity       | TitanIIIB | TitanIIID | ThorLV-2F |
|-----|----------------|-----------|-----------|-----------|
| I   | V <sub>e</sub> | .317298   | .373693   | .311618   |
|     | M              | 4.479637  | 4.551364  | 5.142138  |
| II  | V <sub>e</sub> | .393557   | .393557   | .360036   |
|     | M              | 3.521417  | 3.528396  | 15.928772 |
| III | V <sub>e</sub> | .362519   | .551228   |           |
|     | M              | 3.007333  | 1.506670  |           |
| IV  | V <sub>e</sub> |           | .352587   |           |
|     | M              |           | 15.634947 |           |

where

V<sub>e</sub> is in DU/TU  
M is non dimensional

$$\bar{p}_{SEZ} = \begin{bmatrix} \hat{S} \\ \hat{E} \\ \hat{Z} \end{bmatrix} \bar{p}_{IJK} \quad (2-17)$$

Then, recalling Figure 4, the data is assembled as:

$$\begin{aligned} \text{range} &= \rho \\ \text{azimuth} &= \tan^{-1} (y / x) \\ \text{elevation} &= \tan^{-1} (z / \sqrt{x^2 + y^2}) \end{aligned} \quad (2-18)$$

This result is general in that the mechanization of finding the unit vectors SEZ is the same for both the radar site and the orbiting sensor. The only difference is the initial calculation of  $\bar{r}_s$ .

#### Truth Model Development

Programming the equations of motion and numerically integrating them provide the numerical integration and truth model data. Appendix A lists the programs which were used to generate the data. In order to look at the launch vehicle in particular however, something must be known about their characteristics, specifically, the exhaust velocity  $V_e$ , and the mass ratio,  $M = \dot{m}$  over initial mass. This data was obtained for a few different missiles from the U.S. Space Launch Systems document, published by the Navy, Reference 5. Recalling from basic propulsion (Sutton Reference 4) that:

$$\dot{m} = F/V_e \quad \text{and} \quad V_e = I_{sp} g \quad \text{and} \quad M = \dot{m}/m_0 \quad (2-19)$$

only  $I_{sp}$ ,  $m_0$ , and  $F$  need to be obtained. The values are as follows.

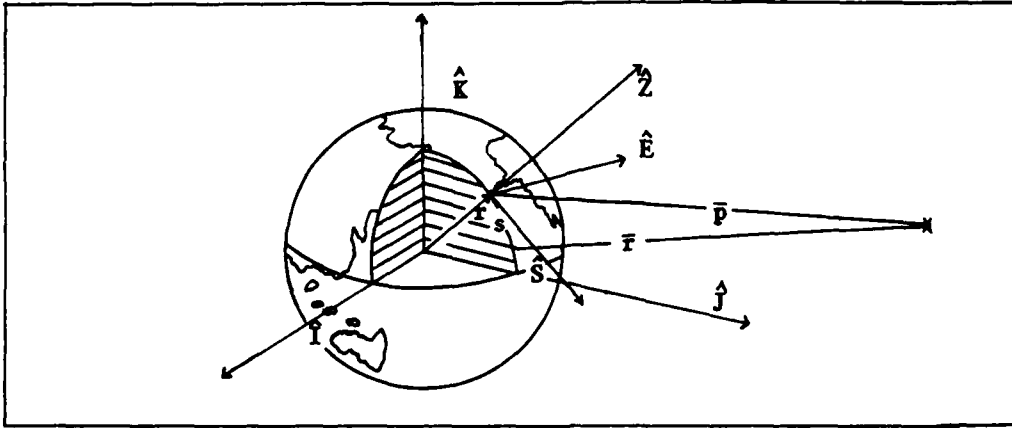


Figure 5. Observation Geometry

$$\hat{Z} = \bar{r}_s / |\bar{r}_s| \quad (2-11)$$

the east unit vector then is

$$\hat{E} = \hat{k} \times \hat{Z} / |\hat{k} \times \hat{Z}| \quad (2-12)$$

and:

$$\hat{S} = \hat{E} \times \hat{Z} / |\hat{E} \times \hat{Z}| \quad (2-13)$$

The transformation matrix is then formed as

$$\begin{bmatrix} I \\ J \\ K \end{bmatrix} = \begin{bmatrix} \hat{S} & | & \hat{E} & | & \hat{Z} \end{bmatrix} \begin{bmatrix} S \\ E \\ Z \end{bmatrix} \quad (2-14)$$

Recalling that the inverse of an orthogonal basis vector is the same as the transpose, this is also:

$$\begin{bmatrix} S \\ E \\ Z \end{bmatrix} = \begin{bmatrix} -\hat{S} & \hat{E} & \hat{Z} \end{bmatrix} \begin{bmatrix} I \\ J \\ K \end{bmatrix} \quad (2-15)$$

To complete the process of finding range, azimuth and elevation, the launch vehicle's position vector is computed as:

$$\bar{\rho}_{IJK} = \bar{r} - \bar{r}_s \quad (IJK) \quad (2-16)$$

transforming to SEZ:

process.) The position vector that is calculated from the orbit elements represents the  $\bar{r}_s$  of the orbiting sensor at that time.

With the site vector for both cases, calculations for the range, azimuth, elevation, and a local coordinate system need to be developed. As will be seen later, the calculation of the data matrix  $G$  and the observation matrix  $H$  (partial of  $G$  with respect to the state vector) is much simplified by the proper choice of coordinate systems. For this reason, the SEZ system was chosen. (Reference 1) With this type of system, range, azimuth and elevation are depicted as follows:

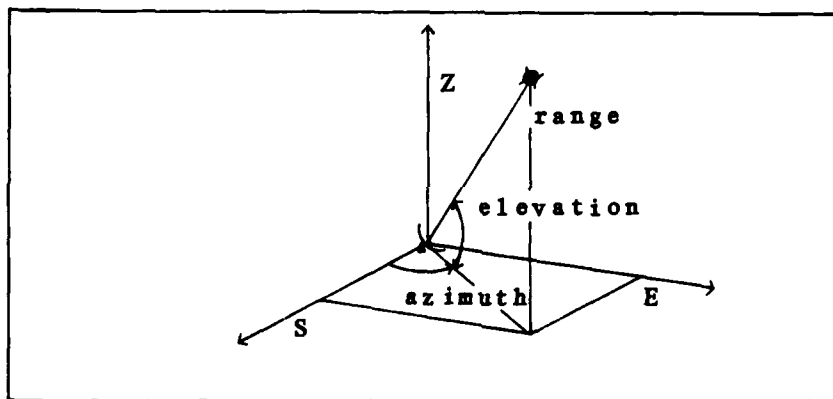


Figure 4. Radar Site geometry

Notice that azimuth is defined as being measured from the South unit vector, rather than the North. Since all of the data is input in the IJK frame, an orthogonal set of unit vectors, SEZ, must be found so that a transformation can be set up. To obtain these unit vectors, Figure 5 is used. The local vertical unit vector is obtained by making the site a unit vector so:

The coordinate system for a land based radar site is shown in Figure 3. Notice that given only latitude, local sidereal time and elevation, the site position vector is obtained.

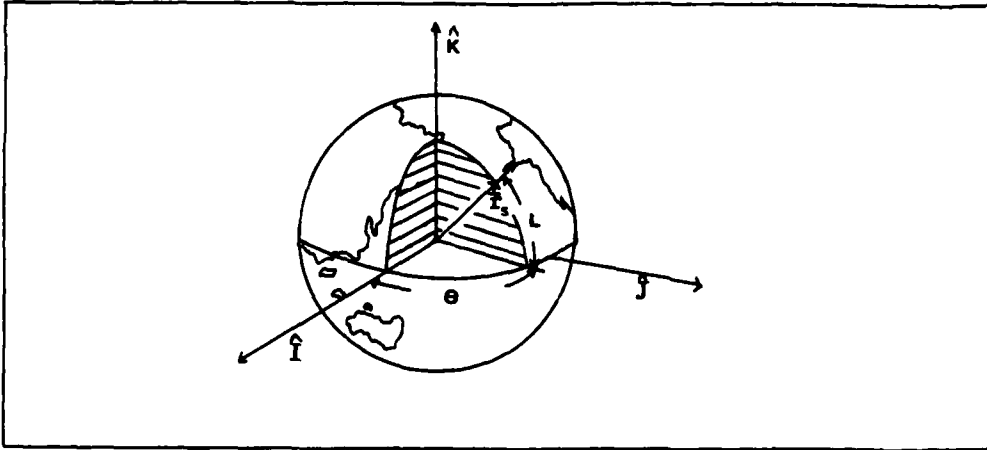


Figure 3. Radar Site Coordinate System

Looking now at an orbiting sensor, the orbit of the sensor must be known. The 5 classical orbit elements are input (Reference 1), and position and velocity vectors can be calculated. From these, the mean motion is calculated as:

$$n = \sqrt{\mu / a^3} \quad (2-9)$$

where

$n$  = mean motion  
 $a$  = semi major axis

The mean anomaly  $M$  is then calculated as:

$$M = n ( t - t_0 ) \quad (2-10)$$

where  $(t - t_0)$  = the time in days past the initial time

Position and velocity vectors can then be calculated. (See Appendix A for a more complete description of the  $\bar{r}$  and  $\bar{v}$

examined an orbiting sensor looking down upon the ICBM trajectory. The observation relationships can be considered identical once the position vector of the observer (radar site, or orbiting sensor) are known.

For a radar site, the position vector can be determined given latitude, longitude, elevation, and universal time. The first step is to calculate the local sidereal time for the site.

$$\theta = \theta_g + \lambda_e \quad (2-6)$$

where

$\theta$  = Local sidereal time  
 $\theta_g$  = greenwich sidereal time  
 $\lambda_e$  = longitude of the site

The Greenwich Sidereal Time is then calculated as:

$$\theta_g = \theta_{go} + 1.0027379093 (t - t_0) 2\pi \quad (2-7)$$

where

$\theta_g$  = Greenwich Sidereal Time (deg)  
 $\theta_{go}$  = the value in deg on 1 Jan 1984, (98.85481°)  
 $(t - t_0)$  = the time in days past the initial time

The position vector of the site can then be calculated as:

$$\bar{r}_s = \begin{bmatrix} h \cos (L) \cos (\theta) \\ h \cos (L) \sin (\theta) \\ h \sin (L) \end{bmatrix} \quad (2-8)$$

where

$h$  = the distance from center of the earth to the site  
 $\bar{r}_s$  = the site vector  
 $L$  = latitude of the site  
 $\theta$  = local sidereal time

that would be considered.

The vehicle will also undergo an acceleration due to thrusting during the propulsive phase of the flight which is equal to:

$$\bar{a} = V_e \frac{\dot{m}}{m_0 - \dot{m}t} \quad (2-3)$$

where

$\bar{a}$  = vehicle acceleration due to thrust  
 $\dot{m}$  = mass flow rate  
 $m_0$  = initial mass  
 $t$  = time  
 $V_e$  = Vehicle exhaust velocity

But since absolute masses are not observable from the trajectory data, let  $M = \dot{m}/m_0$ , and:

$$\bar{a} = V_e \frac{M}{1 - Mt} \quad (2-4)$$

To obtain the total vehicle acceleration, equations (2-1) and (2-4) are added to get:

$$\ddot{\bar{r}} = -\bar{r}^3/r^3 + V_e \frac{M}{1 - Mt} \quad (2-5)$$

where the  $\ddot{\bar{r}}$  denotes the total vehicle acceleration.

These equations constitute the equations of motion for the launch vehicle when they are numerically integrated. The next task is to develop the observation relationships.

#### Observation Relationships

Two cases were considered for the observer. The first case considered a radar site observing the trajectory of an ICBM as it could be seen above the horizon. The second case

## II DYNAMICS FORMULATION

### Equations of Motion

For the specific problem, the equations of motion for the trajectory of the launch vehicle must be generated. Numerically integrating these equations on the computer, will simulate the data that the radar sites would be observing and providing to the sensor system.

The underlying assumption of this work is the spherical earth model and the use of the two body equation of motion (Reference 1), and Newton's Law that the mass times the acceleration is equal to the sum of the forces. In general, the two body equation of motion is written as:

$$\ddot{\vec{r}} + \vec{r} \mu / r^3 = 0 \quad (2-1)$$

where

$\ddot{\vec{r}}$  = vehicle acceleration  
 $\vec{r}$  = radius vector from center of Earth to vehicle  
 $r$  = radius vector magnitude  
 $\mu$  = gravitational parameter defined by:

$$\mu = GM \quad (2-2)$$

where

$G$  = Gravitational potential  
 $M$  = Mass of the Earth

Notice that only gravitational forces are considered since in this paper, all other external forces are assumed to be zero. The assumption is made since the acceleration due to thrust is several orders of magnitude larger than the other forces

the use of radar data in conjunction with the infrared data so that range, azimuth and elevation measurements will be available for computation. In addition, a different local coordinate system is adopted to make the computations easier in the derivation of the various observation relationships. The analysis starts with a specific look at the problem and what data is available. The dynamics are then formulated, yielding the equations of motion and the filter algorithm that will produce tangible results. In order to assess the results of the filter algorithm, a truth model was developed which simulated the data for each parameter that was modeled. Specifically, a reference thrust profile was developed through the computer programs contained in the appendices, and this was used as the 'true' profile. The simulation then proceeds to try to observe this quantity, and the performance of the estimator can thus be observed.

make the target acquisition process slower.

Previous work was done using infrared sensors to determine the position and velocity vectors and vehicle acceleration for the launch vehicle ( Reference 3 ). The method used azimuth and elevation measurements and concluded that a 7-state filter could solve for vehicle acceleration. One of the problems with this particular approach was the extreme complexity of the data and observation relationships. This made for extremely difficult checkout and computer coding. Gross's presentation (Reference 2), provided additional work centered on a space-based infrared sensor system in an attempt to get additional data from the orbiting sensor. The main emphasis here centered on a Bayes filter to observe exhaust velocity and vehicle mass, and the same filter to observe vehicle acceleration. The simulation did not yield significant results for the first case, but was able to observe the vehicle acceleration. The main result seemed to conclude that a fading memory differential corrector might be useful and that additional input data could change the poor results in the expanded estimation system. As before however, the observation relationships were extremely involved.

The majority of this paper will be based on trying to extend the Bayes filter analysis already started in an attempt to develop a filter that will provide as much data as possible. The primary difference with this attempt will be

observations. The dynamics model could incorporate very complex expressions to represent the variations, but this was not done in this paper.

The previous relations establish the laws that will govern the motion of the launch vehicle. The next problem is to determine how to correct the estimate of the state from the input data (coming from the radar sites or satellites), which the estimator will be processing. To accomplish this, a vector is found through the observation relationships which were developed in the previous chapter, and the following operations are performed.

In order to estimate the state, a nominal trajectory is assumed as a function of time ( $\bar{x}(t)$ ), with initial conditions, and the true trajectory is written as:

$$\bar{x}(t) = \bar{x}_0(t) + \delta \bar{x}(t) \quad (3-4)$$

where

$\bar{x}(t)$  = the true solution

$\delta \bar{x}(t)$  = the difference between the nominal and true trajectory

Differentiating yields:

$$\dot{\bar{x}}(t) = \dot{\bar{x}}_0(t) + \delta \dot{\bar{x}}(t) \quad (3-5)$$

and adding to Equation (3-2):

$$\dot{\bar{x}}_0(t) + \delta \dot{\bar{x}}(t) = F(\bar{x}_0(t) + \delta \bar{x}(t), t) \quad (3-6)$$

To solve this we expand the right hand side of Equation 3-6 with Taylor's theorem, and obtain:

$$\begin{aligned} \dot{\bar{x}}_0(t) + \delta \dot{\bar{x}}(t) &= F(\bar{x}_0(t), t) + A(t) \Big|_{\bar{x}_0(t)} \delta \bar{x}(t) \\ &+ \frac{1}{2} \nabla_x (\nabla_x \delta) \Big|_{\bar{x}_0(t)} (\delta \bar{x}(t))^2 + \text{H.O.T.} \end{aligned} \quad (3-7)$$

where  $A(t) = \partial F / \partial \bar{x}$  (the  $A$  matrix is derived in Appendix D)

Now assuming that  $\delta \bar{x}$  is small, Equation (3-7) becomes:

$$\dot{\bar{x}}_0(t) = F(\bar{x}_0(t), t) \quad (3-8)$$

Ignoring higher order terms in Equation (3-7) and subtracting Equation (3-8), we obtain:

$$\delta \dot{\bar{x}}(t) = A(t) \Big|_{\bar{x}_0(t)} \delta \bar{x}(t) \quad (3-9)$$

Recalling the idea of state transition matrices, the state variations are moved through time as follows:

$$\delta \bar{x}(t) = \phi(t, t_0) \delta \bar{x}(t_0) \quad (3-10)$$

where  $\phi(t, t_0)$  is the state transition matrix

$\phi$  is then calculated from:

$$\dot{\phi}(t, t_0) = A(t) \Big|_{\bar{x}_0(t)} \phi(t, t_0) \quad (3-11)$$

The state transition matrix definition also prescribes the initial conditions as:

$$\phi(t_0, t_0) = I \quad (3-12)$$

The preceding equations will enable the calculation of the state vector, propagation through time, and the estimation of the errors from the true trajectory as a function of time.

The next part of the problem is to process the data that will be coming in from the sensors. The predicted data for each observation is given by:

$$\bar{z}(t_i) = G(\bar{x}(t_i), t_i) \quad (3-13)$$

Evaluating this at the initial time, the initial conditions become:

$$\bar{z}_0(t_i) = G(\bar{x}_0(t_i), t_i) \quad (3-14)$$

Also, knowing that there will be some difference between this and the true measurement, let:

$$\bar{z}(t_i) = G(\bar{x}_0(t_i) + \delta\bar{x}(t_i), t_i) \quad (3-15)$$

where  $\delta\bar{x}(t_i)$  is the difference from the true data

This equation can be expanded in a Taylor series as follows:

$$\begin{aligned} \bar{z}(t_i) = G(\bar{x}_0(t_i), t_i) + \frac{\partial G(\bar{x}_0(t_i), t_i)}{\partial \bar{x}(t_i)} \bigg|_{\bar{x}_0(t_i)} \delta\bar{x}(t_i) \\ + \text{H. O. T.} \end{aligned} \quad (3-16)$$

Subtracting this 'true' relation from the calculated relation gives the residual of the observation:

$$\begin{aligned} \bar{r}(t_i) &= \bar{z}(t_i) - \bar{z}_0(t_i) \\ &= \frac{\partial G}{\partial \bar{x}} \bigg|_{\bar{x}_0(t)} \delta\bar{x}(t_i) \\ &= H(\bar{x}_0(t_i), t_i) \delta\bar{x}(t_i) \end{aligned} \quad (3-17)$$

Note that here, as before, the higher order terms have been ignored.

In the previous chapters, observation relationships were developed. The data vector  $G$  consists of the range, azimuth and elevation.

$$[G] = \begin{bmatrix} \text{range} \\ \text{azimuth} \\ \text{elevation} \end{bmatrix} \quad (3-18)$$

The H matrix (observation relation) is simply the partial derivative of G with respect to the state vector, so:

$$[H] = \partial G / \partial \bar{x} \quad (3-19)$$

Using the observation relationships that were developed in the last chapters:

$$\begin{aligned} \text{range} &= \sqrt{x^2 + y^2 + z^2} \\ \text{azimuth} &= \tan^{-1} (y / x) \\ \text{elevation} &= \tan^{-1} (z / \sqrt{x^2 + y^2}) \end{aligned} \quad (3-20)$$

Since only x, y, and z appear in the G matrix, only the first 3 x 3 block of the H matrix will have non zero elements. Using the following definition then:

$$d/dx \tan^{-1} u = 1/(1+u^2) (du/dx) \quad (3-21)$$

The first 3x3 of the H matrix is then:

$$H = \begin{bmatrix} \frac{x}{(x^2+y^2+z^2)^{3/2}} & \frac{y}{(x^2+y^2+z^2)^{3/2}} & \frac{z}{(x^2+y^2+z^2)^{3/2}} \\ \frac{-y/x^2}{1+(y/x)^2} & \frac{1/x}{1+(y/x)^2} & 0 \\ \frac{-xz/(x^2+y^2)^{3/2}}{1+z^2/(x^2+y^2)} & \frac{-yz/(x^2+y^2)^{3/2}}{1+z^2/(x^2+y^2)} & \frac{1/(x^2+y^2)^{1/2}}{1+z^2/(x^2+y^2)} \end{bmatrix} \quad (3-22)$$

The final step is to move the residuals to a single epoch time. Using the state transition matrix which was developed before:

$$\begin{aligned} \bar{r}(t_i) &= H(\bar{x}_0(t_i), t_i) \phi(t_i, t_0) \delta \bar{x}(t_0) \\ &= T(t_i) \delta \bar{x}(t_0) \end{aligned} \quad (3-23)$$

With the residuals calculated, the necessary matrices are available and the filter can be derived.

### Least Squares Filter Development

Summarizing the data already obtained, the state vector is given by:

$$\dot{\bar{x}} = F(\bar{x}, t) \quad (3-24)$$

deviations of the state vector are known as:

$$\delta\bar{x}(t) = \phi(t, t_0) \delta\bar{x}(t_0) \quad (3-25)$$

The observation relationships were developed as G, and the residual data was:

$$\bar{r}(t_i) = T(t_i) \delta\bar{x}(t_0) \quad (3-26)$$

The sensors will not input perfect data, and the covariance matrix Q tells us how accurate the data is (for each measurement of range, azimuth and elevation), and how each quantity affects the other. (See chapter 2 for numerical values. ) The residual vector, including this error is written as:

$$\bar{r}(t_i) = T(t_i) \delta\bar{x}(t_0) + \bar{e}(t_i) \quad (3-27)$$

To calculate this error, rearrange as:

$$\bar{e}(t_i) = \bar{r}(t_i) - T(t_i) \delta\bar{x}(t_0) \quad (3-28)$$

Assuming that for each observation, a random errors in range, azimuth or elevation are uncorrelated with each other, a data covariance matrix (Q) defined which contains the information about the accuracy of the measurements. Using Gaussian error

statistics, the probability density function for the error vector  $\bar{e}(t_1)$  is:

$$P(\bar{e}) = (2\pi)^{-N/2} [Q]^{-1/2} \exp(-J/2) \quad (3-29)$$

where

$N$  = number of measurements  
 $Q$  = data covariance matrix  
 $J = \bar{e}^T Q^{-1} \bar{e}$  (a scalar) weighted least squares function

Using the principle of maximum likelihood,  $J$  is minimized to make  $P$  a maximum. ( $P$  is a maximum when the residual errors  $e$  are the smallest) Thus:

$$\frac{\partial J}{\partial \bar{x}} = \frac{\partial}{\partial \bar{x}} (\bar{e}^T Q^{-1} \bar{e}) = 0 \quad (3-30)$$

Now substituting into  $J$  as:

$$\begin{aligned} J &= (\bar{r} - T \delta \bar{x})^T Q^{-1} (\bar{r} - T \delta \bar{x}) \\ &= \bar{r}^T Q^{-1} \bar{r} - \bar{r}^T Q^{-1} T \delta \bar{x} - \delta \bar{x}^T T^T Q^{-1} \bar{r} + \delta \bar{x}^T T^T Q^{-1} T \delta \bar{x} \end{aligned} \quad (3-31)$$

Note here that the functional dependence on time is intentionally left out to enhance clarity. Thus, Equation (3-30) becomes:

$$\begin{aligned} 0 &= -(\bar{r}^T Q^{-1} T) - T^T Q^{-1} \bar{r} + (\delta \bar{x}^T T^T Q^{-1} T) + T^T Q^{-1} T \delta \bar{x} \\ &= -2T^T Q^{-1} \bar{r} + 2T^T Q^{-1} T \delta \bar{x} \end{aligned} \quad (3-32)$$

and solving for  $\delta \bar{x}$ :

$$\delta \bar{x} = (T^T Q^{-1} T)^{-1} T^T Q^{-1} \bar{r} \quad (3-33)$$

This result is valid when the trajectories are very close. Iteration is needed to get the trajectories to have a very small difference.

Next, the covariance is to be calculated. In general,

the  $\delta \bar{x}$  can be written as:

$$\delta \bar{x} = W \bar{r} \quad (3-34)$$

where  $W = (T^T Q^{-1} T)^{-1} T^T Q^{-1}$

The covariance of  $\bar{x}$  which produced this error, given  $\bar{x}(t_0)$  at the epoch time (assuming  $\delta \bar{x}$  as zero mean) is:

$$P_{\bar{x}}(t_0) = E(\delta \bar{x}, \delta \bar{x}^T) \quad (3-35)$$

where  $\delta \bar{x}$  is  $W \bar{r} = \bar{x} - \bar{x}_0$

Substituting and recognizing that  $W$  can be calculated (the assumption of deterministic dynamics for the problem), it is pulled out of the expectation operator:

$$P_{\bar{x}}(t_0) = W E(\bar{r} \bar{r}^T) W^T \quad (3-36)$$

But recall that  $E(\bar{r} \bar{r}^T)$  is defined as the covariance matrix  $Q$ , where  $\bar{r}$  is the zero mean, thus:

$$P_{\bar{x}}(t_0) = W Q W^T \quad (3-37)$$

Expanding,

$$\begin{aligned} P_{\bar{x}}(t_0) &= (T^T Q^{-1} T)^{-1} T^T Q^{-1} Q [(T^T Q^{-1} T)^{-1} T^T Q^{-1}]^T \\ &= (T^T Q^{-1} T)^{-1} T^T Q^{-1} T (T^T Q^{-1} T)^{-1} \\ &= (T^T Q^{-1} T)^{-1} \end{aligned} \quad (3-38)$$

The final step which is needed is to define when the estimator has reached convergence. Under perfect conditions, the  $\delta x$  will converge to zero. However, it is sufficient to stop the iteration when the state corrections are all less than the square root of their individual covariance values. Knowing this, the algorithm for the least squares filter can

be summarized and written as shown in Appendix H.

### Bayes Filter Development

The process of changing to a Bayes filter is relatively easy since the only real difference is that the Bayes filter will process segments of data in a least squares mode, and then update it's state and covariance matrix to some time in the future. The major changes are detailed below.

Again recalling the Least Squares development, the matrices that will change are as follows, realizing that the previous estimate and matrices can now be treated as 'data':

$$T = \begin{bmatrix} I \\ - \\ T_n \end{bmatrix} \quad Q = \begin{bmatrix} P(-) & | & 0 \\ \hline 0 & | & Q_n \end{bmatrix}$$

$$\bar{r} = \begin{bmatrix} \bar{x}(-) - \bar{x}_{ref} \\ \hline Z_n - G(\bar{x})_n \end{bmatrix} \quad (3-39)$$

where the subscript n denotes the new portion of data

Then:

$$P^{-1}(+) = (P^{-1}(-) \quad T_n^T Q_n^{-1}) \begin{bmatrix} I \\ - \\ T_n \end{bmatrix}$$

$$= P^{-1}(-) + T_n^T Q_n^{-1} T_n \quad (3-40)$$

and the corrections are:

$$\delta \bar{x}(t_0) = P(+)^T Q_n^{-1} \bar{r} \quad (3-41)$$

$$= P(+)^T (P^{-1}(-) \quad T_n^T Q_n^{-1}) \begin{bmatrix} \bar{x}(-) - \bar{x}_{ref} \\ \hline \bar{r}_n \end{bmatrix}$$

and finally:

$$\delta \bar{x}(t_0) = P(+)^T (P^{-1}(-) (\bar{x}(-) - \bar{x}_{ref}) + T_n^T Q_n^{-1} \bar{r}_n) \quad (3-42)$$

One additional device must be added to the least squares development to incorporate the fading memory effect. The object here is to have the filter retain full memory about components which do not change drastically during the flight, (position and velocity vectors) and to retain little memory of those elements which will change rapidly at certain times, (exhaust velocity and mass ratio). The staging event is the primary cause of the changes. The fading memory filtering is accomplished by multiplying the covariance matrix by a matrix of scalar  $\beta$ 's after convergence has been achieved.

$$P^{-1}(-) = \beta P^{-1}(-) \beta^T \quad (3-43)$$

Values of  $\beta$  range from 0 to 1. When  $\beta = 0$ , the filter does not retain memory about the previous states, and when  $\beta = 1$ , the filter retains all previous data.

The Summary for the Bayes Filter Algorithm is shown in Appendix I.

## IV TESTING

### Computer Program Development

The computer programs were developed by simply coding the equations and formulas which were developed in the previous sections. The programs also relied on basis programs which were demonstrated during the Modern Methods of Orbit Determination Course. (Reference 7) The appendices give a brief description of each of the programs, with their inputs and outputs.

To ensure the programs were correct, a succession of checks were run to determine that each stage of the program was indeed functional.

The first check was of the numerical integrator. A program was written which accomplished this by numerically integrating an orbit, once around. Appendix A details the inputs and outputs, and the procedures used. It was noted that the number of steps was crucial in solving the problem. A step size that was too large could not be used with a high altitude orbit, or an elliptical orbit. The starting point for the integration was also very important since at perigee, the spacecraft is moving faster, thereby requiring smaller step sizes. Table 3 lists numerical integration results for an orbit giving position (DU) and velocity (DU/TU) vectors through one revolution. If the numerical integrator were perfect, and there were no roundoff errors, the first and

Table 3. Satellite Orbit, Numerical Integration Data

| a                                             | e                 | i                 | omega             | argp          | nuo           | m    | period  | # it |
|-----------------------------------------------|-------------------|-------------------|-------------------|---------------|---------------|------|---------|------|
| 2.588                                         | .000              | .785              | .000              | .000          | .000          | .000 | 333.973 | 8    |
| the specific mech energy and ang momentum are |                   |                   |                   |               |               |      |         |      |
| -2.2222222222222222 1.5811388333333333        |                   |                   |                   |               |               |      |         |      |
| x                                             | y                 | z                 | xdot              | ydot          | zdot          |      |         |      |
| 2.5880000000000000                            | .0000000000000000 | .0000000000000000 | .0000000000000000 | .44721359551  | .44721359549  |      |         |      |
| 2.48828675328                                 | .22155994842      | .22155994841      | -.07926769692     | .44368718273  | .44368718271  |      |         |      |
| 2.42145798282                                 | .43962576488      | .43962576486      | -.15728529498     | .43316355884  | .43316355882  |      |         |      |
| 2.32444121472                                 | .65875841883      | .65875841882      | -.23282248982     | .41588868527  | .41588868526  |      |         |      |
| 2.19876678811                                 | .85162822456      | .85162822452      | -.38468777626     | .39189626115  | .39189626113  |      |         |      |
| 2.02254248594                                 | 1.03986734445     | 1.03986734441     | -.37174883446     | .36188339888  | .36188339887  |      |         |      |
| 1.82242156856                                 | 1.21811975162     | 1.21811975156     | -.43294568488     | .32688468888  | .32688468887  |      |         |      |
| 1.59355997439                                 | 1.36288784741     | 1.36288784735     | -.48731536289     | .28586467432  | .28586467431  |      |         |      |
| 1.33956698747                                 | 1.49257584188     | 1.49257584111     | -.53399986733     | .23962982756  | .23962982755  |      |         |      |
| 1.06444822895                                 | 1.59952336151     | 1.59952336144     | -.57226287486     | .19841428788  | .19841428787  |      |         |      |
| -2.2222222222222222 1.5811388333333333        |                   |                   |                   |               |               |      |         |      |
| .77254248598                                  | 1.68124627992     | 1.68124627984     | -.68158895558     | .13819668113  | .13819668113  |      |         |      |
| .46845328652                                  | 1.73645494817     | 1.73645494809     | -.62125388577     | .08379947143  | .08379947143  |      |         |      |
| .15697629889                                  | 1.76427866871     | 1.76427866863     | -.63128752551     | .02888877481  | .02888877481  |      |         |      |
| -.15697629875                                 | 1.76427866871     | 1.76427866863     | -.63128752551     | -.02888877481 | -.02888877481 |      |         |      |
| -.46845328638                                 | 1.73645494818     | 1.73645494810     | -.62125388577     | -.08379947142 | -.08379947142 |      |         |      |
| -.77254248585                                 | 1.68124627993     | 1.68124627985     | -.68158895551     | -.13819668112 | -.13819668112 |      |         |      |
| -1.06444822881                                | 1.59952336152     | 1.59952336145     | -.57226287487     | -.19841428787 | -.19841428786 |      |         |      |
| -1.33956698734                                | 1.4925758419      | 1.49257584112     | -.53399986734     | -.23962982755 | -.23962982754 |      |         |      |
| -1.59355997426                                | 1.36288784742     | 1.36288784736     | -.48731536291     | -.28586467432 | -.28586467431 |      |         |      |
| -1.82242156844                                | 1.21811975163     | 1.21811975158     | -.43294568489     | -.32688468889 | -.32688468887 |      |         |      |
| -2.2222222222222222 1.5811388333333333        |                   |                   |                   |               |               |      |         |      |
| -2.02254248582                                | 1.03986734446     | 1.03986734442     | -.37174883446     | -.36188339889 | -.36188339887 |      |         |      |
| -2.19876669999                                | .85162822456      | .85162822453      | -.38468777626     | -.39189626116 | -.39189626114 |      |         |      |
| -2.32444121459                                | .65875841882      | .65875841880      | -.23282248982     | -.41588868528 | -.41588868527 |      |         |      |
| -2.42145798269                                | .43962576488      | .43962576486      | -.15728529489     | -.43316355885 | -.43316355883 |      |         |      |
| -2.48828675314                                | .22155994841      | .22155994840      | -.07926769698     | -.44368718275 | -.44368718273 |      |         |      |
| -2.49999999984                                | .00000000000      | .00000000000      | .00000000000      | -.44721359552 | -.44721359550 |      |         |      |
| -2.48828675312                                | .22155994841      | -.22155994840     | .07926769694      | -.44368718275 | -.44368718273 |      |         |      |
| -2.42145798264                                | .43962576488      | -.43962576486     | .15728529494      | -.43316355885 | -.43316355883 |      |         |      |
| -2.32444121452                                | .65875841882      | -.65875841880     | .23282248987      | -.41588868528 | -.41588868526 |      |         |      |
| -2.19876669988                                | .85162822456      | -.85162822452     | .38468777632      | -.39189626115 | -.39189626114 |      |         |      |
| -2.2222222222222222 1.5811388333333333        |                   |                   |                   |               |               |      |         |      |
| -2.02254248569                                | 1.03986734445     | 1.03986734441     | .37174883452      | -.36188339888 | -.36188339886 |      |         |      |
| -1.82242156827                                | 1.21811975161     | 1.21811975156     | .43294568415      | -.32688468887 | -.32688468885 |      |         |      |
| -1.59355997486                                | 1.36288784739     | 1.36288784733     | .48731536297      | -.28586467438 | -.28586467428 |      |         |      |
| -1.33956698711                                | 1.4925758415      | 1.49257584088     | .53399986741      | -.23962982752 | -.23962982751 |      |         |      |
| -1.06444822855                                | 1.59952336146     | 1.59952336138     | .57226287494      | -.19841428783 | -.19841428782 |      |         |      |
| -.77254248555                                 | 1.68124627984     | 1.68124627976     | .68158895557      | -.13819668187 | -.13819668186 |      |         |      |
| -.46845328685                                 | 1.73645494805     | 1.73645494398     | .62125388583      | -.08379947136 | -.08379947135 |      |         |      |
| -.15697629839                                 | 1.76427866855     | 1.76427866847     | .63128752556      | -.02888877392 | -.02888877392 |      |         |      |
| .15697629927                                  | 1.76427866858     | 1.76427866843     | .63128752555      | .02888877418  | .02888877409  |      |         |      |
| .46845328691                                  | 1.73645494392     | 1.73645493984     | .62125388579      | .08379947153  | .08379947153  |      |         |      |
| -2.2222222222222222 1.5811388333333333        |                   |                   |                   |               |               |      |         |      |
| .77254248638                                  | 1.68124627962     | 1.68124627954     | .68158895558      | .13819668124  | .13819668123  |      |         |      |
| 1.06444822934                                 | 1.59952336115     | 1.59952336188     | .57226287484      | .19841428799  | .19841428798  |      |         |      |
| 1.33956698784                                 | 1.4925758376      | 1.4925758369      | .53399986728      | .23962982768  | .23962982766  |      |         |      |
| 1.59355997472                                 | 1.36288784693     | 1.36288784687     | .48731536281      | .28586467444  | .28586467443  |      |         |      |
| 1.82242156885                                 | 1.21811975188     | 1.21811975182     | .43294568397      | .32688468188  | .32688468187  |      |         |      |
| 2.02254248616                                 | 1.03986734385     | 1.03986734381     | .37174883431      | .36188339988  | .36188339987  |      |         |      |
| 2.19876678825                                 | .85162822388      | .85162822386      | .38468777689      | .39189626126  | .39189626124  |      |         |      |
| 2.32444121476                                 | .65875841812      | .65875841809      | .23282248962      | .41588868537  | .41588868535  |      |         |      |
| 2.42145798274                                 | .43962576333      | .43962576331      | .15728529466      | .43316355811  | .43316355809  |      |         |      |
| 2.48828675388                                 | .22155994764      | .22155994763      | .07926769666      | .44368718279  | .44368718277  |      |         |      |
| -2.2222222222222222 1.5811388333333333        |                   |                   |                   |               |               |      |         |      |
| x                                             | y                 | z                 | xdot              | ydot          | zdot          |      |         |      |
| 2.49999999966                                 | .00000000079      | .00000000079      | -.00000000028     | .44721359554  | .44721359552  |      |         |      |

last state vectors would be identical. The very small,  $10^{-9}$  errors, represent the imperfections.

Table 4 lists sample output for the launch trajectory. The data is for a Titan IIIB, launched from a site at  $53.7^{\circ}$  N,  $158.2^{\circ}$  E. The initial velocity was 200 ft/sec straight up (local coordinate system), and to displace the velocity so that the vehicle would execute a gravity turn, this initial value was perturbed by .06 DU/TU. In a gravity turn there is a pre-programmed displacement to perturb the vehicle's direction so that gravity will cause the vehicle to fall over and reach burnout in the correct orientation. The output contains the position and velocity magnitudes,  $\gamma$ , and the time,  $V_e$  and M. Note that with a .06 displacement, at the end of data, the ICBM is virtually horizontal ( $\gamma = 80^{\circ}$ ), and in orbit. This is easily changed by altering the amount of the displacement.

The next step was to check the A matrix. Appendix D lists the A matrix, and the program which accomplished the check. Given an initial state vector, the A matrix was calculated. Then, each element of the input state was perturbed, and the A matrix was calculated by columns as:

$$A_{ij} = \frac{F_i(\bar{x} + \delta, t) - F_i(\bar{x}, t)}{\delta} \quad (4-1)$$

Table 5 lists the results for a trial case. Note that each state was perturbed by about  $1 \times 10^{-4}$ , and the observed delta

Table 4. ICBM Launch Trajectory, Numerical Integration Data

The initial state vector for the missile is

```

- .4579893757416e-08 - .5625717569683e+08 .6883545756938e-08 - .3531837967384e-02
      ydot      zdot      xdot
- .4338112164634e-02 .5626533647914e-02 .3172982551738e-08 .4479637558632e+01
the initial time is .0020 TU      M
r (km)      v (ft/sec)      gamma (deg)      time (sec)      ve (DU/TU)      m
6378.430588      263.814633      2.8953242033      5.6136237488      .3172982552      4.4796375586
6378.789127      326.359237      4.4846479838      9.6136237488      .3172982552      4.4796375586
6379.225596      393.765854      6.4144524548      13.6136237488      .3172982552      4.4796375586
6379.744871      466.471967      8.6493455674      17.6136237488      .3172982552      4.4796375586
6380.351603      544.993952      11.1451502446      21.6136237488      .3172982552      4.4796375586
6381.050878      629.919834      13.8525358355      25.6136237488      .3172982552      4.4796375586
6381.844101      721.896054      16.7208663601      29.6136237488      .3172982552      4.4796375586
6382.756897      821.618423      19.6967954942      33.6136237488      .3172982552      4.4796375586
6383.731848      929.773986      22.7344241792      37.6136237488      .3172982552      4.4796375586
6384.828431      1047.183788      25.7899883878      41.6136237488      .3172982552      4.4796375586
6386.030381      1174.318352      28.8219788486      45.6136237488      .3172982552      4.4796375586
6387.337255      1312.888445      31.8818569378      49.6136237488      .3172982552      4.4796375586
6388.749345      1461.114587      34.7881278663      53.6136237488      .3172982552      4.4796375586
6390.266165      1622.448931      37.4991481939      57.6136237488      .3172982552      4.4796375586
6391.886952      1795.543147      40.1835516549      61.6136237488      .3172982552      4.4796375586
6393.610783      1982.258598      42.7436382888      65.6136237488      .3172982552      4.4796375586
6395.436292      2182.848797      45.1736818688      69.6136237488      .3172982552      4.4796375586
6397.362579      2398.815889      47.4712761883      73.6136237488      .3172982552      4.4796375586
6399.388524      2628.528764      49.6365374388      77.6136237488      .3172982552      4.4796375586
6401.494541      2823.542549      51.6896474784      81.6136237488      .3929844800      3.5272654570
6403.655588      3027.878349      53.6529187642      85.6136237488      .3929844800      3.5272654570
6405.868128      3242.866483      55.5259659388      89.6136237488      .3929844800      3.5272654570
6408.129193      3468.884445      57.3893457148      93.6136237488      .3929844800      3.5272654570
6410.435582      3707.938547      59.8843861852      97.6136237488      .3929844800      3.5272654570
6412.784312      3959.672688      62.6138887733      101.6136237488      .3929844800      3.5272654570
6415.172571      4224.576523      62.1375427256      105.6136237488      .3929844800      3.5272654570
6417.597755      4503.195472      63.5886516542      109.6136237488      .3929844800      3.5272654570
6420.057503      4796.139981      64.9451664714      113.6136237488      .3929844800      3.5272654570
6422.549729      5104.897117      66.2348269751      117.6136237488      .3929844800      3.5272654570
6425.072654      5427.843164      67.4582859825      121.6136237488      .3929844800      3.5272654570
6427.624844      5768.258228      68.5966544919      125.6136237488      .3929844800      3.5272654570
6430.205238      6126.343318      69.6762594849      129.6136237488      .3929844800      3.5272654570
6432.813194      6503.248598      70.6918114876      133.6136237488      .3929844800      3.5272654570
6435.448525      6908.257717      71.6459815223      137.6136237488      .3929844800      3.5272654570
6438.111553      7318.897373      72.5413828788      141.6136237488      .3929844800      3.5272654570
6440.803161      7760.893661      73.3881589795      145.6136237488      .3929844800      3.5272654570
6443.524858      8228.257251      74.1647743763      149.6136237488      .3929844800      3.5272654570
6446.278862      8723.322899      74.8972882936      153.6136237488      .3929844800      3.5272654570
6449.068189      9248.868313      75.5793496879      157.6136237488      .3929844800      3.5272654570
6451.896772      9808.114288      76.2129129198      161.6136237488      .3929844800      3.5272654570
6454.769599      10404.939722      76.7994331915      165.6136237488      .3929844800      3.5272654570
6457.692897      11043.996945      77.3482688925      169.6136237488      .3929844800      3.5272654570
6460.674355      11730.939759      77.8365538476      173.6136237488      .3929844800      3.5272654570
6463.723417      12472.724757      78.2892662998      177.6136237488      .3929844800      3.5272654570
6466.851662      13278.835739      78.6991338153      181.6136237488      .3929844800      3.5272654570
6470.073319      14157.896975      79.0666462888      185.6136237488      .3929844800      3.5272654570
6473.405962      15126.585985      79.3928223679      189.6136237488      .3929844800      3.5272654570
6476.871492      16203.840581      79.6751521371      193.6136237488      .3929844800      3.5272654570
6480.497557      17413.128586      79.9155267793      197.6136237488      .3929844800      3.5272654570
6484.319783      18793.434422      80.1121229189      201.6136237488      .3929844800      3.5272654570
end of data run

```



## V Results and Conclusions

### ICBM Performance Parameters Estimation

With confidence that the programs are functioning correctly, as obtained from the results in the last chapter, the original problem of how to estimate the performance parameters of an ICBM in flight now begins. Due to time constraints caused by heavy computer usage, only one general case was examined, however the trends are quite certain to extend to the other cases which were programmed into the filter.

Sample test results are listed in Appendix J. The case which is presented uses 100 data point segments. Convergence is shown for all regions, and the covariance is listed. The case uses a radar site at  $52.6^{\circ}$  North,  $174.1^{\circ}$  East, 1 DU in elevation, and a launch point at  $43^{\circ}$  North,  $132^{\circ}$  East. A portion of the truth model observation data for the test case is listed in Table 15.

Table 15. Truth model data, ICBM test case

| range<br>(km) | azimuth<br>(deg) | elevation<br>(deg) | time<br>(sec) |
|---------------|------------------|--------------------|---------------|
| 1065.0411     | 257.08497        | -4.7879401         | 2.0136233     |
| 1065.0601     | 257.08497        | -4.7866484         | 2.4131237     |
| 1065.0801     | 257.08497        | -4.7853194         | 2.8136233     |
| 1065.0998     | 257.08495        | -4.7839529         | 3.2136237     |
| 1065.1196     | 257.08494        | -4.7825492         | 3.6136232     |
| 1065.1394     | 257.08494        | -4.7811076         | 4.0136236     |

The truth model also numerically integrated the trajectory shown in Table 16. Notice that only the first 200

the simulated noisy data, with the covariance data for both trials listed in Table 13.

The tabulated data contains the corrections to the state from the last iteration of each Bayes loop. Again, all units are canonical unless specified. It is interesting to note that the filter converged on the perfect state within 2 iterations, and took only 1 additional iteration to converge on the other cases for the perfect data. Also the magnitude of the state corrections were almost all  $10^{-15}$  which is very good. As with the least squares results, the data converged almost identically, with the only difference being the amount of time required to reach the solution, and the difference of order  $10^{-8}$  is only slightly above the results from the least squares perfect data. The covariance matrices are again very small, with the error ellipsoid axis lengths decreasing as the filter processed more data. This is expected since the confidence in the estimate should go up as additional data is processed.

The simulated noisy calculations show the same trend as the noisy results for the least squares. The convergence took about one extra iteration, and the final errors were of order  $10^{-3}$ , which is just above the least squares, noisy data results. The error ellipsoid axes again decreased with time, and were comparable with the perfect data results.

Table 14. Bayes Filter Test Results, Noisy Data

| iteration                       |              |               |               |
|---------------------------------|--------------|---------------|---------------|
|                                 | # 3          | # 4           | # 4           |
|                                 | loop 1       | loop 2        | loop 3        |
| CASE # 1                        | -.1737332e-8 | .3037524e-9   | -.4274729e-9  |
|                                 | .5348930e-10 | .1155548e-9   | -.1121525e-9  |
|                                 | .7905271e-9  | .4252964e-11  | -.4957654e-10 |
|                                 | .2448065e-8  | -.1910199e-8  | .2261524e-9   |
|                                 | .7611670e-10 | -.4510833e-9  | -.1429128e-10 |
|                                 | -.1175545e-8 | -.4293026e-10 | -.3024271e-10 |
|                                 | # 3          | # 4           | # 4           |
| Case # 2                        | -.8691772e-9 | .3037522e-9   | -.4274728e-9  |
|                                 | -.3560272e-9 | .1155548e-9   | -.1121525e-9  |
|                                 | .3095512e-9  | .4252958e-11  | -.4957650e-10 |
|                                 | .7122785e-8  | -.1910192e-8  | .2261527e-9   |
|                                 | .2991415e-9  | -.4510835e-9  | -.1429111e-10 |
|                                 | -.1698415e-8 | -.4293023e-10 | -.3024270e-10 |
|                                 | # 3          | # 4           | # 4           |
| CASE # 3                        | .6331097e-9  | .3037522e-9   | -.4274728e-9  |
|                                 | -.1559395e-9 | .1155548e-9   | -.1121525e-9  |
|                                 | .5717885e-9  | .4252939e-11  | -.4957651e-10 |
|                                 | -.8614662e-8 | -.1910198e-8  | .2261527e-9   |
|                                 | .8116607e-9  | -.4510833e-9  | -.1429150e-10 |
|                                 | .4998575e-9  | -.4293025e-10 | -.3024260e-10 |
|                                 | # 3          | # 4           | # 4           |
| CASE # 4                        | .8821206e-7  | .3037518e-9   | -.4274729e-9  |
|                                 | -.1524874e-7 | .1155548e-9   | -.1121525e-9  |
|                                 | -.2799510e-7 | .4252960e-11  | -.4957651e-10 |
|                                 | -.8882739e-7 | -.1910197e-8  | .2261525e-9   |
|                                 | .1766154e-7  | -.4510830e-9  | -.1429134e-10 |
|                                 | .3044111e-7  | -.4293025e-10 | -.3024286e-10 |
| the final corrected states are: |              |               |               |
|                                 | Time 1       | Time 2        | Time 3        |
|                                 | .2499575e+1  | .2424369e+1   | .2194183e+1   |
|                                 | -.2669495e-3 | .4396153e+0   | .8536396e+0   |
|                                 | -.8122994e-4 | .4388443e+0   | .8521426e+0   |
|                                 | .2021882e-2  | -.1570918e+0  | -.3146224e+0  |
|                                 | .4481786e+0  | .4336677e+0   | .3864206e+0   |
|                                 | .4470668e+0  | .4334313e+0   | .3892277e+0   |

Table 13. Covariance data for Bayes Filter Tests (continued)

Noisy Observation Data

Covariance Matrix at epoch is:

|               |               |               |               |               |               |
|---------------|---------------|---------------|---------------|---------------|---------------|
| .5180505e-13  | -.1095052e-13 | -.1781172e-13 | -.1731339e-13 | .1035341e-13  | .1243912e-13  |
| -.1095052e-13 | .2469536e-14  | .3835462e-14  | .2185800e-14  | -.2429973e-14 | -.2439921e-14 |
| -.1781172e-13 | .3835462e-14  | .6281015e-14  | .4552161e-14  | -.3752743e-14 | -.4224094e-14 |
| -.1731339e-13 | .2185800e-14  | .4552161e-14  | .3859435e-13  | -.3337219e-14 | -.0616303e-14 |
| .1035341e-13  | -.2429973e-14 | -.3752743e-14 | -.3337219e-14 | .2500323e-14  | .2503209e-14  |
| .1243912e-13  | -.2439921e-14 | -.4224094e-14 | -.0616303e-14 | .2503209e-14  | .3841942e-14  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.137829e+01 meters  
 .105405e+02 meters  
 .000000e+00 meters

the error ellipsoid axis lengths for the position components are

.156509e+01 meters  
 .893509e-01 meters  
 .536925e-01 meters

Covariance Matrix at epoch is:

|               |               |               |               |               |               |
|---------------|---------------|---------------|---------------|---------------|---------------|
| .5689967e-13  | .2893223e-15  | -.7439639e-14 | -.1623987e-13 | .9627018e-14  | .1052307e-13  |
| .2893223e-15  | .1110523e-15  | .1536843e-16  | -.1103610e-14 | -.2911966e-15 | -.5215173e-16 |
| -.7439639e-14 | .1536843e-16  | .1080156e-14  | .1241479e-14  | -.1496019e-14 | -.1554642e-14 |
| -.1623987e-13 | -.1103610e-14 | .1241479e-14  | .2469651e-13  | .2350799e-14  | -.1108601e-14 |
| .9627018e-14  | -.2911966e-15 | -.1496019e-14 | .2350799e-14  | .3067733e-14  | .2271053e-14  |
| .1052307e-13  | -.5215173e-16 | -.1554642e-14 | -.1188691e-14 | .2271053e-14  | .2293103e-14  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.147560e+01 meters  
 .421826e-03 meters  
 .421826e-03 meters  
 .627327e-03 meters/sec

the error ellipsoid axis lengths for the position components are

.153441e+01 meters  
 .472052e-01 meters  
 .808000e-01 meters

Covariance Matrix at epoch is:

|               |               |               |               |              |               |
|---------------|---------------|---------------|---------------|--------------|---------------|
| .5294432e-13  | .1316609e-13  | .5108883e-14  | -.1688632e-13 | .8200553e-14 | .8774566e-14  |
| .1316609e-13  | .3348274e-14  | .1315473e-14  | -.4725914e-14 | .1694544e-14 | .1954117e-14  |
| .5108883e-14  | .1315473e-14  | .5723077e-15  | -.2019637e-14 | .5300090e-15 | .6101017e-15  |
| -.1688632e-13 | -.4725914e-14 | -.2019637e-14 | .1213249e-13  | .1470759e-14 | -.1545602e-15 |
| .8200553e-14  | .1694544e-14  | .5300090e-15  | .1470759e-14  | .3094437e-14 | .3032544e-14  |
| .8774566e-14  | .1954117e-14  | .6101017e-15  | -.1545602e-15 | .3032544e-14 | .2652000e-14  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00 | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00 | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.142217e+01 meters  
 .544872e-00 meters  
 .309400e-03 meters  
 .482731e-03 meters/sec

the error ellipsoid axis lengths for the position components are

.151900e+01 meters  
 .684679e-01 meters

Table 13. Covariance Data for Bayes Filter Tests

Perfect Observation Data

Covariance Matrix at epoch is:

|               |               |               |               |               |               |
|---------------|---------------|---------------|---------------|---------------|---------------|
| .5185616e-13  | -.1095270e-13 | -.1782283e-13 | -.1738451e-13 | .1084102e-13  | .1245239e-13  |
| -.1095270e-13 | .2467963e-14  | .3834707e-14  | .2199219e-14  | -.2424886e-14 | -.2440795e-14 |
| -.1782283e-13 | .3034737e-14  | .6202652e-14  | .4572740e-14  | -.3740550e-14 | -.4226799e-14 |
| -.1738451e-13 | .2199219e-14  | .4572740e-14  | .3367913e-13  | -.3407667e-14 | -.3640367e-14 |
| .1084102e-13  | -.2424886e-14 | -.3740550e-14 | -.3407667e-14 | .2492565e-14  | .2592311e-14  |
| .1245239e-13  | -.2440795e-14 | -.4226799e-14 | -.3640367e-14 | .2592311e-14  | .3046068e-14  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.137874e+01 meters  
 .105428e-02 meters  
 .000000e+00 meters

the error ellipsoid axis lengths for the position components are

.156656e+01 meters  
 .093346e-01 meters  
 .586845e-01 meters

Covariance Matrix at epoch is:

|               |               |               |               |               |               |
|---------------|---------------|---------------|---------------|---------------|---------------|
| .5686771e-13  | .2867172e-15  | -.7425419e-14 | -.1618743e-13 | .9636801e-14  | .1052816e-13  |
| .2867172e-15  | .1106718e-15  | .1562240e-16  | -.1098916e-14 | -.2904672e-15 | -.5265900e-16 |
| -.7425419e-14 | .1562240e-16  | .1076660e-14  | .1234741e-14  | -.1494939e-14 | -.1552548e-14 |
| -.1618743e-13 | -.1098916e-14 | .1234741e-14  | .2458098e-13  | .2335850e-14  | -.1100364e-14 |
| .9636801e-14  | -.2904672e-15 | -.1494939e-14 | .2335850e-14  | .3067210e-14  | .2274167e-14  |
| .1052816e-13  | -.5265900e-16 | -.1552548e-14 | -.1100364e-14 | .2274167e-14  | .2294373e-14  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.147520e+01 meters  
 .421604e-03 meters  
 .421604e-03 meters  
 .625206e-03 meters/sec

the error ellipsoid axis lengths for the position components are

.153295e+01 meters  
 .471102e-01 meters  
 .806210e-01 meters

Covariance Matrix at epoch is:

|               |               |               |               |              |               |
|---------------|---------------|---------------|---------------|--------------|---------------|
| .5262459e-13  | .1305212e-13  | .5072951e-14  | -.1654805e-13 | .3405727e-14 | .9808699e-14  |
| .1305212e-13  | .3311402e-14  | .1303232e-14  | -.4633724e-14 | .1716350e-14 | .1955901e-14  |
| .5072951e-14  | .1303232e-14  | .5682836e-15  | -.1986832e-14 | .5495307e-15 | .6121819e-15  |
| -.1654805e-13 | -.4633724e-14 | -.1986832e-14 | .1200593e-13  | .1510901e-14 | -.1065965e-15 |
| .3405727e-14  | .1716350e-14  | .5495307e-15  | .1510901e-14  | .3056334e-14 | .3076475e-14  |
| .9808699e-14  | .1955901e-14  | .6121819e-15  | -.1065965e-15 | .3076475e-14 | .2679237e-14  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00 | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00 | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.141731e+01 meters  
 .141731e+01 meters  
 .547602e-03 meters  
 .387014e-03 meters/sec  
 .470700e-03 meters/sec

the error ellipsoid axis lengths for the position components are

.151416e+01 meters  
 .684823e-01 meters  
 .356544e-01 meters

Table 12. Bayes Filter Test Results, Perfect Observations

| Corrections to the state from the last iteration: |               |               |               |
|---------------------------------------------------|---------------|---------------|---------------|
| iteration                                         | loop 1        | loop 2        | loop 3        |
| # 2                                               | # 2           | # 2           | # 2           |
| CASE # 1                                          | -.3524090e-15 | .3775959e-15  | .7204185e-16  |
|                                                   | .1607043e-15  | -.3591601e-16 | .1625420e-16  |
|                                                   | .7630261e-16  | .4048324e-17  | .8456722e-17  |
|                                                   | .7847741e-15  | -.1044663e-14 | -.6064243e-15 |
|                                                   | -.1693562e-15 | .6318910e-16  | -.4719534e-15 |
|                                                   | -.1939084e-15 | .3291155e-18  | -.1421949e-15 |
| # 3                                               | # 2           | # 2           | # 2           |
| CASE # 2                                          | -.6998720e-13 | .5343816e-15  | .7503202e-16  |
|                                                   | .1195828e-14  | .1436418e-15  | .7101780e-16  |
|                                                   | .1974012e-14  | -.6164276e-16 | .4390330e-16  |
|                                                   | .9340976e-13  | -.1522799e-14 | -.4117666e-15 |
|                                                   | .4582638e-14  | -.2906382e-15 | -.6517259e-15 |
|                                                   | .4104247e-14  | .6863711e-16  | -.1882977e-15 |
| # 3                                               | # 2           | # 2           | # 2           |
| CASE # 3                                          | .2342559e-14  | .4178976e-15  | .2650948e-15  |
|                                                   | -.5523967e-15 | .6595235e-16  | .1126490e-15  |
|                                                   | -.1012912e-14 | -.6635110e-16 | .3337257e-16  |
|                                                   | -.3881522e-14 | -.1287997e-14 | -.8903416e-15 |
|                                                   | .1229017e-14  | -.1083272e-15 | -.7648698e-15 |
|                                                   | .2243644e-14  | .7944296e-16  | -.1241624e-15 |
| # 3                                               | # 2           | # 2           | # 2           |
| CASE # 4                                          | -.1011434e-8  | .1709862e-15  | .3549278e-16  |
|                                                   | .5786753e-9   | -.9786427e-19 | -.5614404e-17 |
|                                                   | .7691572e-9   | -.5444362e-17 | .1247841e-16  |
|                                                   | .1788590e-8   | -.3533691e-15 | -.1854145e-15 |
|                                                   | -.8509662e-10 | .5039547e-16  | -.3902865e-15 |
|                                                   | -.7039212e-9  | .4078983e-16  | -.6353586e-16 |
| The final corrected states are:                   |               |               |               |
|                                                   | Time 1        | Time 2        | Time 3        |
|                                                   | .2500000e+1   | .2421457e+1   | .2190766e+1   |
|                                                   | .2511528e-8   | .4396257e+0   | .8516282e+0   |
|                                                   | -.1926930e-8  | .4396257e+0   | .8516282e+0   |
|                                                   | -.7865713e-8  | -.1572853e+0  | -.3046877e+0  |
|                                                   | .4472135e+0   | .4331635e+0   | .3918962e+0   |
|                                                   | .4472135e+0   | .4331635e+0   | .3918962e+0   |

Table 11. Least Squares Test Results, Noisy Observations

| CASE # 1 Corrections for iteration # |              |               |               |
|--------------------------------------|--------------|---------------|---------------|
| 1                                    | 2            | 3             |               |
| .7924433e-4                          | -.4206869e-8 | .1216706e-11  |               |
| .1077903e-4                          | -.3045991e-8 | -.1852942e-12 |               |
| .9042774e-5                          | -.7908962e-9 | .2482118e-11  |               |
| -.1686651e-4                         | .1018113e-8  | -.4850761e-12 |               |
| -.1781953e-4                         | .1238650e-8  | -.9031411e-13 |               |
| -.3911945e-5                         | .3462006e-9  | -.3183258e-12 |               |
| CASE # 2 Corrections for iteration # |              |               |               |
| 1                                    | 2            | 3             | 4             |
| .1538294e-2                          | .4096174e-4  | -.1647061e-7  | -.1540987e-10 |
| .2617755e-5                          | .8171601e-5  | -.1336570e-7  | .2158184e-11  |
| -.1334271e-3                         | .1424602e-3  | .8914906e-8   | -.3576532e-10 |
| -.2302934e-5                         | -.1456163e-4 | -.9364774e-9  | .6372505e-11  |
| .3163803e-4                          | -.4945985e-4 | .3522301e-8   | .5392571e-12  |
| -.3960326e-4                         | .3569532e-4  | -.3660874e-8  | .4893863e-11  |
| CASE # 3 Corrections for iteration # |              |               |               |
| 1                                    | 2            | 3             | 4             |
| .6927874e-4                          | .9976133e-5  | -.1474618e-7  | -.1859578e-11 |
| .1848499e-4                          | -.7699773e-5 | -.9227279e-8  | -.2557389e-11 |
| -.4008284e-4                         | .4913705e-4  | -.1221827e-7  | -.1033730e-11 |
| -.1013049e-2                         | -.3820235e-5 | .3970187e-8   | .5265535e-12  |
| -.3238347e-5                         | -.1458009e-4 | .1520327e-9   | -.7616239e-13 |
| -.1385391e-4                         | .9938250e-5  | .4062703e-8   | .6695699e-12  |
| CASE # 4 Corrections for iteration # |              |               |               |
| 1                                    | 2            | 3             | 4             |
| .1465780e-2                          | .1135676e-3  | -.1081471e-6  | -.1593583e-10 |
| -.1008484e-1                         | .9563121e-4  | -.1354342e-7  | -.6378967e-11 |
| -.1009848e-1                         | .1076447e-3  | -.1127688e-6  | -.2897397e-10 |
| -.9720443e-3                         | -.4485862e-4 | .3746891e-7   | .6037466e-11  |
| .5100931e-3                          | -.8061368e-4 | -.2256346e-8  | -.4146842e-12 |
| .4021184e-3                          | .4123910e-4  | .2633033e-7   | .5925330e-11  |
| The final corrected state was:       |              |               |               |
| .2500079e+1                          |              |               |               |
| .1077599e-4                          |              |               |               |
| .9041986e-5                          |              |               |               |
| -.1686550e-4                         |              |               |               |
| .4471957e+0                          |              |               |               |
| .4472096e+0                          |              |               |               |

the actual solution. Finally, the relative magnitude of the differences between the exact and the estimated state are all about order  $10^{-9}$  which are the same order as seen in the numerical integration checkout.

To complete checkout of the least squares filter, the same four cases were tested with the simulated noisy data from the truth model. Table 11 lists the results. In general, the results were similar to those with the perfect data: however, it generally took longer for the filter to eliminate the noise which was present in the data, and the final state had errors of order  $10^{-4}$ . This represents a significant difference to the least squares results. As can be seen in Table 10, the covariance matrix was also the same for all the cases, and almost identical to the perfect data case.

#### **Bayes Filter Checkout**

The last check which was performed was to check the Bayes filter estimator. Again, for consistency, the same truth model data was used. Here however, there was an effort to limit the computational time due to extremely heavy usage of the computer, therefore, only 3 iterations of the Bayes filter were tested, with each of the runs processing 30 data points. This would require only a few matrix inversions, and proved adequate to show trends in the performance of the filter. The four cases were tested against the perfect data, and then the simulated noisy data. Table 12 lists the perfect data tests and Table 14 lists the Bayes filter estimations of

Table 10. Covariance Data for Least Squares Tests

PERFECT OBSERVATION DATA

Covariance Matrix at epoch is:

|               |               |               |               |               |               |
|---------------|---------------|---------------|---------------|---------------|---------------|
| .1894375e-17  | .1906222e-17  | .1949674e-17  | -.7362794e-18 | -.2880636e-18 | -.2487960e-18 |
| .1906222e-17  | .3105600e-17  | .1954685e-17  | -.8456409e-18 | -.3394847e-18 | -.2068683e-18 |
| .1949674e-17  | .1954605e-17  | .3374542e-17  | -.9002667e-18 | -.2249767e-18 | -.3332718e-18 |
| -.7362794e-18 | -.8456409e-18 | -.9002667e-18 | .3262177e-18  | .1080523e-18  | .9734323e-19  |
| -.2880636e-18 | -.3394847e-18 | -.2249767e-18 | .1080523e-18  | .6719715e-19  | .1457902e-19  |
| -.2487960e-18 | -.2068688e-18 | -.3332718e-18 | .9784323e-19  | .1457902e-19  | .5707239e-19  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.772805e-02 meters  
 .565357e-05 meters  
 .565357e-05 meters

the error ellipsoid axis lengths for the position components are

.165522e-01 meters  
 .380495e-02 meters  
 .722606e-02 meters

NOISY OBSERVATION DATA

Covariance Matrix at epoch is:

|               |               |               |               |               |               |
|---------------|---------------|---------------|---------------|---------------|---------------|
| .1894579e-17  | .1906361e-17  | .1949884e-17  | -.7363331e-18 | -.2880884e-18 | -.2480251e-18 |
| .1906361e-17  | .3105715e-17  | .1954824e-17  | -.8456602e-18 | -.3395106e-18 | -.2068873e-18 |
| .1949884e-17  | .1954824e-17  | .3374814e-17  | -.9003164e-18 | -.2250010e-18 | -.3333097e-18 |
| -.7363331e-18 | -.8456602e-18 | -.9003164e-18 | .3262245e-18  | .1080591e-18  | .9785107e-19  |
| -.2880884e-18 | -.3395106e-18 | -.2250010e-18 | .1080591e-18  | .6719833e-19  | .1458403e-19  |
| -.2480251e-18 | -.2068873e-18 | -.3333097e-18 | .9785107e-19  | .1458403e-19  | .5707367e-19  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |
| .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  | .0000000e+00  |

the error ellipsoid axis lengths for the covariance matrix are

.772809e-02 meters  
 .565409e-05 meters  
 .565409e-05 meters

the error ellipsoid axis lengths for the position components are

.165529e-01 meters  
 .380508e-02 meters  
 .722700e-02 meters



Each of the cases was run with the least squares filter and the results are shown in Table 9. (units are canonical and only 6 states are shown since the satellite data consisted only of position and velocity vectors.) Note that the filter converged in only 2 iterations on the perfect data case, yet it required 4 iterations on the perturbed initial state vectors. In each case, the final state was the same since the estimator was using the same truth model data. The filter, after processing all 500 data points of one period of the orbit, improves the initial guess by a very small amount,  $10^{-9}$ , which can really be treated as truncation and machine error. A measure of the accuracy of the filter is the covariance matrix, from which the error ellipsoid axis lengths can be determined. The procedure here is to calculate the eigenvalues of the covariance matrix. The square root of each eigenvalue is then the error ellipsoid axis length. Table 10 lists the covariance matrix and the error ellipsoid axis lengths. The covariance was the same for all the different cases using the same truth model data, which simply means that the same degree of confidence can be placed with each estimate. This of course assumes that the dynamics model is valid. The second set of axis lengths are considered to be more accurate since they represent the upper left  $3 \times 3$  partition of the covariance matrix. This is the best measure of the accuracy of the position components. The filter seemed to adapt rapidly to the errors, correct them, and converge on

The truth model program generated the numerical integration data shown in Table 3 and also produced the truth model (range, azimuth and elevation) data for the orbit. Table 7 lists the first few lines, and the last few lines of the truth model data as it was used. (both perfect and noisy data simulations are included)

Table 7. Truth model data for satellite orbit

| simulated perfect observation data |                  |                    |               |
|------------------------------------|------------------|--------------------|---------------|
| range<br>(DU)                      | azimuth<br>(rad) | elevation<br>(rad) | time<br>(TU)  |
| 2.10828311                         | 5.43568181       | .190435864e+0      | .496729413e-1 |
| 2.09648962                         | 5.43864117       | .203551598e+0      | .993458827e-1 |
| 2.08473224                         | 5.44165050       | .216743101e+0      | .149018824e+0 |
| 2.07301339                         | 5.44471208       | .230011201e+0      | .198691765e+0 |
| ...                                |                  |                    |               |
| 2.29091161                         | 5.21078168       | -.714578847e-3     | .246874518e+2 |
| 2.27868763                         | 5.21196854       | .115080426e-1      | .247371248e+2 |
| simulated noisy observation data:  |                  |                    |               |
| 2.10828823                         | 5.43455972       | .189748349e+0      | .496729413e-1 |
| 2.09648944                         | 5.43893145       | .204563277e+0      | .993458827e-1 |
| 2.08473754                         | 5.44310397       | .216215598e+0      | .149018824e+0 |
| 2.07303439                         | 5.44533297       | .231198678e+0      | .198691765e+0 |
| ...                                |                  |                    |               |
| 2.29090397                         | 5.21055858       | .940219542e-3      | .246874518e+2 |
| 2.27869430                         | 5.21154640       | .127423844e-1      | .247371248e+2 |

For each test, four cases of initial input data vectors were run. Table 8 summarizes the test cases.

Table 8. Filter test cases

| case 1      | case 2      | case 3      | case 4      |
|-------------|-------------|-------------|-------------|
| .2500000e+1 | .2498000e+1 | .2500000e+1 | .2498000e+1 |
| .0000000e+0 | .0000000e+0 | .0000000e+0 | .1000000e-2 |
| .0000000e+0 | .0000000e+0 | .0000000e+0 | .1000000e-2 |
| .0000000e+0 | .0000000e+0 | .1000000e-2 | .1000000e-2 |
| .4472135e+0 | .4472135e+0 | .4472135e+0 | .4467663e+0 |
| .4472135e+0 | .4472135e+0 | .4472135e+0 | .4467663e+0 |

It is important to realize that all the calculations must be made in the IJK frame. There are several places in the programs where conversions between SEZ and IJK frames are performed, and later when runs are made with the Least Squares and the Bayes programs, their input data, in order to be consistent, must be all in the IJK frame.

#### Least Squares Checkout

The next step was to run a trial program of Least Squares. This would serve only to check the combination of all the different matrices and formulations that were developed thus far. The program estimates the original state vector, given only an initial guess, and the range, azimuth, elvation, and time data from the truth model. The program is listed in Appendix H. For consistency, all of the test cases for the Least Squares and the Bayes filter were identical, using the following input data.

Table 6. Test Cases, Input Data

|                             |
|-----------------------------|
| Satellite orbit             |
| a = 2.5 DU                  |
| e = 0                       |
| i = 45 deg                  |
| $\Omega$ = 0 deg            |
| $\omega$ = 0 deg            |
| M = 0 deg                   |
| radar site                  |
| 45 deg North                |
| 60 deg West                 |
| 1.0225DU elevation          |
| dt = period/500 TU          |
| t <sub>0</sub> = 0 TU       |
| initial velocity 200 ft/sec |
| displacement of .06 DU/TU   |

in the calculated matrix was also about  $1 \times 10^{-4}$ .

Next, the  $\phi$  matrix was calculated and checked. See Appendix E for the checkout program. This was accomplished by first setting the state vector to an initial value and saving it. The state was then moved through time (which worked out to be about  $1/4$  of the orbit), and again saved. Individually then, each of the elements of the state vector were perturbed at the original time and translated through time. The columns of the  $\phi$  matrix could then be calculated as:

$$\phi = \frac{X(\bar{x}(t_0) + \delta, t) - X(\bar{x}(t_0), t)}{\delta} \quad (4-2)$$

It should be noted here that after each perturbation, the state and the  $\phi$  matrix were reinitialized. Table 5 lists the output and shows that, for input errors to the state vector of about  $1 \times 10^{-4}$ , the same order of magnitude errors were observed in the calculated  $\phi$  matrix.

Once the G and H matrices were programmed, a check was performed on the H matrix. The program is listed in Appendix F. The state vector is input, and one call to obser calculates the H matrix directly. Then, similar to the check for the A matrix, each one of the state vector elements are perturbed, and the columns of the H matrix are calculated as:

$$[H] = \frac{G(\bar{x} + \delta, t) - G(\bar{x}, t)}{\delta} \quad (4-3)$$

Table 16. Numerical Integration of ICBM Test Case

The initial state vector for the missile is

$x$   $y$   $z$   $\dot{x}$   
 $-1.1326113891294e-08$   $-5.769695352121e-08$   $.8859282822486e-08$   $-1.822594488953e-02$   
 $\dot{y}$   $\dot{z}$   $V_e$   $M$   
 $-4.449136538335e-02$   $.6276833691368e-02$   $.3172982551738e-08$   $.4479637558632e+01$   
 the initial time is .8828 TU

| r (km)      | v (ft/sec)   | gamma (deg)    | time (sec)     | ve (DU/TU)  | m            |
|-------------|--------------|----------------|----------------|-------------|--------------|
| 6378.425311 | 259.243619   | .4765268348    | 5.6136237488   | .3172982552 | 4.4796375586 |
| 6378.778878 | 321.528788   | .7435895547    | 9.6136237488   | .3172982552 | 4.4796375586 |
| 6379.211182 | 388.369675   | 1.8696148558   | 13.6136237488  | .3172982552 | 4.4796375586 |
| 6379.727641 | 459.996128   | 1.4495886812   | 17.6136237488  | .3172982552 | 4.4796375586 |
| 6380.334418 | 536.656672   | 1.8778713579   | 21.6136237488  | .3172982552 | 4.4796375586 |
| 6381.837648 | 618.628189   | 2.3458886248   | 25.6136237488  | .3172982552 | 4.4796375586 |
| 6381.843819 | 706.177191   | 2.8481994289   | 29.6136237488  | .3172982552 | 4.4796375586 |
| 6382.759789 | 799.642473   | 3.3782526931   | 33.6136237488  | .3172982552 | 4.4796375586 |
| 6383.792733 | 899.356393   | 3.9294871189   | 37.6136237488  | .3172982552 | 4.4796375586 |
| 6384.958288 | 1005.687698  | 4.4957822772   | 41.6136237488  | .3172982552 | 4.4796375586 |
| 6386.248181 | 1119.836281  | 5.0728895664   | 45.6136237488  | .3172982552 | 4.4796375586 |
| 6387.671861 | 1239.836538  | 5.6532571869   | 49.6136237488  | .3172982552 | 4.4796375586 |
| 6389.251743 | 1368.561347  | 6.2352294852   | 53.6136237488  | .3172982552 | 4.4796375586 |
| 6390.989882 | 1505.726783  | 6.8141584768   | 57.6136237488  | .3172982552 | 4.4796375586 |
| 6392.989882 | 1651.897727  | 7.3867361718   | 61.6136237488  | .3172982552 | 4.4796375586 |
| 6394.989882 | 1807.694513  | 7.9581599517   | 65.6136237488  | .3172982552 | 4.4796375586 |
| 6397.278258 | 1973.888849  | 8.5281517588   | 69.6136237488  | .3172982552 | 4.4796375586 |
| 6399.754111 | 2158.973268  | 9.0982768696   | 73.6136237488  | .3172982552 | 4.4796375586 |
| 6402.464652 | 2348.852397  | 9.6632588935   | 77.6136237488  | .3172982552 | 4.4796375586 |
| 6405.356695 | 2498.385197  | 10.2747264442  | 81.6136237488  | .3929844888 | 3.5272654578 |
| 6408.436116 | 2645.832961  | 10.8795952776  | 85.6136237488  | .3929844888 | 3.5272654578 |
| 6411.781678 | 2809.651417  | 11.4765767476  | 89.6136237488  | .3929844888 | 3.5272654578 |
| 6415.162762 | 2982.238499  | 11.5645842968  | 93.6136237488  | .3929844888 | 3.5272654578 |
| 6418.829345 | 3164.112753  | 12.8423381122  | 97.6136237488  | .3929844888 | 3.5272654578 |
| 6422.711963 | 3355.839115  | 12.5891584371  | 101.6136237488 | .3929844888 | 3.5272654578 |
| 6426.821817 | 3558.834961  | 12.9641583753  | 105.6136237488 | .3929844888 | 3.5272654578 |
| 6431.178828 | 3771.377151  | 13.4866362988  | 109.6136237488 | .3929844888 | 3.5272654578 |
| 6435.771714 | 3996.618781  | 13.8359888178  | 113.6136237488 | .3929844888 | 3.5272654578 |
| 6440.638878 | 4234.556584  | 14.2516988823  | 117.6136237488 | .3929844888 | 3.5272654578 |
| 6445.784583 | 4486.126839  | 14.6533359155  | 121.6136237488 | .3929844888 | 3.5272654578 |
| 6451.226663 | 4752.335875  | 15.0485488725  | 125.6136237488 | .3929844888 | 3.5272654578 |
| 6456.981461 | 5034.314665  | 15.4138186941  | 129.6136237488 | .3929844888 | 3.5272654578 |
| 6463.867178 | 5333.335523  | 15.7785381986  | 133.6136237488 | .3929844888 | 3.5272654578 |
| 6469.583616 | 5658.834831  | 16.1129169784  | 137.6136237488 | .3929844888 | 3.5272654578 |
| 6476.312379 | 5988.436427  | 16.4408186731  | 141.6136237488 | .3929844888 | 3.5272654578 |
| 6483.517843 | 6347.998892  | 16.7517455262  | 145.6136237488 | .3929844888 | 3.5272654578 |
| 6491.143488 | 6731.648165  | 17.0488328723  | 149.6136237488 | .3929844888 | 3.5272654578 |
| 6499.228286 | 7141.846485  | 17.3288389399  | 153.6136237488 | .3929844888 | 3.5272654578 |
| 6507.778798 | 7581.456181  | 17.5941467558  | 157.6136237488 | .3929844888 | 3.5272654578 |
| 6516.854418 | 8053.836767  | 17.8439588588  | 161.6136237488 | .3929844888 | 3.5272654578 |
| 6526.486452 | 8562.967167  | 18.0782518354  | 165.6136237488 | .3929844888 | 3.5272654578 |
| 6536.719313 | 9113.687185  | 18.2978538935  | 169.6136237488 | .3929844888 | 3.5272654578 |
| 6547.683469 | 9711.517858  | 18.50883537347 | 173.6136237488 | .3929844888 | 3.5272654578 |
| 6559.196754 | 10363.759287 | 18.6881367886  | 177.6136237488 | .3929844888 | 3.5272654578 |
| 6571.566111 | 11079.123248 | 18.8683627994  | 181.6136237488 | .3929844888 | 3.5272654578 |
| 6584.789988 | 11868.739931 | 19.0169586114  | 185.6136237488 | .3929844888 | 3.5272654578 |
| 6598.961134 | 12746.994767 | 19.1578822818  | 189.6136237488 | .3929844888 | 3.5272654578 |
| 6614.191898 | 13732.936869 | 19.2827844876  | 193.6136237488 | .3929844888 | 3.5272654578 |
| 6630.628886 | 14852.537972 | 19.3913813462  | 197.6136237488 | .3929844888 | 3.5272654578 |
| 6648.418941 | 16142.528699 | 19.4834158843  | 201.6136237488 | .3929844888 | 3.5272654578 |

seconds are shown since it was desired to limit the computational time. The staging event between stage I and stage II could then be observed.

A main concern was to show that the filter could not only observe the first stage parameters, since it would surely work on these, given the results of the last chapter, but that it could also observe the staging event, and continue to estimate the second stage parameters as additional data was processed. If the filter could observe the staging event of the first and second stages, it would also be able to observe the staging events throughout the rest of the flight.

Several test cases, such as those listed in Appendix J were run. It was desired to have the filter retain good memory about the position and velocity, since the changes in these quantities, even during staging events, would be small. However, the  $V_e$  and  $M$  would be changing rapidly only at the staging events. To account for this phenomena, the test cases were run with  $\beta$  values of .8 - 1.0 for the position and velocity vectors, and .5 - .7 for the  $V_e$  and  $M$ . The effect of this was to have the filter try to throttle the ICBM in order to make the estimation correct when it was close to a staging event. The compiled data is graphed in Figures 6 thru 14, and shown numerically in Appendix J. Notice that in the figures, the best results are shown in Figures 6 and 10 (exhaust velocity and mass ratio respectively), and the

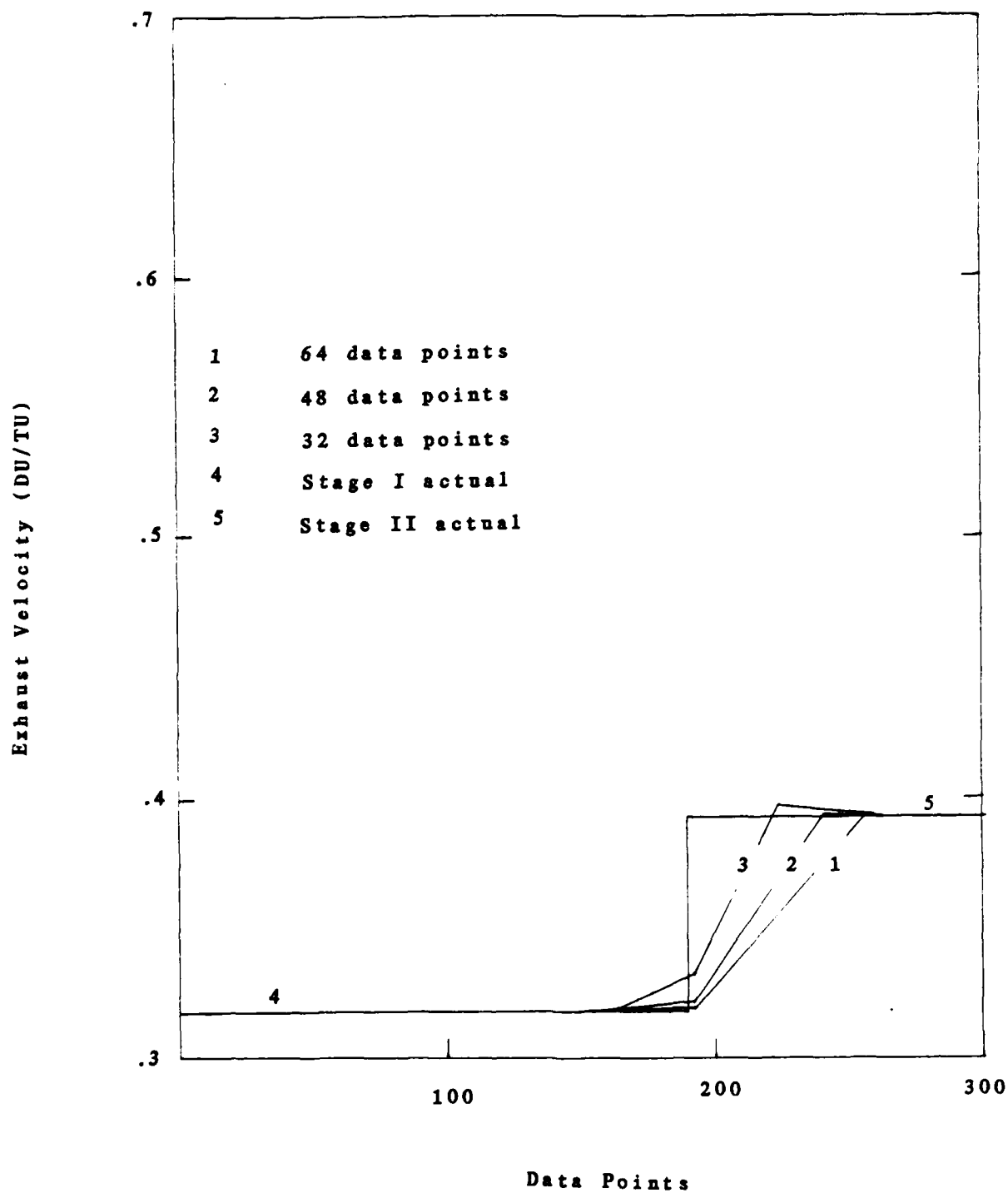


Figure 6. Test Cases: 64, 48, 32

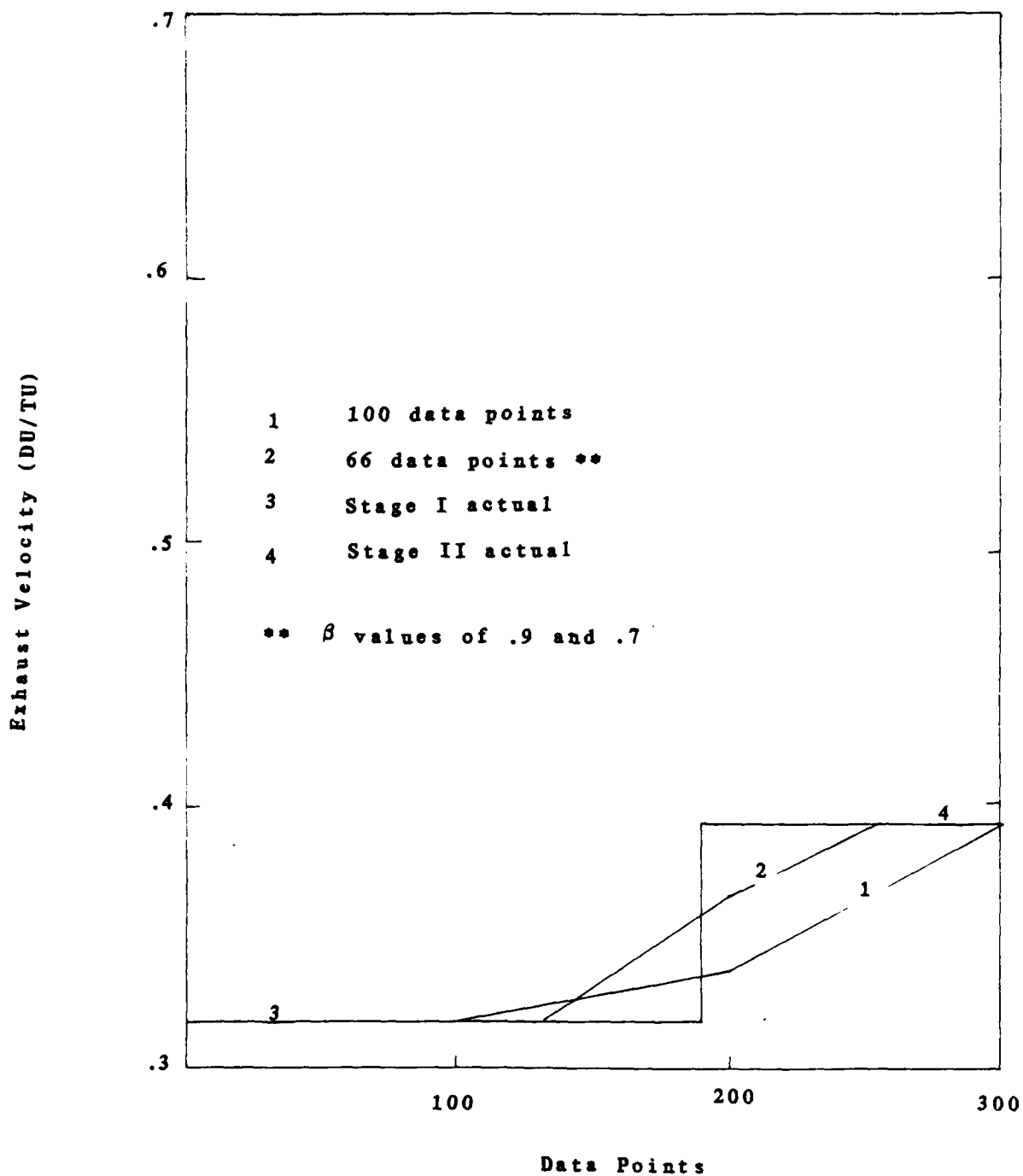


Figure 7. Test Cases: 100, 66

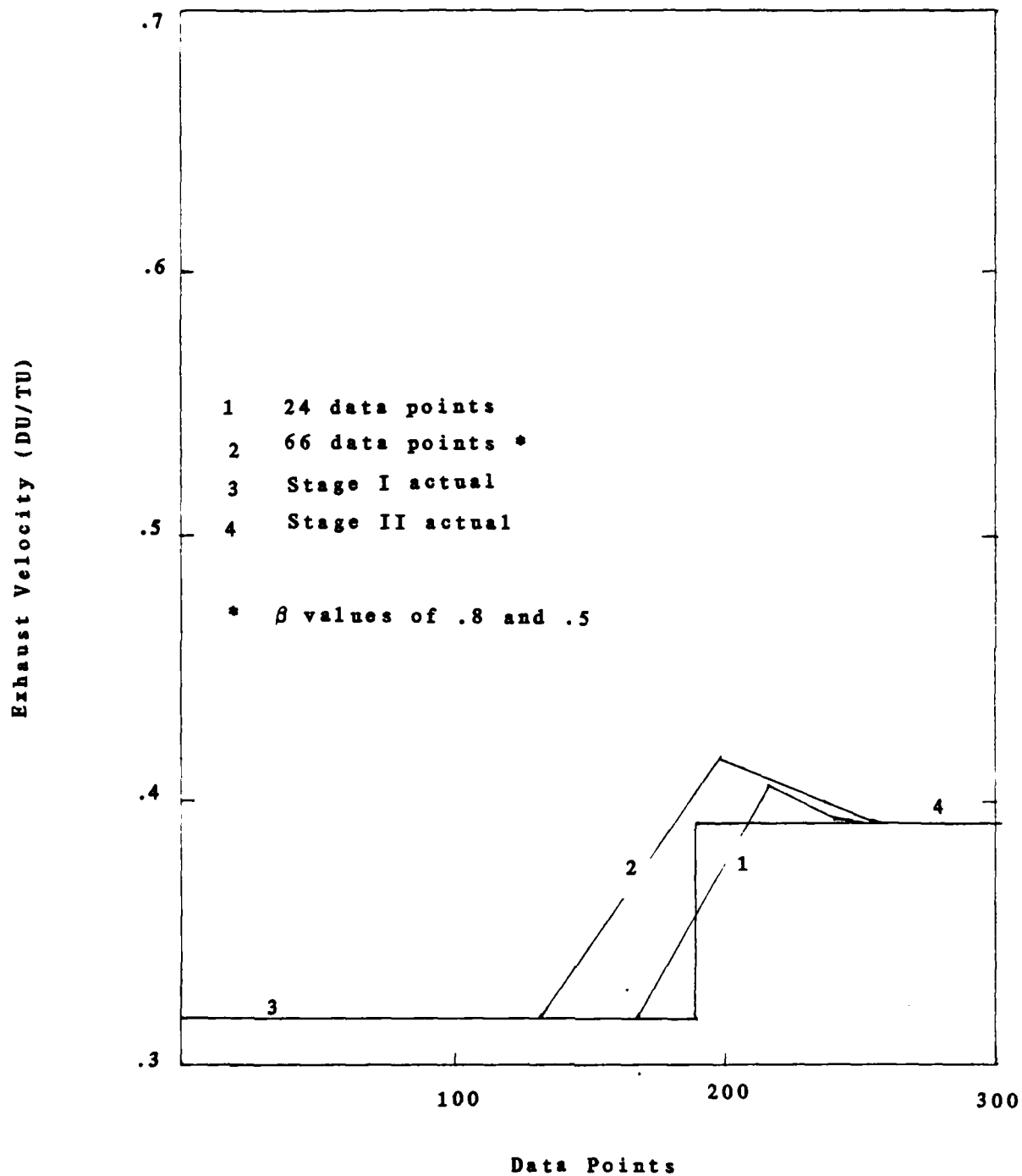


Figure 8. Test Cases: 24, 66

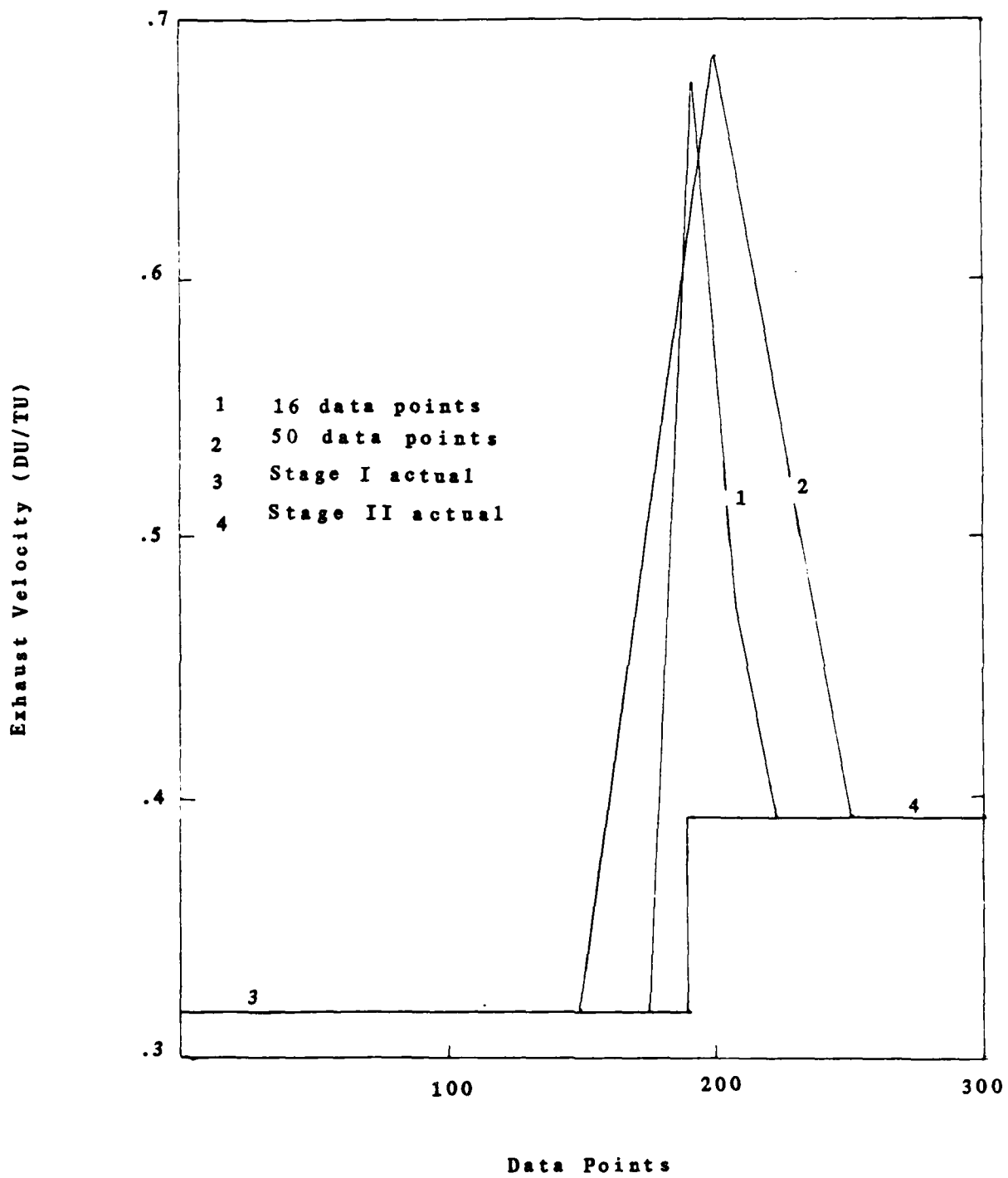


Figure 9. Test Cases: 16, 50

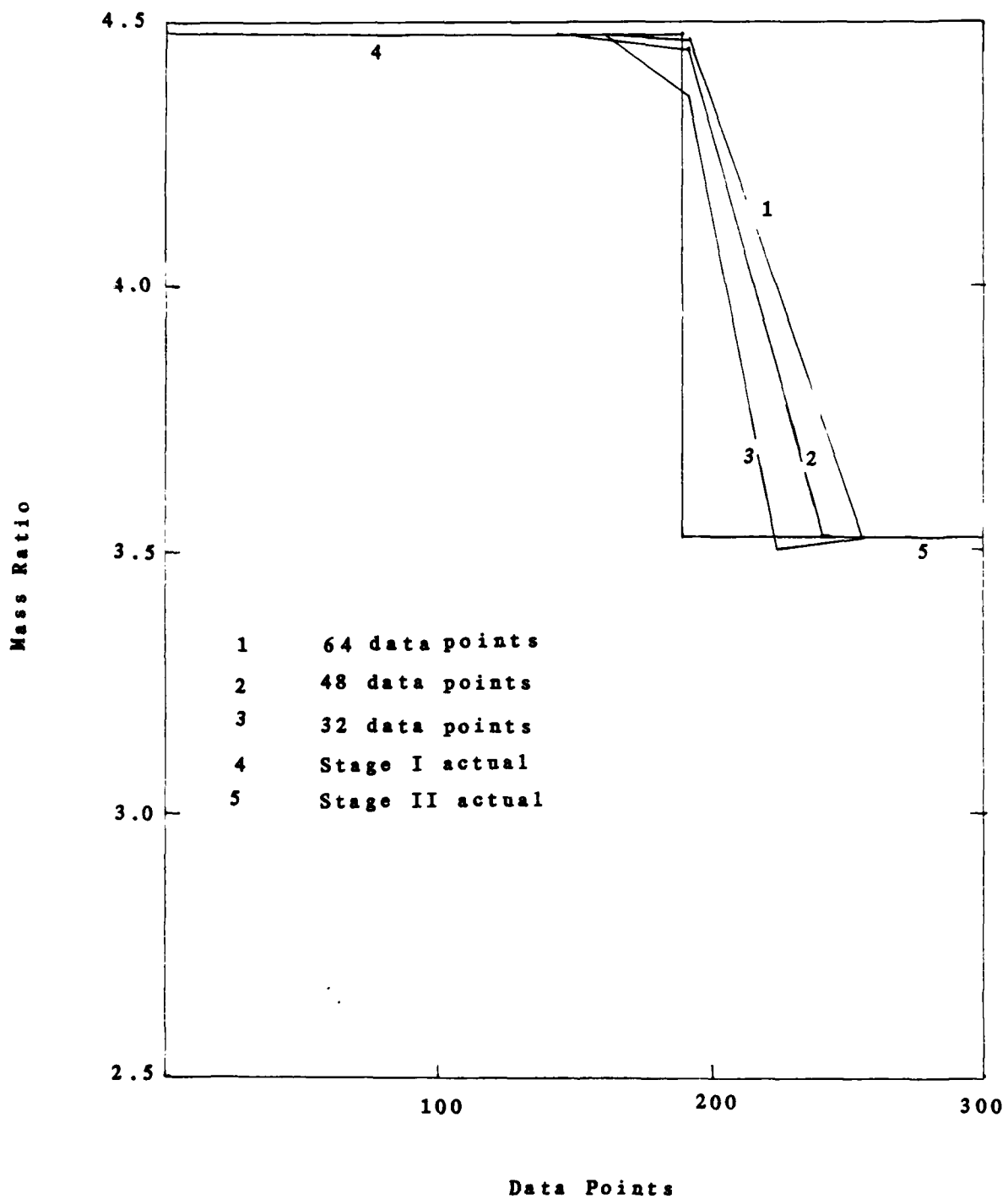


Figure 10. Test Cases: 64, 48, 32

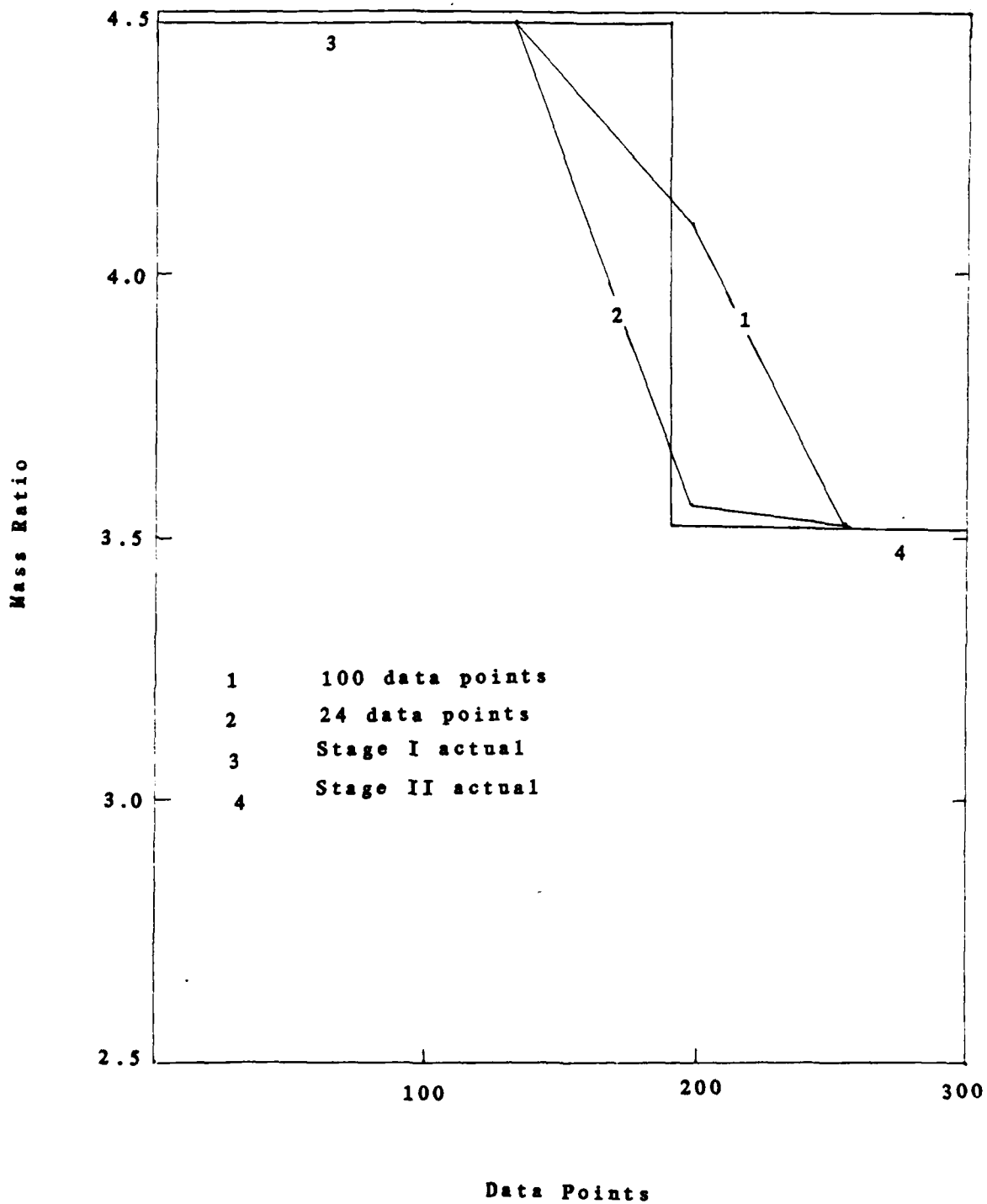


Figure 11. Test Cases: 100, 24

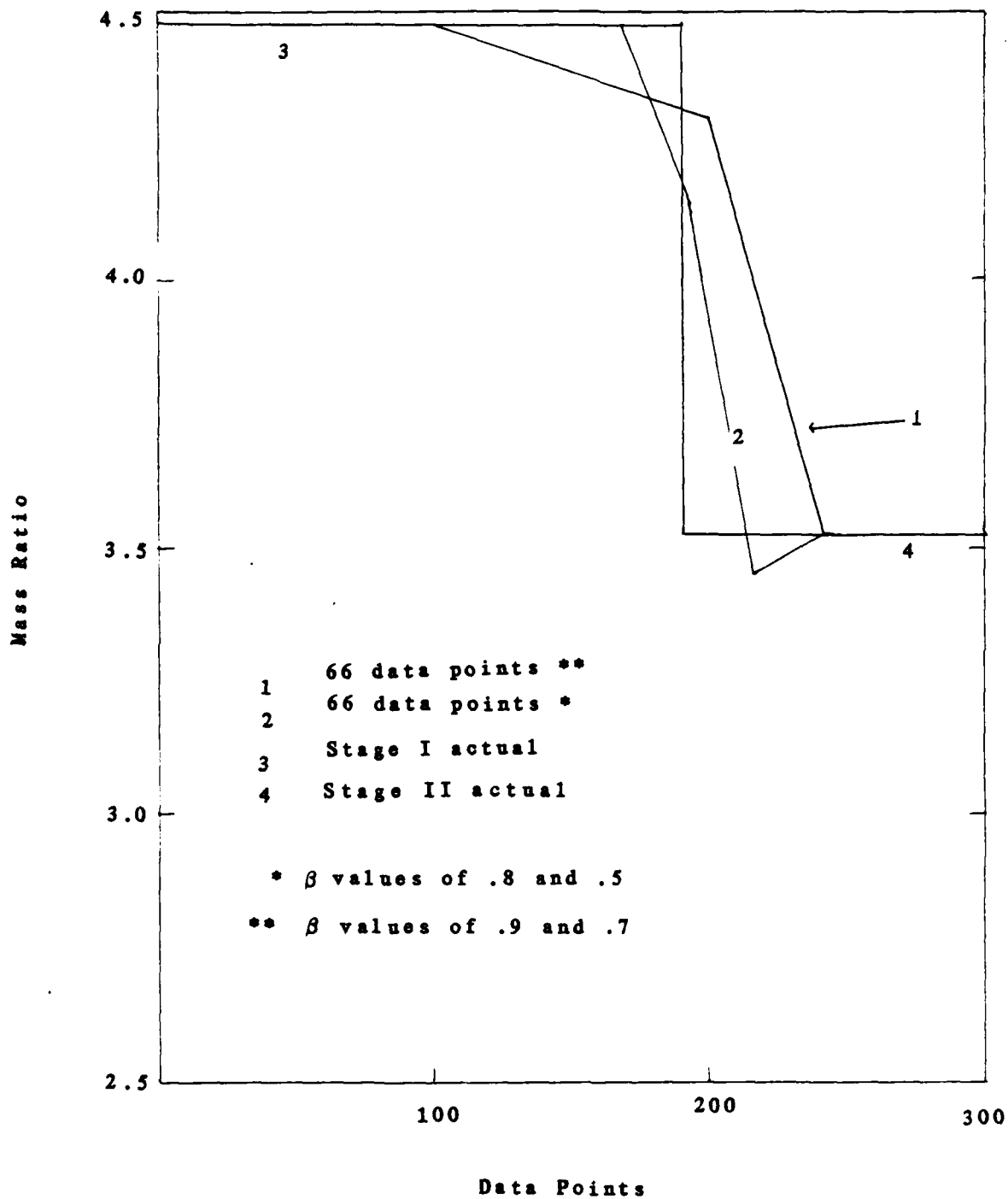


Figure 12. Test Cases: 66, 66

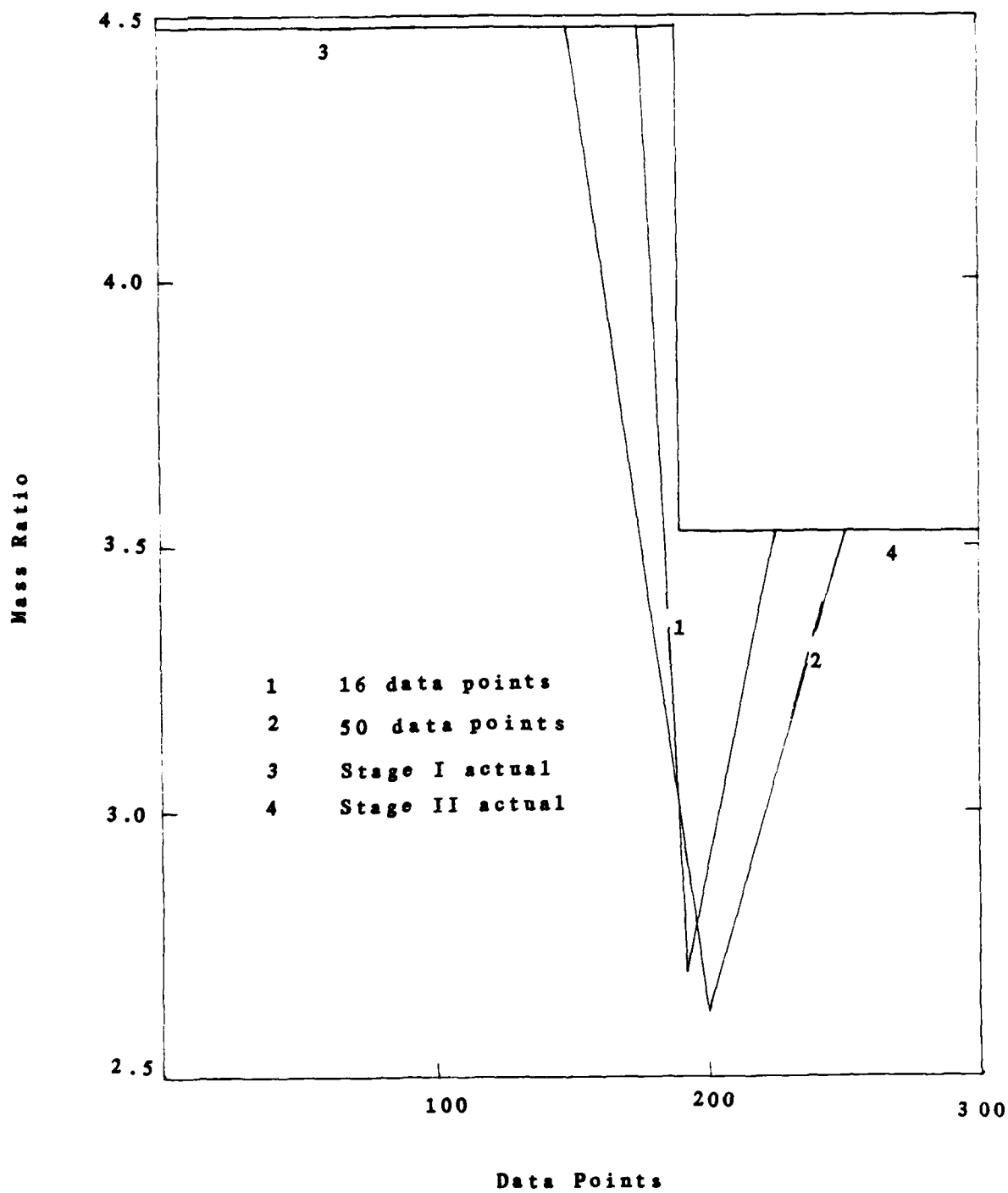


Figure 13. Test Cases: 16, 50

poorest performance are shown in Figures 9 and 14.

It was determined that the filter was only able to observe the staging event successfully when the event occurred within one of the Bayes loop segments of data. In addition, the staging event needed to occur in the latter portion of the data, and the more stage I data that was present, the better the performance of the filter. Table 17 lists the summary of the data segments which were run.

Table 17. Summary of Data Points

| # of points | Staging event data segment | # of Stg I points | # of Stg II points | Ratio Stg I/Stg II |
|-------------|----------------------------|-------------------|--------------------|--------------------|
| 16          | 176-192                    | 14                | 2                  | 7                  |
| 24          | 168-192                    | 22                | 2                  | 11                 |
| 32          | 160-192                    | 30                | 2                  | 15                 |
| 48          | 144-192                    | 46                | 2                  | 23                 |
| 50          | 150-200                    | 40                | 10                 | 4                  |
| 64          | 128-192                    | 62                | 2                  | 31                 |
| 66          | 132-198                    | 58                | 8                  | 7.25               |
| 100         | 100-200                    | 90                | 10                 | 9                  |

\* Note that the staging event occurs at point # 190

Notice that the smallest ratio that successfully converged was the 50 point case. Ratios of less than 4 did not converge. This was apparently due to the fact that the estimator could not change the performance parameters instantaneously, as the stage effectively does. Therefore, it seemed to need a good memory (the previous stage's data), along with a little new data, to enable a partial correction of the data. On the next data segment which was processed, the entire set of stage II data enabled the further

refinement of the estimate, and in most cases, was very close to the actual value.

A further correlation existed between the ratio and the performance of the estimator. Careful examination of the figures, and the information in Table 17 shows that the higher ratio numbers produced the most accurate results. The case which used 64 data points for each segment very closely approximated the actual data for both exhaust velocity, and mass ratio, and it had the highest Stg I/Stg II ratio. This further supports the contention that the filter needed to process the data slightly differently during the staging event in order to assure success.

Examination of the figures, and the data for the test cases run with 66 data points indicate a trend with the  $\beta$  values. The case which included  $\beta$  values of .9 and .7 seemed to perform a little better than the case with  $\beta$  equal .8 and .5. This tends to indicate that the filter needs to keep a reasonable amount of memory to correctly estimate the performance parameters. The value at .7 does this, allowing for just enough change to incorporate the variations caused by the staging event.

The figures and the data show that the filter estimated the performance parameters very well during most of the stages thrusting. As the staging event was approached however, each of the cases diverged, proceeded to initiate an estimate of the second stage's performance, and then

converged on the second stage's performance.

## Conclusions

A method for the estimation of launch vehicle performance parameters and the position and velocity vectors has been examined. The specific parameters were exhaust velocity and mass ratio. The data obtained from the test cases shows that the filter will estimate the performance parameters. The occurrence of the staging event within a segment of data did not appear to present a problem as long as the event occurred significantly towards the end of a particular data segment. This provided a means by which the filter could perform a staged change from one state to another.

Recommendations for additional work could be accomplished by using data already generated at this point. The added features could include a form of residual monitoring in the program which would watch for a quadratic departure in the in-track position residuals. This would suggest the probability that the time was close to a staging event. The program could then shift the data segment so that the staging event would always occur towards the latter portion of the data. This also suggests the possibility for iterating back over the staging event to try and refine the guess. Another change could include an alteration of the accuracy of the measurements. These could be altered to determine if there is a limit for the resolution of the radar

and infrared system that must be obtained to allow for convergence. Finally, additional values could be input for the  $\beta$  values to see if additional changes would alter the results obtained, especially with the smaller data segment size (10-30). The overall objective, however, of observing exhaust velocity and mass ratio was successful.

```

print*, 'y=' .y(1,nxt)*6378.145d+00,y(2,nxt)*6378.145d
+      +00,y(3,nxt)*6378.145d+00,y(4,nxt)*7.90536828d+00
print*, 'y=' .y(5,nxt)*7.90536828d+00,y(6,nxt)*7.90536828
+      d+00,y(7,nxt).y(8,nxt)

write(17,2)

cccccc begin loop to integrate
do 10 incc= 0,nit
  call haming(nxt)
  inb= inb + 1
  do 30 ind=1,3
    r(ind)= y(ind,nxt)
    v(ind)= y(ind+3,nxt)
  continue
  call razel(r,v,rho,az,e1,to,t,rs,term,ino)
  if (ioh.ne.10) then
    print*, 'do you want noisy data, y or n'
    read*, type
    if (type.eq. 'y') then
      print*, 'input a seed number from 1-21483647d+00'
      read*, dseed
    endif
    ioh= 10
  endif
  if (type.eq. 'y') then
    rho= rho + ggnqf(dseed)*sigrho
    az= az + ggnqf(dseed)*sigaz
    e1= e1 + ggnqf(dseed)*sigel
  endif
  write(14,*) rho,az,e1,t

  if (inb.eq.10) then
    do 34 ins=1,3
      rm(ins)= r(ins)*6378.145d+00
      vm(ins)= v(ins)*25936.2647d+00
    continue
    call mag(r)
    call mag(rm)
    call mag(vm)
    call mag(v)
    dot = r(1)*v(1)+r(2)*v(2)+r(3)*v(3)
    gamma = dacos(dot/(r(0)*v(0)))
    tm= t*806.8118744d+00
    write(17,6) rm(0),vm(0),gamma/rad,tm,y(7,nxt),y(8,nxt)
    inb= 0
  endif
  endif
  if (incc.eq.nit-1) then
cccccc
    print out final data
    print*, 'the final values for r and v are='
    print*, 'y=' .y(1,nxt)*6378.145d+00,y(2,nxt)*6378.145d+00
+      .y(3,nxt)*6378.145d+00,y(4,nxt)*7.90536828d+00
    print*, 'y=' .y(5,nxt)*7.90536828d+00,y(6,nxt)*7.90536828
+      d+00,y(7,nxt),y(8,nxt)
    write(17,12)
    write(17,4) (y(inf,nxt),inf=1,8),t
    write(17,*)
  endif
  continue
endfile(unit=17)
endfile(unit=14)

```

```

print*, 'input the launch point number'
read*, lp
if (lp.eq.0) then
    llat=53.7d+00*rad
    llon=158.2d+00*rad
endif
if (lp.eq.1) then
    llat=43.5d+00*rad
    llon=132.0d+00*rad
endif
if (lp.eq.2) then
    llat=1.0d+00*rad
    llon=1.0d+00*rad
endif
if (lp.eq.3) then
    llat=1.0d+00*rad
    llon=1.0d+00*rad
endif
lrs(0)=1.0d+00
call ltime(llst,t,to,llon)
ans= '1'
call radst(lrs,llat,llst,t,to,ans,ino)
ino=0
print*, 'input initial velocity in ft/s'
read*, ivel
ivel= ivel/25936.24764d+00
do 40 inr=1,3
    lvv(inr)=ivel*lrs(inr)
print*, 'how much do you want to nudge the velocity'
read*, nudge
lvv(3)= lvv(3)+lvv(3)*nudge
do 44 inw= 1,3
    y(inw,nxt)= lrs(inw)
    y(inw+3,nxt)= lvv(inw)
44 continue
endif

tp= tp/806.8118744d+00
print*, 'input the number of iterations'
read*, nit
dt=tp/nit
nxt=0

cccccc write initial header data

write(17,*) 'The initial state vector for the missile is'
write(17,12)

inb= 0
dseed= 38888.d+00
sigrho= .00001d+00
sigaz= .001d+00
sigel= .001d+00
type= 'n'

cccccc initialize haming and reset the time

nxt = 0
call haming(nxt)
t= tepoch
if (nxt.eq.0) stop
write(17,4) (y(inf,nxt),inf=1,8),t
print*,y(1,nxt),y(2,nxt),y(3,nxt),y(4,nxt),y(5,nxt),y(6,nxt),
+      y(7,nxt),y(8,nxt)

```



## PROGRAM TLAN.F

### Description

This program accomplishes the numerical integration check for the ICBM launch trajectory. The basic operation is very similar to the operation of tn.f, however, the program is set up to calculate the launch point for the ICBM, of which several choices are available. The program then calculates the initial velocity for the missile, assumed to be staright up from the local coordinate system, and then proceeds to nudge the vehicle over so the gravity turn can be executed. The numerical integration by Haming is identical, and instead of one period being integrated, the user can input how long, in seconds, the trajectory is to be integrated.

The variables and usage are almost identical to the program tn.f









```

cccccc calculate initial angular momentum and specific mech energy
      call mag(r)
      call mag(v)
      angm= dsqrt(a*(1-e*e))
      se= v(0)*v(0)/2.0d +00 - 1.0d +00/r(0)
      tp= tp*13.44686457d +00

cccccc write initial header data
      write(17,12)
      write(17,6)a,e,inc,omega,argp,nuo,mo,tp,itm
      write(17,5)
      write(17,7)se,angm
      write(17,2)

      ina= 0
      inb= 0
      dseed= 38888.d+00
      sigrho= .00001d+00
      sigaz= .001d+00
      sigel= .001d+00
      type= 'n'

cccccc initialize haming and reset the time
      call haming(nxt)
      t= 0.0d+00
      write(17,4) (y(1nf,nxt),1nf=1,6)

cccccc begin loop to integrate
      do 10 incc= 0,nit
        call haming(nxt)
        inb= inb + 1
        do 30 ind=1,3
          r(ind)= y(ind,nxt)
          v(ind)= y(ind+3,nxt)
          continue
        call razel(r,v,rho,az,el,to,t,rs,rm)

        if (loh.ne.10) then
          print*,'do you want noisy data,y or n'
          read*,type
          if (type.eq.'y') then
            print*,'input a seed number from 1-21483647d+00'
            read*,dseed
          endif
          loh= 10
        endif
        if (type.eq.'y') then
          rho= rho + ggnqf(dseed)*sigrho
          az= az + ggnqf(dseed)*sigaz
          el= el + ggnqf(dseed)*sigel
        endif
        write(14,*) rho,az,el,t
        if ((itm.ne.50).and.(inb.eq.10)) then
          ina= ina + 1
          if (ina.eq.10) then
            call mag(r)
            call mag(v)
            se= v(0)*v(0)/2.0d +00 - 1.0d +00/r(0)
            call cross(r,v,rcv)
            angm= rcv(0)
            write(17,7)se,angm
            ina= 0
            print*,'the rho az el time is',rho,az/rad,el/rad
          endif
          inb=0
          if (incc.lt.nit-10) then
            write(17,4) (y(1ne,nxt),1ne=1,6)
          endif
          if (incc.eq.nit-1) then

```

```

cccccc This program checks out the numerical integrator for
      haming. It does this by an integration of an orbit, once
      around the orbit.
cccccc Capt. Dave Vallado 1984
cccccc the common terms
      common /ham/ t,y(72,4),f(72,4),err(72),n,d,t,mode
      double precision t,y,f,err,d,t
      integer n,nxt,mde
cccccc all the other variables
      integer nit,ina,inb,inc,ind,ine,inf
      double precision a,e,inc,omega,argp,m,tm11,tm12,tm21,tm22,
+         tm31,tm32,rad,se,angm,tp,rho,az,el,to,dseed,sigrho,
+         r(8:3),v(8:3),rs(8:3),rcv(8:3),trm(3,3),sigel,sigaz
      character type
cccccc begin the program
2 format(9x,'x',14x,'y',14x,'z',12x,'xdot',12x,'ydot',12x,'zdot')
4 format(6(1x,f14.11))
5 format('the specific mech energy and ang momentum are')
6 format(7(1x,f6.3),1x,f8.3,1x,16)
7 format(2(8x,f14.11))
12 format(3x,'a',6x,'e',6x,'i',5x,'omega',3x,'argp',4x,'nuo',4x,'m'
+       ,4x,'period',3x,'# it')

      open(unit=17,file='product',access='sequential',status='new')
      open(unit=14,file='tdata',access='sequential',status='new')

      print*, 'input the data a,e,i,w.w,m for the orbit'
      read*,a,e,inc,omega,argp,m
      call randv(a,e,inc,omega,argp,nuo,m,r,v)

cccc    convert from PQW to IJK

      rad= 3.14159265359d +00/180.d +00
      nxt=1
      t= 0.d +00
      to= 0.d +00
      mode= 0
      n= 6

      tm11= dcos(omega)*dcos(argp)-dsin(omega)*dsin(argp)*dcos(inc)
      tm12= -dcos(omega)*dsin(argp)-dsin(omega)*dcos(argp)*dcos(inc)
      tm21= dsin(omega)*dcos(argp)+dcos(omega)*dsin(argp)*dcos(inc)
      tm22= -dsin(omega)*dsin(argp)+dcos(omega)*dcos(argp)*dcos(inc)
      tm31= dsin(argp)*dsin(inc)
      tm32= dcos(argp)*dsin(inc)

      y(1,nxt)= tm11*r(1)+tm12*r(2)
      y(2,nxt)= tm21*r(1)+tm22*r(2)
      y(3,nxt)= tm31*r(1)+tm32*r(2)
      y(4,nxt)= tm11*v(1)+tm12*v(2)
      y(5,nxt)= tm21*v(1)+tm22*v(2)
      y(6,nxt)= tm31*v(1)+tm32*v(2)

      print*, 'the initial value for r and v ='
      print*, 'r=',y(1,nxt),y(2,nxt),y(3,nxt)
      print*, 'v=',y(4,nxt),y(5,nxt),y(6,nxt)

cccccc start program

      tp= 2.d +00*3.1415926535d +00*sqrt(a*a*a)
      print*, 'Input the number of iterations'
      read*,nit
      dt=tp/nit
      nxt=0
```

```

C      This program checks out the numerical integrator for
C      haming. It does this by an integration of an orbit, once
C      around the orbit.

```

[illegible]

```

cccccc the common terms

      common /ham/ t,y(72,4),f(72,4),err(72),n,dt,mode
      double precision t,y,f,err,dt
      integer n,nxt,mode

```

```

***** all the other variables

integer nit,ina,lnb,lncc,lnnd,lnne,lnnf
double precision a,e,inc,omega,argp,m,tml1,tml2,tm21,tm22,
+      tm31,tm32,rad,se,angm,tp,rho,az,e1,to,dseed,sigrho,
+      r(8:3),v(8:3),rs(8:3),rcv(8:3),trm(3,3),sigel,sigaz
character type

```

```

cccccc begin the program
2      format(9x,'x',14x,'y',14x,'z',12x,'x_dot',12x,'y_dot',12x,'z_dot')
4      format(6(1x,f14.11))
5      format('the specific mech energy and ang momentum are')
6      format(7(1x,f6.3),1x,f8.3,1x,f6)
7      format(2(8x,f14.11))
12     format(3x,'a',6x,'e',6x,'f',5x,'omega',3x,'argp',4x,'nuo',4x,'m'
+      ,4x,'period',3x,'#',1t')

```

```
open(unit=17,file='product',access='sequential',status='new')
open(unit=14,file='tdata',access='sequential',status='new')

print*, 'input the data a,e,i,w,w,m for the orbit'
read*,a,e,inc,omega,argp,m
call randv(a,e,inc,omega,argp,nuo,m,r,v)
```

```

cccc      convert FROM PQW to IJK

      rad= 3.14159265359d +##/18#.##d +##
      nzt=1
      t= 0.##d +##
      to= 0.##d +##
      mode= 0
      n= 6

```

```

tm11= dcos(omega)*dcos(argp)-dsin(omega)*dsin(argp)*dcos(inc)
tm12= -dcos(omega)*dsin(argp)-dsin(omega)*dcos(argp)*dcos(inc)
tm21= dsin(omega)*dcos(argp)+dcos(omega)*dsin(argp)*dcos(inc)
tm22= -dsin(omega)*dsin(argp)+dcos(omega)*dcos(argp)*dcos(inc)
tm31= dsin(argp)*dsin(inc)
tm32= dcos(argp)*dsin(inc)

```

```

y(1,nxt)=tm11*r(1)+tm12*r(2)
y(2,nxt)=tm21*r(1)+tm22*r(2)
y(3,nxt)=tm31*r(1)+tm32*r(2)
y(4,nxt)=tm11*v(1)+tm12*v(2)
y(5,nxt)=tm21*v(1)+tm22*v(2)
y(6,nxt)=tm31*v(1)+tm32*v(2)

```

```
print*, 'the initial value for r and v =',  
print*, 'r=', y(1,nxt), y(2,nxt), y(3,nxt),  
print*, 'v=', y(4,nxt), y(5,nxt), y(6,nxt)
```

```

cccccc start program

tp= 2.0d+000*3.1415926535d+000*sqrt(a*a*a)
print*, 'input the number of iterations'
read*, nit
dt=tp/nit
nxt=0

```

#### **RANDV**

This subroutine calculates the position and velocity vectors given the initial orbit data. The algorithm uses the eccentric anomaly calculation, and the Newton Rhapsion iteration to find the mean anomaly. Transformations from the Perifocal coordinate system are used to convert the result into the IJK frame.

#### **MAG**

This subroutine simply calculates the magnitude of a vector.

#### **CROSS**

This subroutine calculates the cross product of 2 vectors.

#### **RAZEL**

This subroutine calculates the range, azimuth and elevation for the the truth model and leastsquares, and Bayes filter programs. It uses the next 2 subroutines that are described to accomplish this. Note that 2 seperate versions are used, one shown with the numerical integrator tn.f, and the other shown with obser. The difference here is the inclusion of a transformation to IJK, which is documented in the subroutine.

#### **RADST**

This subroutine calculates the position vector of the site. For a land based site, the user is asked to input latitude and longitude in degrees. The elevation is input in DU's. If the site is a satellote, the user is asked to input the orbit parameters for the calculation of the orbit. Notice that for any follow on effort, it would be advisabler to incorporate a seperate time to the satellite observer so that the satellite could be positioned over the launch point. The program, as written now will place the satellite at the same local sidereal time, no matter what orbit parameters are used.

#### **LSTIME**

This subroutine calculates the local sidereal time for the site. Notice that the value is input in degrees for 1984.

the haming common  
 rad           radians to degrees conversion  
 dseed       input # to IMSL routine for random numbers  
 sigrho      range  
 sigaz       azimuth       standard deviation  
 sigel       elevation

#### Remaining Variables

r(0:3)      Position vector  
 v(0:3)      Velocity vector  
 tml1, ...   Transformation matrix from PQW to IJK  
 se          Specific Mechanical Energy  
 angm       Angular Momentum  
 tp          Time period (TU's)  
 rho         Range  
 az          Azimuth  
 el          Elevation  
 rcv(3)      r cross v  
 trm(3,3)   Transformation from SEZ to IJK  
 rad         degree to radian conversion  
 t<sub>0</sub>       initial time  
 Counters and misc Holders  
 ina, inb, incc, ind, ine, inf, ioh

#### Subroutines used

randv  
 mag  
 cross  
 razel  
 radst  
 lstime  
 dhaming   (See Appendix B)  
 rhstru    (See Appendix C)

#### Notes

mode = 0 so EOM only  
 time step is critical for obtaining convergence on the orbit  
 starting the integration at perigee is difficult since the vehicle is moving the fastest.

## APPENDIX A

### PROGRAM TN.F

#### Description

This program checks out the numerical integrator that is programmed in Haming. It accomplishes this by integrating the two-body equation, once around it's orbit.

The subroutine randv, takes input orbit elements and converts them to position and velocity vectors in the PQW system (reference 1). A Newton Rhaphson iteration is employed to convert the mean anomaly to the eccentric anomaly. This is then input to find the true anomaly, from which the postion and velocity vectors are readily obtained.

The main program then rotates the postion and velocity vectors to the IJK frame and assigns these values to the state vector y. The program then calculates the period of the orbit and divides the iteration as 1/500th of the period. Haming is then called to iointegrate the orbit around, with the only stops being to caluculate the specific mechanical energy and the angular momentum from the position and velocity vectors at 10 step intervals. The main data is placed in the system file with only minimal input directed to the screen.

#### User Inputs

|          |                                          |
|----------|------------------------------------------|
| a        | Semi Major axis (DU)                     |
| e        | eccentricity                             |
| i        | inclination (deg)                        |
| Omega    | Longitude of ascending node (deg)        |
| Argp     | Argument of Perigee (deg)                |
| M        | Mean anomaly (deg)                       |
| itn      | number of iterations                     |
| 'l', 's' | Land or space based sensor               |
| 'y', 'n' | Type, whether or not you want noisy data |

#### If land based

|       |                             |
|-------|-----------------------------|
| Lat   | Lat of the site (deg)       |
| Lon   | Longitude of the site (deg) |
| rs(0) | Elevation of the site (DU)  |

#### If space based

|        |                                       |
|--------|---------------------------------------|
| sta    | tracker semi major axis (DU)          |
| ste    | tracker eccentricity                  |
| sti    | tracker inclination (deg)             |
| stonga | tracker long. of ascending node (deg) |
| stargp | tracker argument of perigee (deg)     |

#### Variables To be Set

## BIBLIOGRAPHY

1. Bate, Roger R., Mueller, Donald D., and White, Jerry E. Fundamentals of Astrodynamics. New York : Dover Publications, Inc., 1971.
2. Gross, Donald W., "Estimation of Launch Vehicle Performance Parameters from Two Orbiting Sensors", Masters Thesis. Wright Patterson AFB, Ohio : Air Force Institute of Technology, December 1982.
3. Miller, Capt. Gregory D., "Estimation of Launch Vehicle Performance Parameters from an Orbiting Sensor", Masters Thesis. Wright Patterson AFB, Ohio : Air Force Institute of Technology, December 1981.
4. Sutton, George P., and Ross, Donald M., " Rocket Propulsion Elements ", New York: Wiley Publishing, 1976.
5. U.S. Space Launch Systems (u), Navy Space Systems Activity. Report No. NSSA-R-20-72-2. P.O. Box 92960, Worldway Postal Center, Los Angeles, California, 90009.
6. Wiesel, William E., Jr. Lecture material distributed in MC731, Modern Methods of Orbit Determination. School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio, 1984.
7. Wiesel, William E., Jr. Lecture Material distributed in MC533, Problems in Spaceflight. School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio, 1984.

## APPENDIX B

### SUBROUTINE HAMING

#### Description

This program is a fourth order differential equations integrator. It carries four copies of the state vector along, and extrapolates them to find the next value. It then corrects this answer to find the new value of the state vector.

To use the predictor-corrector algorithm, an initial state vector must be stored in `y(*,1)`. `nxt` is then set to 0 and one call is made to `haming` to initialize the EOM EOY etc. The time is then reset to the epoch, and normal use can proceed, as long as `nxt` does not still = 0 upon exit from `haming`. This would mean that `haming` was unable to use the initial state vector as a guess. (maybe the step size is too big)

#### User Inputs

none

#### Variables to be Set

|                     |                                                                       |      |
|---------------------|-----------------------------------------------------------------------|------|
| <code>t</code>      | independent variable                                                  | time |
| <code>dt</code>     | time step                                                             |      |
| <code>y(*,1)</code> | 1 copy of the state vector to be input                                |      |
| <code>n</code>      | number of equations to be integrated                                  |      |
| <code>errest</code> | estimate of truncation error (generally not used)                     |      |
| <code>mode</code>   | 0 - EOM only<br>1 - EOM and EOY                                       |      |
| <code>nxt</code>    | sets the transitions between <code>Haming</code> and <code>Rhs</code> |      |

( Collectively I call these variables The `Haming Common` )

#### Remaining Variables

|                     |                                                                                                             |
|---------------------|-------------------------------------------------------------------------------------------------------------|
| <code>f(*,4)</code> | 4 copies of the equations of motion. <code>Rhs</code> updates these on each call from <code>haming</code> . |
| <code>tol</code>    | a tolerance parameter                                                                                       |
| <code>hh</code>     | step size holder                                                                                            |
| <code>xo</code>     | time holder                                                                                                 |

Counters and misc holders,  
`ida, idb, idc, idd, ide, idf, idg, idh, idi, idj, idk, idl, idn`





## APPENDIX C

### SUBROUTINE RHS

#### Description

This program calculates the equations of motion and equation of variation (the A matrix) for the problem which is evaluated. It serves merely as a data source for Haming and is called using nxt, thus no specific inputs are needed other than those in the common block with Haming.

#### User Inputs

type of vehicle parameters  
'0' Satellite in orbit  
'1' Titan IIIB  
'2' Titan IIID  
'3' Thor LV-2F

#### Variables to be set

The Haming common  
Remaining variables

xmu Gravitational Parameter ( $1 \text{ DU}/\text{TU}$ )  
r32, r52  $r^3, r^5$   
v32  $v^3$   
vel Vehicle velocity  
vat Combination of vel, acc, t ve and m  
vve (velocity)(Exhaust velocity)  
vem (exhaust velocity)(m)  
mass mdot/initial mass  
acc Vehicle acceleration  
am A matrix  
masso initial mass  
mdot mass flow rate  
V<sub>e</sub> Exhaust velocity  
Counters and misc holders  
ira, irb, irc, ird, ire, irf, irg, irh, iri, irj, irk, ii, jj

#### Subroutines used

vehd

#### Notes

3 different versions of this program were run.  
rhstru contained subroutine vehd and is listed here  
rhsam did not contain vehd  
instead of the call,  $V_e = y(7, \text{nxt})$ ,  $M = y(8, \text{nxt})$   
printed the a matrix before  $\text{phidot} = a * \text{phi}$   
rhsib the same as rhs am except the A matrix was not  
printed



```

cccccc      Set the constants which will be used in the A matrix

xmu= 1.0d+00
if (ire.eq.0) then
  print*, 'input the type of missile to be evaluated'
  print*, ' the coices are as follows'
  print*, 'Satellite in orbit      0'
  print*, 'Titan-III8              1'
  print*, 'Titan-IIID              2'
  print*, 'Thor-LV-2f short tank   3'
  read*, irf
  ire= 10
endif
if (irf.eq.0) then
  vve= 1.0d+00
  vam= 1.0d+00
  acc=0.0d+00
  mass=1.0d+00
  goto 6
endif

call vehd(ve,mdot,masso,t,irf)

mass= mdot/masso
vve= vel*ve
vam= ve*mass
acc= ve*mass/(-mass*t)

y(7,nxt)= ve
y(8,nxt)= mass

6  f(4,nxt) = - xmu * y(1,nxt) / r32 + acc*y(4,nxt)/vel
f(5,nxt) = - xmu * y(2,nxt) / r32 + acc*y(5,nxt)/vel
f(6,nxt) = - xmu * y(3,nxt) / r32 + acc*y(6,nxt)/vel
f(7,nxt) = 0.0d+00
f(8,nxt) = 0.0d+00

c  end of equations of motion
c  is this all ?

if( mode .eq. 0) return

c  it isnt all ... calculate the
c
c  *****
c
c  EQUATIONS OF VARIATION
c
c  *****
c
c  FIRST, calculate a matrix.... only lower 3x3 isnt hard wired
c
r52 = r32 ** ( 5.0d+00/3.0d+00 )

cccccc      diagonal terms in a matrix

an(4,1) = -xmu/r32 + 3.0d+00*xmu*y(1,nxt)*y(1,nxt)/r52
an(5,2) = -xmu/r32 + 3.0d+00*xmu*y(2,nxt)*y(2,nxt)/r52
an(6,3) = -xmu/r32 + 3.0d+00*xmu*y(3,nxt)*y(3,nxt)/r52

c  off diagonal terms in a matrix
c  use symmetry to avoid as much calculation
c  as possible...this point is deep within lots of loops!!!!

```

```

am(4,2) = 3.0d-08*xmu*y(1,nxt)*y(2,nxt)/r52
am(5,1) = am(4,2)
am(4,3) = 3.0d-08*xmu*y(1,nxt)*y(3,nxt)/r52
am(6,1) = am(4,3)
am(5,3) = 3.0d-08*xmu*y(2,nxt)*y(3,nxt)/r52
am(6,2) = am(5,3)

```

cccccc now same stuff for the other terms

```

am(4,4) = -y(4,nxt)*y(4,nxt)*acc/v32 + acc/vel
am(5,5) = -y(5,nxt)*y(5,nxt)*acc/v32 + acc/vel
am(6,6) = -y(6,nxt)*y(6,nxt)*acc/v32 + acc/vel

```

```

am(4,5) = -y(4,nxt)*y(5,nxt)*acc/v32
am(5,4) = am(4,5)

```

```

am(4,6) = -y(4,nxt)*y(6,nxt)*acc/v32
am(6,4) = am(4,6)

```

```

am(5,6) = -y(5,nxt)*y(6,nxt)*acc/v32
am(6,5) = am(5,6)

```

```

am(4,7) = y(4,nxt)*acc/vve
am(5,7) = y(5,nxt)*acc/vve
am(6,7) = y(6,nxt)*acc/vve

```

```

vat = acc*acc*t/vem + acc/mass

```

```

am(4,8) = y(4,nxt)*vat/vel
am(5,8) = y(5,nxt)*vat/vel
am(6,8) = y(6,nxt)*vat/vel

```

c the a matrix is now calculated

c NOW, calculate phi dot = a \* phi and put into last  
c 64 slots of f matrix

```

do 800 irg = 1,8
  do 800 irh = 1,8
    iri = 8*irh + irg
    f(iri,nxt) = 0.00d+00
    do 700 irj = 1,8
      irk = 8*irh + irj
      f(iri,nxt) = f(iri,nxt) + am(irg,irj)*y(irk,nxt)
    700 continue
  800 continue

```

cccccc phi dot = a \* phi is now done

```

return
end

```



## APPENDIX D

### PROGRAM TNA.F

#### Description

This program checks out the A matrix as described in chapter 3. The inputs are simply the state vector  $y(*,1)$ , which can either be left unchanged, or re-input by the user at run time. The program then calls rhs which calculates the A matrix directly and prints it out to a separate file. Then, using equation (4-1) the A matrix is again calculated using the different method. The results are printed out to the same output file, and the results can then be compared.

#### User inputs

'y' or 'n'            ans- whether or not the user wants to  
                      change the given state vector.  
                      If yes, input the new state vector y(8)

#### Variables to be set

the haming common

#### Remaining variables

xu(8,nxt)           unperturbed state vector  
fu(8,nxt)           unperturbed F matrix  
amat(8,8)           A matrix  
delta                delta of each iteration  
Counters and misc holders  
ias,iab,iac,iad,iae,iaf,iag,iah

#### Subroutines used

rhsam (See Appendix C)

#### Notes

watch using nxt in and out of dhaming, it is not always  
equal to 1.

# THE A MATRIX

$$\begin{bmatrix}
 0 & 0 & 0 & 1 \\
 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 \\
 \frac{-\mu}{r^3} + \frac{3\mu x^2}{r^5} & \frac{3\mu xy}{r^5} & \frac{3\mu xz}{r^5} & \frac{-v_x^2 a}{v^3} + \frac{a}{v} \\
 \frac{3\mu yx}{r^5} & \frac{-\mu}{r^3} + \frac{3\mu y^2}{r^5} & \frac{3\mu yz}{r^5} & \frac{-v_y v_x a}{v^3} \\
 \frac{3\mu xz}{r^5} & \frac{3\mu yz}{r^5} & \frac{-\mu}{r^3} + \frac{3\mu z^2}{r^5} & \frac{-v_x v_z a}{v^3} \\
 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 \\
 1 & 0 & 0 & 0 \\
 0 & 1 & 0 & 0 \\
 \frac{-v_x v_y a}{v^3} & \frac{-v_x v_z a}{v^3} & \frac{v_x a}{v V_e} & \frac{v_x}{v} \left( \frac{a^2 t}{V_{em}} + \frac{a}{M} \right) \\
 \frac{-v_y^2 a}{v^3} + \frac{a}{v} & \frac{-v_y v_z a}{v^3} & \frac{v_y a}{v V_e} & \frac{v_y}{v} \left( \frac{a^2 t}{V_{em}} + \frac{a}{M} \right) \\
 \frac{-v_y v_z a}{v^3} & \frac{-v_z^2 a}{v^3} + \frac{a}{v} & \frac{v_z a}{v V_e} & \frac{v_z}{v} \left( \frac{a^2 t}{V_{em}} + \frac{a}{M} \right) \\
 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0
 \end{bmatrix}$$

where

$$A_{ij} = \partial F_i / \partial \bar{x}_j$$



```

cccccc perturb each element

write(16,*)'now perturb each element of the state vector'
mode=0
do 10 iae= 1,8
  delta= -y(iae,nxt)*.0001d +00
  if (dabs(delta).lt.1.0d-14) goto 9
  y(iae,nxt) = delta + y(iae,nxt)
  call rhs(nxt)
  do 9 iaf= 1,8
    amat(iaf,iae)= (f(iaf,nxt)-fu(iaf,nxt))/delta
  continue
  y(iae,nxt)= xu(iae,nxt)
0 continue

write(16,4) ((amat(iag,iah),iah=1,8),iag=1,8)
endfile(unit=16)
end

yee haw!!!!!!!!!!!!!!
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
include '/en/en84d/dvallado/rhsam'

```

## APPENDIX E

**PROGRAM TNPH.F**

### Description

This program accomplishes much the same function as tna.f. It really only mechanizes the discussion in chapter 3. The input is a state vector  $y(8)$  which can be altered by the user, however, one should note that the time step is based on an orbit from the family where  $a=2.5DU$ . The program calls haming which numerically integrates the state through about  $1/5$  of the orbit. The program then prints out the  $\phi$  matrix to a file called phprod. Then, one by one, the states are perturbed, translated through time, and reinitialized until equation 4-2 has been used to successively calculate the columns of the  $\phi$  matrix. The results of the second calculation are then output to the same file for comparison.

## User inputs

```

'y' or 'n'          ans- whether or not you want to change
                        the given state vector
If yes, input the new state vector

```

### Variables to be set

the having common  
to time period

### Remaining Variables

```

xu(8)                unperturbed state vector
cphi(8,8)            calculated o matrix
xut(8)               unperturbed state vector, moved through time
delta                delta on each iteration
Counters and misc holders
ipa,ipb,ipc,ipd,ipe,ipf,ipg,iph,ipi,ipj,ipk,ipl,ipm,
ipn,ipo,ipp,ipq,ipr,ips,ipt,ipu,ipv

```

### Subroutines used

**dhaming** (See Appendix B)  
**rhs1b** (See Appendix C)

## Notes

```

be sure to set n=72, t=0, dt=
Remember that a= 2.5 DU's !!
must reset o and state on each iteration
watch values of ngt in and out of the subroutine calls

```

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

This program checks out the Phi matrix for the problem of estimation of launch vehicle performance parameters. It does this by having rhs calculate the Phi matrix. Then, each element of the input state vector is perturbed, and moved through time. The columns of the Phi matrix are calculated by subtracting the original state, from the perturbed state, and dividing by delta.

Capt. Dave Vallado 1984

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

```
common /ham/ t,y(72,4),f(72,4),errest(72),n,dt,mode
double precision t,y,f,errest,dt
integer n,mode,nxt

double precision xu(8,1),cphi(8,8),xut(8,1),tp,delta,tstart
integer ipa,ipb,ipc,ipd,ipe,ipf,ipg,iph,ipi,ipj,ipk,ipl,ipm,
+ ipn,ipo,ipp,ipq,ipr,ips,ipt,ipu,ipv
character ans
```

```
format(3(1x,f12.6))
open(unit=15,file='phprod',access='sequential',status='new')
```

```
n=72
mode=1
nxt=1
tstart= .496729413d +000
```

```
cccc initialize the state vector
```

```
y(1,nxt)= 2.48402827630d +000
y(2,nxt)= 0.19950378763d +000
y(3,nxt)= 0.19950378763d +000
y(4,nxt)= -.07137664494d +000
y(5,nxt)= .44435648672d +000
y(6,nxt)= .44435648672d +000
y(7,nxt)= 0.0d +000
y(8,nxt)= 0.0d +000
tp= 2.0d +000*3.1415962535d +000*dsqrt(15.625d+000)
```

```
print*,'the current values of y -12345678 are',(y(ipcc,nxt),
+ ipcc= 1,8)
print*,'do you want to change the values?'
read*,ans
if (ans.eq.'n') goto 5
```

```
print*,'input new state vector,time and tp for tp/500'
read*,(y(ipa,nxt),ipa=1,8),tstart,tp
```

```
cccc set unperturbed state vector
```

```
do 6 ipb=1,8
xu(ipb,nxt)= y(ipb,nxt)
continue
```

```
dt=tp/500.0d +000
t= tstart
```

```
cccc initialize phi matrix
```

```
do 7 ipc= 9,72
y(ipc,nxt)= 0.0d +000
continue
do 8 ipd= 9,72,9
y(ipd,nxt)= 1.0d +000
continue
```

```
write(15,*) 'this data is from the state vector y ='
write(15,*) (y(ipw,nxt),ipw=1,8),t
write(15,*) 'the initial phi matrix is'
write(15,4) ((y(ipc,nxt),ipe=ipf,72,8),ipf=9,16)
```

AD-A151 927

ESTIMATION OF LAUNCH VEHICLE PERFORMANCE PARAMETERS(U)  
AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL  
OF ENGINEERING D A VALLADO DEC 84 AFIT/GA/AA/84D-10

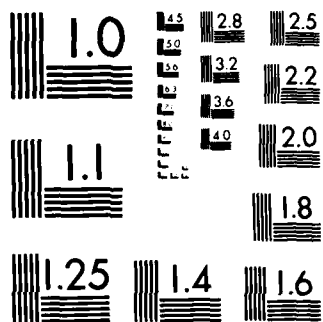
2/2

UNCLASSIFIED

F/G 12/1

NL

|  |  |  |  |  |  |  |  |  |  |       |  |  |  |
|--|--|--|--|--|--|--|--|--|--|-------|--|--|--|
|  |  |  |  |  |  |  |  |  |  |       |  |  |  |
|  |  |  |  |  |  |  |  |  |  |       |  |  |  |
|  |  |  |  |  |  |  |  |  |  |       |  |  |  |
|  |  |  |  |  |  |  |  |  |  | END   |  |  |  |
|  |  |  |  |  |  |  |  |  |  | FILED |  |  |  |
|  |  |  |  |  |  |  |  |  |  | DTIC  |  |  |  |



MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS 1963-A



## APPENDIX F

**PROGRAM TNH.F**

### Description

This program checks out the H matrix for the specific problem. It does this by use of equation 4-3 in chapter 3. The program starts by inputting an initial state vector which the user can change at run time. Then, similar to the A matrix check done in tna.f, the program has observer calculate the H matrix directly, and then calculates the columns individually by perturbing each element in the state vector, and dividing out the resulting differences in the calculated G matrices.

## User Inputs

'y' or 'n'      ans- whether or not the given state vector  
                                 should be changed

If yes, input the new state vector

### Variables to be set

the haming common

### Remaining Variables

```

xu(8)                unperturbed state vector
hm(3,8)              H matrix
zpredu(3)            unperturbed G matrix
zpred                G matrix
delta                delta on each iteration
Counter and misc holders
iha, ihb, ihc, ihd, ihe, ihf, ihg, ihh, ihi, ihj

```

### Subroutines used

```

obser | (See Appendix G)
razel |
randv |
mag |
cross } (See Appendix A)
radst }
lstime }

```

# G MATRIX

$$G = \begin{matrix} \text{rho} & \sqrt{x^2 + y^2 + z^2} \\ \text{az} & \tan^{-1}(y/x) \\ \text{el} & \tan^{-1}(z/\sqrt{x^2 + y^2}) \end{matrix}$$

# H MATRIX

$$\delta \begin{pmatrix} \text{rho} \\ \text{az} \\ \text{el} \end{pmatrix} = \underbrace{H} \delta \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

$$\delta \begin{pmatrix} \text{rho} \\ \text{az} \\ \text{el} \end{pmatrix} = \underbrace{\frac{\partial \begin{pmatrix} \text{rho} \\ \text{az} \\ \text{el} \end{pmatrix}}{\partial \begin{pmatrix} S \\ E \\ Z \end{pmatrix}}}_A \underbrace{\frac{\partial \begin{pmatrix} S \\ E \\ Z \end{pmatrix}}{\partial \begin{pmatrix} I \\ J \\ K \end{pmatrix}}}_B \delta \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

$$A = \begin{bmatrix} (\frac{x}{x^2+y^2+z^2})^{1/2} & (\frac{y}{x^2+y^2+z^2})^{1/2} & (\frac{z}{x^2+y^2+z^2})^{1/2} & \dots 0 \dots \\ \frac{-y/x^2}{1+(y/x)^2} & \frac{1/x}{1+(y/x)^2} & 0 & \dots 0 \dots \\ \frac{-xz/(x^2+y^2)^{3/2}}{1+z^2/(x^2+y^2)} & \frac{-yz/(x^2+y^2)^{3/2}}{1+z^2/(x^2+y^2)} & \frac{1/(x^2+y^2)^{1/2}}{1+z^2/(x^2+y^2)} & \dots 0 \dots \end{bmatrix}$$

$$B = \begin{bmatrix} \hat{S} & | & \hat{E} & | & \hat{Z} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\hat{E} \cdot x \cdot \hat{Z}}{|\hat{E} \cdot x \cdot \hat{Z}|} & \frac{\hat{k} \cdot x \cdot \hat{Z}}{|\hat{k} \cdot x \cdot \hat{Z}|} & \frac{\bar{r} \cdot s}{|\bar{r} \cdot s|} \end{bmatrix}$$



```

cccccc perturb each element
mode=0
do 10 ihe= 1,8
    delta= -y(ihe,nxt)*.0001d +00
    y(ihe,nxt) = delta + y(ihe,nxt)
    call obser(tob,q1,zpred,h,nxt)
    if (abs(delta).lt.1.0d -14) goto 10
    do 9 ihf= 1,3
        hm(ihf,ihe)= (zpred(ihf)-zpredu(ihf))/delta
    9      continue
    y(ihe,nxt)= xu(ihe,nxt)
10      continue

cccccc Print out the result

write(13,*)
write(13,*) 'the perturbed calculation for H is as follows'
write(13,4) ((hm(ihg,ihh),ihh=1,8),ihg=1,3)

write(13,*)

endfile(unit=13)
end

c      yee haw!!!!!!!!!!!!!!!!!!!!
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
include '/en/en84d/dvallado/obser'

```

## APPENDIX G

### SUBROUTINE OBSEB

#### Description

This program calculates the observation relationships. The main function is to calculate the G matrix, the predicted data, and the H matrix which is simply the partial of G with respect to the state vector. I have called the G matrix in the derivations by G and z, and the obser program uses zpred and z. They are the same thing.

#### User inputs

none

#### Variables to be set

##### The Haming Common

|          |                                |
|----------|--------------------------------|
| tob      | time of each observation       |
| trm(3,3) | transformation from SEZ to IJK |
| to       | initial time                   |

#### Remaining Variables

|                                                               |                                 |
|---------------------------------------------------------------|---------------------------------|
| q1                                                            | Q inverse                       |
| zpred(3)                                                      | G matrix                        |
| h(3,8)                                                        | H matrix                        |
| r(3)                                                          | position vector                 |
| v(3)                                                          | velocity vector                 |
| rho                                                           | range                           |
| az                                                            | azimuth                         |
| el                                                            | elevation                       |
| rs(3)                                                         | site vector                     |
| sigrho                                                        | range                           |
| sigaz                                                         | azimuth      standard deviation |
| sigel                                                         | elevation                       |
| Counters and misc holders                                     |                                 |
| ioa, ioc, iod, iof, iou, iov, iow, ohm1, azdnom, eldnom, ioe, |                                 |
| elbtm, hit(3,3)                                               |                                 |

#### Subroutines used

|        |                    |
|--------|--------------------|
| razel  | } (See Appendix A) |
| randv  |                    |
| mag    |                    |
| cross  |                    |
| radst  |                    |
| lstime |                    |

#### Notes

do not transform r to SEZ as in tn.f





```

do 100 inl=1,3
  rhove(inl)= r(inl) - rs(inl)
100 continue
call mag(rhove)
rho= rhove(0)
do 110 inj= 1,3
  zvec(inj)= rs(inj)/rs(0)
110 continue
call cross(kvec,zvec,evec)
do 112 inn=1,3
  evec(inn)= evec(inn)/evec(0)
112 continue
call cross(evec,zvec,svec)
do 114 inn= 1,3
  svec(inn)= svec(inn)/svec(0)
114 continue
cccccc Set up the transformation for IJK = trm SEZ

do 120 inl=1,3
  trm(inl,1)= svec(inl)
  trm(inl,2)= evec(inl)
  trm(inl,3)= zvec(inl)
120 continue
do 121 inn=1,3
  re(inn)= r(inn)
  rse(inn)= rs(inn)
121 continue
cccccc Convert to SEZ for calculations

do 130 ink= 1,3
  rhovec(ink)= rhove(1)*trm(1,ink) + rhove(2)*trm(2,ink)
+ rhove(3)*trm(3,ink)
  r(ink)= re(1)*trm(1,ink) + re(2)*trm(2,ink)
+ re(3)*trm(3,ink)
  rs(ink)= rse(1)*trm(1,ink) + rse(2)*trm(2,ink)
+ rse(3)*trm(3,ink)
130 continue

if (rhovec(1).eq.0.0d+00) then
  if (rhovec(2).gt.0.0d+00) az= 90.0d+00*rad
  if (rhovec(2).lt.0.0d+00) az= 270.0d+00*rad
  if (rhovec(2).eq.0.0d+00) then
    az= 0.0d+00
    if (rhovec(3).gt.0.0d+00) e1= 90.0d+00*rad
    if (rhovec(3).lt.0.0d+00) e1= -90.0d+00*rad
  endif
endif
if ((rhovec(1).ne.0.0d+00).and.(rhovec(2).ne.0.0d+00)) then
  az= datan(rhovec(2)/rhovec(1))
  e1= datan(rhovec(3)/dsqrt(rhovec(1)*rhovec(1) + rhovec(2)*
+ rhovec(2)))
  if (rhovec(1).lt.0.0d+00) az= az + 180.0d+00*rad
  if ((rhovec(1).gt.0.0d+00).and.(rhovec(2).lt.0.0d+00)) az=
+ az + 360.0d+00*rad
endif
return
end

```

## APPENDIX H

### PROGRAM LSTSQ.F

#### Description

This program checks out the initial runs for a least squares estimation of the input problem. The cases used for trial runs consisted of the satellite orbits which were numerically integrated around one orbit. The estimator works by mechanizing the summary shown on the following page. Basically, the data is input, along with the truth model data that is run from program tn.f separately. After an initial state vector is input, the initial guess, the least squares program takes the guess along with the observation data and tries to estimate the true initial state.

#### User inputs

|         |                                                                                             |
|---------|---------------------------------------------------------------------------------------------|
| tePOCH  | initial starting time                                                                       |
| Xref(8) | initial guess for the state vector                                                          |
| maxit   | number of iterations that least squares will run through while trying to estimate the state |
| nob     | number of observations to be read.                                                          |
| trop    | rank of p matrix (used in inversions)                                                       |

#### Variables to be set

##### The Haming Common

#### Remaining variables

|            |                                                            |
|------------|------------------------------------------------------------|
| timeob( )  | time, rho, az and el for the storage                       |
| rho( )     | of the truth model data                                    |
| az( )      |                                                            |
| el( )      |                                                            |
| phi(8,8)   | $\phi$                                                     |
| h(3,8)     | H matrix                                                   |
| tmat(3,8)  | T matrix                                                   |
| tob        | holds time of each observation                             |
| z(3)       | holds G matrix values, each observation                    |
| zpred(3)   | holds G matrix values, calculated from xref in the program |
| dx(8,1)    | state vector corrections                                   |
| q1(3,3)    | $Q^{-1}$                                                   |
| resid(3)   | residual vector                                            |
| work       | storage for IMSL inverse routine                           |
| htq1(8,3)  | $T^T Q^{-1} T$                                             |
| pinv(8,8)  | $P^{-1}$                                                   |
| htqlr(8,1) | $T^T Q^{-1} \bar{r}$                                       |
| xref(8)    | state vector which gets updated                            |

|            |                       |
|------------|-----------------------|
| p(8,8)     | P matrix (covariance) |
| tepoeh     | Initial start time    |
| tmatt(8,3) | T transpose           |

#### Subroutines used

|         |                  |
|---------|------------------|
| mmpy    |                  |
| mtrans  |                  |
| mprint  |                  |
| eigen   |                  |
| dhaming | (See Appendix B) |
| rslb    | (See Appendix C) |
| obser   |                  |
| razel   | (See Appendix G) |
| randv   |                  |
| mag     |                  |
| cross   | (See Appendix A) |
| radst   |                  |
| lstime  |                  |

#### Notes

The value of trop was important in calculating the eigenvalues and eigenvectors. The satellite had a rank of 6, whereas the ICBM had a rank of 8. Difficulties were encountered with using trop=8, so it was left at 6

#### MMPY

This subroutine multiplies 2 matrices together, and outputs the result. Note that the 2 matrices must be declared identically in and out of the routine, and they must both be 2 dimensional, i.e. (0:3,0:4)

#### MTRANS

This subroutine calculates the transpose of a matrix.

#### MPRINT

This subroutine prints a matrix.

#### EIGEN

This subroutine calculates the eigenvalues and eigenvectors for the filter estimation problems. Note that copies are made to pass to the IMSL routines since these routines destroy the original matrix.

## Non-Linear Least Squares Algorithm

1. pick  $\bar{x}_{ref}(t_0)$ , initial guess for state vector

- Set Q and read in data for all observations (G)
- initialize
  - $\phi = I$
  - $P^{-1} = 0$
  - $T^T Q^{-1} \bar{r} = 0$

2. for each observation time

- move  $\bar{x}_{ref}(t_0)$  to  $\bar{x}_{ref}(t_i)$ 
  - Haming and rhs do this for  $x_{ref}$ , also 0
- calculate predicted data using  $\bar{x}_{ref}(t_i)$ , zpred  
obser does this
- calculate residual  $\bar{r}_i = \bar{z}_i - \bar{z}_{pred_i}$
- calculate H, obser does this
- calculate  $T = H \phi$
- sum  $\sum T^T Q^{-1} T$ 
  - since  $P^{-1}$  gets reset each iteration, it is summed up inside the observation loop
- sum  $\sum T^T Q^{-1} \bar{r}$

- loop back until all the data is processed

3. Calculations

- $P = [T^T Q^{-1} T]^{-1}$
- $\delta \bar{x} = P T^T Q^{-1} \bar{r}$
- update the  $\bar{x}_{ref}(t_0) = \bar{x}_{ref}(t_0) + \delta \bar{x}(t_0)$
- check convergence (See page 23,  $\delta \bar{x}(i) < \sqrt{P_{ii}}$ )
- if good, end with  $\bar{x}_{ref}(t_0)$
- if not, begin at start with  $\bar{x}_{ref}(t_0)$  and reset

$\phi = I : T^T Q^{-1} \bar{r} = 0 : P^{-1} = 0 : t = t_0$   
 $y(*,1) = \bar{x}_{ref}(t_0)$



```

do 420 ibe=1,8
  do 420 ibf= 1,8
    pinvn(ibc,ibf)= pinvo(ibc,ibf) + htqlt(ibc,ibf)
420    continue
  call meql(pinvn,8,8,pinvnc)
cccccc have we just finished printing last pass residuals ?
  if( idone .eq. 1) go to 5000
cccccc data is processed....improve estimate
cccccc invert matrix H transpose Q inverse H to find
cccccc covariance P
  call llnvlf(pinvnc,trop,8,p,0,work,ier)
  call meql(p,8,8,pc)
cccccc from matrix ***** dx = P * T transpose Q inverse r
  do 500 ibc=1,8
    xmxref(ibc,1)= xrefu(ibc,1)- xref(ibc,1)
500    continue
    call mmpy(pinvo,8,8,xmxref,1,pndx)
    do 640 ibd=1,8
      pndxph(ibd,1)= pndx(ibd,1)+htqlr(ibd,1)
640    continue
    call mmpy(p,8,8,pndxph,1,dx)
cccccc add in state corrections..
  do 700 ilv = 1,8
    xref(ilv,1) = xref(ilv,1) + dx(ilv,1)
700
cccccc print iteration, and current guess
720 format(/.2x,"iteration ",i3,/,2x,"state corrections"
+      ,/,2x,4e20.13,/,2x,4e20.13 )
  print 720,ilc,dx
740 format(/.2x,"current reference trajectory state vector at
+      at epoch:",/,2x,4e20.13,/,2x,4e20.13,/)
  print 740,xref
cccccc SUCCESS ??????????
cccccc check convergence
  ifail = 0
  do 800 ilu = 1,8
    if( dabs(dx(ilu,1)).gt.0.1*dsqrt(dabs(p(ilu,ilu))))
+      ifail = 1
800    continue
  if((ifail .eq. 0) .and. idone = 1)
9999 continue
c----- LOOP BACK FOR NEXT ITERATION OF LEAST SQUARES -----c
cccccc FAILURE for the least squares !!!!!!!!!!!!!
900 format(2x,"maximum iteration limit exceeded
+      without convergence.")
  print 900
  stop
cccccc SUCCESS for the least squares !!!!!!!!!!!!!
5000 continue
940 format(/.2x,"CONVERGENCE ACHIEVED.",/
+      .2x,"In nomina Gaussian trajectorye referentia",/,
+      2x,"declarium est estimat(a)",/)
  print 940
  ilnobs = ilnobs + 1lnob

```

```

----- OBSERVATION PROCESSING LOOP -----c
      do 1000 i1i = 1lnobs,1lnob + 1lnobs - 1
cccccc      extract each observation
              tob = timeob(i1i)
              z(1)= rho(i1i)
              z(2)= az(i1i)
              z(3)= e1(i1i)
cccccc      NUMERICALLY INTEGRATE STATE AND PHI TO OBS TIME
c            the number of steps here is equal to 1 since we
cccccc      have dt set exactly the same as the truth data we read
              nstp= 1
              do 100 i1k = 1,nstp
100          call haming(nxt)
cccccc      OBTAIN MATRICES FOR THIS OBSERVATION
              call obser(tob,q1,zpred,h,nxt)
cccccc      MATRIX STUFF - THIS OBSERVATION
              do 120 i1i = 1,ndata
120          resid(i1i) = z(i1i) - zpred(i1i)
              if( i1i .lt. 1lnobs+5) go to 200
              if(( idone .eq. 1).and.(i1i.lt.1lnobs+5)) go to 200
              if(( idone .eq. 1).and.(i1i.ge.1lnobs+5)) go to 240
              go to 250
200          continue
              print*, 'time. res =', tob, (resid(i1i), i1i=1,ndata)
cccccc      if this is last pass. weve already converged.
cccccc      so skip matrix calculations
240          if( idone .eq. 1 ) go to 9000
250          continue
cccccc      extract phi matrix in normal form
              do 260 i1n = 1,8
              do 270 i1o = 1,8
270          phi(i1n,i1o) = y(8*i1o+i1n,nxt)
260          continue
cccccc      form matrix ***** tmat = h * phi
              call mmpy(h,3,8,phi,8,tmat)
cccccc      form matrix ***** htq1 = T transpose * Q inverse
              call mtrans(tmat,3,8,tmatt)
              call mmpy(tmatt,8,3,q1,3,htq1)
cccccc      form matrix ***** htq1t = T transpose Q inverse T
cccccc      (sum through the observations)
              do 290 i1p = 1,8
              do 290 i1q = 1,8
              do 280 i1r = 1,ndata
280          htq1t(i1p,i1q)= htq1t(i1p,i1q)+htq1(i1p,i1r)
              +          *tmat(i1r,i1q)
290          continue
cccccc      form matrix ***** htq1r = T transpose Q inverse r
cccccc      (sum through the observations)
              do 150 i1s = 1,8
              do 150 i1t = 1,ndata
150          htq1r(i1s,1)= htq1r(i1s,1)+htq1(i1s,i1t)*
              +          resid(i1t)
9000          continue
1000         continue
----- LOOP BACK FOR OBSERVATION LOOP OF LEAST SQUARES -----c

```

```

cccccc READ IN OBSERVATIONS

open(unit=14,file='tdata',access='sequential',status='old')
rewind(unit=14)
print*, 'input the total number of observations to be read'
read*, nob
do 30 ilb = 1, nob
    read (14,*,end=30) rho(ilb), az(ilb), el(ilb), timeob(ilb)
30    continue
endfile(unit=14)
ndata= 3
print 10, xrefu, tepoch, nob, maxit, ilnob, ibloop, trop.
+    beta(1,1), beta(2,2), beta(3,3), beta(4,4), beta(5,5),
+    beta(6,6), beta(7,7), beta(8,8)

cccccc set last iteration flag

idone = 0
ilnobs = 1
call meq1(xrefu, 8, 1, xref)
do 40 ibj=1, 8
    do 40 ibi=1, 8
40        pinvo(ibj, ibi) = 0.0d+00

c----- BEGIN BAYES FILTER LARGE LOOP -----c

do 100000 ibg= 1, ibloop

c----- BEGIN ITERATION LOOP - NONLINEAR LEAST SQUARES -----c

dt = timeob(2) - timeob(1)
do 9999 ilc = 1, maxit

cccccc REINITIALIZE NUMERICAL INTEGRATION PARAMETERS

t = tepoch
mode = 1
n = 72

cccccc fcs are new reference traj guess

do 50 ild = 1, 8
    y(ild,1) = xref(ild,1)
50    continue

cccccc phi initial conditions

do 60 ile = 9, 72
    y(ile,1) = 0.0d+00
60    do 70 ilf = 9, 72, 9
    y(ilf,1) = 1.0d+00
70    continue

cccccc initialize haming and reset the time

nxt = 0
call haming(nxt)
t = tepoch

cccccc INITIALIZE BUFFERS FOR MATRIX PRODUCT ACCUMULATION

do 80 ilg = 1, 8
    htqlr(ilg,1) = 0.0d+00
do 80 ilh = 1, 8
    htqlt(ilg, ilh) = 0.00d+00
80    continue

cccccc print first or last pass residual headers when necessary

90    format(2x, "First Pass Residuals: ", /)
95    format(2x, "Last Pass Residuals: ", /)
if(ilc .eq. 1) print 90
if(idone .eq. 1) print 95

```



# Bayes Filter Algorithm

1. Pick  $\bar{x}_{refu}(t_{epoch})$ , initial guess for state vector

Set  $Q$  and  $\bar{x}_{ref} = \bar{x}_{refu}$   
read all data and store in  $z$

set  $P^{-1}(-) = 0$                        $t_{epoch} = 0$

2. For each Bayes iteration

3. For each least squares iteration

Set  $\phi = [I]$  :  $T^T Q^{-1} T = 0$  :  $T^T Q^{-1} \bar{r} = 0$

4. For each observation time

move  $\bar{x}_{ref}(t_{epoch})$  to  $\bar{x}_{ref}(t_i)$

hanning and rhs do this to  $\bar{x}_{ref}$  also  $\phi$

calculate predicted data using  $\bar{x}_{ref}(t_i) - \bar{z}_{pred}$   
obser does this

calculate residuals  $\bar{r}_i = \bar{z}_i - \bar{z}_{pred_i}$

calculate  $H$  - obser does this

$T = H \phi$   
sum  $\sum T^T Q^{-1} T$

sum  $\sum T^T Q^{-1} \bar{r}$

5. loop back until each data segment is processed

$P^{-1}(+) = P^{-1}(-) + T^T Q^{-1} T$

$\delta \bar{x} = P(+)(P^{-1}(-)(\bar{x}(-) - \bar{x}_{ref}) + T^T Q^{-1} \bar{r})$

$\bar{x}_{ref}(t_0) = \bar{x}_{ref}(t_0) - \bar{x}$

determine convergence (See page 23)

no

yes

loop back for another L.S.  
iteration

$P^{-1}(-) = P^{-1}(+)$

$\bar{x}_{ref}(t_i) = y(*, \text{nxt})$

$\bar{x}_{refu}(t_i) = \bar{x}_{ref}(t_i)$

$P^{-1}(-) = \phi^{-1} T P^{-1} \phi^{-1}$

$t_{epoch} = t$

$P^{-1}(-) = \beta P^{-1}(-) \beta^T$

|                                                                  |                                          |
|------------------------------------------------------------------|------------------------------------------|
| pinvn(8,8)                                                       | $P^{-1}(+)$                              |
| pinvnc(8,8)                                                      | copy of pinvn                            |
| pinvol(8,8)                                                      | update of pinvo after each least squares |
|                                                                  | run                                      |
| htqlr(8,1)                                                       | $T^T Q^{-1} \bar{r}$                     |
| xref(8,1)                                                        | state vector which gets updated          |
| xrefu(8,1)                                                       | initial state for each bayes loop        |
| p(8,8)                                                           | P matrix (covariance)                    |
| pc(8,8)                                                          | copy of p matrix                         |
| tePOCH                                                           | Initial start time                       |
| tmatt(8,3)                                                       | $T$                                      |
| mxmref(8,1)                                                      | $\bar{x}refu - \bar{x}ref$               |
| Counters and misc holders                                        |                                          |
| ilb, ilj, ibi, ibg, ibe, ibf, ibc, ibr, ibv, ibw, ilc, ild, ile, |                                          |
| ilf, ilg, ilh, ili, ilk, ilm, iln, ilo, ilp, ilq, ilr, ils, ilt, |                                          |
| ilv, ilu, pndx(8,1), pndxph(8,1)                                 |                                          |

#### Subroutines used

|         |                    |
|---------|--------------------|
| meql    |                    |
| mmpy    |                    |
| mtrans  | { (See Appendix H) |
| mprint  |                    |
| eigen   |                    |
| dhaming | (See Appendix B)   |
| rhlb    | (See Appendix C)   |
| obser   |                    |
| razel   | { (See Appendix G) |
| randv   |                    |
| mag     |                    |
| cross   | { (See Appendix A) |
| radst   |                    |
| lstime  |                    |

#### MEQL

This subroutine makes a copy of a matrix. This was used in the Bayes filter program.

## APPENDIX I

### PROGRAM BAYES.F

#### Description

This program mechanizes the Bayes filter discussion in chapter 3 and incorporates the summary that is shown on the following page. It reads the truth model data, proceeds to input user data, and then it calculates the state vector as each segment of data is processed. The program is very similar in operation to the 1stsq.f program, with the major differences discussed in chapter 3.

#### User Inputs

|          |                                                          |
|----------|----------------------------------------------------------|
| tePOCH   | initial start time                                       |
| xrefu(8) | initial state vector guess                               |
| maxit    | max number of least squares iterations                   |
| trop     | rank of P matrix                                         |
| ibloop   | max number of bayes loop iterations                      |
| ilnob    | number of data points read, each least squares iteration |
| beta     | Weigthing matrix for covariance matrix                   |

#### Variables to be set

The haming common

#### Remainging variables

|            |                                                                     |
|------------|---------------------------------------------------------------------|
| timeob( )  | time, rho, az and el for the storage                                |
| rho( )     | of the truth model data                                             |
| az( )      |                                                                     |
| el( )      |                                                                     |
| phi(8,8)   | $\phi$                                                              |
| phiin(8,8) | $\phi^{-1}$                                                         |
| phit(8,8)  | $\phi^T$                                                            |
| phic(8,8)  | copy of $\phi$ matrix                                               |
| h(3,8)     | H matrix                                                            |
| tmat(3,8)  | T matrix                                                            |
| tob        | holds time of each observation                                      |
| z(3)       | holds G matrix values, each observation                             |
| zpred(3)   | holds G matrix values, calculated from $\bar{x}$ ref in the program |
| dx(8,1)    | state vector corrections                                            |
| q1(3,3)    | $Q^{-1}$                                                            |
| resid(3)   | residual vector                                                     |
| work       | storage for IMSL inverse routine                                    |
| htq1(8,3)  | $T^T Q^{-1} T$                                                      |
| pinvo(8.8) | $P^{-1}(-)$                                                         |

```

do 2020 ilae= 1,3
  stgp(ilae,1)= real(wp(ilae))
2020  continue
  print*, 'note that these values are all in random order'
  print*, 'eigenvalues equal for the covariance matrix'
  call matprt(stig,8,1)
  print*, 'eigenvectors equal for the covariance matrix'
  call matprt(stigv,8,8)

cccccc now calculate error ellipsoid axis lengths, do conversions

do 2040 ilad=1,8
  stiger(ilad,1)= dsqrt(stig(ilad,1))
2040  continue

do 2060 ilaf=1,3
  stiper(ilaf,1)= (dsqrt(stigp(ilaf,1)))*6378.145d+00
  stiger(ilaf,1)= stiger(ilaf,1)*6378.145d+00
2060  continue

do 2080 ilag=4,6
  stiger(ilag,1)= stiger(ilag,1)*7.90563828d+00
2080  continue
  print*, 'the axis lengths for the covariance matrix are'
  call matprt(stiger,8,1)
  print*, 'the axis lengths for the position components are'
  call matprt(stiper,3,1)
  return
end

include '/en/en84d/dvallado/obser'
include '/en/en84d/dvallado/dhaming'
include '/en/en84d/dvallado/rhs1b'

```





```

      if( i11 .lt. 10) go to 200
      if( (idone .eq. 1).and.(i11.lt.10)) go to 200
      if( (idone .eq. 1).and.(i11.ge.10)) go to 240
      go to 250
      continue
      print*, 'time. res = ', tob, (resid(i1m), i1m=1, ndata)

cccccc      if this is last pass, weve already converged,
cccccc      so skip matrix calculations

240      if( idone .eq. 1 ) go to 9000
250      continue

cccccc      extract phi matrix in normal form
      do 105 i1n = 1,8
        do 106 i1o = 1,8
          phi(i1n,i1o) = y(8*i1o+i1n,nxt)
106        continue
105

cccccc      form matrix ***** tmat = h * phi
      call mmpy(h,3,8,phi,8,tmat)

cccccc      form matrix ***** htq1 = T transpose * Q inverse
      call mtrans(tmat,3,8,tmat)
      call mmpy(tmat,8,3,q1,3,htq1)

cccccc      form matrix ***** pinv = T transpose Q inverse T
cccccc      (sum through the observations)
      do 140 i1p = 1,8
        do 140 i1q = 1,8
          do 130 i1r = 1, ndata
            pinv(i1p,i1q) = pinv(i1p,i1q) + htq1(i1p,i1r)
            *tmat(i1r,i1q)
130          continue
140

cccccc      form matrix ***** htq1r = T transpose Q inverse r
cccccc      (sum through the observations)
      do 150 i1s = 1,8
        do 150 i1t = 1, ndata
          htq1r(i1s,1) = htq1r(i1s,1) + htq1(i1s,i1t) *
150          resid(i1t)
          +

c----- LOOP BACK FOR OBSERVATION LOOP -----c
9000      continue
1000      continue

cccccc      have we just finished printing last pass residuals ?
      if( idone .eq. 1) go to 5000

cccccc      data is processed....improve estimate
cccccc      invert matrix H transpose Q inverse H to find
cccccc      covariance P
      call l1nvlf(pinv,trop,8,p,8,work,ier)

cccccc      from matrix ***** dx = P * T transpose Q inverse r
      call mmpy(p,8,8,htq1r,1,dx)

cccccc      add in state corrections..
      do 800 i1v = 1,8
        xref(i1v) = xref(i1v) + dx(i1v,1)
800

cccccc      print iteration. and current guess
720      format(/,2x,"iteration ",i3,/,2x,"state corrections"
      +      ,/,2x,4e20.13,/,2x,4e20.13 )
      print 720,i1c,dx

```

```

c----- BEGIN ITERATION LOOP - NONLINEAR LEAST SQUARES -----c
      dt = timeob(2) - timeob(1)
      do 9999 ilc = 1,maxit

cccccc  REINITIALIZE NUMERICAL INTEGRATION PARAMETERS

          t = tepoch
          mode = 1
          n = 72
cccccc  lcs are new reference traj guess

          do 50 ild = 1,8
            y(ild,1) = xref(ild)
50      continue

cccccc  phi initial conditions

          do 51 ile = 1,72
            y(ile,1) = 0.0d+00
51      do 52 ilf = 1,72,9
            y(ilf,1) = 1.0d+00
52

cccccc  initialize haming and reset the time

          nxt = 0
          call haming(nxt)
          t = tepoch

cccccc  INITIALIZE BUFFERS FOR MATRIX PRODUCT ACCUMULATION

          do 60 ilg = 1,8
            htqlr(ilg,1) = 0.0d+00
          do 60 ilh = 1,8
            pinv(ilg,ilh) = 0.00d+00
60

cccccc  print first or last pass residual headers when necessary

63      format(2x,"First Pass Residuals: ",/)
64      format(2x,"Last Pass Residuals: ",/)
          if(ilc .eq. 1) print 63
          if(idone .eq. 1) print 64

c----- OBSERVATION PROCESSING LOOP -----c

      do 1000 ill = 1,nob

cccccc  extract each observation

          tob = timeob(ill)
          z(1) = rho(ill)
          z(2) = az(ill)
          z(3) = al(ill)

cccccc  NUMERICALLY INTEGRATE STATE AND PHI TO OBS TIME
c      the number of steps here is equal to 1 since we
cccccc  have dt set exactly the same as the truth data we read

          nstp = 1
          do 80 ilk = 1,nstp
            call haming(nxt)
80

cccccc  OBTAIN MATRICES FOR THIS OBSERVATION

          call obser(tob,q1,zpred,h,nxt)

cccccc  MATRIX STUFF - THIS OBSERVATION

          do 100 ill = 1,ndata
            resid(ill) = z(ill) - zpred(ill)
100

cccccc  selectively print out the iteration data

```



# APPENDIX J

## ICBM Test Results 100 Data Point Case

initial state vector :  

|                      |                     |                    |                     |
|----------------------|---------------------|--------------------|---------------------|
| x                    | y                   | z                  | xdot                |
| - .1326113791290e+00 | -.5769695352120e+00 | .8059282248600e-00 | -.1025944089500e-02 |
| xdot                 | zdot                | Ve                 | M                   |
| -.4449136538300e-02  | .6276833691300e-02  | .3172982551730e-00 | .4479637558632e+01  |

initial time : .002000 sec # of data points : 400  
max LS iterations : 8 # of each bayes chunk : 100  
max bayes iterations : 3 rank of P : 8  
Beta matrix = 1.000 1.000 1.000 1.000 1.000 1.000 .500 .500  
Titan-IIIB launched from 43 N, 132 E 1 DU elevation  
Radar site at 52.6 N, 174.1 E, 1 DU elevation

### First Pass Residuals:

|             |                |                 |                 |                |
|-------------|----------------|-----------------|-----------------|----------------|
| time, res = | .249577852e-02 | .209186890e-08  | -.204688340e-06 | .275503849e-06 |
| time, res = | .299155705e-02 | .611410787e-09  | -.207843734e-06 | .275112060e-06 |
| time, res = | .348733557e-02 | -.120013763e-08 | -.209707052e-06 | .274805515e-06 |
| time, res = | .398311409e-02 | -.342912500e-08 | -.209561260e-06 | .274554077e-06 |
| time, res = | .447889262e-02 | -.616290148e-08 | -.216663032e-06 | .274406923e-06 |

### iteration 1

state corrections  

|                    |                     |                    |                     |
|--------------------|---------------------|--------------------|---------------------|
| .7524464312042e-08 | -.1468811488669e-08 | .5692291303397e-07 | .3335708465224e-05  |
| .2730453261049e-07 | -.3186857906561e-07 | .2176948652099e-04 | -.2843510444043e-03 |

### current reference trajectory state vector at etime 1:

|                      |                     |                    |                     |
|----------------------|---------------------|--------------------|---------------------|
| - .1326113716045e+00 | -.5769695366808e+00 | .8059282317829e+00 | -.1022608381035e-02 |
| -.4449109233767e-02  | .6276801822721e-02  | .3173200246595e-00 | .4479352707588e+01  |

|             |                |                |                 |                 |
|-------------|----------------|----------------|-----------------|-----------------|
| time, res = | .249577852e-02 | .751060410e-08 | .110439458e-08  | -.571607390e-09 |
| time, res = | .299155705e-02 | .788293635e-08 | .392187394e-10  | -.461485322e-09 |
| time, res = | .348733557e-02 | .809102872e-08 | .457333726e-09  | -.358729600e-09 |
| time, res = | .398311409e-02 | .805920647e-08 | .308801629e-08  | -.298545372e-09 |
| time, res = | .447889262e-02 | .771118528e-08 | -.131271349e-08 | -.237021385e-09 |

### iteration 2

state corrections  

|                     |                    |                     |                    |
|---------------------|--------------------|---------------------|--------------------|
| -.5665596635395e-08 | .1099084931790e-08 | .2988739891713e-09  | .1067280853176e-07 |
| -.2792618244811e-07 | .3405576184837e-07 | -.2288203826774e-04 | .2996219419247e-03 |

### current reference trajectory state vector at time 1:

|                      |                     |                    |                     |
|----------------------|---------------------|--------------------|---------------------|
| - .1326113072701e+00 | -.5769695355817e+00 | .8059282320818e+00 | -.1022597708226e-02 |
| -.4449137150950e-02  | .6276835878483e-02  | .3172971426213e-00 | .4479652329530e+01  |

|             |                |                |                 |                 |
|-------------|----------------|----------------|-----------------|-----------------|
| time, res = | .249577852e-02 | .173993853e-08 | .111107259e-08  | .521426131e-10  |
| time, res = | .299155705e-02 | .212014891e-08 | .473456829e-10  | .479431581e-10  |
| time, res = | .348733557e-02 | .233710567e-08 | .466052752e-09  | .450908946e-10  |
| time, res = | .398311409e-02 | .231518487e-08 | .309726567e-08  | .804860865e-11  |
| time, res = | .447889262e-02 | .197815492e-08 | -.130300015e-08 | -.196081641e-10 |

### iteration 3

state corrections  

|                     |                     |                     |                     |
|---------------------|---------------------|---------------------|---------------------|
| .1918458332356e-12  | -.3375004459828e-13 | -.1479374160914e-13 | -.5513572843075e-12 |
| -.2165436884948e-12 | .4637224828259e-12  | .1079067081321e-08  | .6122699995441e-08  |

### current reference trajectory state vector at time 1:

|                      |                     |                     |                     |
|----------------------|---------------------|---------------------|---------------------|
| - .1326113072699e+00 | -.5769695355818e+00 | .80592820280818e+00 | -.1022597708778e-02 |
| -.4449137160166e-02  | .6276835878947e-02  | .3172971437003e-00  | .4479652335653e+01  |

### Last Pass Residuals:

|             |                |                |                 |                 |
|-------------|----------------|----------------|-----------------|-----------------|
| time, res = | .249577852e-02 | .174013353e-08 | .111107204e-08  | .521455015e-10  |
| time, res = | .299155705e-02 | .212034339e-08 | .473452338e-10  | .479293376e-10  |
| time, res = | .348733557e-02 | .233729948e-08 | .466052419e-09  | .450502413e-10  |
| time, res = | .398311409e-02 | .231537783e-08 | .309726567e-08  | .797097111e-11  |
| time, res = | .447889262e-02 | .197834686e-08 | -.130299982e-08 | -.197327623e-10 |

### CONVERGENCE ACHIEVED.

In nomina Gaussian trajectorum referentia  
declarium est estimatia

NEXT BAYES LOOP  
tepooh = time 2

First Pass Residuals:

|             |                |                 |                |                 |
|-------------|----------------|-----------------|----------------|-----------------|
| time, res = | .520736309e-01 | -.211587923e-08 | .254245136e-08 | -.539672231e-10 |
| time, res = | .525694094e-01 | -.196634043e-08 | .237330666e-08 | -.115494081e-09 |
| time, res = | .530651879e-01 | -.220760424e-08 | .173436310e-08 | -.513559091e-10 |
| time, res = | .535609665e-01 | -.303361597e-08 | .398389188e-08 | -.496343557e-10 |
| time, res = | .540567450e-01 | -.264041814e-08 | .250295296e-08 | -.138615302e-09 |

Iteration 1

state corrections

|                    |                     |                    |                     |
|--------------------|---------------------|--------------------|---------------------|
| .4772394463467e-05 | -.1089107115169e-05 | .3038021824766e-07 | -.7321866990798e-04 |
| .8064792395159e-04 | -.1029113298157e-03 | .1786224106006e-01 | -.1676136958536e+00 |

current reference trajectory state vector at time 2:

|                     |                     |                    |                     |
|---------------------|---------------------|--------------------|---------------------|
| -.1327393361293e+00 | -.5775486202332e+00 | .8068165467056e+00 | -.4650639050595e-02 |
| -.1983379893742e-01 | .3285147722390e-01  | .3351593047604e+00 | .4312038639799e+01  |

|             |                |                |                |                 |
|-------------|----------------|----------------|----------------|-----------------|
| time, res = | .520736309e-01 | .483175390e-05 | .194892165e-06 | -.238352329e-05 |
| time, res = | .525694094e-01 | .479114335e-05 | .163652054e-06 | -.203051882e-05 |
| time, res = | .530651879e-01 | .474885516e-05 | .132980260e-06 | -.169085454e-05 |
| time, res = | .535609665e-01 | .470468343e-05 | .106250640e-06 | -.136435899e-05 |
| time, res = | .540567450e-01 | .466042017e-05 | .768105582e-07 | -.105069666e-05 |

Iteration 2

state corrections

|                    |                     |                     |                     |
|--------------------|---------------------|---------------------|---------------------|
| .7615219313611e-06 | -.1334827757042e-06 | -.5861886219136e-07 | -.1400620533620e-04 |
| .2556661828496e-05 | .9145331031774e-06  | .9494045557528e-03  | -.2037972626567e-02 |

current reference trajectory state vector at time 2:

|                     |                     |                    |                     |
|---------------------|---------------------|--------------------|---------------------|
| -.1327386246073e+00 | -.5775407537160e+00 | .8068164380868e+00 | -.4664645255932e-02 |
| -.1983124227559e-01 | .3285239175701e-01  | .3361087093161e+00 | .4310000667172e+01  |

|             |                |                |                |                 |
|-------------|----------------|----------------|----------------|-----------------|
| time, res = | .520736309e-01 | .559989979e-05 | .195332241e-06 | -.237775035e-05 |
| time, res = | .525694094e-01 | .555182063e-05 | .164775805e-06 | -.203369292e-05 |
| time, res = | .530651879e-01 | .550179592e-05 | .135278702e-06 | -.170892141e-05 |
| time, res = | .535609665e-01 | .544961655e-05 | .110197235e-06 | -.140327332e-05 |
| time, res = | .540567450e-01 | .539707127e-05 | .829090652e-07 | -.111642211e-05 |

Iteration 3

state corrections

|                    |                     |                      |                     |
|--------------------|---------------------|----------------------|---------------------|
| .8150937176092e-08 | -.1410478678937e-08 | -.66690963503761e-09 | -.1379014888982e-06 |
| .1071002770686e-07 | .2004530430905e-07  | -.2418850477316e-05  | .2907349647999e-04  |

current reference trajectory state vector at time 2:

|                     |                     |                    |                     |
|---------------------|---------------------|--------------------|---------------------|
| -.1327386164564e+00 | -.5775407551264e+00 | .8068164074199e+00 | -.4664703157420e-02 |
| -.1983122356556e-01 | .3285241180231e-01  | .3361063704657e+00 | .4310029740669e+01  |

|             |                |                |                |                 |
|-------------|----------------|----------------|----------------|-----------------|
| time, res = | .520736309e-01 | .560812816e-05 | .195204734e-06 | -.237744213e-05 |
| time, res = | .525694094e-01 | .555997694e-05 | .164732813e-06 | -.203342796e-05 |
| time, res = | .530651879e-01 | .550987841e-05 | .135240531e-06 | -.170870285e-05 |
| time, res = | .535609665e-01 | .545762345e-05 | .110164184e-06 | -.140310430e-05 |
| time, res = | .540567450e-01 | .540500077e-05 | .828014448e-07 | -.111630609e-05 |

Iteration 4

state corrections

|                     |                     |                    |                     |
|---------------------|---------------------|--------------------|---------------------|
| -.2834382545790e-09 | .4966473091265e-10  | .2185191770470e-10 | .5306844893774e-08  |
| -.9005844325026e-09 | -.4537295038619e-09 | .4248777543284e-07 | -.4465267579966e-06 |

current reference trajectory state vector at epoch:

|                     |                     |                    |                     |
|---------------------|---------------------|--------------------|---------------------|
| -.1327386167398e+00 | -.5775407550763e+00 | .8068164074417e+00 | -.4664777350576e-02 |
| -.1983122406614e-01 | .3205241134858e-01  | .3361064129534e+00 | .4310029294142e+01  |

Last Pass Residuals:

|             |                |                |                |                 |
|-------------|----------------|----------------|----------------|-----------------|
| time, res = | .520736309e-01 | .560784228e-05 | .195204667e-06 | -.237744529e-05 |
| time, res = | .525694094e-01 | .555969384e-05 | .164732734e-06 | -.203343071e-05 |
| time, res = | .530651879e-01 | .550959815e-05 | .135240432e-06 | -.170870526e-05 |
| time, res = | .535609665e-01 | .545734610e-05 | .110164071e-06 | -.140310647e-05 |
| time, res = | .540567450e-01 | .540472639e-05 | .828013184e-07 | -.111630787e-05 |

CONVERGENCE ACHIEVED.

In nomina Gaussian trajectorum referentia

declarium est estimatia

NEXT BAYES LOOP  
tepochn = time 3

First Pass Residuals:

time, res = .101651483e+00 -.100673452e-04 .332567295e-05 -.185167969e-04  
time, res = .102147262e+00 -.104727284e-04 .430990151e-05 -.241225845e-04  
time, res = .102643040e+00 -.108936810e-04 .541680948e-05 -.303255215e-04  
time, res = .103138819e+00 -.113308403e-04 .662202038e-05 -.371336172e-04  
time, res = .103634597e+00 -.117838512e-04 .794113728e-05 -.445512757e-04

iteration 1

state corrections

.1408942335930e-05 .1019466118635e-05 -.2264490910225e-05 .7804349800153e-03  
.5697722345062e-03 -.1476070141468e-02 .1468127904633e-01 -.5050602625640e+00

current reference trajectory state vector at time 3:

-.1330975613393e+00 -.5790404145171e+00 .8096544890475e+00 -.9459250899923e-02  
-.4135244386006e-01 .8619843067786e-01 .3507876919998e+00 .3804161031578e+01

time, res = .101651483e+00 -.823612411e-05 .780864914e-06 .966100091e-06  
time, res = .102147262e+00 -.821767890e-05 .779175096e-06 .768552275e-06  
time, res = .102643040e+00 -.819930637e-05 .796032783e-06 .549569884e-06  
time, res = .103138819e+00 -.818143254e-05 .805838625e-06 .305388616e-06  
time, res = .103634597e+00 -.816348789e-05 .823014784e-06 .358946038e-07

iteration 2

state corrections

.7099607587070e-05 -.1527531318060e-05 -.6699753658038e-06 -.3336368992457e-04  
.3022512340808e-04 -.4015400423896e-04 .3521001280592e-01 -.2515317166610e+00

current reference trajectory state vector at time 3:

-.1330896617317e+00 -.5790499420484e+00 .8096538190721e+00 -.9492614589848e-02  
-.4131422373966e-01 .8615827667362e-01 .3859985048057e+00 .3552629314917e+01

time, res = .101651483e+00 -.182580406e-06 .101704796e-08 .962569567e-07  
time, res = .102147262e+00 -.180872879e-06 .829302715e-08 .108306370e-06  
time, res = .102643040e+00 -.178641234e-06 .735337102e-08 .142274295e-06  
time, res = .103138819e+00 -.176235749e-06 .211995642e-07 .195500296e-06  
time, res = .103634597e+00 -.173210684e-06 .356493463e-07 .269018163e-06

iteration 3

state corrections

.1785326751436e-06 -.3962636388119e-07 -.2576102003298e-08 -.2332915298275e-05  
.3150244560652e-05 -.4974173782535e-05 .6839410314836e-02 -.2533005817415e-01

current reference trajectory state vector at time 3:

-.1330894331990e+00 -.5790499816747e+00 .8096538172960e+00 -.9494947505146e-02  
-.4131107349509e-01 .8615330249904e-01 .3928379151205e-00 .3527299256743e+01

time, res = .101651483e+00 -.184599170e-08 .383802701e-03 .1700003333e-08  
time, res = .102147262e+00 -.188518304e-08 -.357831602e-08 .107068482e-08  
time, res = .102643040e+00 -.178554985e-08 .360917363e-08 .273275281e-08  
time, res = .103138819e+00 -.194957889e-08 -.425014024e-09 .408342949e-08  
time, res = .103634597e+00 -.178375517e-08 -.149100080e-08 .618284950e-08

iteration 4

state corrections

.2441997527549e-08 -.5038107129709e-09 -.1332160705860e-09 -.2425202470404e-07  
.2674560279695e-07 -.3642365757096e-07 .6585595590269e-04 -.3034730539397e-04

current reference trajectory state vector at time 3:

-.1330894307570e+00 -.5790499821735e+00 .8096538171628e+00 -.9494971757471e-02  
-.4131104674941e-01 .8615326607618e-01 .3929037710764e-00 .3527208909438e+01

time, res = .101651483e+00 .630525410e-09 .373884612e-08 .803733154e-09  
time, res = .102147262e+00 .565372294e-09 -.356359153e-08 -.413223567e-09  
time, res = .102643040e+00 .629075565e-09 .382533771e-08 .183410289e-09  
time, res = .103138819e+00 .421454777e-09 .792317323e-10 .102279045e-10  
time, res = .103634597e+00 .533530029e-09 -.610926421e-09 .649873827e-10

iteration 5  
state corrections  
.2816313341378e-11 -.5133465718550e-12 -.2058123034134e-12 -.3341240477990e-10  
.8633814355239e-11 .4240745285131e-11 -.4860173745442e-09 .7662202059526e-08

current reference trajectory state vector at time 3:  
-.1330894307542e+00 -.5790499821791e+00 .8096533171625e-00 -.9494971790884e-02  
-.4131104674077e-01 .3615326608042e-01 .3929037705904e-00 .3527260917100e+01

Last Pass Residuals:

|             |                |                |                 |                 |
|-------------|----------------|----------------|-----------------|-----------------|
| time, res = | .101651483e+00 | .633384685e-09 | .373855746e-08  | .803691522e-09  |
| time, res = | .102147262e+00 | .568213934e-09 | -.356386831e-08 | -.413276981e-09 |
| time, res = | .102643040e+00 | .632699247e-09 | .382507415e-08  | .183338540e-09  |
| time, res = | .103138819e+00 | .424260183e-09 | .789824872e-10  | -.103245433e-10 |
| time, res = | .103634597e+00 | .536316831e-09 | -.611159900e-09 | .648592881e-10  |

CONVERGENCE ACHIEVED.  
In nomina Gaussian trajectorye referentia  
declarium est estimatia

[illegible]

```

the xis lengths for the covariance matrix are
.553892e-01
.656473e-03
.146556e-05
.431656e-05

the xis lengths for the position components are
.656473e-03
.146556e-05
.431656e-05

```

[illegible]

```

the axis lengths for the covariance matrix are
.171632e-01
.183764e-02
.171609e-05
.428545e-05

```

[illegible]

```

the axis lengths for the covariance matrix are
.269372e-01
.999999e+00
the axis lengths for the position components are
.119875e-02
.191221e-05
.427634e-05

```

# SUMMARY OF ALL CASES

|                      | $V_e$ (DU/TU)  | M              |                           |
|----------------------|----------------|----------------|---------------------------|
| Exact                |                |                |                           |
| Stage I              | .317298255e+00 | .447963755e+01 |                           |
| Stage II             | .392904480e+00 | .352726545e+01 |                           |
| Data Segments = 16   |                |                |                           |
| Point #:             |                |                | Converged on iteration #: |
| 16                   | .317315725e+00 | .447938985e+01 | 2                         |
| 32                   | .317193607e+00 | .448102999e+01 | 1                         |
| 48                   | .317368164e+00 | .447874052e+01 | 1                         |
| 64                   | .317439486e+00 | .447790352e+01 | 1                         |
| 80                   | .317345804e+00 | .447907412e+01 | 1                         |
| 96                   | .317478537e+00 | .447760474e+01 | 1                         |
| 112                  | .317518937e+00 | .447727873e+01 | 1                         |
| 128                  | .317471044e+00 | .447786183e+01 | 1                         |
| 144                  | .317562390e+00 | .447706513e+01 | 1                         |
| 160                  | .317353960e+00 | .447912668e+01 | 1                         |
| 176                  | .316971039e+00 | .448249374e+01 | 1                         |
| 192                  | .675400244e+00 | .269968034e+01 | 4                         |
| 208                  | .473677628e+00 | .311701023e+01 | 3                         |
| 224                  | .392931745e+00 | .352711260e+01 | 3                         |
| 240                  | .393296229e+00 | .352519154e+01 | 2                         |
| Data Segments = = 24 |                |                |                           |
| 24                   | .317305053e+00 | .447954511e+01 | 2                         |
| 48                   | .317304049e+00 | .447955637e+01 | 1                         |
| 72                   | .317283474e+00 | .447981709e+01 | 1                         |
| 96                   | .317353812e+00 | .447900588e+01 | 1                         |
| 120                  | .317308991e+00 | .447952553e+01 | 1                         |
| 144                  | .317392974e+00 | .447871385e+01 | 1                         |
| 168                  | .317268608e+00 | .447990303e+01 | 1                         |
| 192                  | .361079891e+00 | .414022556e+01 | 3                         |
| 216                  | .406305787e+00 | .345281348e+01 | 4                         |
| 240                  | .393116024e+00 | .352614137e+01 | 3                         |
| Data Segments = 32   |                |                |                           |
| 32                   | .317308409e+00 | .447949574e+01 | 2                         |
| 64                   | .317306635e+00 | .447953176e+01 | 1                         |
| 96                   | .317305226e+00 | .447955992e+01 | 1                         |
| 128                  | .317324984e+00 | .447935543e+01 | 1                         |
| 160                  | .317285360e+00 | .447976097e+01 | 1                         |
| 192                  | .331250475e+00 | .436375095e+01 | 3                         |
| 224                  | .396913417e+00 | .350494767e+01 | 4                         |
| 256                  | .392841545e+00 | .352758616e+01 | 2                         |

Continued:

$V_e$  (DU/TU) M

Point #: Data Segments = 48 Converged on iteration #

|     |                |                |   |
|-----|----------------|----------------|---|
| 48  | .317303338e+00 | .447956672e+01 | 2 |
| 96  | .317296296e+00 | .447966247e+01 | 1 |
| 144 | .317295326e+00 | .447966670e+01 | 1 |
| 192 | .320304527e+00 | .445333709e+01 | 3 |
| 240 | .393636292e+00 | .352327942e+01 | 5 |

Data segments = 50

|     |                |                |   |
|-----|----------------|----------------|---|
| 50  | .317304110e+00 | .447955585e+01 | 2 |
| 100 | .317296361e+00 | .447966253e+01 | 1 |
| 150 | .317291091e+00 | .447970933e+01 | 1 |
| 200 | .686866862e+00 | .261674361e+01 | 3 |
| 250 | .392921506e+00 | .352717312e+01 | 5 |

Data Segments = 64

|     |                |                |   |
|-----|----------------|----------------|---|
| 64  | .317299985e+00 | .447961325e+01 | 2 |
| 128 | .317300626e+00 | .447960982e+01 | 1 |
| 192 | .318331699e+00 | .447028787e+01 | 2 |
| 256 | .393127634e+00 | .352608074e+01 | 5 |
| 320 | .392905747e+00 | .352725914e+01 | 2 |

Data Segments = 66 \*

|     |                |                |   |
|-----|----------------|----------------|---|
| 66  | .317298046e+00 | .447964018e+01 | 2 |
| 132 | .317302351e+00 | .447958587e+01 | 1 |
| 198 | .416586494e+00 | .356484600e+01 | 4 |
| 254 | .392918156e+00 | .352720315e+01 | 4 |
| 320 | .392871250e+00 | .352740128e+01 | 2 |

Data segments = 66 \*\*

|     |                |                |   |
|-----|----------------|----------------|---|
| 66  | .317300432e+00 | .447960725e+01 | 2 |
| 132 | .317301918e+00 | .447959537e+01 | 1 |
| 198 | .364060396e+00 | .409734550e+01 | 4 |
| 254 | .392912142e+00 | .352722456e+01 | 4 |
| 320 | .392902400e+00 | .352727440e+01 | 2 |

Data Segments = 100

|     |                |                |   |
|-----|----------------|----------------|---|
| 100 | .317297143e+00 | .447965233e+01 | 1 |
| 200 | .336106412e+00 | .431002929e+01 | 4 |
| 300 | .392903770e+00 | .352726891e+01 | 5 |

\* Values of .9 .9 .9 .9 .9 .9 .7 .7

\*\* Values of .8 .8 .8 .8 .8 .8 .5 .5

## VITA

David A. Vallado was born on May 14, 1958 in Winchester Massachusetts. He graduated from Parsippany Hills High School in 1976. He attended the United States Air Force Academy and graduated in 1980. He received a Bachelor of Science in Astronautical Engineering. His first assignment was as the project officer for the M-X Stage I at the Ballistic Missile Office at Norton AFB, California. While stationed at Norton AFB, he also completed a Master of Science in Systems Management from the University of Southern California in 1982. He entered the School of Engineering, Air Force Institute of Technology in June 1983. He met and married his wife Laura while at AFIT.

Permanent address: 3 Glencove Road

Morris Plains, New Jersey 07950

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD-A151 927

## REPORT DOCUMENTATION PAGE

|                                                                                                                                                                                   |                                                 |                                                                                                  |                               |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------|--------------------------------------------------------------------------------------------------|-------------------------------|
| REPORT SECURITY CLASSIFICATION<br>UNCLASSIFIED                                                                                                                                    |                                                 | 1b. RESTRICTIVE MARKINGS                                                                         |                               |
| SECURITY CLASSIFICATION AUTHORITY                                                                                                                                                 |                                                 | 3. DISTRIBUTION/AVAILABILITY OF REPORT<br>Approved for Public Release;<br>Distribution Unlimited |                               |
| DECLASSIFICATION/DOWNGRADING SCHEDULE                                                                                                                                             |                                                 | 5. MONITORING ORGANIZATION REPORT NUMBER(S)                                                      |                               |
| PERFORMING ORGANIZATION REPORT NUMBER(S)<br>AFIT/GA/AA/84D-10                                                                                                                     |                                                 | 7a. NAME OF MONITORING ORGANIZATION                                                              |                               |
| 1. NAME OF PERFORMING ORGANIZATION<br>School of Engineering                                                                                                                       | 8b. OFFICE SYMBOL<br>(If applicable)<br>AFIT/AA | 7b. ADDRESS (City, State and ZIP Code)                                                           |                               |
| 2. ADDRESS (City, State and ZIP Code)<br>Air Force Institute of Technology<br>Wright-Patterson AFB, Ohio 45433                                                                    |                                                 | 9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER                                                  |                               |
| 3. NAME OF FUNDING/SPONSORING ORGANIZATION                                                                                                                                        | 8b. OFFICE SYMBOL<br>(If applicable)            | 10. SOURCE OF FUNDING NOS.                                                                       |                               |
| 4. ADDRESS (City, State and ZIP Code)                                                                                                                                             |                                                 | PROGRAM ELEMENT NO.                                                                              | PROJECT NO.                   |
| 1. TITLE (Include Security Classification)<br>See Box 19                                                                                                                          |                                                 | TASK NO.                                                                                         | WORK UNIT NO.                 |
| 2. PERSONAL AUTHOR(S)<br>David A. Vallado, M.S., B.S., Capt., USAF                                                                                                                |                                                 |                                                                                                  |                               |
| 3a. TYPE OF REPORT<br>MS Thesis                                                                                                                                                   | 13b. TIME COVERED<br>FROM TO                    | 14. DATE OF REPORT (Yr., Mo., Day)<br>1984 December                                              | 15. PAGE COUNT<br>128         |
| 6. SUPPLEMENTARY NOTATION                                                                                                                                                         |                                                 |                                                                                                  |                               |
| 7. COSATI CODES                                                                                                                                                                   |                                                 | 18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)                |                               |
| FIELD                                                                                                                                                                             | GROUP                                           | SUB. GR.                                                                                         |                               |
| 19                                                                                                                                                                                | 04                                              | Estimation Rocket and Missile Trajectory                                                         |                               |
| 16                                                                                                                                                                                | 02                                              | Bayes Filter Parameter Identification                                                            |                               |
| 9. ABSTRACT (Continue on reverse if necessary and identify by block number)                                                                                                       |                                                 |                                                                                                  |                               |
| Title: ESTIMATION OF LAUNCH VEHICLE PERFORMANCE PARAMETERS                                                                                                                        |                                                 |                                                                                                  |                               |
| Thesis Chairman: Dr. William E. Niesel                                                                                                                                            |                                                 |                                                                                                  |                               |
| 10. DISTRIBUTION/AVAILABILITY OF ABSTRACT<br>UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS <input type="checkbox"/> |                                                 | 21. ABSTRACT SECURITY CLASSIFICATION<br>UNCLASSIFIED                                             |                               |
| 22a. NAME OF RESPONSIBLE INDIVIDUAL<br>Dr. William E. Niesel, Professor                                                                                                           |                                                 | 22b. TELEPHONE NUMBER<br>(Include Area Code)<br>513-255-3517                                     | 22c. OFFICE SYMBOL<br>AFIT/AA |

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

The estimation of launch vehicle performance parameters was explored through the use of a Bayes Filter. The main emphasis was to use an eight state model that would include the vehicle position and velocity vectors, the vehicle exhaust velocity, and the ratio of the mass flow rate to the initial mass. A primary objective was to be able to observe these quantities through the staging events, where the last two elements would be changing very rapidly. The results indicated that indeed the staging event was observable. However, as would be expected, the data processed at the exact time of staging included errors which diminished as the filter processed more data. A fading memory was added in an attempt to improve the filters performance in the area of a staging event. This proved to be marginally successful as several Bayes loop iterations had to be performed to notice the effect of the fading memory addition. Care was taken to show each step of the filter development and its checkout. Several numerical tables are presented including the input and output data.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

**END**

**FILMED**

4-85

**DTIC**