AD-A082 827 NAVAL SURFACE WEAPONS CENTER DAHLGREN LAB VA F/G 22/3
DATA GENERATOR FOR A SATELLITE-BORNE THREE-AXIS ACCELEROMETER.(U)
MAR 80 C G BRANCH
UNCLASSIFIED NSWC/DL/TR-3849 NL

1 OF 1
AD A
082827

END
DATE
FILMED
5-80
DTIC

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NSWC/DL TR-3849	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DATA GENERATOR FOR A SATELLITE-BORNE THREE-AXIS ACCELEROMETER		5. TYPE OF REPORT & PERIOD COVERED Final
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Carol G. Branch		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Surface Weapons Center Dahlgren Laboratory (K11) Dahlgren, VA 22448		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 63701B:3204:240: CK10RC
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Mapping Agency Washington, DC 20360		12. REPORT DATE March 1980
		13. NUMBER OF PAGES 64
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Data Generator
Three-Axis Accelerometer
Gauss-Markov Random Noise
Satellite-Borne Accelerometer
Orbit Determination Studies

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes a series of computer programs which generate simulated accelerometer data for filter validation and orbit determination studies. Attitude thruster induced accelerations and Gauss–Markov random noise are modeled.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

FOREWORD

Interest has been demonstrated in the use of accelerometer data for satellite orbit and gravity field model improvement. These programs generate accelerometer data for the purpose of filter checkout and orbit simulations.

Mr. Phillip L. Young formulated the Gauss-Markov process for the random noise. Dr. Bruce R. Hermann modeled the resonance responses in the Host Vehicle Vibrations Program. The equations for the damping constants were derived by Dr. Jeffrey N. Blanton.

The work was accomplished in the Satellite Geodesy Branch of the Astronautics and Geodesy Division.

Released by:

R. T. RYLAND, JR., Head
Strategic Systems Department

Accession For
NTIS GRA&I
DDC TAB
Unannounced
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or special
A

CONTENTS

PRECEDING PAGE BLANK-NOT FILMED

CONTENTS (Continued)

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

Atmospheric drag has a significant effect on low-altitude trajectories. To measure this force, an accelerometer will be tested which senses this nonconservative acceleration. The information obtained will then be used to improve the orbit and obtain a more accurate gravity field model.

When the raw data is first received, all known systematic effects will be removed. These include gravity gradient and pitch-rate induced accelerations. However, the preprocessed data will still contain anomalies of which there is little or no *a priori* knowledge. Satellite attitude thruster-induced vibrations are a major source of unmodeled noise. As for the instrument itself, each axis of the accelerometer adds an undetermined bias to the output accelerations. In addition, the conversion of the digital output to engineering units introduces a scale factor error. Other effects may take the form of additive correlated noise and white noise.

The first instrument to be flown is the miniature electrostatic accelerometer (MESA) built by Bell Aerospace Textron under contract to the Air Force Geophysics Laboratory. The MESA senses accelerations along the vehicle pitch, roll, and yaw axes. Relying on the accuracy of the vehicle's attitude control system, these axes are assumed to be along the $\vec{R}$, $\vec{R} \times \vec{V}$, and $-\vec{R}$ vectors, respectively. Hence, any attitude errors will be reflected in the measurements. The MESA also introduces undetermined scale factor errors and biases.

Consequently, a computer program was designed which applies a filter to the processed accelerations. This filter should statistically remove most of the unmodeled corruptions. The filtered data is then input to a special version of the Celest orbit determination program. In order to check out the filter and perform special case studies prior to launch, simulated processed data had to be created. This required a series of computer programs which will be referred to as the "Data Generator." The individual programs include "Drag Accelerations," "Host Vehicle Vibrations," "Merge," and "Gauss-Markov."

COMPUTER PROGRAM DESCRIPTIONS AND FORMULATIONS

DRAG ACCELERATIONS

Updates were inserted in the "Orbgen" and "Atdrag" subroutines of Celest[1] Version 16. During integration, the atmospheric drag accelerations are converted from the inertial frame to a perpendicular local frame $(\hat{U}_1, \hat{U}_2, \hat{U}_3)$:

$$\hat{U}_1 = \widehat{R \times (V \times R)}$$

$$\hat{U}_2 = \widehat{V \times R}$$

$$\hat{U}_3 = -\hat{R}$$

A 2-sec step size is used to approximate the MESA data averaging interval.

In the integration start routine, there are multiple iterations of the first 10 time lines. Any duplicate set of time lines must be omitted. Hence, only the final set of accelerations are retained.

The output file of drag accelerations is described in the section of this report entitled "Computer Program Utilization" (p. 15). It can be used as input to either the Host Vehicle Vibrations Program or the Gauss-Markov Program.

HOST VEHICLE VIBRATIONS

Run once for each of the accelerometer coordinate axes, this program simulates the response of the MESA to the attitude thruster firings of a satellite in orbit. The attitude thrusters are assumed to be coupled. This means that, when a pair of thrusters fire, they create a torque about the axis perpendicular to the thruster plane. The resultant force is zero, causing purely rotational vehicle motion.

The magnitude and shape of the initial impulse can be specified, with a maximum duration of 0.5 sec. All impulses are assumed to be positive along the respective axes. For the purposes of the simulations to be performed, the direction of the initial impulses is inconsequential.

Eleven vehicle resonances may be activated by each thruster pair. A different amplitude and damping constant is input for each frequency. Because of the resolution of the MESA, a resonance is assumed to be totally damped once its amplitude is below 10^{-9} g (one g is 9.80621×10^{-3} km/sec^2).

The satellite is assumed to experience maximum drag acceleration at perigee, the time of closest approach to the earth. The thrusters fire most often during this period because of the high aerodynamic torques resulting from the increased magnitude and gradient of the gravity force. The opposite is true at apogee, the satellite's farthest distance from the earth. At this point, the drag acceleration and the frequency of thrusts are assumed to be at a minimum.

Consequently, for each thruster pair, the probability of a thrust is linearly interpolated with respect to the true drag acceleration along the velocity vector. The time and drag accelerations at apogee and perigee are obtained from the Drag Accelerations Program. To vary the frequency of the attitude thrusts, the

PRECEDING PAGE BLANK-NOT FILMED

number of seconds between thrusts at apogee and perigee is input. The execution time of the program can be reduced by increasing the time between probability recalculations.

To test for a thruster firing, a random number (N) from the interval [0, 1) is compared to the probability (P) derived for that time. If $0 \leq N < P$, the thruster is assumed to have fired.

Frequency Response

Since the MESA sampling interval is approximately 2 sec, the highest frequency that can be unambiguously filtered from the data is 0.25 Hz (the Nyquist frequency). An electronic filter was incorporated in the MESA to attenuate any higher frequencies. However, virtual blockage of the signal does not occur until the 10-Hz level. Any resonances above 0.25 Hz will alias into those below 0.25 Hz as a result of the size of the sample interval.

Simulated accelerometer data must reflect this condition. Hence, an iteration interval of 0.05 sec ($1/2f$, where $f = 10$) is used in the Host Vehicle (HV) Vibrations Program to include resonance at the higher frequencies. Letting K be the number of 0.05-sec time lines since a given thrust, the MESA response at a particular vehicle resonance (referenced by 'ℓ') is

$$\begin{aligned} r_\ell[KT] &= A_\ell\, e^{-a_\ell T} (\sin 2\pi f_\ell T)\, U(K) \\ &\quad + 2e^{-a_\ell T} (\cos 2\pi f_\ell T)\, r_\ell[(K-1)\,T] \\ &\quad - e^{-a_\ell T}\, r_\ell[(K-2)\,T] \end{aligned}$$

where

A_ℓ = the maximum amplitude

a_ℓ = the damping constant

T = 0.05 sec

f_ℓ = the resonance frequency

U = an array of 10 terms which determines the size and shape of the initial thruster impulse.

The total response at KT seconds after a thruster fires is then

$$r[KT] = \sum_{\ell=1}^{N_f} r_\ell[KT]$$

where N_f is the number of resonance frequencies excited. For each consecutive 0.05-sec time line, the response to the thrust is computed until it is damped below 10^{-9} g. A different sequence of accelerations is computed for each of the four pairs of thrusters.

Using the method described earlier, the program checks for thrusts at specified intervals. If a thrust is indicated, the appropriate values are added to the subsequent accelerations. The aggregate responses are then averaged over a 2-sec period and time-tagged. A flowchart of the program logic is presented in Figure 1.

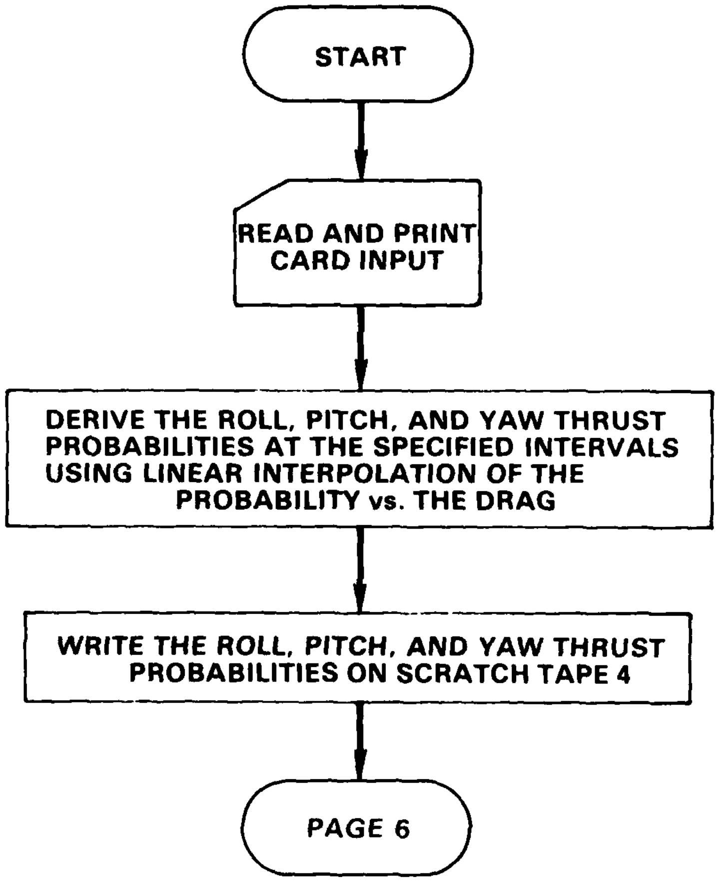

Figure 1. Host Vehicle Vibrations Flowchart

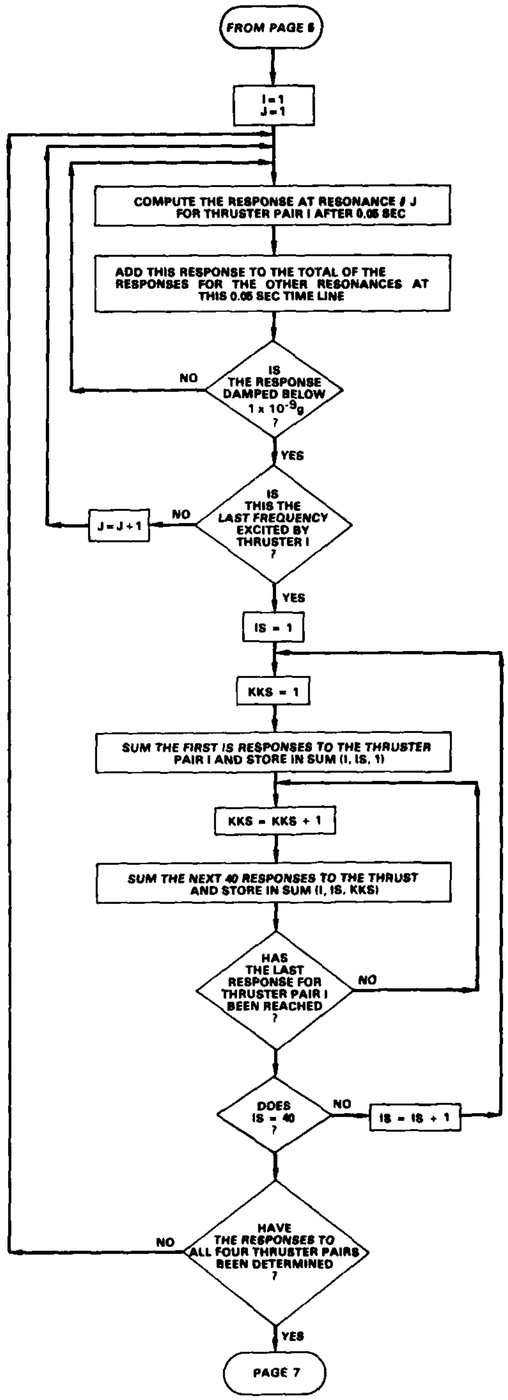

Figure 1. Host Vehicle Vibrations Flowchart (Continued)

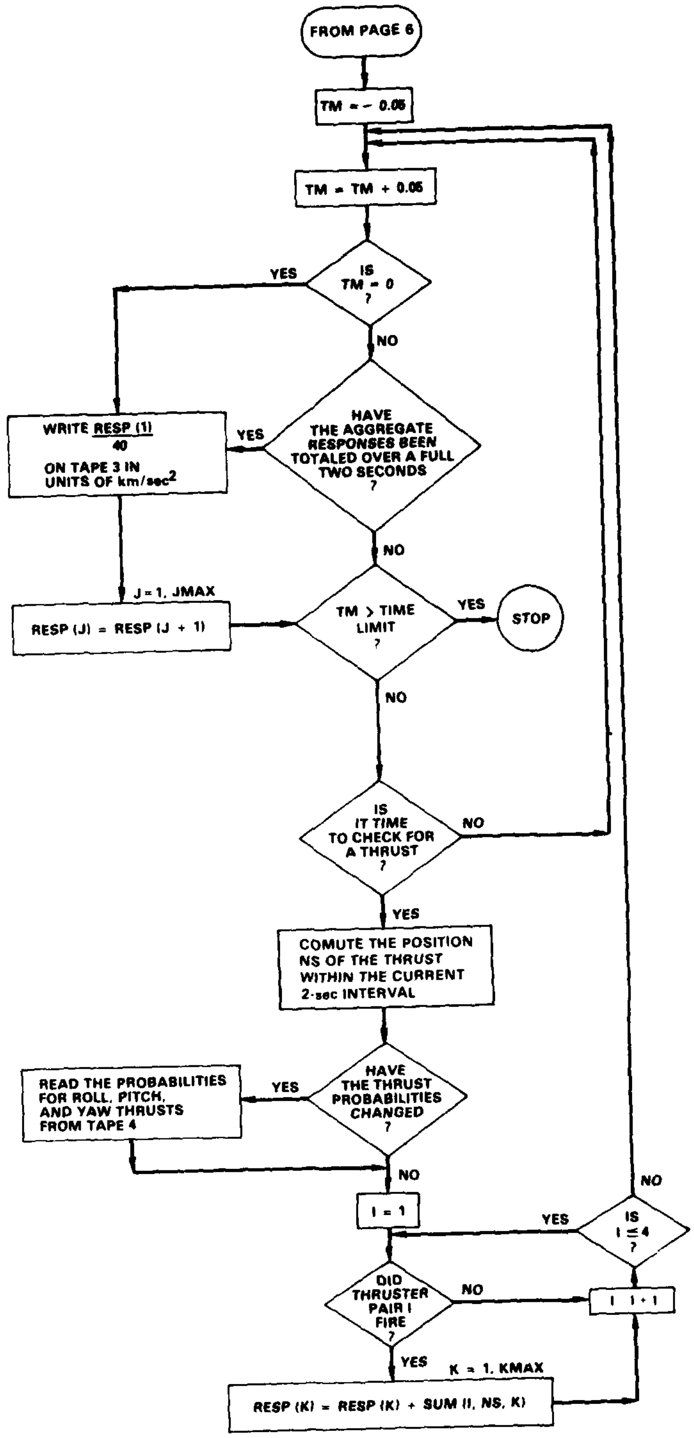

Figure 1. Host Vehicle Vibrations Flowchart (Continued)

Derivation of Damping Constants

The host vehicle behaves like a damped oscillator when excited by an attitude thrust. The equation of motion for this disturbance is

$$F(t) = m\ddot{x} + c\dot{x} + k\dot{x} = 0$$

where m and c are the coefficients of mass and damping, and k is the spring constant.

One solution to this equation is to let $x = \phi(t)$, such that x, $\dot{x}$, and $\ddot{x}$ differ only by constant multiplicative factors. The general exponential form Ae^{qt}, where A and q are constants, fills this requisite perfectly. Substituting in $F(t)$,

$$\begin{aligned} F(t) &= m(Ae^{qt})'' + c(Ae^{qt})' + k(Ae^{qt}) \\ &= Ae^{qt}[mq^2 + cq + k] = 0 \end{aligned}$$

Since e^{qt} never vanishes, this implies that

$$mq^2 + cq + k = 0$$

The general solution to this quadratic equation is, of course,

$$q = \frac{-c \pm \sqrt{c^2 - 4mk}}{2m}$$

The real part of this root is the damping term, whereas the imaginary part is the oscillatory term. Since our system is characterized by damped oscillatory motion, q must be a complex number of the form $-\alpha \pm \omega i$, such that α and ω are real numbers. Hence,

$$q = -\alpha \pm i\omega = \frac{-c \pm \sqrt{c^2 - 4km}}{2m}$$

where $c^2 - 4km < 0$. Consequently, $\alpha = c/2m$ and $\omega = \sqrt{4km - c^2}/2m$.

The solution is then

$$\begin{aligned} x &= A_1 e^{q_1 t} + A_2 e^{q_2 t} \\ &= A_1 e^{(-\alpha + i\omega)t} + A_2 e^{(-\alpha - i\omega)t} \\ &= e^{-\alpha t}(A_1 e^{i\omega t} + A_2 e^{-i\omega t}) \end{aligned}$$

Requiring that $\dot{x}(t_o) = 0$ (i.e.,$A_1 = A_2 = A/2$) and $x(t_o) = x_o$ (i.e., $x_o = A$),

$$x = e^{-\alpha t} \cdot (A/2) \cdot (e^{i\omega t} + e^{-i\omega t})$$

$$= x_o e^{-\alpha t}\left(\frac{e^{i\omega t} + e^{-i\omega t}}{2}\right)$$

$$= x_o e^{-\alpha t} \cos \omega t$$

The number (N) of cycles completed at time t is $n = \omega t/2\pi$ or, equivalently, $t = 2\pi n/\omega$.

The maximum ratio of displacement at time t to the initial displacement is

$$\frac{x(t)}{x_o} = e^{-\alpha t} \quad \text{since} \quad |\cos \omega t| \leq 1$$

Substituting for t,

$$\frac{x(t)}{x_o} = e^{-2\pi n\alpha/\omega} \quad \text{or} \quad ln\left(\frac{x(t)}{x_o}\right) = \frac{-2\pi n\alpha}{\omega}$$

Hence,

$$n = \left(\frac{-\omega}{2\pi\alpha}\right) ln\left(\frac{x(t)}{x_o}\right)$$

$$= \frac{-\sqrt{4km - c^2}}{4\pi mc/2m}\, ln\left(\frac{x(t)}{x_o}\right)$$

$$= \frac{\sqrt{4km - c^2}}{2\pi c}\, ln\left(\frac{x_o}{x(t)}\right)$$

$$= \frac{1}{2\pi}\left[\frac{4km - c^2}{c^2}\right]^{1/2} ln\left(\frac{x_o}{x(t)}\right)$$

The system is said to be critically damped when $\sqrt{c^2 - 4km} = 0$. The critical damping coefficient (C_c) is then

$$C_c = 4km$$

Rewriting n,

$$n = \frac{1}{2\pi}\left[\frac{1 - (c/C_c)^2}{(c/C_c)^2}\right]^{1/2} ln\left(\frac{x_o}{x(t)}\right)$$

c/C_c is the percentage of critical damping inherent in the system. If $x(t)$ is set to 1×10^{-9} g (minimum amplitude), n would be the total number of cycles required to damp out the oscillation. The damping coefficient α can be determined from n using the formula

$$\alpha = -\left(\frac{\omega}{2\pi n}\right) \ln \left(\frac{x_o}{x(t)}\right)$$

In summary, the following two equations should be used to determine the respective damping coefficients:

$$n = \frac{1}{2\pi}\left[\frac{1 - (c/C_c)^2}{(c/C_c)^2}\right]^{1/2} \ln\left(\frac{x_{max}}{x_{min}}\right)$$

and

$$\alpha = -\left(\frac{\omega}{2\pi n}\right) \ln \left(\frac{x_{max}}{x_{min}}\right)$$

where

x_{max} = the maximum amplitude of the disturbance

x_{min} = the minimum sensed acceleration (1×10^{-9} g)

c/C_c = the percentage of critical damping inherent in the host vehicle.

α corresponds to A_g in the HV vibrations formulation.

MERGE

This program adds the thruster-induced vibrational accelerations computed for each axis to the corresponding true drag accelerations from Celest. If only one axis is to be affected, the input files for the undisturbed axes are such that all the accelerations are set to zero. These files should have the correct time tags and format, however (p. 20).

GAUSS-MARKOV RANDOM NOISE

Using a Gauss-Markov statistical process, all three components of random noise are computed simultaneously. The composite error vector results from a random bias and scale factor, additive correlated noise, axis misalignment, and white noise. The magnitudes of the errors are assumed to be small and, hence, are retained only to the first order.

The random noise is computed at each time line. It is then added to the drag accelerations corrupted by the thruster-induced vehicle vibrations. That is,

$$a_s(t) = a_I(t) + e_1(t) + e_2(t) + e_3(t) + e_4(t)$$

such that

a_s = the output acceleration

a_I = the input acceleration

e_1 = the error due to the random bias and scale factor

e_2 = the time-dependent orientation error

e_3 = the additive correlated noise

e_4 = the white noise.

The models of the individual error sources follows.

Bias and Scale Factor

Much of the unmodeled system noise may result from an unknown bias or scale factor. Some types of instrument error are of this form.

$$\vec{e_1} = \vec{b} + S\vec{a}(t)$$

where b and s remain constant for the duration of the run. $S = \| s_{ij} \|$ where $s_{ij}; i,j = 1,3$ are input quantities. $\vec{b} = b_x\hat{i} + b_y\hat{j} + b_z\hat{k}$.

b_x is randomly chosen from a Gaussian (normal) population of mean μ_{b_x} and variance $\sigma^2_{b_x}$, where μ_{b_x} and σ_{b_x} are input constants. In the same manner, b_y and b_z are independently selected on the basis of the input values $\mu_{b_y}, \sigma^2_{b_y}$ and $\mu_{b_z}, \sigma^2_{b_z}$, respectively.

Time-Dependent Orientation Error

Since the satellite has an attitude sensing and control system, the axis orientation errors (β) are assumed to be small angles (i.e., $\sin \beta \approx \beta$).

$$\vec{e_2}(t_n) = \mathcal{M}(t_n)\vec{a}(t_n)$$

where $\vec{a}(t_n)$ is the input acceleration vector at time t_n.

The nonzero elements of $\mathfrak{M}(t_n)$ are assumed to be pair-wise uncorrelated for a given time t_n.

$$\mathfrak{M}(t_n) = \begin{bmatrix} 0 & M_z(t_n) & -M_y(t_n) \\ -M_z(t_n) & 0 & M_x(t_n) \\ M_y(t_n) & -M_x(t_n) & 0 \end{bmatrix}$$

such that $M_K(t_n)$ are the current orientation error angles for the respective axes.

$M(t_n)$ itself is a time-dependent vector described by

$$M(t_1) = S_m W_m(t_1)$$

and

$$M(t_{n+1}) = R_m M(t_n) + (I - R_m^2)^{1/2} S_m W_m(t_n)$$

Reflecting the length of the correlation period,

$$R_m = \begin{bmatrix} \rho_{m_x} & 0 & 0 \\ 0 & \rho_{m_y} & 0 \\ 0 & 0 & \rho_{m_z} \end{bmatrix}$$

such that

$$\rho_{m_k} = \exp(-\Delta t/\tau_{m_k})$$

Δt *is the time between data points and* τ_{m_k} is the correlation time for each axis k.

$\vec{W}_m(t_n)$ is a vector of standard normal variates of mean zero and $\sigma = 1$.

$$S_m = \begin{bmatrix} \sigma_{m_x} & 0 & 0 \\ 0 & \sigma_{m_y} & 0 \\ 0 & 0 & \sigma_{m_z} \end{bmatrix}$$

where σ_{m_k} *are the standard deviations (in radians) of the orientation angles for the individual axes.*

Additive Correlated Noise

One known source of additive correlated noise is the effect of temperature variation on the instrument. This type of noise is modeled as the sum of a term ($\vec{\Gamma}(t_n)$) of on-axis correlated noise and a second term ($\xi(t_n)$) of cross-axis correlated noise. Both are obtained using Gauss-Markov vector sequences: i.e.,

$$\vec{e}_3(t_n) = \vec{\eta}(t_n) + \vec{\xi}(t_n)$$

For every t_n, $E\left\{\vec{\eta}(t_n) \cdot \vec{\eta}^T(t_n)\right\}$ (the expected value) is a diagonal matrix.

$$\vec{\eta}(t_1) = S_\eta \vec{W}_\eta(t_1)$$

and

$$\vec{\eta}(t_{n+1}) = R_\eta \vec{\eta}(t_n) + (I - R_\eta^2)^{1/2} S_\eta \vec{W}_\eta(t_n).$$

$$R_\eta(t_n) = \begin{bmatrix} \rho_{\eta_x} & 0 & 0 \\ 0 & \rho_{\eta_y} & 0 \\ 0 & 0 & \rho_{\eta_z} \end{bmatrix}$$

where $\rho_{\eta_k} = \exp(-\Delta t/\tau_{\eta_k})$. Δt, again, is the time between data points and $\tau_{\eta k}$ is the correlation time.

$$S_\eta = \begin{bmatrix} \sigma_{\eta_x} & 0 & 0 \\ 0 & \sigma_{\eta_y} & 0 \\ 0 & 0 & \sigma_{\eta_z} \end{bmatrix}$$

$\sigma_{\eta k}$ is the standard deviation of the correlated noise. $\vec{W}_\eta(t_n)$ is a vector of standard normal variates of mean zero and $\sigma = 1$.

For the second term, $E\left\{\vec{\xi}(t_n) \cdot \vec{\xi}^T(t_n)\right\} = \rho_\xi$, where ρ_ξ is a nondiagonal symmetric input covariance matrix.

$$\vec{\xi}(t_1) = C_\xi \vec{W}_\xi(t_1)$$

and

$$\vec{\xi}(t_{n+1}) = R_\xi \vec{\xi}(t_n) + B_\xi \vec{W}_\xi(t_n)$$

where

$$R_\xi = \begin{bmatrix} \rho_{\xi_x} & 0 & 0 \\ 0 & \rho_{\xi_y} & 0 \\ 0 & 0 & \rho_{\xi_z} \end{bmatrix}$$

$\rho_{\xi k} = \exp(-\Delta t/\tau_{\xi_k})$ where Δt is the time between data points and $\tau_{\xi k}$ is the correlation time for each axis.

In terms of the input matrix P_ξ,

$$C_\xi = P_\xi^{1/2}$$

and

$$B_\xi = (P_\xi - R_\xi P_\xi R_\xi)^{1/2}$$

W_ξ, of course, is a random vector like that of $\vec{W}_\eta$.

Additive White Noise

The uncorrelated random noise is modeled as

$$\vec{e_4}(t_n) = S_\omega \vec{W}(t_n)$$

where

$$S_\omega = \begin{bmatrix} \sigma_{\omega_x} & 0 & 0 \\ 0 & \sigma_{\omega_y} & 0 \\ 0 & 0 & \sigma_{\omega_z} \end{bmatrix}$$

σ_{ω_k} is the standard deviation, or *a priori* magnitude, of the white noise for each axis. The elements of $\vec{W}(t_n)$ are random numbers from a set of standard normal variates of mean zero, and a standard deviation of one.

COMPUTER PROGRAM UTILIZATION

DRAG ACCELERATIONS PROGRAM

Input/Output

The orbit parameters are left to the user's discretion. Reference 1 describes the necessary input values. However, an integration step size of 2 sec is required.

The accelerations, in the orthogonal local frame, are written on TAPE51. Table 1 shows the format for each record.

Table 1. Drag Accelerations Output File Format

Word	Description	Unit	Format
1	Time from Epoch	sec	Floating Point
2	Drag Along $\hat{U}_1$ Axis	km/sec^2	Floating Point
3	Drag Along $\hat{U}_2$ Axis	km/sec^2	Floating Point
4	Drag Along $\hat{U}_3$ Axis	km/sec^2	Floating Point

The times of apogee and perigee are printed out, along with the corresponding drag accelerations along $\hat{U}_1$. This information will be needed as input to the HV Vibrations Program.

HOST VEHICLE VIBRATIONS PROGRAM

Input

A file (TAPE2) of uncorrupted drag accelerations must be input to compute the probability of thruster firings during the orbit. This is the same file as that output by the Drag Accelerations Program previously described.

The data cards are grouped in sections, which can be arranged in any order in the deck. All card identifiers are left-justified, while the data words are right-justified in the field. Table 2 describes the card format and Table 3 describes input parameters.

Table 2. HV Vibrations Card Input Format

Word	Format	Columns
1	A10	1-10
2	E15.9	11-25
3	E15.9	26-40
4	E15.9	41-55
5	E15.9	56-70

Table 3. HV Vibrations Card Input

Section Identifier (Cols 1-10)	Card #	Item #1 (Cols 11-25)	Item #2 (Cols 26-40)	Item #3 (Cols 41-55)	Item #4 (Cols 56-70)
NOMTLBTEST	1	Number of 0.05 sec time lines between tests for a thrust	–	–	–
PROBCALINT	1	Interval (in sec) for re-calculation of thrust probability	–	–	–
TBTFAPO	1	Seconds between thrusts at apogee for 1^{ST} pair of roll thrusters	Seconds between thrusts at apogee for 2^{ND} pair of roll thrusters	Seconds between thrusts at apogee for the pair of pitch thrusters	Seconds between thrusts at apogee for the pair of yaw thrusters
APOACCELS	1	–	–	–	–
	2	Number (NA) of apogee passages during the orbit time span	–	–	–
	3	Time of 1^{ST} apogee passage from epoch	x-axis drag acceleration (in km/sec^2) at that time	–	–
	.				
	.				
	.				
	NA+2	Time of NA^{TH} apogee passage in sec from epoch	x-axis drag acceleration (in km/sec^2) at that time	–	–
TBTFPERI	1	Seconds between thrusts at perigee for 1^{ST} pair of roll thrusters	Seconds between thrusts at perigee for 2^{ND} pair of roll thrusters	Seconds between thrusts at perigee for the pair of pitch thrusters	Seconds between thrusts at perigee for the pair of yaw thrusters

Table 3. HV Vibrations Card Input (Continued)

Section Identifier (Cols 1-10)	Card #	Item #1 (Cols 11-25)	Item #2 (Cols 26-40)	Item #3 (Cols 41-55)	Item #4 (Cols 56-70)
PERACCELS	1	–	–	–	–
	2	Number (NP) of perigee passages during the orbit time span	–	–	–
	3 . . .	Time of 1^{ST} perigee passage in sec from epoch	x-axis drag acceleration (in km/sec^2) at that time	–	–
	NP+2	Time of NP^{TH} perigee passage in sec from epoch	x-axis drag acceleration (in km/sec^2) at that time	–	–
FREQUENCY	1	–	–	–	–
	2 . . .	Resonance frequency #1 (Hz) for 1^{ST} pair of roll thrusters	Resonance frequency #1 (Hz) for 2^{ND} pair of roll thrusters	Resonance frequency #1 (Hz) for the pair of pitch thrusters	Resonance frequency #1 (Hz) for the pair of yaw thrusters
	12	Resonance frequency #11 (Hz) for 1^{ST} pair of roll thrusters	Resonance frequency #11 (Hz) for 2^{ND} pair of roll thrusters	Resonance frequency #11 (Hz) for the pair of pitch thrusters	Resonance frequency #11 (Hz) for the pair of yaw thrusters
AMPLITUDES	1	–	–	–	–
	2 . . . 12	Amplitude (km/sec^2) of resonance on corresponding "FREQUENCY" section card	Amplitude (km/sec^2) of resonance on corresponding "FREQUENCY" section card	Amplitude (km/sec^2) of resonance on corresponding "FREQUENCY" section card	Amplitude (km/sec^2) of resonance on corresponding "FREQUENCY" section card

Table 3. HV Vibrations Card Input (Continued)

Section Identifier (Cols 1-10)	Card #	Item #1 (Cols 11-25)	Item #2 (Cols 26-40)	Item #3 (Cols 41-55)	Item #4 (Cols 56-70)
DAMPING	1	–	–	–	–
	2 . . . 12	Damping constant of resonance on corresponding "FREQUENCY" section card	Damping constant of resonance on corresponding "FREQUENCY" section card	Damping constant of resonance on corresponding "FREQUENCY" section card	Damping constant of resonance on corresponding "FREQUENCY" section card
UM ARRAY	1	–	–	–	–
	2 . .(J) . 11	$(J\text{-}1)^{ST}$ term specifying the shape & magnitude of the initial thrust/1^{ST} pair of roll thrusters	$(J\text{-}1)^{ST}$ term specifying the shape & magnitude of the initial thrust/2^{ND} pair of roll thrusters	$(J\text{-}1)^{ST}$ term specifying the shape & magnitude of the initial thrust/the pair of pitch thrusters	$(J\text{-}1)^{ST}$ term specifying the shape & magnitude of the initial thrust/the pair of yaw thrusters
TIME LIMIT	1	Seconds from the epoch of the trajectory at which to terminate the program			

Output

The output file (TAPE3) is a time history of the thruster-induced vibrational accelerations. Table 4 shows the format for each record of the file.

Table 4. HV Vibrations Output File Format

Word	Description	Units	Format
1	Time from Epoch	sec	Floating Point
2	Vibrational Acceleration	km/sec^2	Floating Point

The printed output is arranged in four sections. First of all, the card input is printed out and labeled accordingly. The derived probability for a thrust is then printed at the specified interval for each thruster pair. The vehicle responses to each thruster are printed out at a 0.05-sec step size. Finally, the 2-sec averages of the aggregate vehicle responses over the orbit span are given. In addition, the number of thrusts per pair is given for the previous 2-sec period.

MERGE PROGRAM

Input

The input data card specifies the epoch of the accelerations. The format is shown in Table 5.

Table 5. Merge Input Card Format

Word	Description	Columns	Format
1	Year	1-10	F10.0
2	Day	11-20	F10.0
3	Second	21-30	F10.0

Four binary files are input to this program. The first file (TAPE4) contains the true drag in the perpendicular local frame. This file is the same as that output by the "Drag Accelerations" version of Celest.

The remaining three files (TAPE1, TAPE2, and TAPE3) are produced by separate runs of the HV Vibrations Program. For each MESA axis, there is a file of thruster-induced vehicle vibrations. These files, of course, have the same format as that output by HV vibrations.

Output

The time and corresponding corrupted accelerations (km/sec^2) are printed out at each 2-sec interval. If the corresponding times on two files are not equivalent, an error message is printed out and the program is terminated. The same action is taken when an end-of-file is reached on one of the input files.

The output file (TAPE5) contains the true drag corrupted by the host vehicle vibrations. The accelerations are in the perpendicular local frame in units of km/sec^2.

The first record has three words describing the epoch (year, day, second) of the accelerations. All time tags on the file are referenced to this epoch.

Each subsequent record has the format described in Table 6. The first word is an integer specifying the number (N) of pertinent observations in that record. Although there are always 1000 observations per record, only N of them are valid. Each observation consists of four words: a time tag, and the x-, y-. amd z-axis accelerations.

Table 6. Merge Output File Format

Word	Value		Units	Type
1	N		--	Integer
2	Time of Observation	1	Seconds	Floating Point
3	A_x		km/sec^2	Floating Point
4	A_y		km/sec^2	Floating Point
5	A_z		km/sec^2	Floating Point
6	Time of Observation	2	Seconds	Floating Point
7	A_x		km/sec^2	Floating Point
8	A_y		km/sec^2	Floating Point
9	A_z		km/sec^2	Floating Point
.	.			
.	.			
.	.			
3998	Time of Observation	1000	Seconds	Floating Point
3999	A_x		km/sec^2	Floating Point
4000	A_y		km/sec^2	Floating Point
4001	A_z		km/sec^2	Floating Point

GAUSS-MARKOV PROGRAM

Input

The input file (TAPE1) is the same as that output by the Merge Program. The format was specified earlier.

Every input card must be labeled with the appropriate heading, left-justified in Column 1. For all 17 cards, the format is (A10, 3E20.14). They are described in Table 7.

Table 7. Gauss-Markov Card Input

Section	Card Label	Columns	Item	Units
General	INTINT	11-30	Δt	sec
	TIMELIM	11-30	Time at which to terminate program execution	sec
$\vec{e}_1$	BIASX	11-30	μ_{b_x}	g
		31-50	$\sigma^2_{b_x}$	g^2
	BIASY	11-30	μ_{b_y}	g
		31-50	$\sigma^2_{b_y}$	g
	BIASZ	11-30	μ_{b_z}	g
		31-50	$\sigma^2_{b_z}$	g
	SCAFAC1	11-30	S(1,1)	g
		31-50	S(1,2)	g
		51-70	S(1,3)	g
	SCAFAC2	11-30	S(2,1)	g
		31-50	S(2,2)	g
		51-70	S(2,3)	g
	SCAFAC3	11-30	S(3,1)	g
		31-50	S(3,2)	g
		51-70	S(3,3)	g
$\vec{e}_2$	TAUM	11-30	τ_{m_x}	sec
		31-50	τ_{m_y}	sec
		51-70	τ_{m_z}	sec
	SIGMAM	11-30	σ_{m_x}	–
		31-50	σ_{m_y}	–
		51-70	σ_{m_z}	–
$\vec{e}_3$	TAUETA	11-30	τ_{η_x}	sec
		31-50	τ_{η_y}	sec
		51-70	τ_{η_z}	sec
	SIGMAETA	11-30	σ_{η_x}	–
		31-50	σ_{η_y}	–
		51-70	σ_{η_z}	–

Table 7. Gauss-Markov Card Input (Continued)

Section	Card Label	Columns	Item	Units
$\vec{e}_3$ (Continued)	TAUXI	11-30	τ_{ξ_x}	sec
		31-50	τ_{ξ_y}	sec
		51-70	τ_{ξ_z}	sec
	PXI1	11-30	$P_\xi(1,1)$	g^2
		31-50	$P_\xi(1,2)$	g^2
		51-70	$P_\xi(1,3)$	g^2
	PXI2	11-30	$P_\xi(2,1)$	g^2
		31-50	$P_\xi(2,2)$	g^2
		51-70	$P_\xi(2,3)$	g^2
	PXI3	11-30	$P_\xi(3,1)$	g^2
		31-50	$P_\xi(3,2)$	g^2
		51-70	$P_\xi(3,3)$	g^2
$\vec{e}_4$	SIGMAW	11-30	σ_{ω_x}	g
		31-50	σ_{ω_y}	g
		51-70	σ_{ω_z}	g

Output

The format of the output file (TAPE2) is the same as that produced by the Merge Program. The accelerations are written in units of km/sec^2. As the end product of the Data Generator, this file of data is ready for filtering.

Each data card is printed exactly as it was punched. In the case of an error, an appropriate message is given. The elements of the time-invariant input and computed matrices are then specified. (See Table 8.)

Finally, the time history of the input and output accelerations is given with the corresponding time tags. The data is specified in units of g's for three coordinates.

Table 8. Gauss-Markov Printed Output

Term	Label	Matrix Represented
$\vec{e}_1$	B	b
	S	S
$\vec{e}_2$	RM	R_m
	SM	S_m
	IRMSM	$(I - R_m^2)^{1/2} S_m$
$\vec{e}_3$		
$\vec{\eta}$	RETA	R_η
	SETA	S_η
	IRESE	$(I - R_\eta^2)^{1/2} S_\eta$
$\vec{\xi}$	PXI	P_ξ
	CXI	C_ξ
	BXI	B_ξ
$\vec{e}_4$	SW	S_ω

SAMPLE CASE: NOMINAL CORRUPTIONS

DRAG ACCELERATIONS

The first step was to compute a low-altitude inertial trajectory using a 25 X 25 gravity field. With one iteration per time line, a tenth-order integration was effected at a 2-sec step size. The drag accelerations derived for this orbit were saved for future use.

The drag model implemented was the standard Celest[1] version. It may be noted that there are discontinuities in the plots of the true drag accelerations. These anomalies mark the boundaries of the drag segments applied.

HV VIBRATIONS

A preliminary study was made of the structure of the host vehicle and its solar arrays. Assuming a nonrigid body, six resonance frequencies were determined to be below 10 Hz.[2] These resonances were used for the responses to the pitch thruster pair. For simplicity, this was the only pair allowed to fire.

The amplitude of the sensed acceleration for each resonance was assumed to be 1 μg. To obtain the respective MESA output amplitudes, the instrument attenuation for each frequency was multiplied by this value. The vehicle damping was 3 percent of critical. In Table 9, the actual input values are listed. The thrust impulse itself was assumed to be instantaneous. Therefore, $U(1) = 1$ was the only nonzero element of U.

The time interval between thrusts at apogee was 30 sec, while that at perigee was 10 sec. The thrust probability was recomputed every 200 sec. To minimize aliasing, the check for a thrust was made every 1.05 sec. The same resultant vibrations were added to all three axes.

Table 9. Nominal Case Resonances

Frequency (Hz)	Amplitude (g's)	Damping Constant
0.102	0.97×10^{-6}	$0.306137790 \times 10^{-2}$
0.591	0.26×10^{-6}	$0.177379839 \times 10^{-1}$
1.125	0.54×10^{-7}	$0.337651978 \times 10^{-1}$
1.787	0.15×10^{-7}	$0.536341408 \times 10^{-1}$
2.488	0.64×10^{-8}	$0.746736107 \times 10^{-1}$
3.847	0.16×10^{-8}	$0.115461970 \times 10^{0}$

GAUSS-MARKOV

A bias of 0.1 μg was applied to each of the three axes. The scale factor was set to zero. For the time-dependent orientation error, the standard deviation was 0.15° for each axis. The correlation time was one-fourth of a revolution or 1350 sec.

No additive correlated noise was applied. However, the white noise level was 5×10^{-9} g on all axes.

RESULTS

As expected, the out-of-plane components of the drag were much smaller than the along-track component. Hence, the plots for the y and z axes were magnified to expose their finer structure. The nominal case accelerations were submitted to a Fourier filter, which successfully recovered the true drag. Plots of the simulated MESA data are attached (Figures 2-9).

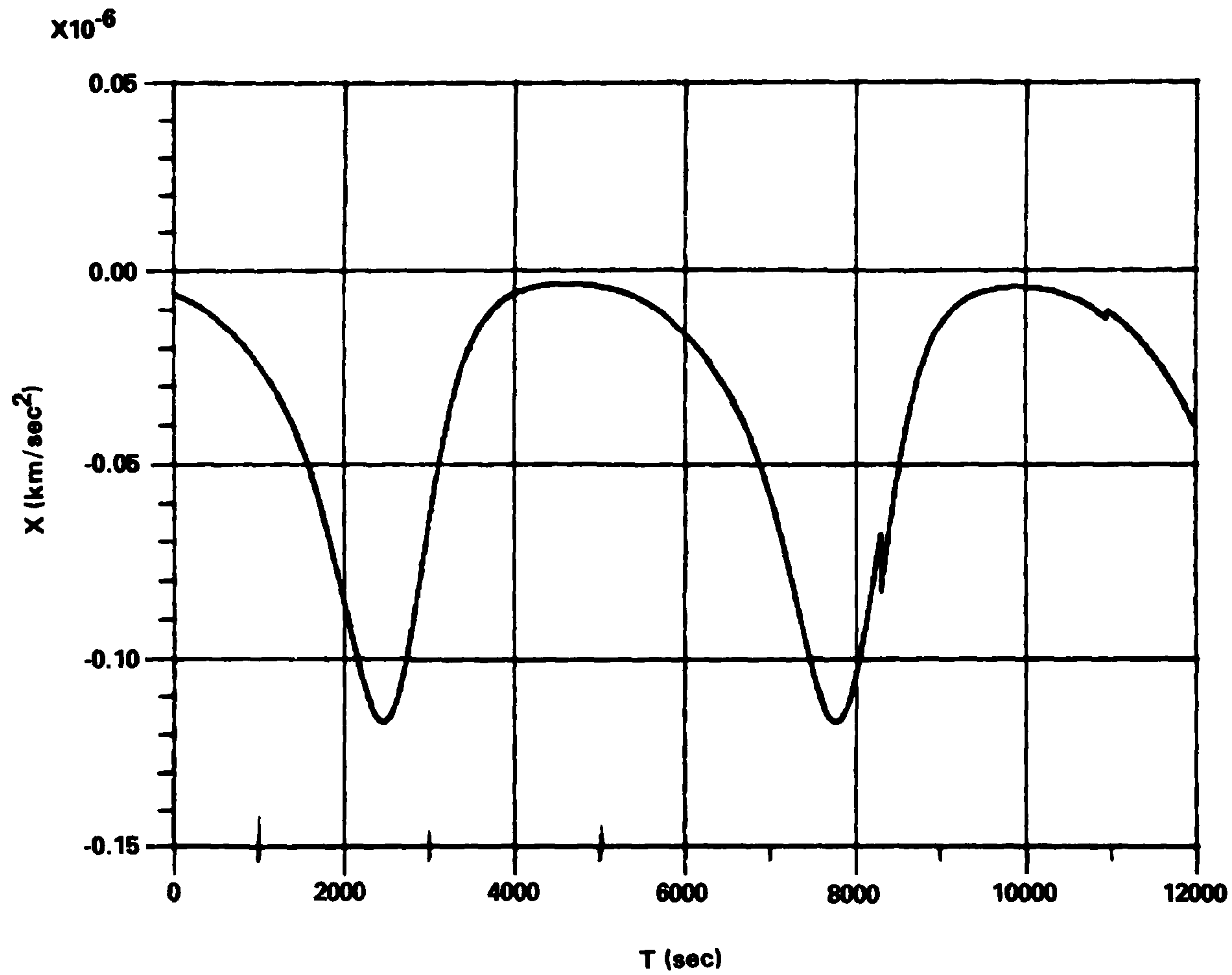

Figure 2. Sample Case: Along-Track Drag Accelerations

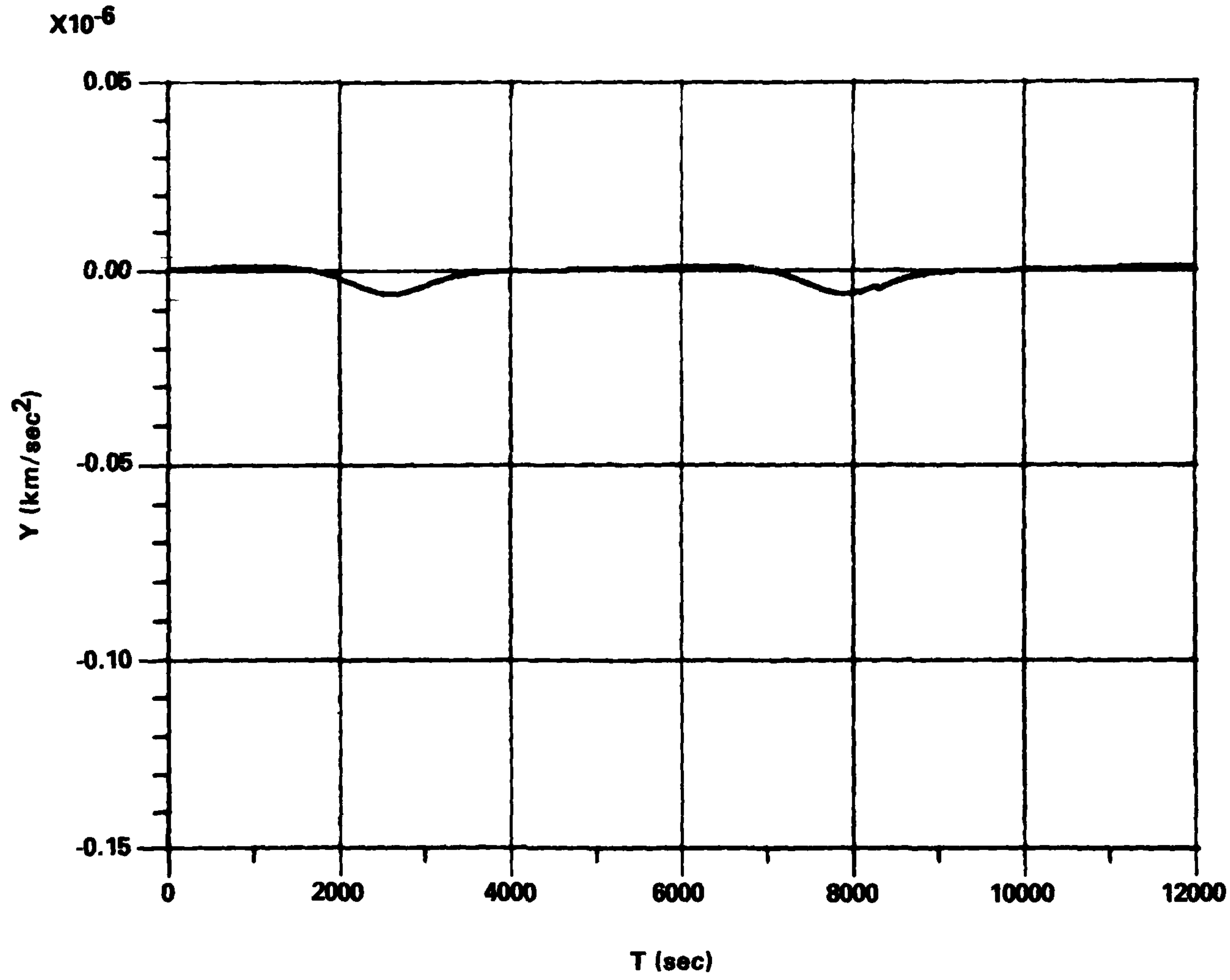

Figure 3. Sample Case: Cross-Track Drag Accelerations

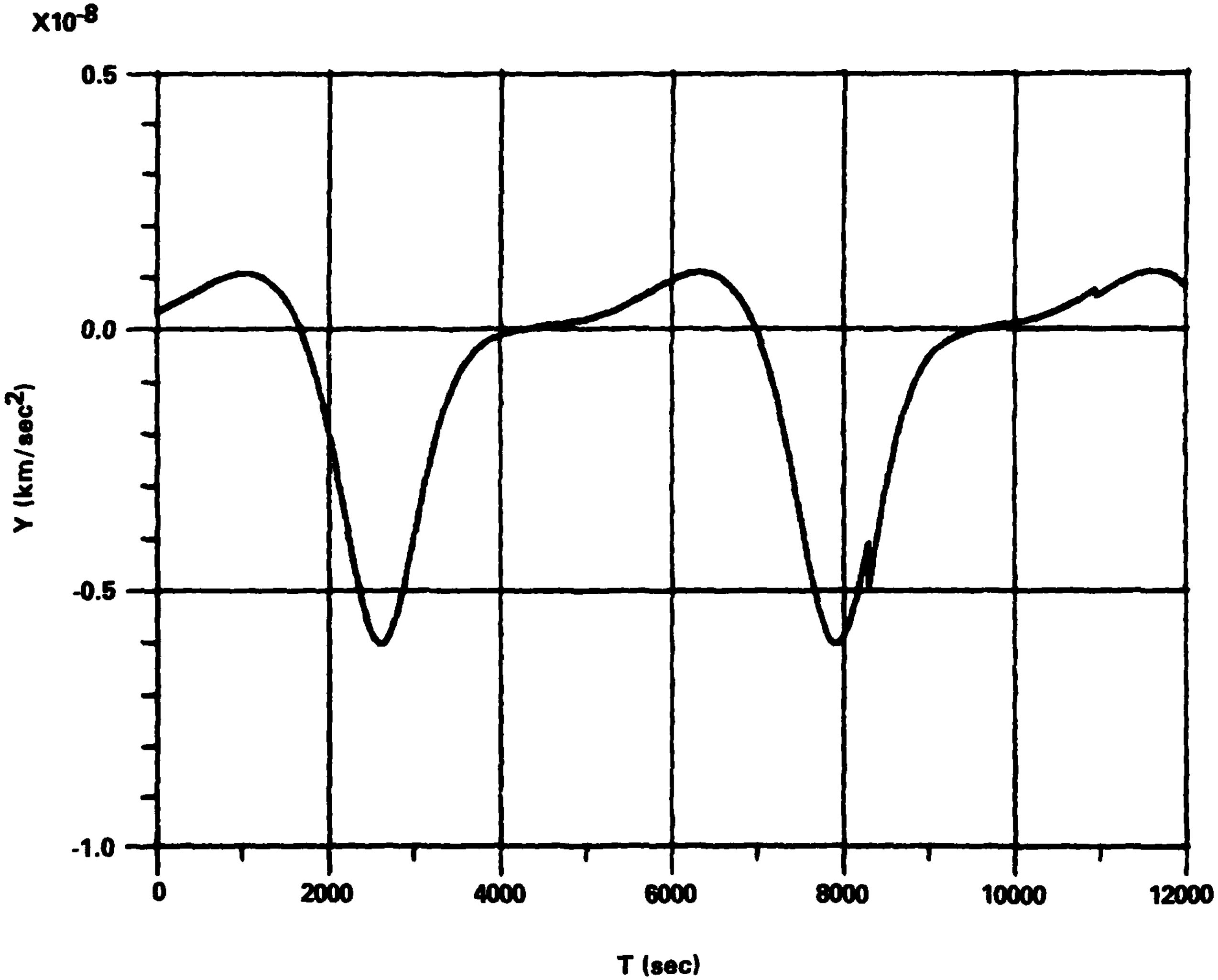

Figure 4. Sample Case: Cross-Track Drag Accelerations Magnified

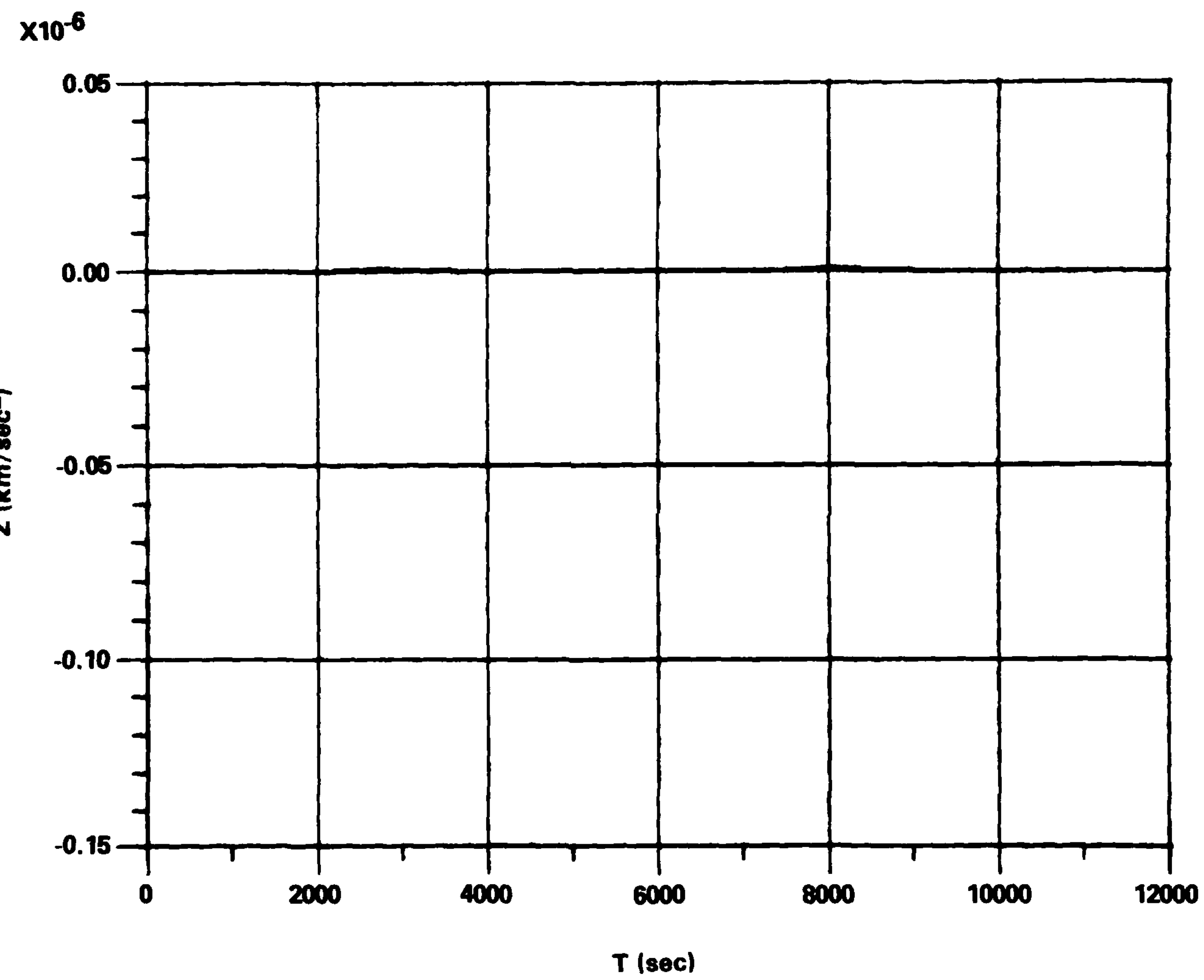

Figure 5. Sample Case: Radial Drag Accelerations

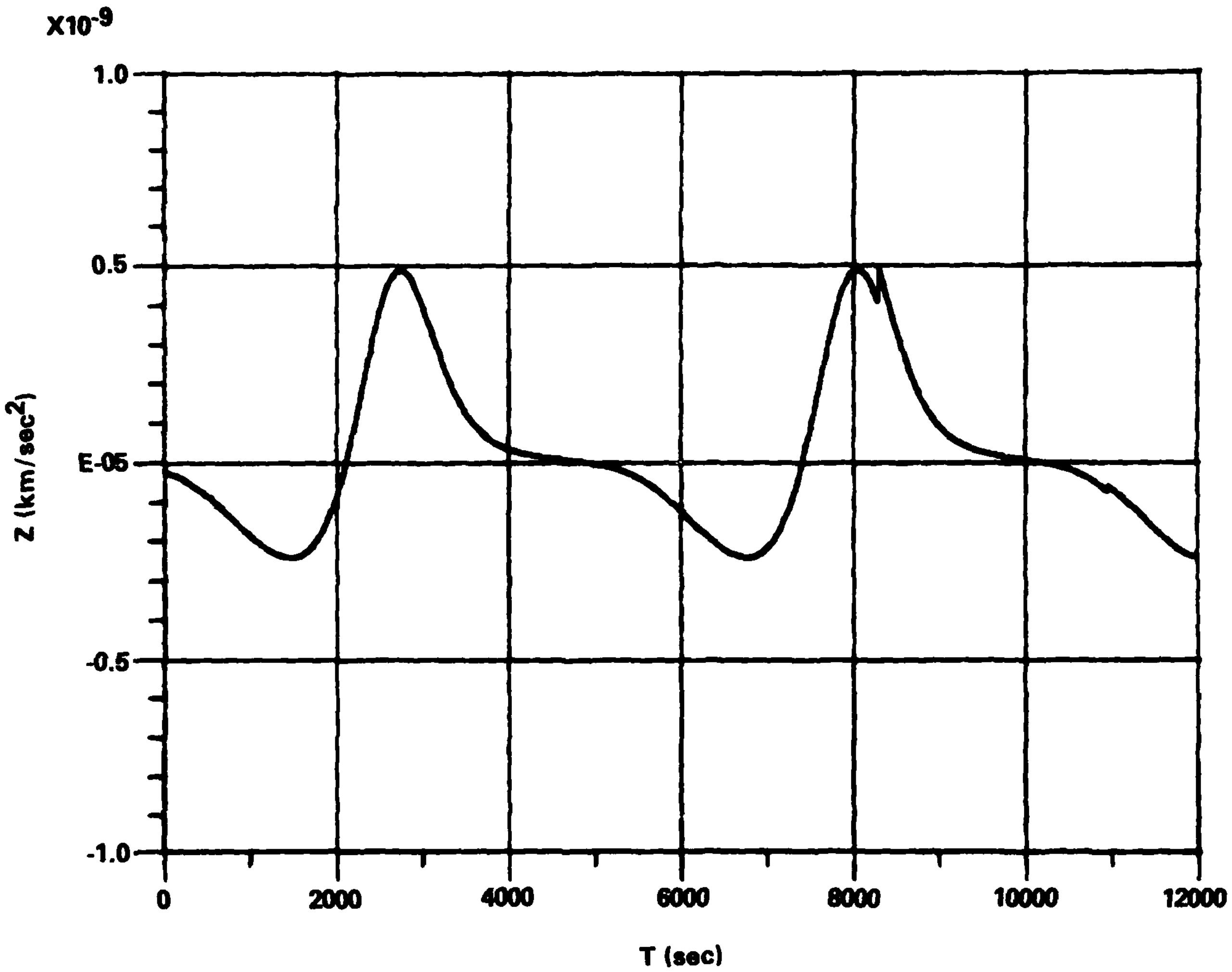

Figure 6. Sample Case: Radial Drag Accelerations Magnified

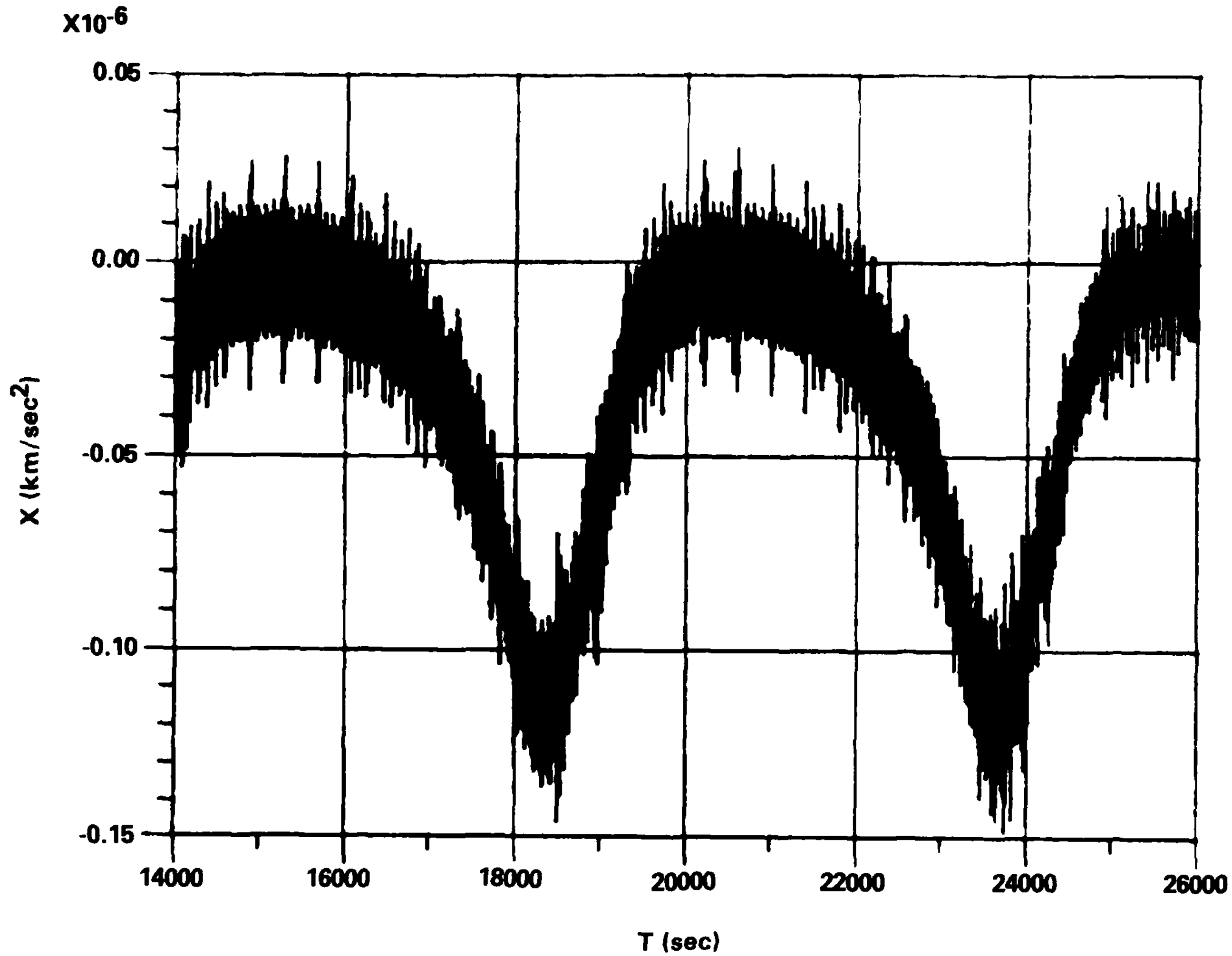

Figure 7. Sample Case Simulated Data: Along-Track

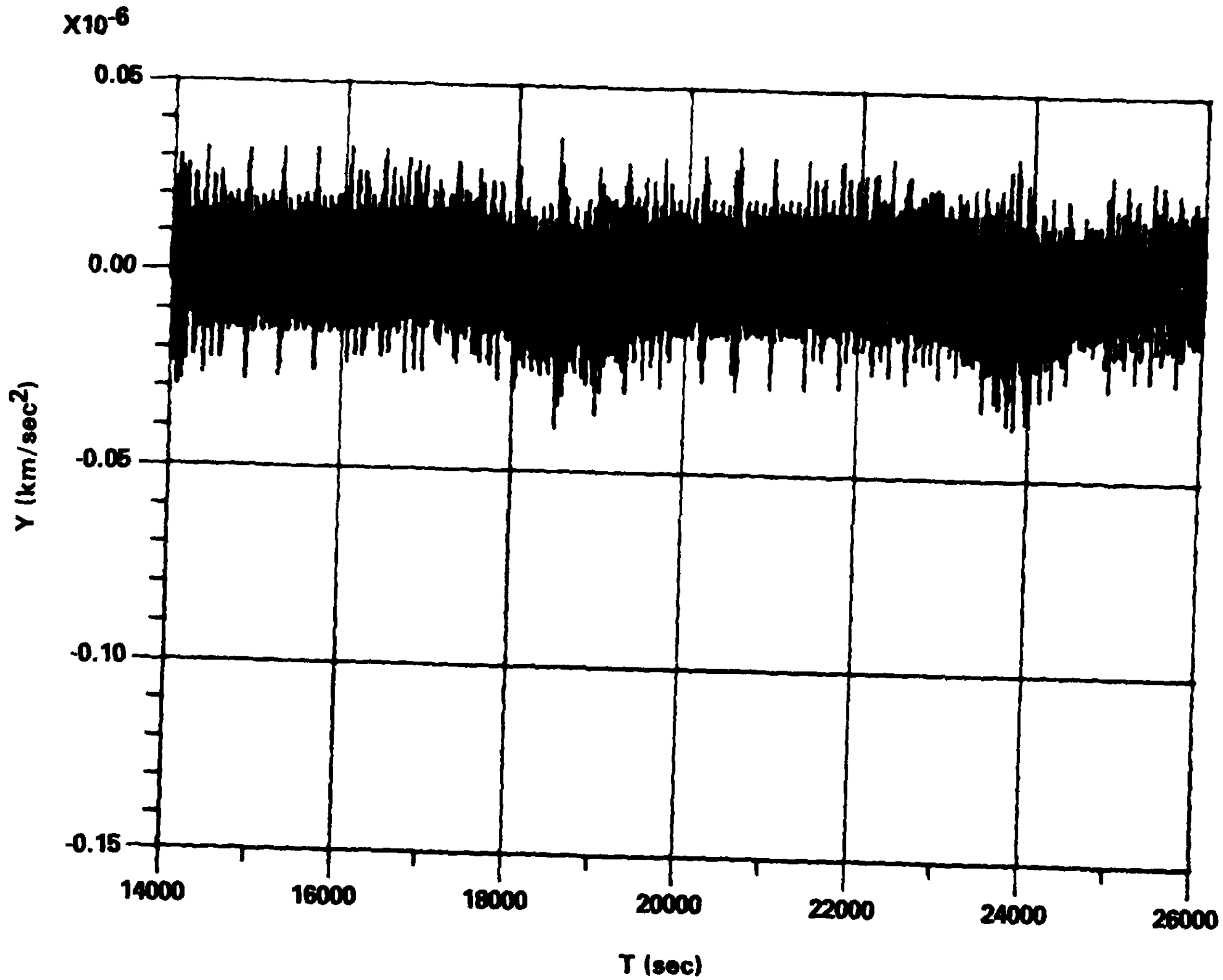

Figure 8. Sample Case Simulated Data: Cross-Track

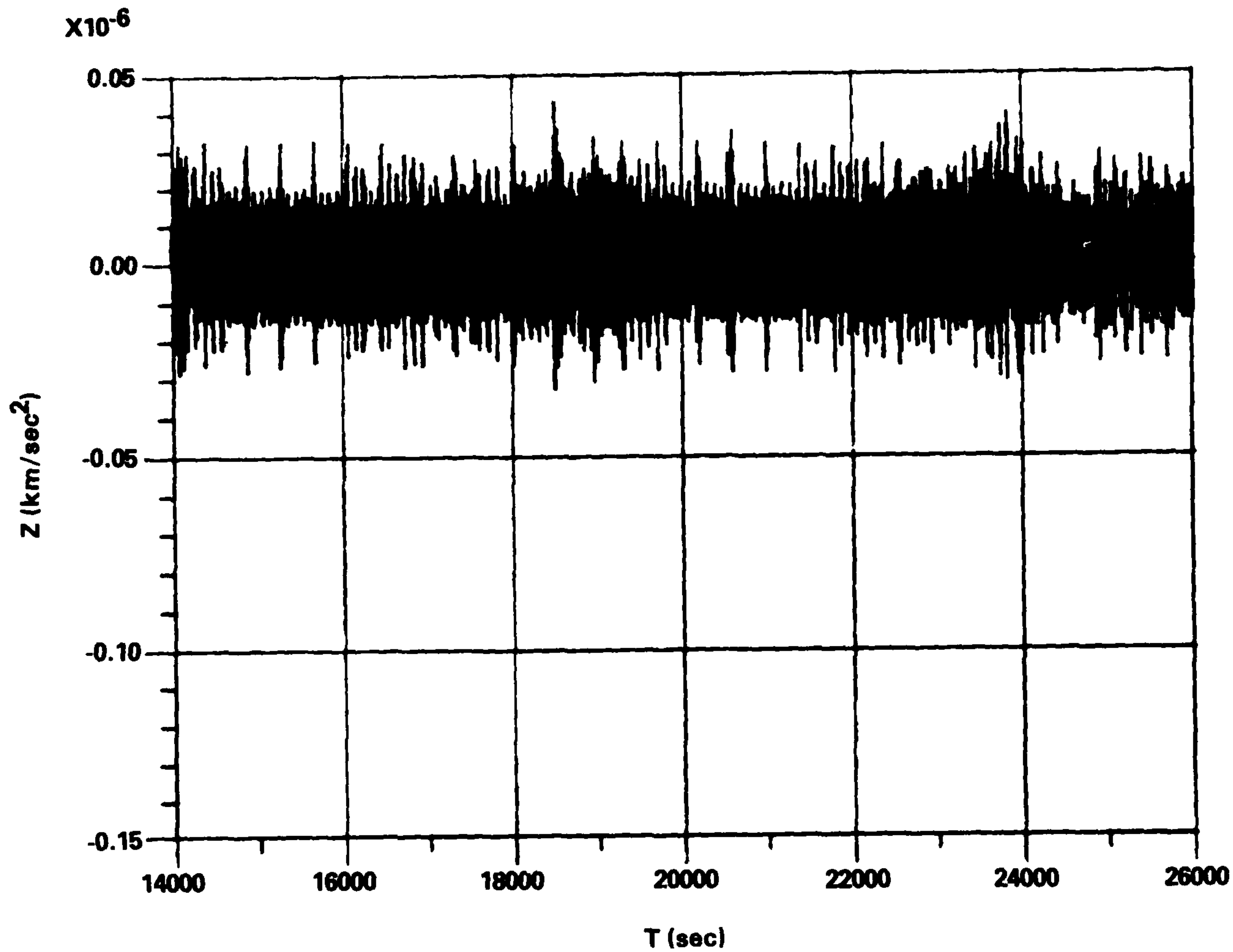

Figure 9. Sample Case Simulated Data: Radial

SUMMARY

A sample case was run simulating data with nominal corruptions. Both vibration-induced accelerations and Gauss-Markov modeled noise were added to the drag computed in Celest. The simulated data appears quite realistic. In fact, it closely resembles the real data received from the first orbiting three-axis MESA. A Fourier filter removed virtually all of the ocrruptions from the true drag accelerations. The details of the runs are described in the section of this report entitled "Sample Case: Nominal Corruptions"(p.25).

The "Data Generator" is quite versatile in that vibration-induced accelerations and Gauss-Markov noise can be added independently and, if desired, exclusively of one another. One can add corruptions to any combination of axes.

In the Host Vehicle Vibrations Program, 11 inherent vehicle resonances can be designated. These vibrational frequencies can be initiated by any one of four pairs of thrusters. The amplitude and damping constants of the resonances are different for each thruster. In addition, an initial thrust impulse can be superimposed on each of the excited vibrational modes. The size and duration of this impulse can be specified.

Up to five types of first-order Gauss-Markov noise may be added to the data. The noise may take the form of a bias or scale-factor induced anomaly, a time-dependent orientation error, on-axis and cross-axis additive correlated noise, or white noise. This model can therefore emulate an instrument bias on each axis, digital-to-engineering unit conversion errors, vehicle attitude errors, temperature variation effects, and uniformly distributed random noise.

FORTRAN LISTINGS

DRAG ACCELERATIONS PROGRAM

```
*IDENT ACCEL
*INSERT ORBGEN.3
      COMMON/ACCELS/IGO,TT1,TT2,AC1,AC2
*INSERT ORBGEN.29
      IGO=0
      PRINT 3345
 3345 FORMAT (1H1)
*COMPILE ORBGEN
*INSERT ATDRAG.3
      DIMENSION AINERT(3,1),APERP(3,1)
      REAL ITOP(3,3)
      COMMON/ACCELS/IGO,TT1,TT2,AC1,AC2
*INSERT ATDRAG.137
*                                                                      *
*                       OMIT DUPLICATE TIME LINES                      *
*                                                                      *
      IF (IGO.EQ.3) GO TO 905
      IF (T1.EQ.0.) GO TO 906
      IF (T1.LT.20.) GO TO 909
      IGO=IGO+1
      GO TO 909
*                                                                      *
*     DERIVE MATRIX FOR CONVERSION FROM INERTIAL TO LOCAL FRAME        *
*                                                                      *
  905 RMAG=SQRT(X(1)*X(1)+X(2)*X(2)+X(3)*X(3))
      VMAG=SQRT(XD(1)*XD(1)+XD(2)*XD(2)+XD(3)*XD(3))
      VXRX=X(3)*XD(2)-X(2)*XD(3)
      VXRY=X(1)*XD(3)-X(3)*XD(1)
      VXRZ=X(2)*XD(1)-X(1)*XD(2)
      VXRMAG=SQRT(VXRX*VXRX+VXRY*VXRY+VXRZ*VXRZ)
      RXVMAG=VXRMAG
      COSGAM=(X(1)*XD(1)+X(2)*XD(2)+X(3)*XD(3))/(RMAG*VMAG)
      SINGAM=RXVMAG/(RMAG*VMAG)
      COTGAM=COSGAM/SINGAM
      CSCGAM=1./SINGAM
      AINERT(1,1)=UX
      AINERT(2,1)=UY
      AINERT(3,1)=UZ
      ITOP(1,1)=XD(1)*CSCGAM/VMAG-X(1)*COTGAM/RMAG
      ITOP(1,2)=XD(2)*CSCGAM/VMAG-X(2)*COTGAM/RMAG
      ITOP(1,3)=XD(3)*CSCGAM/VMAG-X(3)*COTGAM/RMAG
      ITOP(2,1)=VXRX/VXRMAG
      ITOP(2,2)=VXRY/VXRMAG
      ITOP(2,3)=VXRZ/VXRMAG
      ITOP(3,1)=-1.*X(1)/RMAG
      ITOP(3,2)=-1.*X(2)/RMAG
      ITOP(3,3)=-1.*X(3)/RMAG
```

```
*                                                                              *
*         CONVERT INERTIAL ACCELERATIONS TO LOCAL FRAME                        *
*                                                                              *
      CALL MPROD (3,3,1,ITOP,3,AINERT,3,APERP,3)
      WRITE (51) TI,(APERP(I,1),I=1,3),(X(I),I=1,3),(XD(I),I=1,3)
*                                                                              *
*                    DETERMINE APOGEE AND PERIGEE                              *
*                                                                              *
      IF (TI.GT.2.) GO TO 3333
      IF (TI.EQ.2.) GO TO 3331
      TT1=0.  $  AC1=APERP(1,1)  $  GO TO 909
 3331 TT2=2.  $  AC2=APERP(1,1)  $  GO TO 909
 3333 IF (AC2.LT.AC1.AND.AC2.LT.APERP(1,1)) PRINT 111, TT2,AC2
      IF (AC2.GT.AC1.AND.AC2.GT.APERP(1,1)) PRINT 112, TT2,AC2
      TT1=TT2  $  AC1=AC2
      TT2=TI   $  AC2=APERP(1,1)
      GO TO 909
  111 FORMAT (1H0,5X,*PERIGEE -- TIME = *,F5.0,10X,*ACCELERATION = *,
     1E20.14)
  112 FORMAT (1H0,5X,*APOGEE -- TIME = *,F6.0,10X,*ACCELERATION = *,
     1E20.14)
  909 CONTINUE
*COMPILE ATDRAG
```

HOST VEHICLE VIBRATIONS PROGRAM

```
      PROGRAM HVVIBE (INPUT,OUTPUT,TAPE1=INPUT,TAPE2,TAPE3,TAPE4)
      DIMENSION MAXS(4),SUM(4,40,251)
      DIMENSION AMP(4,11),DAMP(4,11),FREQ(11),UM(4,10)
      DIMENSION T(20),AC(20),THR1(20),THR2(20),THP(20),THY(20),MR1(22)
      DIMENSION MR2(22),MP(22)
      DIMENSION MY(22),BR1(22),BR2(22),BP(22),BY(22),TTEST(22),ACC(3,1)
      DIMENSION RESP(10000),MAX(4)
      DIMENSION PERTI(10),PERAC(10),APOTI(10),APOAC(10)
      REAL NOMTLBT
*                                                                       *
*                            READ CARD INPUT                            *
*                                                                       *
      DO 5 I=1,11
      READ (1,10) CARDID,CONST1,CONST2,CONST3,CONST4
   10 FORMAT (A10,E15.9,E15.9,E15.9,E15.9)
      IF (CARDID.NE.10HNOMTLBTEST) GO TO 100
      NOMTLBT=CONST1     $ GO TO 5
  100 IF (CARDID.NE.10HAMPLITUDES) GO TO 101
      DO 15 J=1,11
   15 READ (1,10) CARDID,(AMP(JJJ,J),JJJ=1,4)
      GO TO 5
  101 IF (CARDID.NE.10HDAMPING   ) GO TO 102
      DO 16 K=1,11
   16 READ (1,10) CARDID,(DAMP(KKK,K),KKK=1,4)
      GO TO 5
  102 IF (CARDID.NE.10HPROBCALINT) GO TO 103
      PROBINT=CONST1    $ GO TO 5
  103 IF (CARDID.NE.10HTBTFAPO   ) GO TO 104
      TBTFAR1=CONST1  $  TBTFAR2=CONST2  $  TBTFAP=CONST3
      TBTFAY=CONST4  $  GO TO 5
  104 IF (CARDID.NE.10HTBTFPERI  ) GO TO 105
      TBTPR1=CONST1  $  TBTFPR2=CONST2  $  TBTFPP=CONST3
      TBTFPY=CONST4  $  GO TO 5
  105 IF (CARDID.NE.10HPERACCELS ) GO TO 106
      READ (1,10) CARDID,PN
      NP=PN
      DO 70 M=1,NP
   70 READ (1,10) CARDID,PERTI(M),PERAC(M)
      GO TO 5
  106 IF (CARDID.NE.10HAPOACCELS ) GO TO 107
      READ (1,10) CARDID,AN
      NA=AN
      DO 71 M=1,NA
   71 READ (1,10) CARDID,APOTI(M),APOAC(M)
      GO TO 5
  107 IF (CARDID.NE.10HFREQUENCY ) GO TO 108
      DO 263 JJJ=1,11
  263 READ (1,10) CARDID,FREQ(JJJ)
      GO TO 5
  108 IF (CARDID.NE.10HUM ARRAY  ) GO TO 109
      DO 264 JJJ=1,10
  264 READ (1,10) CARDID,(UM(III,JJJ),III=1,4)
      GO TO 5
```

```
109 IF (CARDID.NE.10HTIME LIMIT) GO TO 110
    TIMELIM=CONST1
    GO TO 5
110 PRINT 3, CARDID
  3 FORMAT (1X,*ERROR ON DATA CARDS-----ID = *,A10)
1666 STOP "ERROR ON DATA CARDS"
  5 CONTINUE
*                                                              *
*                    PRINT CARD INPUT                          *
*                                                              *
    PRINT 200
    PRINT 901, NCMTLBT
901 FORMAT (5X,*NO. OF TIME LINES BETWEEN THRUSTER CHECKS - *,F5.0)
    PRINT 911, ((I,FREQ(I)),I=1,11)
911 FORMAT (///,5X,*FREQUENCIES*,/,(/,5X,I2,10X,E15.9))
    PRINT 903
903 FORMAT (///,5X,*AMPLITUDES*,//)
    DO 634 JJJ=1,4
634 PRINT 902,JJJ,(AMP(JJJ,J),J=1,11)
902 FORMAT (1H0,5X,I1,3(/,5(5X,E15.9)))
    PRINT 904
904 FORMAT (///,5X,*DAMPING CONSTANTS*,//)
    DO 635 KKK=1,4
635 PRINT 902, KKK,(DAMP(KKK,K),K=1,11)
    PRINT 905, PROBINT
905 FORMAT (///,5X,*PROBABILITY CALCULATION INTERVAL - *,F6.0)
    PRINT 200
    PRINT 906
906 FORMAT (///,5X,*TIME BETWEEN THRUSTER FIRINGS*)
    PRINT 907, TBTFAR1,TBTFAR2,TBTFAP,TBTFAY
907 FORMAT (/,5X,*APOGEE*,/,
   15X,*ROLL1 - *,F6.0,10X,*ROLL2 - *,F6.0,10X,*PITCH - *,F6.0,10X,
   2*YAW - *,F6.0)
    PRINT 908, TBTFPR1,TBTFPR2,TBTFPP,TBTFPY
908 FORMAT (/,5X,*PERIGEE*,/,
   15X,*ROLL1 - *,F6.0,10X,*ROLL2 - *,F6.0,10X,*PITCH - *,F6.0,10X,
   2*YAW - *,F6.0)
    PRINT 909
909 FORMAT (///,5X,*ACCELERATIONS AT PERIGEE*,//)
    DO 5050 I=1,NP
5050 PRINT 72, PERTI(I),PERAC(I)
 72 FORMAT (1H0,4X,*TIME = *,F6.0,10X,*ACCELERATION = *,E15.9)
    PRINT 910
910 FORMAT (///,5X,*ACCELERATIONS AT APOGEE*,//)
    DO 5025 I=1,NA
5025 PRINT 72, APOTI(I),APCAC(I)
    PRINT 200
    PRINT 912
912 FORMAT (///,5X,*UM ARRAY*,/)
    DO 640 III=1,4
640 PRINT 902, III,(UM(III,JJJ),JJJ=1,10)
    PRINT 913, TIMELIM
913 FORMAT (///,5X,*TIME LIMIT - *,F6.0)
```

```
************************************************************************
*                                                                      *
*          COMPUTE PROBABILITIES FOR THRUSTER FIRINGS                  *
*                                                                      *
      PRINT 200
  200 FORMAT (1H1)
      TMPTEST=.05+NOMTLBT*.05
*                                                                      *
*         DERIVE THRUST PROBABILITIES AT APOGEE AND PERIGEE            *
*                                                                      *
      ROLLMX1=TMPTEST/TBTFPR1
      ROLLMX2=TMPTEST/TBTFPR2
      ROLLMN1=TMPTEST/TBTFAR1
      ROLLMN2=TMPTEST/TBTFAR2
      PICHMAX=TMPTEST/TBTFPP
      PICHMIN=TMPTEST/TBTFAP
      YAWMAX=TMPTEST/TBTFPY
      YAWMIN=TMPTEST/TBTFAY
      K=0
      IF (PERTI(1).LT.APOTI(1)) GO TO 742
      DO 73 I=1,NA
      K=K+1
      T(K)=APOTI(I)
      AC(K)=APOAC(I)
      THR1(K)=ROLLMN1
      THR2(K)=ROLLMN2
      THP(K)=PICHMIN
      THY(K)=YAWMIN
      KMAX=K
      K=K+1
      IF (I.GT.NP) GO TO 743
      KMAX=K
      T(K)=PERTI(I)
      AC(K)=PERAC(I)
      THR1(K)=ROLLMX1
      THR2(K)=ROLLMX2
      THP(K)=PICHMAX
   73 THY(K)=YAWMAX
      GO TO 743
  742 DO 74 I=1,NP
      K=K+1
      T(K)=PERTI(I)
      AC(K)=PERAC(I)
      THR1(K)=ROLLMX1
      THR2(K)=ROLLMX2
      THP(K)=PICHMAX
      THY(K)=YAWMAX
      KMAX=K
      K=K+1
      IF (I.GT.NA) GO TO 743
      KMAX=K
      T(K)=APOTI(I)
      AC(K)=APOAC(I)
      THR1(K)=ROLLMN1
      THR2(K)=ROLLMN2
      THP(K)=PICHMIN
   74 THY(K)=YAWMIN
```

```
  745 KMAXP1=KMAX+1
      KMAXP2=KMAX+2
      KMAXM1=KMAX-1
      KMAXM2=KMAX-2
*                                                                        *
*  COMPUTE THRUST PROBABILITY VS DRAG ACCELERATION FOR EACH HALF REV     *
*                                                                        *
      IF (T(1).NE.0.) GO TO 7010
      DO 7000 I=1,KMAXM1
      J=I+1
      MR1(I)=(THR1(J)-THR1(I))/(AC(J)-AC(I))
      MR2(I)=(THR2(J)-THR2(I))/(AC(J)-AC(I))
      MP(I)=(THP(J)-THP(I))/(AC(J)-AC(I))
      MY(I)=(THY(J)-THY(I))/(AC(J)-AC(I))
      BR1(I)=THR1(J)-MR1(I)*AC(J)
      BR2(I)=THR2(J)-MR2(I)*AC(J)
      BP(I)=THP(J)-MP(I)*AC(J)
 7000 BY(I)=THY(J)-MY(I)*AC(J)
      MR1(KMAX)=MR1(KMAXM2)
      MR2(KMAX)=MR2(KMAXM2)
      MP(KMAX)=MP(KMAXM2)
      MY(KMAX)=MY(KMAXM2)
      BR1(KMAX)=THR1(KMAX)-MR1(KMAX)*AC(KMAX)
      BR2(KMAX)=THR2(KMAX)-MR2(KMAX)*AC(KMAX)
      BP(KMAX)=THP(KMAX)-MP(KMAX)*AC(KMAX)
      BY(KMAX)=THY(KMAX)-MY(KMAX)*AC(KMAX)
      MP=KMAXP1
      DO 7001 I=1,KMAX
 7001 TTEST(I)=T(I)
      TTEST(KMAXP1)=999999.
      GO TO 7050
 7010 DO 7020 I=2,KMAX
      J=I-1
      MR1(I)=(THR1(I)-THR1(J))/(AC(I)-AC(J))
      MR2(I)=(THR2(I)-THR2(J))/(AC(I)-AC(J))
      MP(I)=(THP(I)-THP(J))/(AC(I)-AC(J))
 7020 MY(I)=(THY(I)-THY(J))/(AC(I)-AC(J))
      MR1(1)=MR1(3)
      MR2(1)=MR2(3)
      MP(1)=MP(3)
      MY(1)=MY(3)
      MR1(KMAXP1)=MR1(KMAXM1)
      MR2(KMAXP1)=MR2(KMAXM1)
      MP(KMAXP1)=MP(KMAXM1)
      MY(KMAXP1)=MY(KMAXM1)
      DO 7021 I=1,KMAX
      BR1(I)=THR1(I)-MR1(I)*AC(I)
      BR2(I)=THR2(I)-MR2(I)*AC(I)
      BP(I)=THP(I)-MP(I)*AC(I)
 7021 BY(I)=THY(I)-MY(I)*AC(I)
      BR1(KMAXP1)=THR1(KMAX)-MR1(KMAXP1)*AC(KMAX)
      BR2(KMAXP1)=THR2(KMAX)-MR2(KMAXP1)*AC(KMAX)
      BP(KMAXP1)=THP(KMAX)-MP(KMAXP1)*AC(KMAX)
      BY(KMAXP1)=THY(KMAX)-MY(KMAXP1)*AC(KMAX)
      MP=KMAXP2
      TTEST(1)=0.   $  TTEST(KMAXP2)=999999.
      DO 7022 I=2,KMAXP1
```

```
      J=I-1
 7022 TTEST(I)=T(J)
 7050 PRINT 801
  801 FORMAT (10X,49H*** PROBABILITIES - THRUSTER FIRINGS PER SECOND *,
     12H**,///)
      TLAST=-2.
*                                                                      *
*               DETERMINE NUMBER OF PROBABILITY SEGMENTS               *
*                                                                      *
      KLIM=TIMELIM/PROBINT+1
*                                                                      *
*     COMPUTE PROBABILITY OF ROLL, PITCH, AND YAW THRUSTER FIRINGS     *
*                          FOR EACH INTERVAL                           *
*                                                                      *
      DO 12 K=1,KLIM
      KK=K+1
      RK=K
*                                                                      *
*            DETERMINE THE CENTER TIME LINE OF THE INTERVAL            *
*                                                                      *
      RCH=(PROBINT*(RK-.5))/2.
      RCK=AINT(RCH)
      IF ((RCH-RCK).GT..5) GO TO 163
      TIME=2.*RCK  $  GO TO 17
  163 TIME=2.*RCK+2.
   17 CONTINUE
      ISKIP=(TIME-TLAST)/2-1
      IF (ISKIP.EQ.0) GO TO 6343
*                                                                      *
*                   SKIP TO THE APPROPRIATE TIME LINE                  *
*                                                                      *
      DO 6342 J=1,ISKIP
      READ (2)
      IF (EOF(2)) 25,6342
 6342 CONTINUE
*                                                                      *
*                     READ THE X-AXIS ACCELERATION                     *
*                                                                      *
 6343 READ (2) TI,(ACC(I,1),I=1,3)
      IF (EOF(2)) 25,150
  150 IF (TIME.EQ.TI) GO TO 675
*                                                                      *
*            PRINT AN ERROR MESSAGE IF THE TIMES DONT MATCH            *
*                                                                      *
      PRINT 300, K,TIME,TI
  300 FORMAT (1X,*K = *,I5,1H,,10X,*TIME = *,F6.0,1H,,10X,*TI = *,F6.0)
 2666 STOP 2666
  675 ACCEL=ACC(1,1)
      TLAST=TIME
*                                                                      *
*       COMPUTE THE PROBABILITY OF ROLL, PITCH, AND YAW THRUSTS        *
*                           FOR THE INTERVAL                           *
*                                                                      *
      DO 47 I=1,MM
      II=I+1
      IF (TTEST(I).GT.TIME.OR.TIME.GT.TTEST(II)) GO TO 47
      PROBR1=BR1(I)+MR(I)*ACCEL
```

```
      PROBR2=BR2(I)+MR2(I)*ACCEL
      PROBP=BP(I)+MP(I)*ACCEL
      PROBY=BY(I)+MY(I)*ACCEL
      WRITE (4) PROBR1,PROBR2,PROBP,PROBY
      PRBR1=PROBR1/TMPTEST
      PRBR2=PROBR2/TMPTEST
      PRBP=PROBP/TMPTEST
      PRBY=PROBY/TMPTEST
      PRINT 800, K,TIME,PRBR1,PRBR2,PRBP,PRBY
  800 FORMAT (1X,I4,5X,*TIME = *,F6.0,1H,,5X,*ROLL1 = *,F8.5,1H,,5X,
     2*ROLL2 = *,F8.5,1H,,5X,
     2*PITCH = *,F8.5,1H,,5X,*YAW = *,F8.5)
      GO TO 12
   47 CONTINUE
 5666 STOP
   12 CONTINUE
      GO TO 27
   25 CONTINUE
   27 REWIND 2
      REWIND 4
*********************************************************************
      PRINT 200
      PRINT 803
  803 FORMAT (15X,24H***THRUSTER RESPONSES***,///)
      PI=ACOS(-1.)
      DO 21 I=1,4
   21 MAXS(I)=0
      DO 23 I=1,4
      DO 23 J=1,40
      DO 23 K=1,251
   23 SUM(I,J,K)=0.
*                                                                   *
*     COMPUTE THE MESA RESPONSES TO THE INDIVIDUAL THRUSTER INPULSES *
*                                                                   *
      DO 1111 I=1,4
      DO 514 JJ=1,10000
  514 RESP(JJ)=0.
      MXOLD=0
*                                                                   *
*          COMPUTE THE RESPONSES AT EACH RESONANCE FREQUENCY        *
*                                                                   *
      DO 1122 J=1,11
      RLMK1=0.  $  RLMK2=0.
      DO 1133 K=1,10000
      IF (K.GT.10) GO TO 45
      E1=AMP(I,J)*EXP(-.05*DAMP(I,J))*SIN(.1*PI*FREQ(J))*UM(I,K)
   45 E2=2.*EXP(-.05*DAMP(I,J))*COS(.1*PI*FREQ(J))*RLMK1
      IF (FREQ(J).NE.5.) GO TO 20
      E2=0.
   20 E3=-1.*EXP(-.1*DAMP(I,J))*RLMK2
      RLM=E1+E2+E3
      RLMK2=RLMK1
      RLMK1=RLM
      RESP(K)=RESP(K)+RLM
      MX=K
      ABRLM=ABS(RLM)
```

```
*                                                                        *
*   THE RESPONSE IS ASSUMED TO BE TOTALLY DAMPED BELOW 1.X10**-9 GS     *
*                                                                        *
      IF (ABRLM.LT.1.E-09) GO TO 1346
 1133 CONTINUE
      PRINT 418, RLMK1,I,J
  418 FORMAT (1H1,1X,*RLM ARRAY TOO SMALL----RLMK1 = *,E15.9,2X,/,1X,
     1*THRUSTER   *,I2,5X,*FREQUENCY   *,I2)
 3666 STOP 3666
*                                                                        *
*          COMPUTE THE NUMBER OF .05 SEC TIME LINES REQUIRED             *
*                   TO DAMP THE RESPONSE TO THE THRUST                   *
*                                                                        *
 1346 MAX(I)=MAX0(MX,MXOLD)
      MXOLD=MAX(I)
 1122 CONTINUE
      MXW=MAX(I)
      PRINT 850, I,(RESP(III),III=1,MXW)
  850 FORMAT (1H1,5X,I1,////,(/,6(5X,E15.9)))
*                                                                        *
*  DEPENDING UPON THE POSITION OF THE THRUST WITHIN A 2 SEC INTERVAL,   *
*                   AVERAGE THE APPROPRIATE RESPONSES                    *
*                  FOR EACH SUBSEQUENT 2 SEC TIME LINE                   *
*                                                                        *
      DO 60 IS=1,40
      ADD=0.  $  KKS=1
*                                                                        *
*              IS REFLECTS THE POSITION OF A THRUST IMPULSE              *
*                     WITHIN A TWO SECOND INTERVAL                       *
*                                                                        *
      DO 22 JS=1,IS
      ADD=ADD+RESP(JS)
      IF (JS.EQ.MAX(I)) GO TO 50
   22 CONTINUE
      SUM(I,IS,1)=ADD
      JS=IS
   40 ADD=0.
*                                                                        *
*     KKS COUNTS THE NUMBER OF TWO SECOND AVERAGES FOR THRUSTER I        *
*                                                                        *
      JJS=0  $  KKS=KKS+1
*                                                                        *
*        JS COUNTS THE NUMBER OF RESPONSES ADDED FOR THRUSTER I          *
*                                                                        *
   46 JS=JS+1
*                                                                        *
*               JJS COUNTS THE NUMBER OF RESPONSES ADDED                 *
*                  FOR THE CURRENT TWO SECOND INTERVAL                   *
*                                                                        *
      JJS=JJS+1
      ADD=ADD+RESP(JS)
      IF (JS.EQ.MAX(I)) GO TO 50
      IF (JJS.NE.40) GO TO 46
   50 SUM (I,IS,KKS)=ADD
      IF (JS.NE.MAX(I)) GO TO 40
   11 MAXS(I)=MAX0(MAXS(I),KKS)
   60 CONTINUE
```

```
1111 CONTINUE
     MAXS1=MAXS(1)
     MAXS2=MAXS(2)
     MAXS3=MAXS(3)
     MAXS4=MAXS(4)
     MX=MAX0(MAXS(1),MAXS(2),MAXS(3),MAXS(4))
     MXX=MX-1
***********************************************************************
*                                                                     *
*          COMPUTE THE AGGREGATE THRUST-INDUCED ACCELERATION          *
*             FOR EACH TWO-SEC TIME LINE OF THE TRAJECTORY            *
*                                                                     *
     PRINT 200
     PRINT 880
 880 FORMAT (15X,31H***VIBRATIONAL ACCELERATIONS***,///)
     NOMTL=NOMTL8T
     IROLL1=IROLL2=IPITCH=IYAW=0
     DO 518 II=1,MX
 518 RESP(II)=0.
     RNDSET=0.
     INT=-1
     IPT=-1
     TM=-.05
     IPIFST=-2
 446 INT=INT+1
     TM=TM+.05
     IF (TM.EQ.0.) GO TO 600
     IF (INT.LT.40) GO TO 4756
 600 RESP(1)=RESP(1)/40.
     PRINT 870, TM,RESP(1)
 870 FORMAT (5X,*TIME = *,F6.0,10X,*ACCELERATION = *,E15.9)
     PRINT 650, IROLL1,IROLL2,IPITCH,IYAW
 650 FORMAT (5X,*THRUSTER FIRINGS:  ROLL1 = *,I2,10X,*ROLL2 = *,I2,10X,
    1*PITCH = *,I2,10X,*YAW = *,I2,/)
*                                                                     *
*    CONVERT ACCELERATIONS TO UNITS OF KM/SEC2 AND WRITE ON TAPE3     *
*                                                                     *
     RESP(1)=RESP(1)*.00980621
     WRITE (3) TM,RESP(1)
     DO 3462 I=1,MXX
     J=I+1
3462 RESP(I)=RESP(J)
     RESP(MX)=0.
     IROLL1=IROLL2=IPITCH=IYAW=0
     INT=0
*                                                                     *
*                 CHECK IF TIME LIMIT HAS BEEN REACHED                *
*                                                                     *
4756 IF (TM.GT.TIMELIM) GO TO 4666
     IPT=IPT+1
     IF (TM.EQ.0.) GO TO 500
     IF (IPT.NE.NOMTL) GO TO 446
*                                                                     *
*                       CHECK FOR THRUSTER FIRING                     *
*                                                                     *
 500 TI2=TM/2.
     TI=AINT(TI2)
```

```
*                                                                    *
*                NS REFLECTS THE TIME OF THE THRUST                  *
*        WITH REFERENCE TO THE OVERLYING TWO-SEC INTERVAL            *
*                                                                    *
      NS=20.*(2.-TM+TI*2.)
      IPI=TM/PROBINT+1
      IF (IPI.EQ.IPIFST) GO TO 333
      READ (4) PROBR1,PROBR2,PROBP,PROBY
      IPIFST=IPI
*                                                                    *
*            DETERMINE IF THE ROLL THRUSTERS HAVE FIRED.             *
*        IF SO, ADD THE APPROPRIATE ACCELERATIONS TO THE TOTAL       *
*           AT EACH OF THE SUBSEQUENT TWO-SEC TIME LINES.            *
*                                                                    *
  333 RNDSET=RNDSET+2.
      CALL RANSET(RNDSET)
      P=RANF(DUM)
      IF (P.GT.PROBR1) GO TO 66
      DO 64 I=1,MAXS1
   64 RESP(I)=RESP(I)+SUM(1,NS,I)
      IROLL1=IROLL1+1
   66 RNDSET=RNDSET+2.
      CALL RANSET(RNDSET)
      P=RANF(DUM)
      IF (P.GT.PROBR2) GO TO 65
      DO 67 I=1,MAXS2
   67 RESP(I)=RESP(I)+SUM(2,NS,I)
      IROLL2=IROLL2+1
*                                                                    *
*            DETERMINE IF THE PITCH THRUSTER HAS FIRED.              *
*        IF SO, ADD THE APPROPRIATE ACCELERATIONS TO THE TOTAL       *
*           AT EACH OF THE SUBSEQUENT TWO-SEC TIME LINES.            *
*                                                                    *
   65 RNDSET=RNDSET+2.
      CALL RANSET(RNDSET)
      P=RANF(DUM)
      IF (P.GT.PROBP) GO TO 343
      DO 342 I=1,MAXS3
  342 RESP(I)=RESP(I)+SUM(3,NS,I)
      IPITCH=IPITCH+1
*                                                                    *
*             DETERMINE IF THE YAW THRUSTER HAS FIRED.               *
*        IF SO, ADD THE APPROPRIATE ACCELERATIONS TO THE TOTAL       *
*           AT EACH OF THE SUBSEQUENT TWO-SEC TIME LINES.            *
*                                                                    *
  343 RNDSET=RNDSET+2.
      CALL RANSET(RNDSET)
      P=RANF(DUM)
      IF (P.GT.PROBY) GO TO 468
      DO 467 I=1,MAXS4
  467 RESP(I)=RESP(I)+SUM(4,NS,I)
      IYAW=IYAW+1
  468 IPT=-1
      GO TO 446
 4666 STOP 4666
      END
```

MERGE PROGRAM

```
      PROGRAM MERGE(INPUT,OUTPUT,TAPE1,TAPE2,TAPE3,TAPE4,TAPE5,
     1TAPE6=INPUT)
      DIMENSION ARRAY(4,1000),HV(3)
      PRINT 1
    1 FORMAT (1H1,10X,*TIME*,16X,*X*,24X,*Y*,24X,*Z*,////)
      DO 5 I=1,4
      DO 5 J=1,1000
    5 ARRAY(I,J)=0.
*                                                                    *
*                     WRITE THE HEADER ON TAPE5.                     *
*                                                                    *
      READ (6,10) YR,DAY,SEC
   10 FORMAT (3F10.0)
      WRITE (5) YR,DAY,SEC
   20 N=0
   30 N=N+1  $  IFILE=4
*                                                                    *
*          READ THE TRUE DRAG ACCELERATIONS OFF OF TAPE4.            *
*                                                                    *
      READ (4) (ARRAY(I,N),I=1,4)
      IF (EOF(4)) 70,40
*                                                                    *
*        READ THE CORRESPONDING THRUSTER-INDUCED ACCLERATIONS        *
*   ON THE X, Y, AND Z AXES OFF OF TAPES 1, 2, AND 3, RESPECTIVELY.  *
*                                                                    *
   40 DO 60 I=1,3
      IFILE=I  $  READ (I) T(I),HV(I)
      IF (EOF(I)) 70,50
   50 J=I+1
*                                                                    *
*              TEST TO INSURE THAT THE TIME TAGS MATCH.              *
*                                                                    *
      TEST=ARRAY(1,N)-T(I)
      IF (TEST.GE..05) GO TO 110
*                                                                    *
*  ADD THE THRUST-INDUCED VIBRATIONS TO THE TRUE DRAG ACCELERATIONS. *
*                                                                    *
   60 ARRAY(J,N)=ARRAY(J,N)+HV(I)
*                                                                    *
*       ONCE A FULL RECORD OF SIMULATED DATA HAS BEEN COMPUTED,      *
*                        WRITE IT ON TAPE 5.                         *
*                                                                    *
      IF (N.NE.1000) GO TO 30
      WRITE (5) N,ARRAY
      PRINT 80, ARRAY  $  GO TO 20
   70 N=N-1
      IF (N.EQ.0) GO TO 90
      WRITE (5) N,ARRAY
      PRINT 80, ((ARRAY(I,J),I=1,4),J=1,N)
   80 FORMAT (10X,F6.0,5X,E20.14,5X,E20.14,5X,E20.14)
```

```
 90 PRINT 100, IFILE
100 FORMAT (1H1,10X,*EOF ENCOUNTERED ON TAPE*,I1)
    STOP " EOF ENCOUNTERED"
110 PRINT 120, IFILE
120 FORMAT (1H1,5X,*TIMES ON TAPE*,I1,* AND TAPE4 DO NOT MATCH*)
    STOP "CORRESPONDING TIMES UNEQUAL"
    END
```

```
      PROGRAM GAUSMK (INPUT,OUTPUT,TAPE4=INPUT,TAPE1,TAPE2,TAPE3)
      DIMENSION A(17,4),RM(3,3),SM(3,3),PXI(3,3),RETA(3,3),SETA(3,3)
      DIMENSION WM(3,1),WETA(3,1),WXI(3,1),W(3,1),PM1(3,1),PM2(3,1)
      DIMENSION ETAOLD(3,1),PE1(3,1),ETA(3,1),PE2(3,1)
      REAL MM(3,1),M(3,3)
      DIMENSION PP1(3,1),PP2(3,1),XI(3,1),XIOLD(3,1),BXI(3,3)
      REAL MMOLD(3,1),MUBX,MUBY,MUBZ
      DIMENSION SQ(3,3),PR(3,3),RPR(3,3),RNDNOS(12),ACC(1000,4)
      REAL ISQ(3,3),IRMSM(3,3),IRESE(3,3)
      DIMENSION RXI(3,3),SW(3,3),S(3,3),B(3,1)
      REAL II(3,3),IRM(3,3),IRSE(3,3)
      DIMENSION PRPR(3,3),CXI(3,3),ATRU(3,1),ASEN(3,1),E1(3,1),E2(3,1)
      DIMENSION E3(3,1),E4(3,1)
      PI=ACOS(-1.)
      ZERO=0.
*                                                                    *
*                     WRITE THE HEADER ON TAPE 2.                    *
*                                                                    *
      READ (1) YR,DAY,SEC
      WRITE (2) YR,DAY,SEC
      PRINT 443
  443 FORMAT (1H1)
*                                                                    *
*                       READ THE INPUT CARDS.                        *
*                                                                    *
      DO 20 I=1,17
      READ(4,10) (A(I,J),J=1,4)
   10 FORMAT (A10,E20.5,E20.5,E20.5)
      PRINT 446, (A(I,J),J=1,4)
  446 FORMAT (//,5X,A10,E20.5,E20.5,E20.5)
   20 CONTINUE
      DO 30 I=1,3
      DO 40 J=1,3
      M(I,J)=RM(I,J)=SM(I,J)=PXI(I,J)=RETA(I,J)=SETA(I,J)=RXI(I,J)=0.
      SW(I,J)=0.
   40 CONTINUE
   30 CONTINUE
      II(1,1)=II(2,2)=II(3,3)=1.
      II(1,2)=II(1,3)=II(2,1)=II(2,3)=II(3,1)=II(3,2)=0.
*                                                                    *
*         ASSIGN THE INPUT VALUES TO THE APPROPRIATE VARIABLES.      *
*                                                                    *
      DO 50 I=1,17
      IF (A(I,1).NE.10HBIASX     ) GO TO 51
      MUBX=A(I,2) $ VARBX=A(I,3) $ GO TO 50
   51 IF (A(I,1).NE.10HBIASY     ) GO TO 52
      MUBY=A(I,2) $ VARBY=A(I,3) $ GO TO 50
   52 IF (A(I,1).NE.10HBIASZ     ) GO TO 53
      MUBZ=A(I,2) $ VARBZ=A(I,3) $ GO TO 50
   53 IF (A(I,1).NE.10HTAUM      ) GO TO 54
      TAUMX=A(I,2) $ TAUMY=A(I,3) $ TAUMZ=A(I,4) $ GO TO 50
   54 IF (A(I,1).NE.10HSIGMAM    ) GO TO 55
      SM(1,1)=A(I,2)*PI/180.  $  SM(2,2)=A(I,3)*PI/180.
```

```
      SM(3,3)=A(I,4)*PI/180.  $  GO TO 50
   55 IF (A(I,1).NE.10HINTINT     ) GO TO 56
      DELTAT=A(I,2) $ GO TO 50
   56 IF (A(I,1).NE.10HTAUETA     ) GO TO 57
      TAUETAX=A(I,2) $ TAUETAY=A(I,3) $ TAUETAZ=A(I,4) $ GO TO 50
   57 IF (A(I,1).NE.10HSIGETA     ) GO TO 58
      SETA(1,1)=A(I,2) $ SETA(2,2)=A(I,3) $ SETA(3,3)=A(I,4) $ GO TO 50
   58 IF (A(I,1).NE.10HTAUXI      ) GO TO 59
      TAUXIX=A(I,2) $ TAUXIY=A(I,3) $ TAUXIZ=A(I,4) $ GO TO 50
   59 IF (A(I,1).NE.10HPXI1       ) GO TO 60
      PXI(1,1)=A(I,2) $ GO TO 50
   60 IF (A(I,1).NE.10HPXI2       ) GO TO 61
      PXI(1,2)=PXI(2,1)=A(I,2) $ PXI(2,2)=A(I,3) $ GO TO 50
   61 IF (A(I,1).NE.10HPXI3       ) GO TO 62
      PXI(1,3)=PXI(3,1)=A(I,2) $ PXI(3,2)=PXI(2,3)=A(I,3)
      PXI(3,3)=A(I,4) $ GO TO 50
   62 IF (A(I,1).NE.10HSIGMAW     ) GO TO 63
      SW(1,1)=A(I,2) $ SW(2,2)=A(I,3) $ SW(3,3)=A(I,4) $ GO TO 50
   63 IF (A(I,1).NE.10HSCAFAC1    ) GO TO 64
      S(1,1)=A(I,2) $ S(1,2)=A(I,3) $ S(1,3)=A(I,4) $ GO TO 50
   64 IF (A(I,1).NE.10HSCAFAC2    ) GO TO 65
      S(2,1)=A(I,2) $ S(2,2)=A(I,3) $ S(2,3)=A(I,4) $ GO TO 50
   65 IF (A(I,1).NE.10HSCAFAC3    ) GO TO 66
      S(3,1)=A(I,2) $ S(3,2)=A(I,3) $ S(3,3)=A(I,4) $ GO TO 50
   66 IF (A(I,1).EQ.10HTIMELIM    ) GO TO 67
      PRINT 444, A(I,1)
  444 FORMAT (1H1,5X,*ERROR ON DATA CARDS----ID = *,A10)
      GO TO 666
   67 TIMELIM=A(I,2)
   50 CONTINUE
*                                                                      *
*                        COMPUTE THE BIAS FOR E1.                      *
*                                                                      *
      CALL TRMSET(50.)
      CALL TRMRNG(RNDNOS,3)
      B(1,1)=SQRT(VARBX)*RNDNOS(1)+MUBX
      B(2,1)=SQRT(VARBY)*RNDNOS(2)+MUBY
      B(3,1)=SQRT(VARBZ)*RNDNOS(3)+MUBZ
*                                                                      *
*                    DERIVE SQRT(I-RM2)SM FOR E2.                      *
*                                                                      *
      RM(1,1)=EXP(-1.*DELTAT/TAUMX)
      RM(2,2)=EXP(-1.*DELTAT/TAUMY)
      RM(3,3)=EXP(-1.*DELTAT/TAUMZ)
      CALL MPROD(3,3,3,RM,3,RM,3,SQ,3)
      CALL MSUBT(3,3,II,3,SQ,3,ISQ,3)
      DO 80 I=1,3
      DO 90 J=1,3
      IRM(I,J)=SQRT(ISQ(I,J))
   90 CONTINUE
   80 CONTINUE
      CALL MPROD(3,3,3,IRM,3,SM,3,IRMSM,3)
*                                                                      *
*                    DERIVE RETA FOR ETA OF E3.                        *
*                                                                      *
      RETA(1,1)=EXP(-1.*(DELTAT/TAUETAX))
      RETA(2,2)=EXP(-1.*(DELTAT/TAUETAY))
```

```
      RETA(3,3)=EXP(-1.*(DELTAT/TAUETAZ))
*                                                                   *
*              DERIVE SQRT(I-RETA2)SETA FOR ETA OF E3.              *
*                                                                   *
      CALL MPROD(3,3,3,RETA,3,RETA,3,SQ,3)
      CALL MSUBT (3,3,II,3,SQ,3,ISQ,3)
      DO 100 I=1,3
      DO 110 J=1,3
      IRSE(I,J)=SQRT(ISQ(I,J))
  110 CONTINUE
  100 CONTINUE
      CALL MPROD(3,3,3,IRSE,3,SETA,3,IRESE,3)
*                                                                   *
*                     DERIVE RXI FOR XI OF E3.                      *
*                                                                   *
      RXI(1,1)=EXP(-1.*DELTAT/TAUXIX)
      RXI(2,2)=EXP(-1.*DELTAT/TAUXIY)
      RXI(3,3)=EXP(-1.*DELTAT/TAUXIZ)
      CALL MPROD(3,3,3,PXI,3,RXI,3,PR,3)
      CALL MPROD(3,3,3,RXI,3,PR,3,RPR,3)
      CALL MSUBT(3,3,PXI,3,RPR,3,PRPR,3)
*                                                                   *
*            FIND BXI AND CXI USING SQUARE ROOT ALGORITHM.          *
*                                                                   *
      IF (PXI(1,1).NE.0.) GO TO 38
      DO 36 I=1,3
      DO 37 J=1,3
      CXI(I,J)=BXI(I,J)=0.
   37 CONTINUE
   36 CONTINUE
      GO TO 39
   38 CXI(1,2)=CXI(1,3)=CXI(2,3)=0.
      CXI(1,1)=SQRT(PXI(1,1))
      CXI(2,1)=PXI(1,2)/CXI(1,1)
      CXI(3,1)=PXI(1,3)/CXI(1,1)
      CXI(2,2)=SQRT(PXI(2,2)-CXI(2,1)**2)
      CXI(3,2)=(PXI(2,3)-CXI(2,1)*CXI(3,1))/CXI(2,2)
      CXI(3,3)=SQRT(PXI(3,3)-CXI(3,1)**2-CXI(3,2)**2)
      BXI(1,2)=BXI(1,3)=BXI(2,3)=0.
      BXI(1,1)=SQRT(PRPR(1,1))
      BXI(2,1)=PRPR(1,2)/BXI(1,1)
      BXI(3,1)=PRPR(1,3)/BXI(1,1)
      BXI(2,2)=SQRT(PRPR(2,2)-BXI(2,1)**2)
      BXI(3,2)=(PRPR(2,3)-BXI(2,1)*BXI(3,1))/BXI(2,2)
      BXI(3,3)=SQRT(PRPR(3,3)-BXI(3,1)**2-BXI(3,2)**2)
*                                                                   *
*         PRINT THE INPUT AND COMPUTED TIME-INVARIANT MATRICES.     *
*                                                                   *
   39 PRINT 901, (B(I,1),I=1,3)
  901 FORMAT (1H1,10X,*BIAS MATRIX*,///,10X,E20.5,10X,E20.5,10X,
     1E20.5)
      PRINT 902
  902 FORMAT (//////////,10X,*S MATRIX*,//)
      DO 1001 I=1,3
 1001 PRINT 1000, (S(I,J),J=1,3)
      PRINT 903
  903 FORMAT (//////////,10X,*RM MATRIX*,//)
```

```
      DO 1002 I=1,3
 1002 PRINT 1000, (RM(I,J),J=1,3)
      PRINT 904
  904 FORMAT (//////////,10X,*SM MATRIX*,//)
      DO 1003 I=1,3
 1003 PRINT 1000, (SM(I,J),J=1,3)
      PRINT 905
  905 FORMAT (//////////,10X,*IRMSM MATRIX*,//)
      DO 1004 I=1,3
 1004 PRINT 1000, (IRMSM(I,J),J=1,3)
      PRINT 906
  906 FORMAT (//////////,10X,*RETA MATRIX*,//)
      DO 1005 I=1,3
 1005 PRINT 1000, (RETA(I,J),J=1,3)
      PRINT 907
  907 FORMAT (//////////,10X,*SETA MATRIX*,//)
      DO 1006 I=1,3
 1006 PRINT 1000, (SETA(I,J),J=1,3)
      PRINT 908
  908 FORMAT (//////////,10X,*IRESE MATRIX*,//)
      DO 1007 I=1,3
 1007 PRINT 1000, (IRESE(I,J),J=1,3)
      PRINT 703
  703 FORMAT (//////////,10X,*PXI MATRIX*,//)
      DO 704 I=1,3
  704 PRINT 1000, (PXI(I,J),J,1,3)
      IF (PXI(1,1).NE.0.) GO TO 450
      PRINT 451
  451 FORMAT (//////////)
      PRINT 452
  452 FORMAT (1X,5H*****,5X,*SINCE PXI(1,1) EQUALS ZERO, ALL ELEMENTS O
     1F*,5X,5H*****,/,1X,5H*****,6X,*THE BXI AND CXI MATRICES WERE SET*,
     11X,*TO ZERO*,5X,5H*****)
  450 PRINT 909
  909 FORMAT (//////////,10X,*BXI MATRIX*,//)
      DO 1008 I=1,3
 1008 PRINT 1000, (BXI(I,J),J=1,3)
      PRINT 701
  701 FORMAT (//////////,10X,*CXI MATRIX*,//)
      DO 1009 I=1,3
 1009 PRINT 1000, (CXI(I,J),J=1,3)
      PRINT 702
  702 FORMAT (//////////,10X,*SW MATRIX*,//)
      DO 1010 I=1,3
 1010 PRINT 1000, (SW(I,J),J=1,3)
 1000 FORMAT (/,10X,E20.5,10X,E20.5,10X,E20.5)
*                                                                      *
*                    READ THE INPUT DRAG ACCELERATION.                 *
*                                                                      *
      PRINT 705
  705 FORMAT (1H1,10X,49H*****      ALL ACCELERATIONS ARE IN (G)S      ***
     1**)
      NOACEL=0
      RNCSET=0.
 1111 READ (1) N,((ACC(J,L),L=1,4),J=1,N)
      IF (EOF(1)) 666,250
```

```
25. DO 200 ICNT=1,N
    TI=ACC(ICNT,1)
    ATRU(1,1)=ACC(ICNT,2)
    ATRU(2,1)=ACC(ICNT,3)
    ATRU(3,1)=ACC(ICNT,4)
*                                                                        *
*                 COUNT THE NUMBER OF TIME LINES.                        *
*                                                                        *
    NOACEL=NOACEL+1
*                                                                        *
*                       CHECK FOR THE TIME LIMIT.                        *
*                                                                        *
    IF (TI.GT.TIMELIM) GO TO 667
*                                                                        *
*            CONVERT INPUT ACCELERATIONS TO UNITS OF GS.                 *
*                                                                        *
    DO 33 I=1,3
    ATRU(I,1)=ATRU(I,1)/(.00980621)
 33 CONTINUE
    IF (TI.GT.20.) GO TO 500
    DO 501 I=1,3
    ASEN(I,1)=ATRU(I,1)
501 CONTINUE
    GO TO 502
*                                                                        *
*            OBTAIN THE ELEMENTS OF THE RANDOM VECTORS                   *
*           FROM A GAUSSIAN SEQUENCE OF NORMAL VARIATES.                 *
*                                                                        *
500 RNDSET=RNDSET+43.
    CALL TBMSET(RNDSET)
    CALL TBMRNG(RNDNOS,12)
    DO 25 I=1,3
    WM(I,1)=RNDNOS(I)
    WETA(I,1)=RNDNOS(I+3)
    WXI(I,1)=RNDNOS(I+6)
    W(I,1)=RNDNOS(I+9)
 25 CONTINUE
*                                                                        *
*                            COMPUTE E1                                  *
*      THE RANDOM BIAS AND SCALE FACTOR-INDUCED ACCELERATIONS            *
*                                                                        *
    CALL MPRCD(3,3,1,S,3,ATRU,3,E1,3)
    CALL MADD(3,1,B,3,E1,3,E1,3)
*                                                                        *
*                       COMPUTE MM FOR E2.                               *
*                                                                        *
    IF (NOACEL.NE.1) GO TO 730
    CALL MPROD(3,3,1,SM,3,WM,3,MM,3)
    GO TO 731
730 CALL MPRCD(3,3,1,RM,3,MMCLD,3,PM1,3)
    CALL MPRCD(3,3,1,IRMSM,3,WM,3,PM2,3)
    CALL MADD(3,1,PM1,3,PM2,3,MM,3)
731 DO 88 I=1,3
    MMCLD(I,1)=MM(I,1)
 88 CONTINUE
```

```
*                                                                         *
*                        COMPUTE M FOR E2.                                *
*                                                                         *
      M(1,2)=MM(3,1)
      M(1,3)=-1.*MM(2,1)
      M(2,1)=-1.*MM(3,1)
      M(2,3)=MM(1,1)
      M(3,1)=MM(2,1)
      M(3,2)=-1.*MM(1,1)
*                                                                         *
*                              COMPUTE E2                                 *
*                   THE TIME-DEPENDENT ORIENTATION ERROR                  *
*                                                                         *
      CALL MPROD(3,3,1,M,3,ATRU,3,E2,3)
*                                                                         *
*                        COMPUTE ETA AND XI FOR E3.                       *
*                                                                         *
      IF (NOACEL.NE.1) GO TO 732
      CALL MPROD(3,3,1,SETA,3,WETA,3,ETA,3)
      CALL MPROD(3,3,1,CXI,3,WXI,3,XI,3)
      GO TO 733
  732 CALL MPROD(3,3,1,RETA,3,ETAOLD,3,PE1,3)
      CALL MPROD(3,3,1,IRESE,3,WETA,3,PE2,3)
      CALL MADD(3,1,PE1,3,PE2,3,ETA,3)
      CALL MPROD(3,3,1,RXI,3,XIOLD,3,PP1,3)
      CALL MPROD(3,3,1,BXI,3,WXI,3,PP2,3)
      CALL MADD(3,1,PP1,3,PP2,3,XI,3)
  733 DO 68 I=1,3
      ETAOLD(I,1)=ETA(I,1)
      XIOLD(I,1)=XI(I,1)
   68 CONTINUE
*                                                                         *
*                              COMPUTE E3                                 *
*                       THE ADDITIVE CORRELATED NOISE                     *
*                                                                         *
      CALL MADD(3,1,ETA,3,XI,3,E3,3)
*                                                                         *
*                              COMPUTE E4                                 *
*                            THE WHITE NOISE                              *
*                                                                         *
      CALL MPROD(3,3,1,SW,3,W,3,E4,3)
*                                                                         *
*          COMPUTE THE RESULTANT CORRUPTED DRAG ACCELERATION.             *
*                                                                         *
      CALL MADD(3,1,E1,3,E2,3,ASEN,3)
      CALL MADD(3,1,E3,3,ASEN,3,ASEN,3)
      CALL MADD(3,1,E4,3,ASEN,3,ASEN,3)
      CALL MADD(3,1,ATRU,3,ASEN,3,ASEN,3)
  502 PRINT 467, NOACEL,(ATRU(I,1),I=1,3)
  467 FORMAT (1H0,1X,1H*,I4,1H*,20X,*TRUE AX = *,E15.5,6X,*TRUE AY = *,
     1E15.5,6X,*TRUE AZ = *,E15.5)
      PRINT 468, TI,(ASEN(I,1),I=1,3)
  468 FORMAT (1X,*TIME = *,F15.5,4X,*SENSED AX = *,E15.5,4X,
     1*SENSED AY = *,E15.5,4X,*SENSED AZ = *,E15.5)
```

```
*                                                                          *
*       CONVERT THE ACCELERATIONS TO UNITS OF KM/SEC2 AND STORE.           *
*                                                                          *
      DO 34 I=1,3
      ASEN(I,1)=ASEN(I,1)*(.00980621)
   34 CONTINUE
      ACC(ICNT,2)=ASEN(1,1)
      ACC(ICNT,3)=ASEN(2,1)
      ACC(ICNT,4)=ASEN(3,1)
  200 CONTINUE
*                                                                          *
*              WRITE ONE RECORD OF SIMULATED DATA ON TAPE 2.               *
*                                                                          *
      WRITE (2) N,((ACC(J,L),L=1,4),J=1,1000)
      GO TO 1111
  666 STOP "EOF ON ACCELERATIONS FILE"
  667 STOP "TIME LIMIT REACHED"
      END
```

REFERENCES

1. James W. O'Toole, *Celest Computer Program for Computing Satellite Orbits*, Naval Surface Weapons Center, Technical Report TR-3565, Dahlgren, VA, October 1976.

2. Thomas A. Butler, *Effect of Attitude Control Thruster-Induced Structural Vibrations on Sensed Accelerations of a Space Vehicle*, Los Alamos Scientific Laboratory, Informal Report LA-6965-MS, Los Alamos, NM, September 1977.

DISTRIBUTION

Defense Mapping Agency Aerospace Center St. Louis, MO 63118 ATTN: Robert Ballew	
Defense Mapping Agency U.S. Naval Observatory Bldg. 56 Washington, DC 20360 ATTN: Charles Martin	
Defense Mapping Agency Topographic Center Washington, DC 20315 ATTN: R. Smith	
Library of Congress Washington, DC 20540 ATTN: Gift and Exchange Division	(4)
Defense Technical Information Center Cameron Station Alexandria, VA 22314	(12)
GIDEP Operations Office Corona, CA 91720	
Local:	
E41	
K02	
K10	(80)
X210	(6)
X211	(2)