NASA TECHNICAL MEMORANDUM

NASA TM X-73342

COMPUTER SIMULATION RESULTS OF ATTITUDE ESTIMATION OF EARTH ORBITING SATELLITES

By Shanying R. Kou
Systems Dynamics Laboratory

July 1976

NASA

George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO.	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.
NASA TM X-73342		

4. TITLE AND SUBTITLE	5. REPORT DATE
Computer Simulation Results of Attitude Estimation of Earth Orbiting Satellites	July 1976
	6. PERFORMING ORGANIZATION CODE

7. AUTHOR(S)	8. PERFORMING ORGANIZATION REPORT #
Shanying R. Kou	

9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. WORK UNIT NO.
George C. Marshall Space Flight Center Marshall Space Flight Center, Alabama 35812	11. CONTRACT OR GRANT NO.
	13. TYPE OF REPORT & PERIOD COVERED
12. SPONSORING AGENCY NAME AND ADDRESS	Technical Memorandum
National Aeronautics and Space Administration Washington, D.C. 20546	14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

Prepared by Systems Dynamics Laboratory, Science and Engineering

16. ABSTRACT

Computer simulation results of attitude estimation of Earth-orbiting satellites (including Space Telescope) subjected to environmental disturbances and noises are presented in this report. Decomposed linear recursive filter and Kalman filter were used as estimation tools. Six programs were developed for this simulation, and all were written in BASIC language and were executed in computers HP 9830A and HP 9866A. Simulation results show that a decomposed linear recursive filter is accurate in estimation and fast in response time. Furthermore, for higher order systems, this filter has computational advantages (i.e., less integration errors and roundoff errors) over Kalman filter.

17. KEY WORDS	18. DISTRIBUTION STATEMENT
	Unclassified — Unlimited

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE
Unclassified	Unclassified	58	NTIS

ACKNOWLEDGMENTS

Sincere appreciation is expressed to my scientific adviser Dr. Sherman M. Seltzer of the Systems Dynamics Laboratory of Marshall Space Flight Center for the discussions and help in writing this report. This research was supported by the National Research Council.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Concluded)

DEFINITION OF SYMBOLS

Symbol	Definition
A	transition matrix of a linear continuous-time system
B	input matrix for composed system
B_1	input matrix for the first subsystem $\underline{x}_1$
C_1	transition matrix of system $\underline{x}_1$
C_2	input matrix corresponding to disturbance $\underline{z}$
C_3	matrix related to the dynamics of disturbance $\underline{z}$
$\underline{f}$	a n-dimensional vector valued function
G	input matrix for a linear continuous-time system
H	output matrix
H_1	output matrix corresponding to state variable $\underline{x}_1$
h	time increment (or step size) in iteration
I	identity matrix
K	desired number of random numbers
$K(k)$	optimal gain matrix of Kalman filter at instant t_k
$\overline{K}_x$	upper part of $K(k)$ for a particular initial condition
K_z	lower part of $K(k)$ with three rows
M	a transformation matrix in decomposed linear recursive filter
$\overline{M}$	mean value of a random variable

DEFINITION OF SYMBOLS (Continued)

Symbol	Definition
$\underline{m}_0$	mean value of initial state $\underline{x}(0)$
N	number of used uniform distributed random numbers
P(k|k-1)	a priori error covariance matrix
P(k|k)	a posteriori error covariance matrix
P_0	covariance matrix of initial state $\underline{x}(0)$
P_x	covariance matrix with respect to variable $\underline{x}_1$
P_{xz}	covariance matrix of variables $\underline{x}_1$ and $\underline{z}$
P_z	covariance matrix with respect to variable $\underline{z}$
p_{ij}	the ith row and jth column element of P
$\bar{P}_x$	a P_x matrix corresponding to a special initial value
Q	value of covariance function of s(t)
R	value of covariance function of v(t)
S	standard deviation
s(t)	aerodynamic torque and solar pressure torque
T	number of iterations
t	time variable
t_0	initial time
Δt	sampling period

DEFINITION OF SYMBOLS (Continued)

Symbol	Definition
U_x	a 2×3 transition matrix of decomposed linear recursive filter
U_z	a 3×3 transition matrix of decomposed linear recursive filter
v	sensor noise
V_x	a 2×3 transition matrix related to U_x
V_z	a 3×3 transition matrix related to U_z
X_i	uniformly distributed random variable
$\underline{x}$	state variable of the composite system
$\underline{x}_0$	initial value of $\underline{x}$
$\hat{\underline{x}}$	estimated state of a filter
$\underline{x}_1$	state variable (having two components) of a system
$\hat{\underline{x}}_1$	estimated state of $\underline{x}_1$
$\bar{x}$	estimated state of $\underline{x}_1$ by setting disturbance $\underline{z}$ equal zero
x_i	the ith component of $\underline{x}_1$
$\hat{x}_i$	the estimated value of x_i
Y'	a normally distributed random variable with zero mean and unity standard deviation
Y	a normally distributed random variable
y	output of a system

DEFINITION OF SYMBOLS (Concluded)

Symbol	Definition
$\underline{z}$	transformed deterministic disturbances
$\hat{\underline{z}}$	estimated value of $\underline{z}$
$\hat{\underline{z}}(0)$	initial value of $\hat{\underline{z}}$
ϕ	transition matrix of a discrete-time linear system

TECHNICAL MEMORANDUM X-73342

COMPUTER SIMULATION RESULTS OF ATTITUDE ESTIMATION OF EARTH ORBITING SATELLITES

I. INTRODUCTION

The computer simulation results of attitude estimation of Earth-orbiting satellites are presented in this report. This is a continuation of previous research work published in NASA TM X-64943 [1]. The theory, derivation, and applications of decomposed linear recursive filter are described in Reference 1 which also shows the practical feasibility and advantages of these filters by computer simulation. Six main simulation programs were developed for this simulation program, written in BASIC language and executed by computers HP 9830A and HP 9866A, and are as follows:

Program I — Gaussian Random Number Generation

Program II — Inverse of Symmetric Positive Definite Matrices

Program III — Solutions of Ordinary Differential Equations

Program IV — Simulation of Dynamic Systems in Stochastic Environments

Program V — Simulation of Attitude Estimation by Kalman Filter

Program VI — Simulation of Attitude Estimation by Decomposed Linear Recursive Filters.

Program I generates Gaussian distributed random numbers with required mean value and standard deviation. This program will be used in the modeling of stochastic environmental disturbances and sensor measurement noises. The formulas of decomposed linear recursive filter involve an inverse of symmetric positive definite matrices which can be calculated by Program II. Programs III and IV are used to find solutions of differential equations and difference equations, respectively. Their results show how well the discrete-time model (difference equations) represents the corresponding continuous-time model

(differential equations) through the discretization process. For comparison, attitude estimation of Earth-orbiting satellites by Kalman filter has also been simulated using Program V. Programs I, II, III, and IV were used to develop the simulation program (Program VI) of attitude estimation by decomposed linear recursive filters. These six programs will be described in the following sections with programs also shown.

II. GAUSSIAN RANDOM NUMBER GENERATION

Using the HP 9830A calculator, uniformly distributed random numbers between 0 and 1 can be easily generated by executing RND(x), where x is a dummy argument. Now the problem is how to generate Gaussian (or normally) distributed random numbers with required mean value and standard deviation. According to [2], an approximation to normally distributed random number Y' can be found from a sequence of uniformly distributed random numbers using

$$Y' = \frac{\sum_{i=1}^{N} X_i - \frac{N}{2}}{\sqrt{\frac{N}{12}}}$$

where X_i is a uniformly distributed random number between 0 and 1, and N is the X_i to be used. It can be shown that Y' approaches a true normally distributed random variable as N approaches infinite. For this program, N was chosen as 12 to reduce execution time; therefore, we have

$$Y' = \sum_{i=1}^{12} X_i - 6.0 \quad .$$

The adjustment for the required mean and standard deviation is then

$$Y = Y' * S + \bar{M}$$

where S is the required standard deviation and $\bar{M}$ is the required mean.

Program I-1 (Fig. 1) is for the generation of Gaussian distributed random numbers by using the previously mentioned method. In Program I-1, three parameters should be specified as inputs: $\bar{M}$ (mean value), S (standard deviation), and K (Gaussian random number needed). An example of generating Gaussian random numbers with $\bar{M} = 0$, $S = 1$, and $K = 30$ has been executed and listed following the program.

Program I-2 (Fig. 2) is for examining the characteristics of generated Gaussian random numbers, i.e., to calculate sample mean value and sample standard deviation. These values can be used as a measure of whether the generated random numbers represent Gaussian random numbers satisfactorily. An example with mean zero and standard deviation 10^{-6} was executed and the results of Program I-2 gave a sample mean of $1.132 \cdot 10^{-7}$ and a sample standard deviation of $1.05 \cdot 10^{-6}$ for $K = 30$. These show that the error in the sample mean value is on the order of 10^{-7} and the error in the sample standard deviation using 30 samples is less than 5 percent of the required standard deviation.

III. INVERSE OF SYMMETRIC POSITIVE DEFINITE MATRICES

The BASIC program to find the inverse of symmetric positive definite matrices is presented in this section. The basic idea is first to compute a triangular factorization of a symmetric positive definite matrix using the square root method of Cholesky [3]. This method is formulated in the following two parts:

a) Given an $n \times n$ symmetric positive definite matrix A, w compute an upper triangular matrix R such that

$$A = R^T R \quad .$$

By Cholesky's method, the elements r_{ik} of R are computed from the following recursive formulas:

$$r_{1k} = \frac{a_{1k}}{r_{11}} \qquad k = 1, 2, \ldots, n$$

```
10 REM GENERATION OF GAUSSIAN RANDOM NUMBER
20 N=1
30 INPUT S,M,K
35 REM S=STANDARD DEVIATION, M=MEAN, K=GAUSSIAN RANDOM NO. NEEDED
40 A=0
50 FOR I=1 TO 12
60 B=RND5
70 A=A+B
80 NEXT I
90 Y=(A-6)*S+M
100 PRINT Y
110 N=N+1
120 IF N<K THEN 40
130 END
```

```
-1.804205262
-0.960825294
 0.898244914
-0.577471438
 1.886124850
 1.283398978
-1.392987854
-0.059498446
-0.242939598
-2.62313211
-1.105240782
 2.050305586
 0.472294194
 0.489608242
 1.260506930
 1.332305458
 0.903455026
-0.979977166
 0.698572082
-0.472558030
-0.341572302
 0.260860466
 0.470087474
 0.622351922
 0.128073010
-0.304474062
 0.776921906
 0.068888114
 0.659347762
```

Figure 1. Program I-1: generation of Gaussian random number.

```
10 REM CHECK OF GAUSSIAN RANDOM NUMBER
20 N=1
30 INPUT S,M,K
35 REM S=STANDARD DEVIATION, M=MEAN, K=GAUSSIAN RANDOM NO. NEEDED
36 U=0
37 V=0
40 A=0
50 FOR I=1 TO 12
60 B=RND5
70 A=A+B
80 NEXT I
90 Y=(A-6)*S+M
100 PRINT Y
102 U=U+Y
104 V=V+Y↑2
110 N=N+1
120 IF N<K THEN 40
122 U=U/K
124 V=SQR(V/K)
126 PRINT U,V
130 END

-1.80421E-06
-9.60825E-07
 8.98245E-07
-5.77471E-07
 1.88612E-06
 1.28339E-06
-1.39299E-06
-5.94984E-08
-2.42940E-07
-2.62313E-06
-1.10524E-06
 2.05031E-06
 4.72294E-07
 4.89608E-07
 1.26051E-06
 1.33231E-06
 9.03455E-07
-9.79977E-07
 6.98572E-07
-4.72558E-07
-3.41572E-07
 2.60860E-07
 4.70087E-07
 6.22352E-07
 1.28073E-07
-3.04474E-07
 7.76922E-07
 6.88881E-08
 6.59343E-07
 1.13215E-07     1.05027E-06
```


Figure 2. Program I-2: check of Gaussian random number.

$$r_{jk} = \frac{1}{r_{jj}} \left(a_{jk} - \sum_{i=1}^{j-1} r_{ij}\, r_{ik} \right) \qquad \begin{matrix} j = 2, 3, \ldots, n \\ k = j, j+1, \ldots, n \end{matrix} \quad .$$

b) Since $A^{-1} = R^{-1} (R^{-1})^T$, to compute A^{-1} we need to find the inverse of upper triangular matrix R. Now the elements r_{ik} of R^{-1} are computed from the following recursive formula:

$$\bar{r}_{ik} = -\frac{1}{r_{ij}} \left(\sum_{m=i+1}^{k} r_{im}\, \bar{r}_{mk} \right) \qquad i < k$$

$$\bar{r}_{ik} = \frac{1}{r_{ii}} \qquad i = k$$

$$\bar{r}_{ik} = 0 \qquad i > k \quad .$$

After we find R^{-1}, we can compute $A^{-1} = R^{-1} (R^{-1})^T$.

Program II (Fig. 3) gives the inverse of any 3×3 symmetric positive definite matrices. This program can also be used to invert any $n \times n$ symmetric positive definite matrix by changing the array dimensions of the matrix and the statement (with statement number 120).

In Program II, the input is the A matrix and outputs are R, R^{-1}, and A^{-1}. (Note that R^{-1} is denoted by symbol P and A^{-1} is denoted by symbol Q.)

Example — A 3×3 positive definite and symmetric matrix A:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 6 \end{bmatrix}$$

```
110 DIM A[3,3],R[3,3]
120 N=3
130 INPUT A[1,1],A[1,2],A[1,3],A[2,2],A[2,3],A[3,3]
140 R[1,1]=SQR(A[1,1])
150 FOR K=2 TO N
160 R[1,K]=A[1,K]/R[1,1]
170 NEXT K
180 J=2
190 S=0
200 FOR I=1 TO J-1
210 S=S+R[I,J]↑2
220 NEXT I
230 R[J,J]=SQR(A[J,J]-S)
240 IF J=N THEN 340
250 FOR K=J+1 TO N
260 U=0
270 FOR I=1 TO J-1
280 U=U+R[I,J]*R[I,K]
290 NEXT I
300 R[J,K]=(A[J,K]-U)/R[J,J]
310 NEXT K
320 J=J+1
330 GOTO 190
340 PRINT R[1,1],R[1,2],R[1,3],R[2,2],R[2,3],R[3,3]
434 DIM P[3,3]
450 FOR J=1 TO N
460 P[J,J]=1/R[J,J]
462 NEXT J
464 I=N-1
466 FOR K=I+1 TO N
468 V=0
470 FOR M=I+1 TO K
472 V=V+R[I,M]*P[M,K]
474 NEXT M
476 P[I,K]=-V/R[I,I]
478 NEXT K
480 I=I-1
482 IF I>0 THEN 466
484 PRINT P[1,1],P[1,2],P[1,3],P[2,2],P[2,3],P[3,3]
510 DIM Q[3,3]
550 FOR I=2 TO N
560 FOR K=1 TO I-1
570 P[I,K]=0
580 NEXT K
590 NEXT I
600 FOR J=1 TO N
610 FOR L=1 TO N
620 Q[J,L]=0
630 FOR K=1 TO N
640 Q[J,L]=Q[J,L]+P[J,K]*P[L,K]
650 NEXT K
660 NEXT L
670 NEXT J
680 PRINT Q[1,1],Q[1,2],Q[1,3],Q[2,2],Q[2,3],Q[3,3]
690 END
```

Figure 3. Program II: inverse of symmetric positive definite matrices.

is used as input to execute Program II. The outputs are

$$R = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix} ,$$

$$R^{-1} = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 1 & -0.5 \\ 0 & 0 & 0.5 \end{bmatrix} ,$$

and

$$A^{-1} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 1.25 & -0.25 \\ 0 & -0.25 & 0.25 \end{bmatrix} .$$

The following matrix multiplications are to check the correctness of this program and results.

$$AA^{-1} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 6 \end{bmatrix} \begin{bmatrix} 2 & -1 & 0 \\ -1 & 1.25 & -0.25 \\ 0 & -0.25 & 0.25 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$RR^{-1} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -0.5 \\ 0 & 0 & 0.5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R^T R = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 6 \end{bmatrix} = A \quad .$$

IV. SOLUTIONS OF ORDINARY DIFFERENTIAL EQUATIONS

Given a set of first-order ordinary differential equations (Note: high-order differential equations can be transformed into a set of first-order differential equations)

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, t)$$

with initial value $\underline{x}(t_0) = \underline{x}_0$ where $\underline{x}(t)$ is an n-dimensional vector and $\underline{f}$ is a vector-valued function of $\underline{x}$ and t. The modified Runge-Kutta method by Gill [3] is used to find the solution to this set of first order differential equations. This method improves the storage requirements and accumulated roundoff errors of the classical Runge-Kutta method. Now the resulting vector $\underline{x}(t_0 + h) \overset{\Delta}{=} \underline{x}_4$ is computed by the following formulas with $\underline{x}(t_0) = \underline{x}_0$ and $Q_0 = 0$:

$$\underline{k}_1 = h \cdot \underline{f}(\underline{x}_0, t_0)$$

$$\underline{x}_1 = \underline{x}_0 + \frac{1}{2}(\underline{k}_1 - 2Q_0)$$

$$\underline{Q}_1 = \underline{Q}_0 + 3\left[\frac{1}{2}(\underline{k}_1 - 2\underline{Q}_0)\right] - \frac{1}{2}\underline{k}_1$$

$$\underline{k}_2 = h \cdot \underline{f}\left(\underline{x}_1, t_0 + \frac{h}{2}\right)$$

$$\underline{x}_2 = \underline{x}_1 + \left(1 - \sqrt{\frac{1}{2}}\right)(\underline{k}_2 - \underline{Q}_1)$$

$$\underline{Q}_2 = \underline{Q}_1 + 3\left[\left(1 - \sqrt{\frac{1}{2}}\right)(\underline{k}_2 - \underline{Q}_1)\right] - \left(1 - \sqrt{\frac{1}{2}}\right)\underline{k}_2$$

$$\underline{k}_3 = h \cdot \underline{f}\left(\underline{x}_2,\ t_0 + \frac{h}{2}\right)$$

$$\underline{x}_3 = \underline{x}_2 + \left(1 + \sqrt{\frac{1}{2}}\right)(\underline{k}_3 - \underline{Q}_2)$$

$$\underline{Q}_3 = \underline{Q}_2 + 3\left[\left(1 + \sqrt{\frac{1}{2}}\right)(\underline{k}_3 - \underline{Q}_2)\right] - \left(1 + \sqrt{\frac{1}{2}}\right)\underline{k}_3$$

$$\underline{k}_4 = h \cdot \underline{f}\,(\underline{x}_3,\ t_0 + h)$$

$$\underline{x}_4 = \underline{x}_3 + \frac{1}{6}(\underline{k}_4 - 2\,\underline{Q}_3)$$

$$\underline{Q}_4 = \underline{Q}_3 + 3\left[\frac{1}{6}(\underline{k}_4 - 2\,\underline{Q}_3)\right] - \frac{1}{2}\,\underline{k}_4 \quad .$$

Here $\underline{Q}_4$ represents approximately three times the roundoff error in $\underline{x}_4$ accumulated during one iteration. It can be used as a check of calculation accuracy. For the next iteration, $\underline{Q}_4$ is used as $\underline{Q}_0$, $t_0 + h$ as t_0, and $\underline{x}_4$ as $\underline{x}_0$.

The adjustment of the step size h is done by comparison of the results due to double and single step size 2h and h.

Program III (Fig. 4) is presented using the modified Runge-Kutta method to solve a fifth-order system:

```
20 DIM X[5],Q[5],K[5]
30 N=5
40 INPUT X[1],X[2],X[3],X[4],X[5]
50 T=0
60 FOR I=1 TO 5
70 Q[I]=0
80 NEXT I
90 H=100
95 W=0
100 A=0
110 FOR I=1 TO 12
120 B=RND5
130 A=A+B
140 NEXT I
150 S=(A-6)*0.00000251
200 L=1
210 K[1]=H*X[2]
220 K[2]=H*(X[3]+X[4]+S)
230 K[3]=0
240 K[4]=H*0.002*X[5]
250 K[5]=H*(-0.002)*X[4]
260 IF L>1.1 THEN 345
270 FOR I=1 TO 5
280 X[I]=X[I]+0.5*  [I]-2*Q[I])
290 NEXT I
300 FOR J=1 TO 5
310 Q[J]=Q[J]+3*0.5  K[J]-2*Q[J])-0.5*K[J]
320 NEXT J
330 L=L+1
340 GOTO 210
345 IF L>2.1 THEN 425
350 FOR I=1 TO 5
360 X[I]=X[I]+(1-SQR(0.5))*(K[I]-Q[I])
370 NEXT I
380 FOR J=1 TO 5
390 Q[J]=Q[J]+3*(1-SQR(0.5))*(K[J]-Q[J])-(1-SQR(0.5))*K[J]
400 NEXT J
410 L=L+1
420 GOTO 210
425 IF L>3.1 THEN 510
430 FOR I=1 TO 5
440 X[I]=X[I]+(1+SQR(0.5))*(K[I]-Q[I])
450 NEXT I
460 FOR J=1 TO 5
470 Q[J]=Q[J]+3*(1+SQR(0.5))*(K[J]-Q[J])-(1+SQR(0.5))*K[J]
480 NEXT J
490 L=L+1
500 GOTO 210
510 FOR I=1 TO 5
520 X[I]=X[I]+(K[I]-2*Q[I])/6
530 NEXT I
540 FOR J=1 TO 5
550 Q[J]=Q[J]+3*(K[J]-2*Q[J])/6-0.5*K[J]
560 NEXT J
570 PRINT T+H,X[1],X[2],X[3],X[4],X[5],Q[1],Q[2],Q[3],Q[4],Q[5]
575 W=0
580 T=T+H
590 IF T<3001 THEN 100
600 END
```

Figure 4. Program III: modified Runge-Kutta method.

$$\dot{\underline{x}} = \frac{d\underline{x}}{dt} = \underline{f}(\underline{x},t)$$

$$= A\,\underline{x} + G\,s$$

$$= \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.002 \\ 0 & 0 & 0 & -0.002 & 0 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} S$$

where s is a random variable having Gaussian distribution with mean 0 and standard deviation $2.51 \cdot 10^{-6}$. It was found that the results are satisfactory for $h = 0.01$ second. In Program III, the input is the initial value $\underline{x}(0)$ and the outputs are time t, $\underline{x}(t)$, and $\underline{Q}_d$.

One of the purposes of solving this differential equation is that its results can be used as a measure of the accuracy of discrete-time model through discretization process which will be described in detail in the next section.

V. SIMULATION OF DYNAMIC SYSTEMS IN STOCHASTIC ENVIRONMENTS

At first we introduce the discretization process [4] in which differential equations are transformed into difference equations. Let us rewrite the fifth-order system considered in Section III as follows:

$$\dot{\underline{x}} = A\,\underline{x} + G\,s \quad .$$

It can be shown that the solution $\underline{x}(t)$ of this equation in terms of initial value $\underline{x}(0)$ and input s can be given as

$$\underline{x}(t) = e^{At}\underline{x}(0) + \int_0^t e^{A(t-z)} G\, s(z)\, dz$$

or equivalently

$$\underline{x}(t+\Delta t) = e^{A\Delta t}\underline{x}(t) + \int_t^{t+\Delta t} e^{A(t+\Delta t-z)} G\, dz \cdot s(t) \quad ,$$

or using discrete-time system notations, we have

$$\underline{x}(k+1) = \phi\, \underline{x}(k) + B\, s(k) \tag{1}$$

where $\underline{x}(k)$ denotes the value of state $\underline{x}$ at the $(k+1)$th sampling instant and matrices ϕ and B are

$$\phi = e^{A\Delta t}$$

$$= \begin{bmatrix} 1 & \Delta t & 0.5\cdot(\Delta t)^2 & 250\ 000\ (1-\cos 0.002\,\Delta t) & 250\ 000\ (0.002\,\Delta t - \sin 0.002\,\Delta t) \\ 0 & 1 & \Delta t & 500 \sin 0.002\,\Delta t & 500\ (1-\cos 0.002\,\Delta t) \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \cos 0.002\,\Delta t & \sin 0.002\,\Delta t \\ 0 & 0 & 0 & -\sin 0.002\,\Delta t & \cos 0.002\,\Delta t \end{bmatrix}$$

$$B = \begin{bmatrix} 0.5 \cdot (\Delta t)^2 \\ \Delta t \\ 0 \\ 0 \\ 0 \end{bmatrix} .$$

Since s is a Gaussian random input, Equation (1) is a discrete-time stochastic system. In Program IV (Fig. 5), the simulation of discrete-time stochastic systems includes the generation of Gaussian random number. When we compare these simulated results with those of continuous-time equations (using Program III modified Runge-Kutta method), it shows that this discrete-time model with $\Delta t = 0.01$ second is a good approximation of the continuous-time model.

```
1020 DIM Y[5]
1030 T=0
1040 H=100
1050 INPUT Y[1],Y[2],Y[3],Y[4],Y[5]
1100 A=0
1110 FOR I=1 TO 12
1120 B=RND5
1130 A=A+B
1140 NEXT I
1150 S=(A-6)*0.00000251
1160 Y[1]=Y[1]+H*Y[2]+0.5*H↑2*Y[3]+250000*(1-COS(0.002*H))*Y[4]
1165 Y[1]=Y[1]+250000*(0.002*H-SIN(0.002*H))*Y[5]+0.5*H↑2*S
1170 Y[2]=Y[2]+H*Y[3]+500*SIN(0.002*H)*Y[4]+500*(1-COS(0.002*H))*Y[5]
1175 Y[2]=Y[2]+H*S
1180 Y[3]=Y[3]
1185 U=Y[4]
1190 Y[4]=COS(0.002*H)*Y[4]+SIN(0.002*H)*Y[5]
1200 Y[5]=-SIN(0.002*H)*U+COS(0.002*H)*Y[5]
1210 PRINT T+H,Y[1],Y[2],Y[3],Y[4],Y[5]
1220 T=T+H
1230 IF T<3001 THEN 1100
1240 END
```

Figure 5. Program IV: discrete-time stochastic systems.

VI. SIMULATION OF ATTITUDE ESTIMATION BY KALMAN FILTER

The previous four programs will be used in this section for the simulation of attitude estimation of Earth-orbiting satellites. The Space Telescope, which is an Earth orbiting satellite, is used as an example in this simulation and the Kalman filter is used as the estimation tool. The discrete-time mathematical model of Space Telescope [1] can be represented by

$$\underline{x}(k+1) = \phi\ \underline{x}(k) + B s(k) \tag{2}$$

where s is a Gaussian random number with mean 0 and standard deviation $2.51 \cdot 10^{-6}$ or covariance $Q = 6.28 \cdot 10^{-12}$. The measurement equation is represented by

$$y(k) = H\ \underline{x}(k) + v \tag{3}$$

where $H = [1\ 0\ 0\ 0\ 0]$ is a row vector and v is the sensor noise which is also modeled as a Gaussian white noise with mean 0 and standard deviation $7.26 \cdot 10^{-3}$ or covariance $R = 52.7 \cdot 10^{-6}$.

Kalman filter for the estimation of state $\underline{x}(k)$ of the above system is

$$\hat{\underline{x}}(k+1) = \phi\ \hat{\underline{x}}(k) + K(k+1)\ [y(k+1) - H \phi\ \hat{\underline{x}}(k)] \tag{4}$$

and

$$\hat{\underline{x}}(0) = \underline{m}_0$$

where $\hat{\underline{x}}(k+1)$ is the estimation of $\underline{x}(k+1)$ and is represented as a combination of two terms: one is the previous estimate $\hat{\underline{x}}(k)$ of the state and the other is the correction by the current measurement $y(k+1)$. $K(k+1)$ is called the optimal gain which is determined by minimizing mean square estimation error and can be calculated by

$$K(k+1) = P(k+1|k)\, H^T (H\, P(k+1|k)\, H^T + R)^{-1} \qquad (5)$$

where $P(k+1|k)$ is called a priori error covariance matrix and is obtained from a posteriori error covariance matrix $P(k|k)$ as

$$P(k+1|k) = \phi\ P(k|k)\ \phi^T + B\,Q\,B^T \qquad (6)$$

and

$$P(0|0) = P_0 \quad .$$

Furthermore $P(k|k)$ is obtained from the previous stage a priori error covariance matrix $P(k|k-1)$ as follows:

$$P(k|k) = (I - K(k)\, H)\ P(k|k-1) \quad . \qquad (7)$$

It is assumed that the random numbers $s(k)$ and $v(k)$ are uncorrelated. The initial state $\underline{x}(0)$ is unknown and is modeled as a Gaussian random variable with mean $\underline{m}_0 = \underline{0}$ and covariance $P_0 = I$ (identity matrix).

The calculation procedures of Kalman filter program (Program V, Fig. 6) are as follows:

a. Calculate $P(1|0)$ using initial value $P(0|0) = P_0$ and equation (6).

b. By $P(1|0)$ and equation (5) to calculate $K(1)$ (in general, this procedure involves a matrix inversion.).

c. Generate Gaussian random number $s(k)$ and, for the simulation purpose, assume an initial value $\underline{x}(0)$ then use equation (2) to calculate the state $\underline{x}(1)$.

d. Generate random variable $v(k)$, then use $\underline{x}(1)$ and equation (3) to obtain the simulated output $y(1)$.

```
10 REM KALMAN FILTER SIMULATION
20 DIM X[10],X[5],P[5,5],C[5,5],B[5,1],H[1,5],M[5,5],E[5],F[5],G[5,5],O[5,5]
22 REM X=C*X+B*S , Y=H*X+V
24 REM X(1),....,X(5) ESTIMATES OF TRUE STATES X(6),....,X(10)
30 INPUT X[6],X[7],X[8],X[9],X[10]
35 REM INITIAL VALUES OF ESTIMATES STATE AND ERROR COVARIANCE MATRIX
40 INPUT X[1],X[2],X[3],X[4],X[5]
50 INPUT P[1,1],P[2,2],P[3,3],P[4,4],P[5,5]
100 FOR I=1 TO (5-1)
110 FOR J=I+1 TO 5
120 P[I,J]=0
130 NEXT J
140 NEXT I
150 FOR I=2 TO 5
160 FOR J=1 TO I-1
170 P[I,J]=P[J,I]
180 NEXT J
190 NEXT I
200 T=0
210 H=0.01
220 N=5
230 C[1,1]=1
240 C[1,2]=H
250 C[1,3]=0.5*H↑2
260 C[1,4]=250000*(1-COS(0.002*H))
270 C[1,5]=250000*(0.002*H-SIN(0.002*H))
280 C[2,1]=0
290 C[2,2]=1
300 C[2,3]=H
310 C[2,4]=500*SIN(0.002*H)
320 C[2,5]=500*(1-COS(0.002*H))
330 C[3,1]=0
340 C[3,2]=0
350 C[3,3]=1
360 C[3,4]=0
370 C[3,5]=0
380 C[4,1]=0
390 C[4,2]=0
400 C[4,3]=0
410 C[4,4]=COS(0.002*H)
420 C[4,5]=SIN(0.002*H)
430 C[5,1]=0
440 C[5,2]=0
450 C[5,3]=0
460 C[5,4]=-SIN(0.002*H)
470 C[5,5]=COS(0.002*H)
480 B[1,1]=0.5*H↑2
490 B[2,1]=H
```

Figure 6. Program V: Kalman filter simulation.

```
500 B[3,1]=0
510 B[4,1]=0
520 B[5,1]=0
530 H[1,1]=1
540 H[1,2]=0
550 H[1,3]=0
560 H[1,4]=0
570 H[1,5]=0
575 REM CALCULATE A PRIORI P(I,J)
580 FOR I=1 TO N
590 FOR J=1 TO N
600 D[I,J]=0
610 NEXT J
620 NEXT I
630 FOR I=1 TO N
640 FOR L=1 TO N
650 FOR K=1 TO N
660 D[I,L]=D[I,L]+C[I,K]*P[K,L]
670 NEXT K
672 NEXT L
674 NEXT I
680 FOR I=1 TO N
690 FOR J=1 TO N
700 Q[I,J]=0
710 NEXT J
720 NEXT I
730 FOR I=1 TO N
740 FOR J=1 TO N
750 FOR L=1 TO N
760 Q[I,J]=Q[I,J]+D[I,L]*C[J,L]
770 NEXT L
780 P[I,J]=Q[I,J]+0.00000000000628*B[I,1]*B[J,1]
790 NEXT J
800 NEXT I
805 REM CALCULATE OPTIMAL KALMAN GAIN
810 FOR I=1 TO N
820 K[I]=P[1,I]/(P[1,1]+0.0000527)
830 NEXT I
832 PRINT K[1],K[2],K[3],K[4],K[5]
834 PRINT P[1,1],P[1,2],P[1,3],P[1,4],P[1,5]
835 REM RANDOM NO. GENERATION
836 PRINT P[3,1],P[3,2],P[3,3],P[3,4],P[3,5]
837 PRINT P[4,1],P[4,2],P[4,3],P[4,4],P[4,5]
838 PRINT P[5,1],P[5,2],P[5,3],P[5,4],P[5,5]
840 A1=0
850 A2=0
860 FOR I=1 TO 12
870 B1=RND5
880 B2=RND6
890 A1=A1+B1
900 A2=A2+B2
910 NEXT I
```

Figure 6. (Continued).

```
920 S=(A1-6)*0.00000251
930 V=(A2-6)*0.00726
935 REM CALCULATE TRUE STATE AND OUTPUT FOR SIMULATION
940 FOR I=1 TO 5
950 E[I]=0
960 NEXT I
970 FOR J=1 TO 5
980 FOR K=1 TO 5
990 E[J]=E[J]+C[J,K]*X[K+5]
1000 NEXT K
1010 E[J]=E[J]+B[J,1]*S
1020 NEXT J
1022 FOR I=1 TO 5
1024 X[I+5]=E[I]
1026 NEXT I
1030 Y=X[6]+V
1035 REM CALCULATE ESTIMATED STATE
1040 FOR I=1 TO 5
1050 F[I]=0
1060 NEXT I
1070 FOR I=1 TO 5
1080 FOR J=1 TO 5
1090 F[I]=F[I]+C[I,J]*X[J]
1100 NEXT J
1110 X[I]=F[I]+K[I]*(Y-F[1])
1120 NEXT I
1130 REM CALCULATE ERROR COVARIANCE MATRIX
1160 FOR I=1 TO 5
1170 FOR J=1 TO 5
1180 G[I,J]=P[I,J]-K[I]*P[1,J]
1185 G[I,J]=G[I,J]-G[I,1]*K[J]+0.0000527*K[I]*K[J]
1190 NEXT J
1200 NEXT I
1202 FOR I=1 TO 5
1204 FOR J=1 TO 5
1206 P[I,J]=G[I,J]
1207 NEXT J
1208 NEXT I
1210 PRINT T+H,X[1],X[2],X[3],X[4],X[5],X[6],X[7],X[8],X[9],X[10]
1220 PRINT P[1,1],P[1,2],P[1,3],P[1,4],P[1,5]
1222 PRINT P[2,1],P[2,2],P[2,3],P[2,4],P[2,5]
1224 PRINT P[3,1],P[3,2],P[3,3],P[3,4],P[3,5]
1226 PRINT P[4,1],P[4,2],P[4,3],P[4,4],P[4,5]
1228 PRINT P[5,1],P[5,2],P[5,3],P[5,4],P[5,5]
1230 PRINT X[6]-X[1],X[7]-X[2],X[8]-X[3],X[9]-X[4],X[10]-X[5]
1240 T=T+H
1250 IF T<1 THEN 580
1260 END
```

Figure 6. (Concluded).

e. By the output $y(1)$ and optimal gain $K(1)$ to obtain the estimated state $\hat{\underline{x}}(1)$ from equation (4) with $\hat{\underline{x}}(0) = \underline{m}_0$.

f. Next calculate $P(1|1)$ by equation (7).

g. Procedures a through f are repeated to find $\hat{\underline{x}}(2)$, $\hat{\underline{x}}(3)$, etc.

Inputs of Program V are as follows:

a. Initial state of this system (five components): $x(6)$, $x(7)$, $x(8)$, $x(9)$ and $x(10)$.

b. Initial values of estimated state: $x(1)$, $x(2)$, $x(3)$, $x(4)$ and $x(5)$. These values are usually selected equal to the mean values of the system's initial state.

c. Initial values of diagonal elements of error covariance matrix P_0.

Outputs of Program V are as follows:

a. $K(1)$, $K(2)$, $K(3)$, $K(4)$ and $K(5)$ — optimal gains

b. T+h, $x(1)$, $x(2)$, $x(3)$, ..., $x(10)$ where

T+h — sampling instant,

$x(1)$, $x(2)$, ..., $x(5)$ — estimated states for the system,

$x(6)$, $x(7)$, ..., $x(10)$ — simulated states of the system,

c. $P(I,J)$, $I,J = 1, 2, 3, 4, 5$ error covariance matrix,

d. $x(6) - x(1)$, $x(7) - x(2)$, $x(8) - x(3)$, $x(9) - x(4)$ and $x(10) - x(5)$ — estimated errors.

A flow chart of this program is shown in Figure 7.

A summary of the simulation results using the Kalman filter is presented in the following paragraphs.

With initial estimated state $\hat{x}_1(0) = 0$, two computer runs have been executed in the simulation study for initial state $x_1(0) = 1$ and 5 respectively. It is shown in Figure 8 that the estimated state $\hat{x}_1(k)$ approaches the true

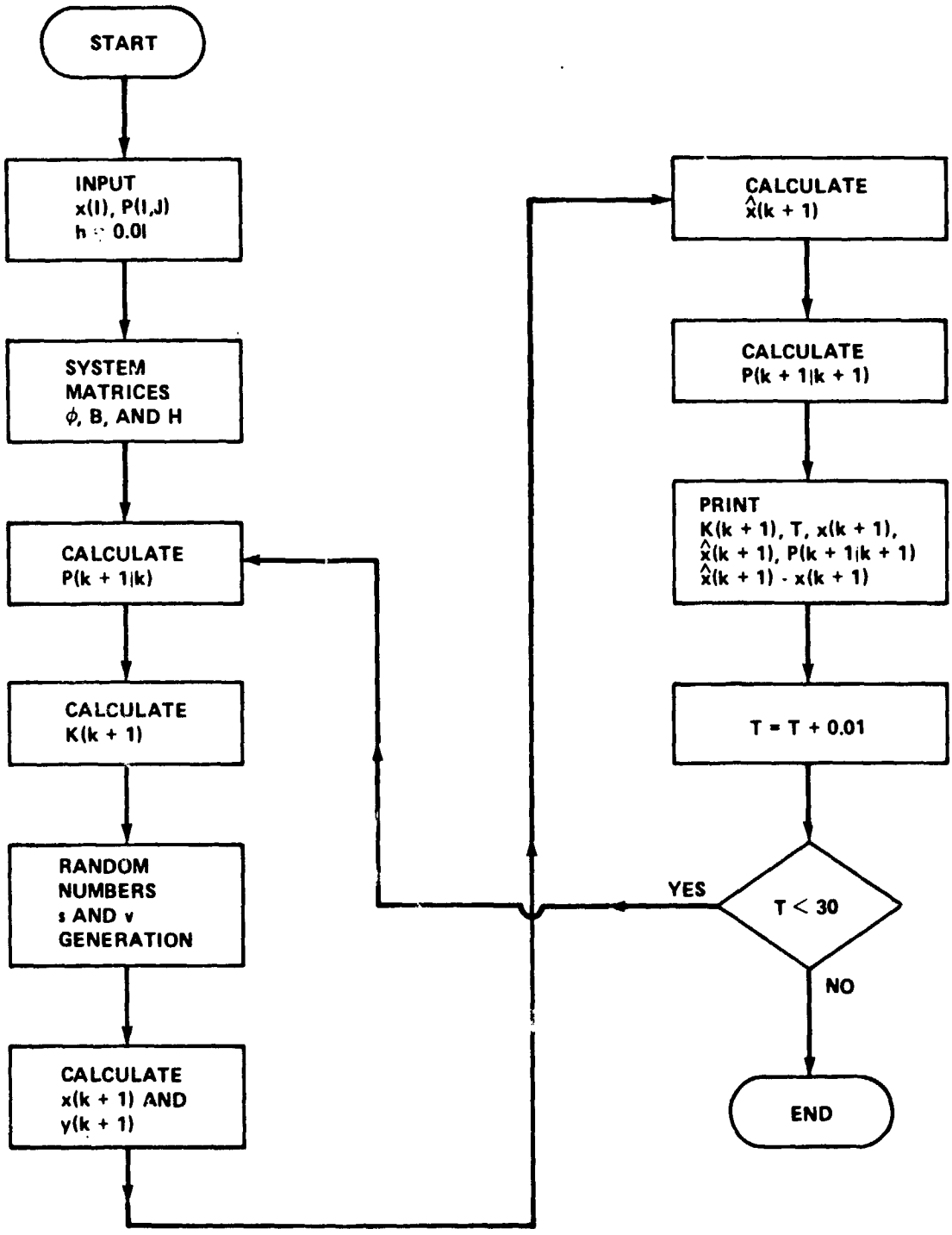

Figure 7. Program V flow chart.

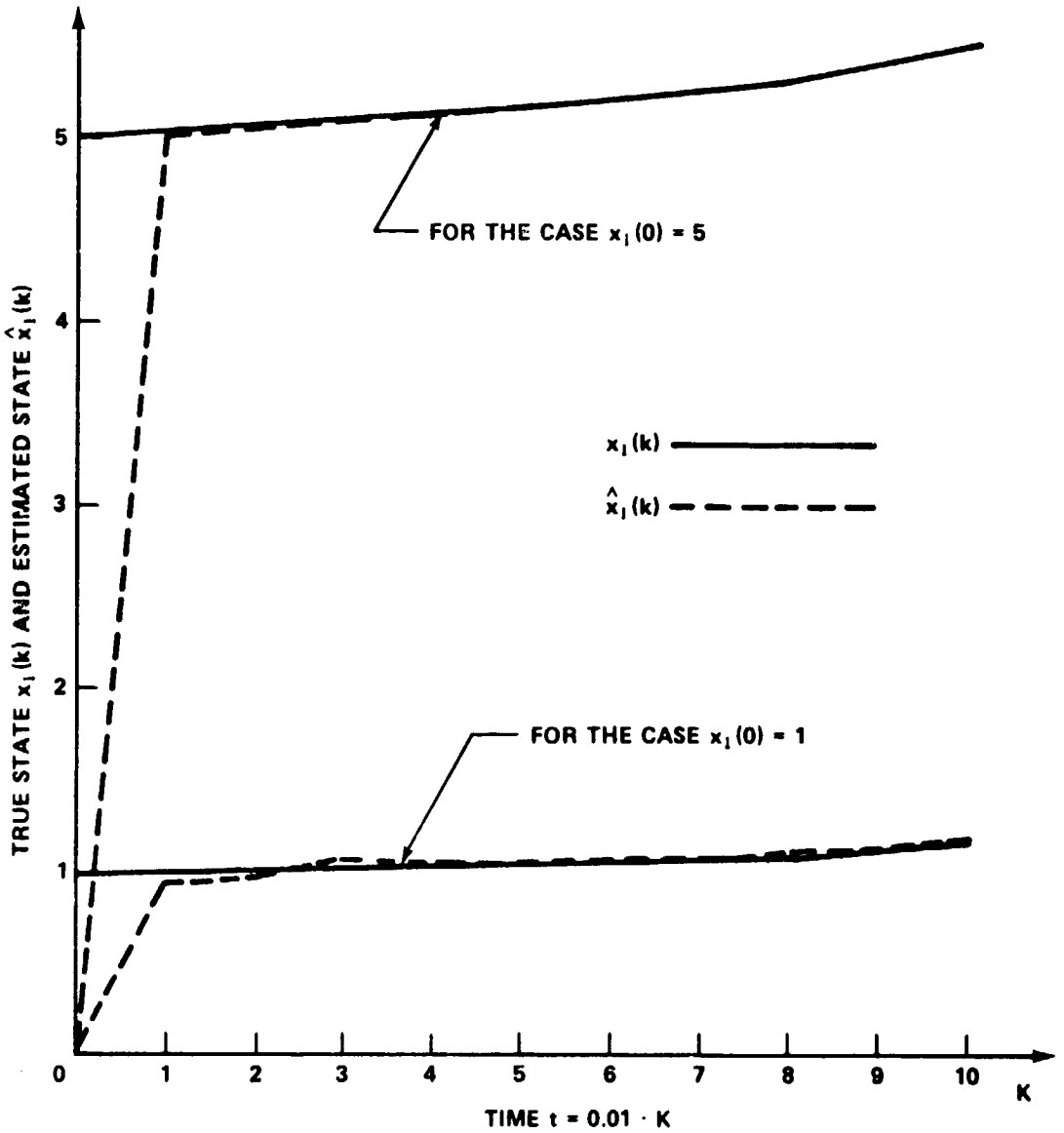

Figure 8. Comparison between true state and estimated state using Kalman filter (with $\hat{x}_1(0) = 0$).

state $x_1(k)$ very fast and then follows the trajectory of the true state precisely. Figure 8 also shows that the accuracy (or characteristics) of the Kalman filter does not depend on the initial value of the system's state, i.e., the simulation results are not sensitive with respect to initial state values.

Figure 9 presents the estimation errors and the rms error curve. The rms error curve is a root mean square error curve obtained from the diagonal elements of error covariance matrix P of Kalman filter. It gives a measure (in the probability sense) of the error at time K. That is, most of the error points in Figure 9 will lie within the two dashed lines (i.e., rms error curves). This curve can be obtained without using measurement information. In addition, the estimated errors shown in Figure 9 are very small and will decrease even more with time.

Figure 10 presents the variation of rms error curves for different Q's and R's which are the covariance matrices of disturbances and sensor noises, respectively. Four computer runs have been executed for this study:

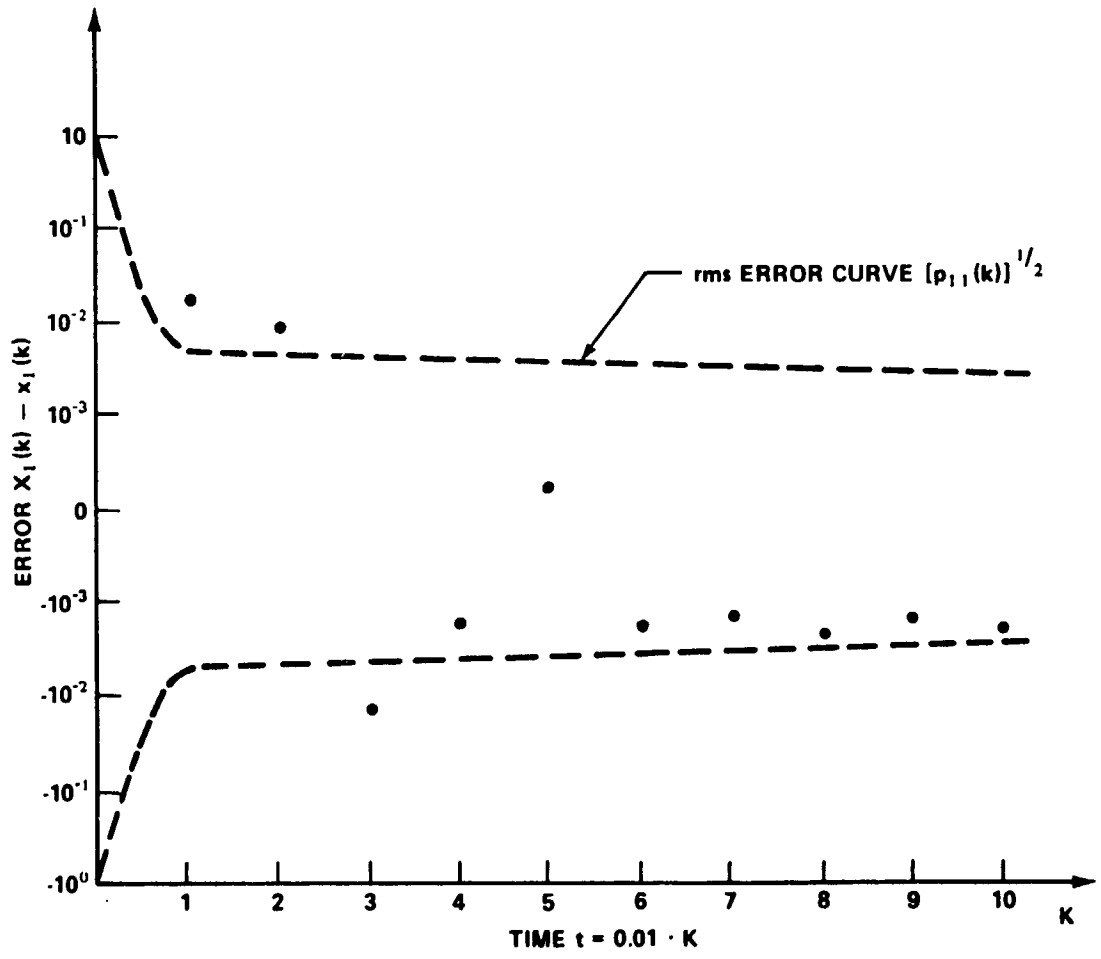

Figure 9. Estimation error $x_1(k) - \hat{x}_1(k)$ by Kalman filter (with $x_1(0) = 1$, $\hat{x}_1(0) = 0$).

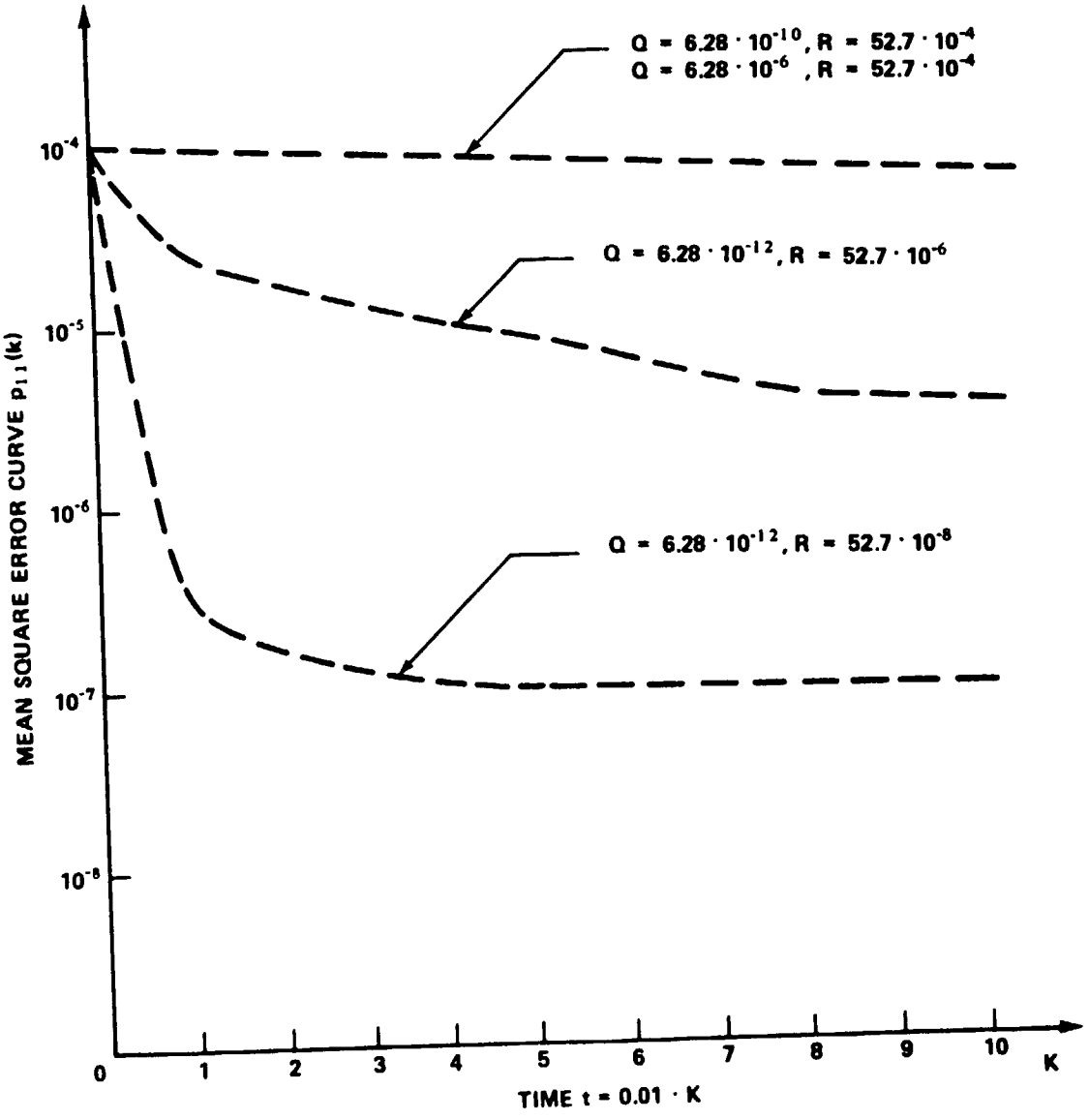

Figure 10. Variations of error curves with respect to Q and R.

$Q = 6.28 \cdot 10^{-}$, $R = 52.7 \cdot 10^{-8}$

$Q \doteq 6.28 \cdot 10^{-}$, $R = 52.7 \cdot 10^{-6}$

$Q = 6.28 \cdot 10^{-10}$, $R = 52.7 \cdot 10^{-4}$

$Q = 6.28 \cdot 10^{-6}$, $R = 52.7 \cdot 10^{-4}$.

From the curves of Figure 10, it is shown that for this system, the rms error curve is not sensitive with respect to Q values, but is sensitive with respect to different R's. To obtain a more accurate estimation requires smaller R values. (Note that all curves are obtained using $p_{11}(0) = 10^{-4}$.)

VII. SIMULATION OF ATTITUDE ESTIMATION BY DECOMPOSED LINEAR RECURSIVE FILTER

Decomposed linear recursive filters were developed previously [1] for the attitude estimation of Earth-orbiting satellites. These filters were shown in Reference 1 to have computational advantages over Kalman filter. The simulation procedures and results of attitude estimation of Space Telescope using decomposed linear recursive filter are presented in this section. The discrete-time mathematical model of Space Telescope [1] can be written as

$$\underline{x}_1(k+1) = C_1 \underline{x}_1(k) + C_2 \underline{z}(k) + B_1 s$$

$$\underline{z}(k+1) = C_3 \underline{z}(k) \qquad (8)$$

where $\underline{x}_1$ and $\underline{z}$ are the state variables of the overall system, i.e., $\underline{x} = [\underline{x}_1 \; \underline{z}]^T$; s is a Gaussian white noise as described in the previous section; and

$$C_1 = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \qquad \Delta t \text{ sampling period,}$$

$$C_2 \begin{bmatrix} 0.5 \cdot (\Delta t)^2 & 250\,000\,(1 - \cos 0.002\,\Delta t) & 250\,000\,(0.002\,\Delta t - \sin 0.002\,\Delta t) \\ \Delta t & 500 \sin 0.002\,\Delta t & 500\,(1 - \cos 0.002\,\Delta t) \end{bmatrix}$$

$$B_1 \cdot \begin{bmatrix} 0.5 \cdot (\Delta t)^2 \\ \Delta t \end{bmatrix},$$

$$C_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos 0.002\,\Delta t & \sin 0.002\,\Delta t \\ 0 & -\sin 0.002\,\Delta t & \cos 0.002\,\Delta t \end{bmatrix} .$$

The measurement equation is

$$y(k) = H \begin{bmatrix} \underline{x}_1(k) \\ \\ \underline{z}(k) \end{bmatrix} + v \qquad (9)$$

where row vector H and Gaussian white noise v are defined as those of the previous section.

To estimate the state of this system, we use decomposed linear recursive filter which is described as follows:

$$\begin{aligned} \hat{\underline{x}}_1(k) &= \bar{x}(k) + V_x(k)\, V_z^{-1}(k)\, \hat{\underline{z}}(k) \\ \hat{\underline{z}}(k+1) &= C_3\, \hat{\underline{z}}(k) + K_z(k+1)\, [y(k+1) - H_1 C_1\, \hat{\underline{x}}_1(k) - H_1 C_2\, \hat{\underline{z}}(k)] \end{aligned} \qquad (10)$$

where $\hat{\underline{x}}_1$ and $\hat{\underline{z}}$ are estimated states of $\underline{x}_1$ and $\underline{z}$, $\hat{\underline{z}}(0) = [10^{-4},\ 4 \cdot 10^{-4},\ 0]^T$, and

a. $\bar{x}(k)$ is obtained from the following formula:

$$\bar{x}(k+1) = C_1\, \bar{x}(k) + \bar{K}_x(k+1)\, [y(k+1) - H_1 C_1\, \bar{x}(k)] \qquad (11)$$

where $H_1 = [1\ \ 0]$, $\bar{x}(0) = [0\ \ 0]^T$, and $\bar{K}_x(k+1)$ satisfy

$$\bar{K}_x(k+1) = \bar{p}_x(k+1|k)\, H_1^T \left[H_1\, \bar{p}_x(k+1|k)\, H_1^T + R\right]^{-1} \tag{12}$$

where

$$\bar{p}_x(k+1|k) = C_1\, \bar{p}_x(k|k)\, C_1^T + B_1\, Q\, B_1^T \tag{13}$$

and

$$\bar{p}_x(0|0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} .$$

b. V_x and V_z are obtained from

$$\begin{aligned} V_x(k+1) &= (I - \bar{K}_x(k+1)\, H_1)\, U_x(k+1) \\ V_z(k+1) &= U_z(k+1) \end{aligned} \tag{14}$$

where

$$\begin{aligned} U_x(k+1) &= C_1\, V_x(k) + C_2\, V_z(k) \\ U_z(k+1) &= C_3\, V_z(k) \end{aligned} \tag{15}$$

and

$$V_x(0) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} , \qquad V_z(0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} .$$

c. $K_z(k+1)$ is given by

$$K_z(k) = V_z(k)\, M(k+1)\, V_x^T(k)\, H_1^T\, R^{-1} \tag{16}$$

where $M(k+1)$ satisfies

$$M(k+1) = M(k) - M(k) \begin{bmatrix} U_x \\ \\ U_z \end{bmatrix}^T H^T \left\{ H \begin{bmatrix} \bar{p}_x(k|k-1) & 0 \\ \\ 0 & 0 \end{bmatrix} H^T + R \right.$$

$$\left. + H \begin{bmatrix} U_x \\ \\ U_z \end{bmatrix} M(k) \begin{bmatrix} U_x \\ \\ U_z \end{bmatrix}^T H^T \right\}^{-1} H \begin{bmatrix} U_x \\ \\ U_z \end{bmatrix} M(k) \tag{17}$$

with initial value

$$M(0) = \begin{bmatrix} 10^{-4} & 0 & 0 \\ 0 & 10^{-4} & 0 \\ 0 & 0 & 10^{-4} \end{bmatrix} .$$

The procedures for calculating the estimated states in this program (Program VI, Fig. 11) are as follows:

a. From equation (13) using initial value $\bar{p}_x(0|0)$ we get $\bar{p}_x(1|0)$,

b. Then by equation (12) and $\bar{p}_x(1|0)$ we can calculate $\bar{K}_x(1)$,

c. System's state and output are obtained by first generating two Gaussian random numbers s and v and then we have $\underline{x}(k)$ and $y(k)$ from equations (8) and (9).

```
1450 REM DECOMPOSED LINEAR RECURSIVE FILTER SIMULATION
1460 REM X=C*X+B*S , Y=H*X+V
1470 REM X(1),....,X(5) ESTIMATES OF TRUE STATES X(6),...,X(10)
1500 DIM X[12],K[5],P[5,5],C[5,5],G[5,5],HL[1,5],U[2,2],E[5],F[5],GL[5,5],C[2,2]
1510 DIM V[5,3],U[5,3],M[3,3],A[3,3],R[3,3],C[5,5],HL[5,5]
1520 H=0.01
1530 C[1,1]=1
1540 C[1,2]=H
1550 C[1,3]=0.5*H↑2
1560 C[1,4]=250000*(1-COS(0.002*H))
1570 C[1,5]=250000*(0.002*H-SIN(0.002*H))
1580 C[2,1]=0
1590 C[2,2]=1
1600 C[2,3]=H
1610 C[2,4]=500*SIN(0.002*H)
1620 C[2,5]=500*(1-COS(0.002*H))
1630 C[3,1]=0
1640 C[3,2]=0
1650 C[3,3]=1
1660 C[3,4]=0
1670 C[3,5]=0
1680 C[4,1]=0
1690 C[4,2]=0
1700 C[4,3]=0
1710 C[4,4]=COS(0.002*H)
1720 C[4,5]=SIN(0.002*H)
1730 C[5,1]=0
1740 C[5,2]=0
1750 C[5,3]=0
1760 C[5,4]=-SIN(0.002*H)
1770 C[5,5]=COS(0.002*H)
1780 B[1,1]=0.5*H↑2
1790 B[2,1]=H
1800 B[3,1]=0
1810 B[4,1]=0
1820 B[5,1]=0
1830 H[1,1]=1
1840 H[1,2]=0
1850 H[1,3]=0
1860 H[1,4]=0
1870 H[1,5]=0
2010 INPUT X[6],X[7],M[3,3],M[3,3],M[1,1]
2014 T=0
2020 N=2
2030 M=3
2035 REM INITIAL VALUES OF ESTIMATES, STATE AND ERROR COVARIANCE MATRIX
2040 P[1,1]=1
2050 P[2,1]=0
2060 P[1,2]=0
2070 P[2,2]=1
2080 X[11]=0
2090 X[12]=0
```

Figure 11. Program VI: decomposed linear recursive filter simulation.

```
2100 FOR I=1 TO N
2110 FOR J=1 TO M
2120 V[I,J]=0
2130 NEXT J
2140 NEXT I
2150 V[3,1]=1
2160 V[4,2]=1
2170 V[5,3]=1
2180 FOR I=1 TO N
2190 FOR J=1 TO N
2200 M[I,J]=0
2210 NEXT J
2220 NEXT I
2230 M[1,1]=0.0001
2240 M[2,2]=0.0001
2250 M[3,3]=0.0001
2260 X[3]=0.0001
2270 X[4]=0.0004
2280 X[5]=0
2290 X[1]=0
2300 X[2]=0
2310 FOR I=1 TO N
2320 FOR J=1 TO N
2330 D[I,J]=0
2340 NEXT J
2350 NEXT I
2360 FOR I=1 TO N
2370 FOR L=1 TO N
2380 FOR K=1 TO N
2390 D[I,L]=D[I,L]+C[I,K]*P[K,L]
2400 NEXT K
2410 NEXT L
2420 NEXT I
2430 FOR I=1 TO N
2440 FOR J=1 TO N
2450 Q[I,J]=0
2460 NEXT J
2470 NEXT I
2480 FOR I=1 TO N
2490 FOR J=1 TO N
2500 FOR L=1 TO N
2510 Q[I,J]=Q[I,J]+D[I,L]*C[J,L]
2520 NEXT L
2530 P[I,J]=Q[I,J]+0.000000000000628*B[I,1]*B[J,1]
2540 NEXT J
2550 NEXT I
2555 REM CALCULATE OPTIMAL KALMAN GAIN
2560 FOR I=1 TO N
2570 K[I]=P[1,I]/(P[1,1]+0.0000527)
```

Figure 11. (Continued).

```
2580 NEXT I
2581 PRINT K[1],K[2]
2582 PRINT P[1,1],P[1,2]
2583 PRINT P[2,1],P[2,2]
2587 REM RANDOM NO. GENERATION
2590 A1=0
2600 A2=0
2610 FOR I=1 TO 12
2620 B1=RND5
2630 B2=RND6
2640 A1=A1+B1
2650 A2=A2+B2
2660 NEXT I
2670 S=(A1-6)*0.00000251
2680 V=(A2-6)*0.00726
2685 REM CALCULATE TRUE STATE AND OUTPUT FOR SIMULATION
2690 FOR I=1 TO 5
2700 E[I]=0
2710 NEXT I
2720 FOR J=1 TO 5
2730 FOR K=1 TO 5
2740 E[J]=E[J]+C[J,K]*X[K+5]
2750 NEXT K
2760 E[J]=E[J]+B[J,1]*S
2770 NEXT J
2772 FOR I=1 TO 5
2774 X[I+5]=E[I]
2776 NEXT I
2780 Y=X[6]+V
2785 REM CALCULATE ESTIMATED STATE
2790 FOR I=1 TO N
2800 F[I]=0
2810 NEXT I
2820 FOR I=1 TO N
2830 FOR J=1 TO N
2840 F[I]=F[I]+C[I,J]*X[J]
2850 NEXT J
2860 X[10+I]=F[I]+K[I]*(Y-F[1])
2870 NEXT I
2880 FOR I=1 TO N+M
2890 FOR J=1 TO M
2900 U[I,J]=0
2910 NEXT J
2920 NEXT I
2930 FOR I=1 TO N
2940 FOR J=1 TO M
2950 FOR K=1 TO N+M
2960 U[I,J]=U[I,J]+C[I,K]*V[K,J]
2970 NEXT K
```

Figure 11. (Continued).

```
2980 NEXT J
2990 NEXT I
3000 FOR I=N+1 TO N+M
3010 FOR J=1 TO M
3020 FOR K=N+1 TO N+M
3030 U[I,J]=U[I,J]+C[I,K]*V[K,J]
3040 NEXT K
3050 NEXT J
3060 NEXT I
3070 FOR I=1 TO N
3080 FOR J=1 TO M
3090 V[I,J]=U[I,J]-K[I]*U[1,J]
3100 NEXT J
3110 NEXT I
3120 FOR I=N+1 TO N+M
3130 FOR J=1 TO M
3140 V[I,J]=U[I,J]
3150 NEXT J
3160 NEXT I
3170 P=0
3180 FOR K=1 TO 3
3190 P=P+U[1,K]*(M[K,1]*U[1,1]+M[K,2]*U[1,2]+M[K,3]*U[1,3])
3200 NEXT K
3210 FOR I=1 TO 3
3220 FOR J=1 TO 3
3230 N[I,J]=0
3240 O[I,J]=0
3250 NEXT J
3260 NEXT I
3270 FOR I=1 TO 3
3280 FOR J=1 TO 3
3290 FOR K=1 TO 3
3300 O[I,J]=O[I,J]+M[I,K]*U[1,K]*U[1,J]
3310 NEXT K
3320 NEXT J
3330 NEXT I
3340 FOR I=1 TO 3
3350 FOR J=1 TO 3
3360 FOR K=1 TO 3
3370 N[I,J]=N[I,J]+O[I,K]*M[K,J]
3380 NEXT K
3390 NEXT J
3400 NEXT I
3410 FOR I=1 TO 3
3420 FOR J=1 TO 3
3430 M[I,J]=M[I,J]-N[I,J]/(P[1,1]+P+0.000527)
3440 NEXT J
3450 NEXT I
3460 FOR I=1 TO 3
```

Figure 11. (Continued).

```
3470 FOR J=1 TO 5
3480 N[I,J]=0
3490 NEXT J
3510 NEXT I
3512 FOR I=6 TO 5
3514 K[I]=0
3516 NEXT I
3520 FOR I=N+1 TO N+M
3530 FOR J=1 TO 3
3540 FOR K=1 TO 3
3550 N[I,J]=N[I,J]+V[I,K]*M[K,J]
3560 NEXT K
3570 K[I]=K[I]+N[I,J]*V[I,J]
3580 NEXT J
3590 K[I]=K[I]/0.0000527
3600 NEXT I
3610 FOR I=1 TO 2
3620 FOR J=1 TO 2
3630 G[I,J]=P[I,J]-K[I]*P[1,J]
3640 NEXT J
3650 NEXT I
3660 FOR I=1 TO 2
3670 FOR J=1 TO 2
3680 P[I,J]=G[I,J]
3690 NEXT J
3700 NEXT I
3710 FOR I=1 TO 5
3720 FOR J=1 TO 5
3730 N[I,J]=0
3740 G[I,J]=0
3750 NEXT J
3760 NEXT I
3770 FOR I=1 TO 2
3780 FOR L=1 TO 2
3790 FOR J=1 TO 3
3800 FOR K=1 TO 3
3810 G[I,J]=G[I,J]+V[I,K]*M[K,J]
3820 NEXT K
3830 N[I,L]=N[I,L]+G[I,J]*V[L,J]
3840 NEXT J
3850 N[I,L]=N[I,L]+P[I,L]
3860 NEXT L
3870 NEXT I
3880 FOR I=1 TO 2
3890 FOR J=1 TO 3
3900 FOR K=1 TO 3
3910 N[I,J+2]=N[I,J+2]+G[I,K]*V[J+2,K]
3920 NEXT K
3930 NEXT J
3940 NEXT I
```

Figure 11. (Continued).

```
3950 FOR I=1 TO 3
3960 FOR J=1 TO 3
3970 G[I,J]=0
3980 NEXT J
3990 NEXT I
4000 FOR I=1 TO 3
4010 FOR J=1 TO 3
4020 FOR L=1 TO 3
4030 FOR K=1 TO 3
4040 G[I,L]=G[I,L]+V[I+2,K]*M[K,L]
4050 NEXT K
4060 N[I+2,J+2]=N[I+2,J+2]+G[I,L]*V[J+2,L]
4062 G[I,L]=0
4070 NEXT L
4080 NEXT J
4090 NEXT I
4100 PRINT N[1,1],N[1,2],N[1,3],N[1,4],N[1,5]
4102 PRINT N[2,1],N[2,2],N[2,3],N[2,4],N[2,5]
4104 PRINT N[3,3],N[3,4],N[3,5]
4106 PRINT N[4,3],N[4,4],N[4,5]
4108 PRINT N[5,3],N[5,4],N[5,5]
4610 FOR I=3 TO 5
4620 F[I]=0
4630 NEXT I
4640 FOR I=3 TO 5
4650 FOR J=3 TO 5
4660 F[I]=F[I]+C[I,J]*X[J]
4670 NEXT J
4680 NEXT I
4690 Y=Y-C[1,1]*X[1]-C[1,2]*X[2]-C[1,3]*X[3]-C[1,4]*X[4]-C[1,5]*X[5]
4700 FOR I=3 TO 5
4710 X[I]=F[I]+K[I]*Y
4720 NEXT I
4730 R[1,1]=SQR(V[3,1])
4740 FOR K=2 TO 3
4750 R[1,K]=V[3,K]/R[1,1]
4760 NEXT K
4770 J=2
4780 S=0
4790 FOR I=1 TO J-1
4800 S=S+R[I,J]↑2
4810 NEXT I
4820 R[J,J]=SQR(V[J+2,J]-S)
4830 IF J=3 THEN 5020
4840 FOR K=J+1 TO 3
4850 U=0
4860 FOR I=1 TO J-1
4870 U=U+R[I,J]*R[I,K]
4880 NEXT I
4890 R[J,K]=(V[J+2,K]-U)/R[J,J]
4900 NEXT K
```

Figure 11. (Continued).

```
5000 J=J+1
5010 GOTO 4780
5020 FOR J=1 TO 3
5030 A[J,J]=1/R[J,J]
5040 NEXT J
5050 I=3-1
5060 FOR K=I+1 TO 3
5070 V=0
5080 FOR L=I+1 TO K
5090 V=V+R[I,L]*A[L,K]
5100 NEXT L
5110 A[I,K]=-V/R[I,I]
5120 NEXT K
5130 I=I-1
5140 IF I>0 THEN 5060
5150 FOR I=2 TO 3
5160 FOR K=1 TO I-1
5170 A[I,K]=0
5180 NEXT K
5190 NEXT I
5200 FOR J=1 TO 3
5210 FOR L=1 TO 3
5220 V[J+2,L]=0
5230 FOR K=1 TO 3
5240 V[J+2,L]=V[J+2,L]+A[J,K]*A[L,K]
5250 NEXT K
5260 NEXT L
5270 NEXT J
5280 PRINT V[3,1],V[3,2],V[3,3],V[4,1],V[4,2],V[4,3],V[5,1],V[5,2],V[5,3]
5300 FOR I=1 TO 2
5310 FOR J=1 TO 3
5320 N[I,J]=0
5330 NEXT J
5340 NEXT I
5350 FOR I=1 TO 2
5360 FOR K=1 TO 3
5370 FOR J=1 TO 3
5380 N[I,K]=N[I,K]+V[I,J]*V[J+2,K]
5390 NEXT J
5400 NEXT K
5410 X[I]=X[I+10]+N[I,1]*X[2+1]+N[I,2]+X[2+2]+N[I,3]*X[2+3]
5420 NEXT I
5430 PRINT T+H,X[1],X[2],X[3],X[4],X[5],X[6],X[7],X[8],X[9],X[10]
5440 PRINT X[6]-X[1],X[7]-X[2],X[8]-X[3],X[9]-X[4],X[10]-X[5]
5450 T=T+H
5460 IF T<0.1 THEN 3310
5470 END
```

Figure 11. (Concluded).

d. Using $\bar{K}_x$ and y(k), equation (11) gives $\bar{x}(k)$. This vector will be used to construct $\hat{\underline{x}}_1$.

e. Calculate $V_x(k)$ and $V_z(k)$ by equations (14) and (15). M(k+1) is obtained by equation (17).

f. Optimal gain $K_z(k)$ can be obtained from equation (16).

g. Finally $\hat{\underline{x}}_1(k)$ and $\hat{\underline{z}}(k)$ are calculated by equation (10).

h. Before the second iteration is executed, we need to find $\bar{p}_x(k|k)$ using

$$\bar{p}_x(k|k) = (I - \bar{K}_x(k)\, H_1)\, \bar{p}_x(k|k-1)$$

i. Then the procedures a to h are repeated.

j. To have more insight concerning the simulation results, we need to calculate the error covariance matrices which can be found by

$$p_x(k|k) = \bar{p}_x(k|k) + V_x(k)\, M(k+1)\, V_x^T(k)$$

$$p_{xz}(k|k) = V_x(k)\, M(k+1)\, V_z^T(k)$$

$$p_z(k|k) = V_z(k)\, M(k+1)\, V_z^T(k) \quad .$$

Inputs of this program (Program [illegible]e

x(6), x(7), x(8), x(9) a[illegible] 0) — initial values of system's state.

Outputs of Program VI are

a. K(1), K(2) — optimal gains of Kalman filter for the bias-free subsystem.

b. $P(1,1)$, $P(1,2)$, $P(2,1)$, $P(2,2)$ — a priori error covariance matrix of the bias-free subsystem.

c. $V_z(K)$ — a 3×3 matrix (this printout is used to check the results of $V_z^{-1}(k)$ shown later).

d. $M(k)$ — a 3×3 matrix used to construct the error covariance matrix.

e. $N(I,J)$ — a 5×5 matrix which is a posteriori error covariance matrix [common symbol is $P(k|k)$].

f. $V(I,J)$ — a 3×3 matrix which is V_z^{-1} used to construct estimation of $\hat{\underline{x}}_1(k)$.

g. $T+h$, $x(1)$, $x(2)$, ..., $x(10)$ where

$T+h$ — sampling instant,

$x(1)$, $x(2)$, ..., $x(5)$ — estimated state,

$x(6)$, $x(7)$, ..., $x(10)$ — state of the system.

h. $x(6) - x(1)$, $x(7) - x(2)$, ..., $x(10) - x(5)$ — estimation errors.

Figure 12 presents a flow chart of this program.

Simulation results using decomposed linear recursive filter are summarized in the following paragraphs.

Figure 13 presents the true state $x_1(k)$ and the estimated state $\hat{x}_1(k)$ using decomposed linear recursive filter for two cases ($x_1(0) = 1$ and 5 respectively). The results show that this filter is accurate in estimation and fast in response. It is interesting to point out that these simulation results and those of using Kalman filter (Fig. 8) match very well. But for higher order systems, this filter will give a better estimation than that of Kalman filter. Decomposed linear recursive filter has computational advantages, i.e., less integration errors and roundoff errors.

Figure 14 gives the results of the state $x_2(k)$ and the estimated state $\hat{x}_2(k)$. $x_2(k)$ is the time rate of pitch angle.

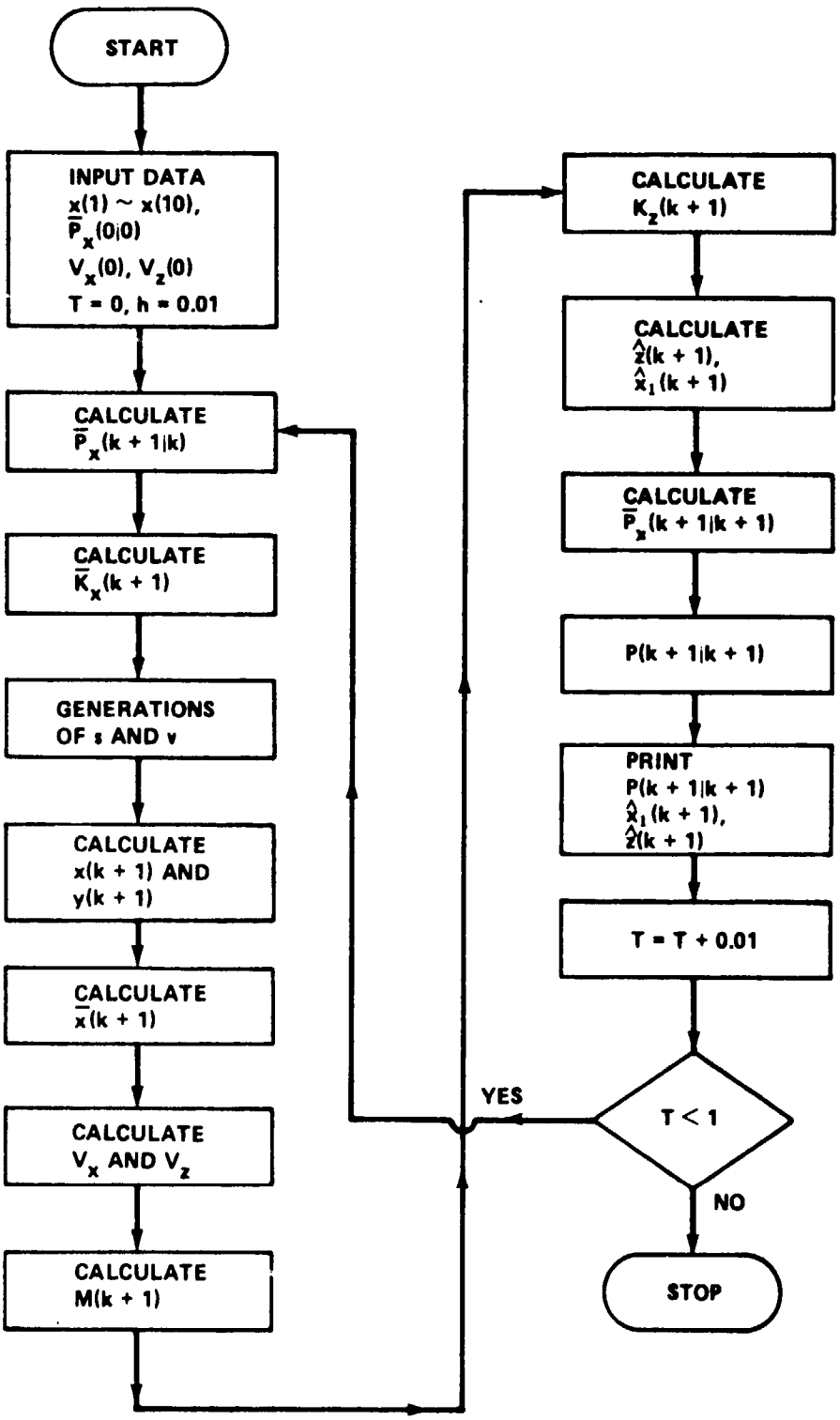

Figure 12. Program VI flow chart.

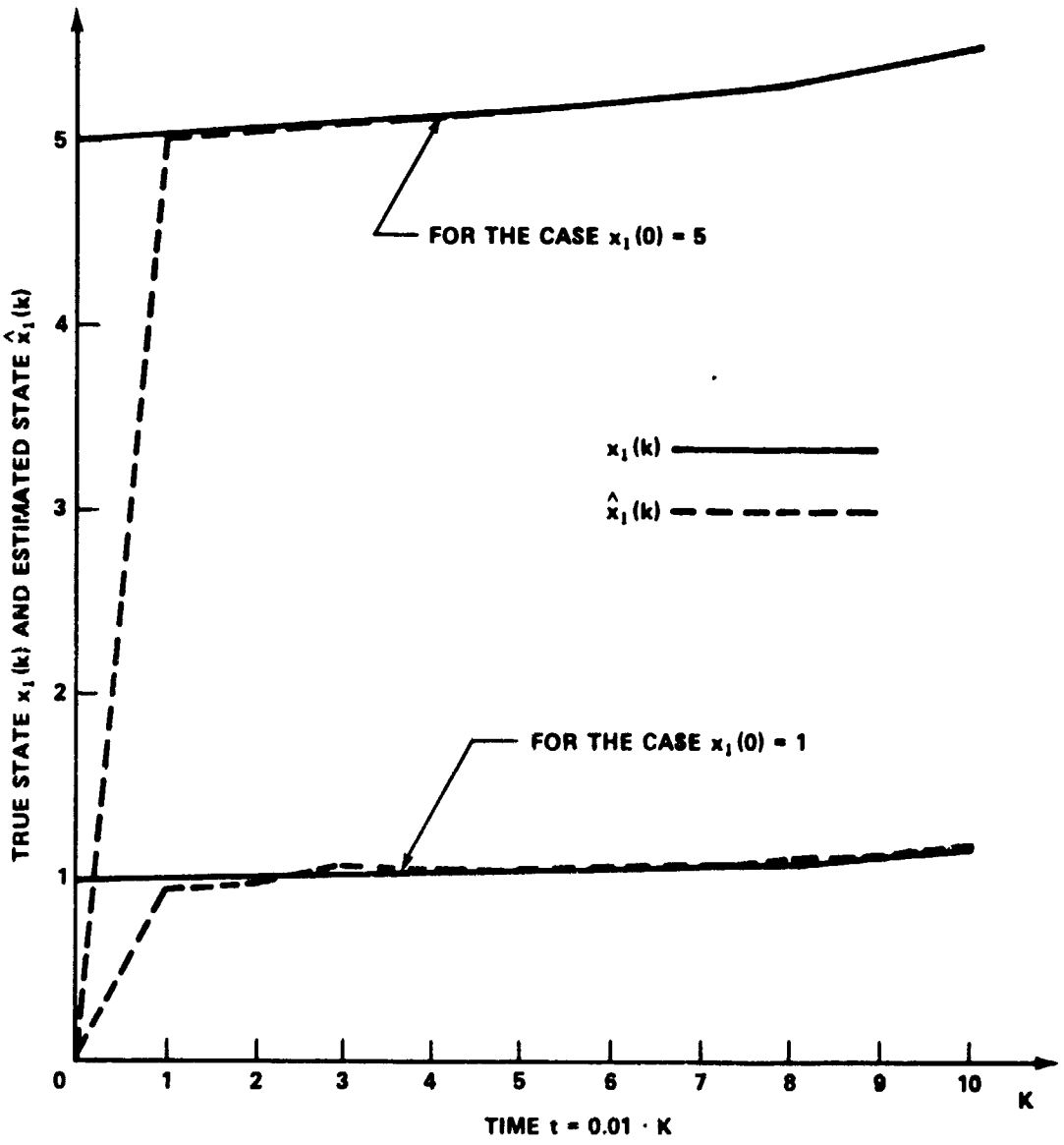

Figure 13. Comparison between true state and estimated state with $\hat{x}_1(0) = 0$.

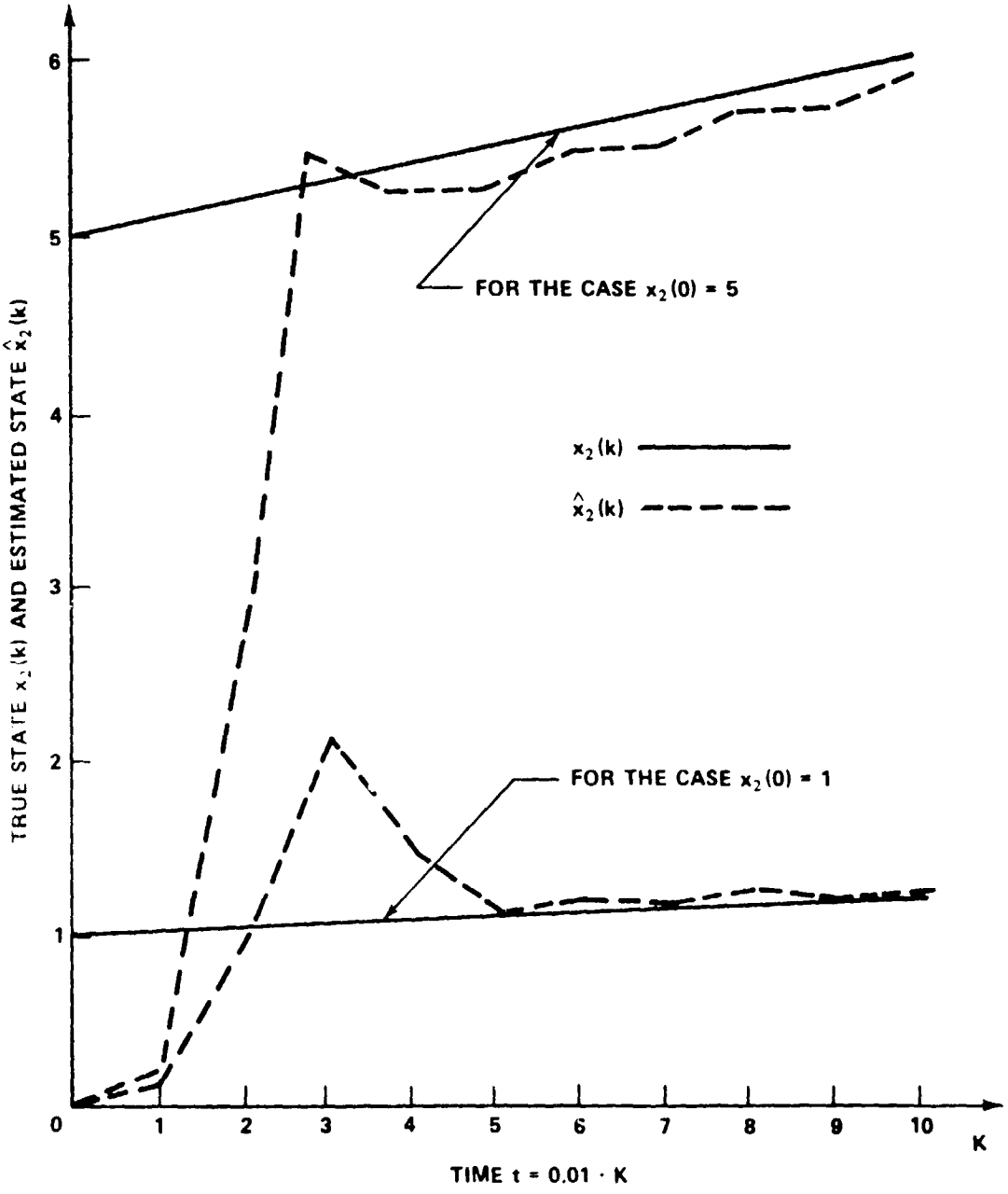

Figure 14. Comparison between true state $x_2(k)$ and estimated state $\hat{x}_2(k)$ with $\hat{x}_2(0) = 0$.

Figures 15 and 16 give the estimation errors $x_1(k) - \hat{x}_1(k)$ and rms error curves. Figure 15 presents the case with $x_1(0) = 1$ and $\hat{x}_1(0) = 0$. Figure 16 presents the case with $x_1(0) = 5$ and $\hat{x}_1(0) = 0$. It is also shown that this filter is not sensitive to the different values of the system's initial state. As expected most of these error points in Figures 15 and 16 are within the area between the two rms error curves. Furthermore, if we compare Figure 15 with Figure 9 the estimation error points match very well.

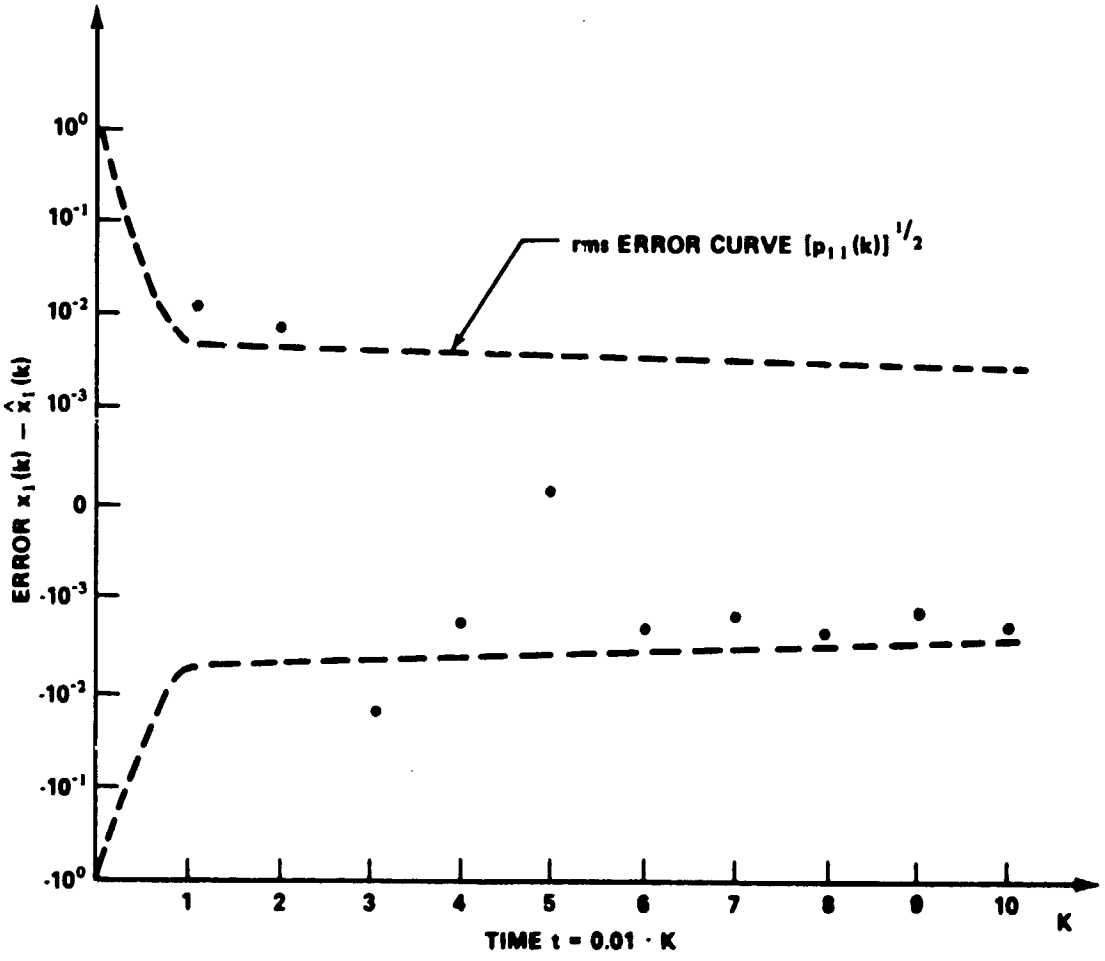

Figure 15. Estimation error $x_1(k) - \hat{x}_1(k)$ (with $x_1(0) = 1$, $\hat{x}_1(0) = 0$).

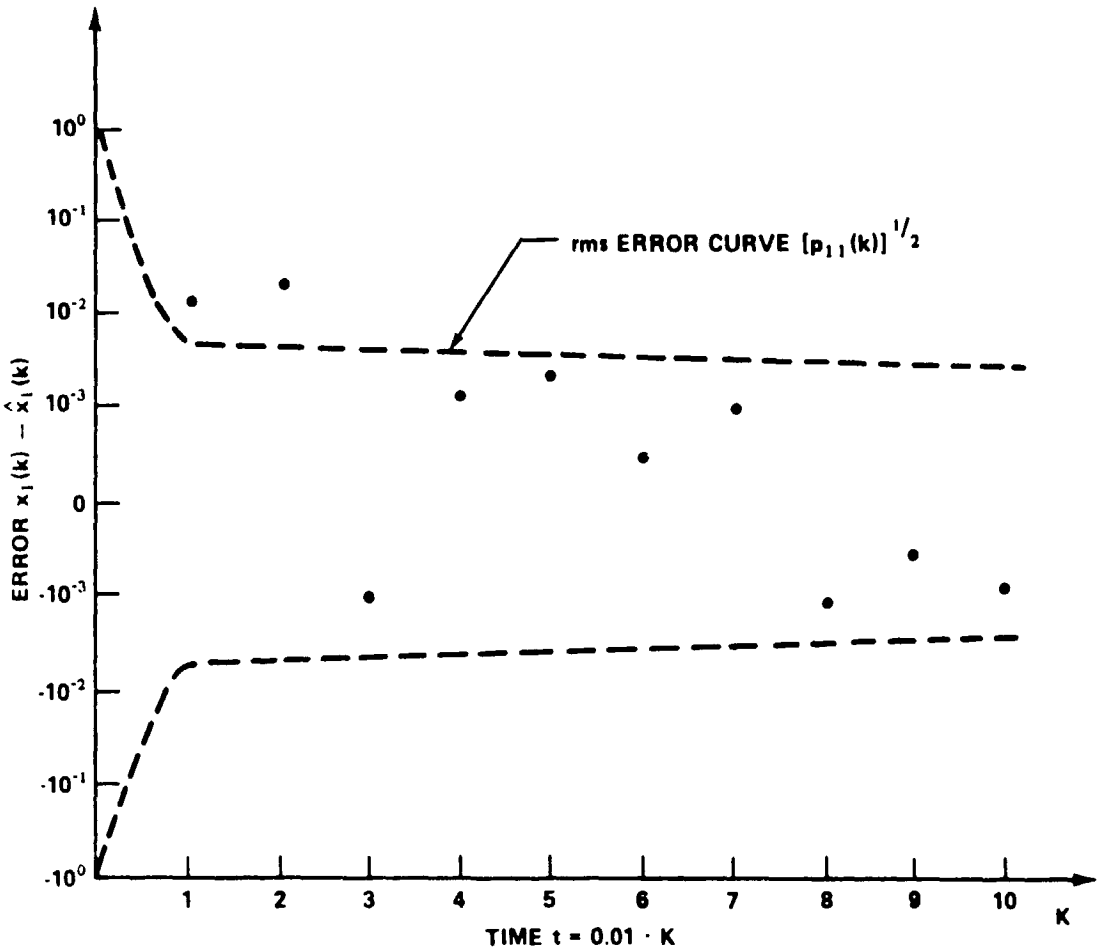

Figure 16. Estimation error $x_1(k) - \hat{x}_1(k)$ (with $x_1(0) = 5$, $\hat{x}_1(0) = 0$).

Figures 17 and 18 present the results for $x_2(k) - \hat{x}_2(k)$. The rms error curve for this case also converges with time; however, the convergence rate is not as fast as that of Figures 15 and 16. To observe the variation of this rms error curve for longer time interval, another computer run has been executed for 100 iterations and the results are shown in Figure 19. It shows that for state $x_2(k)$, the rms error curve also converges at a satisfactory rate.

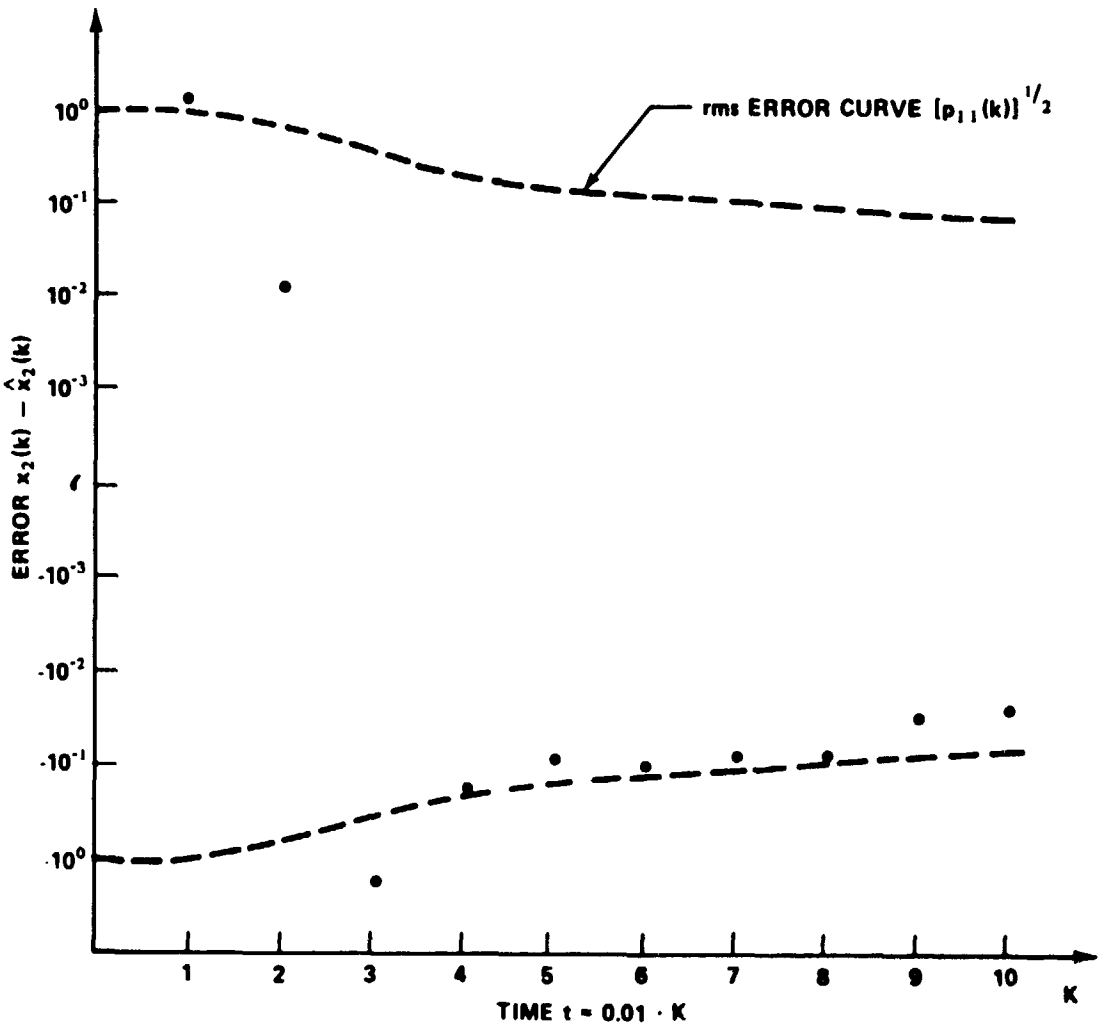

Figure 17. Estimation error $x_2(k) - \hat{x}_2(k)$ (with $x_2(0) = 1$, $\hat{x}_2(0) = 0$).

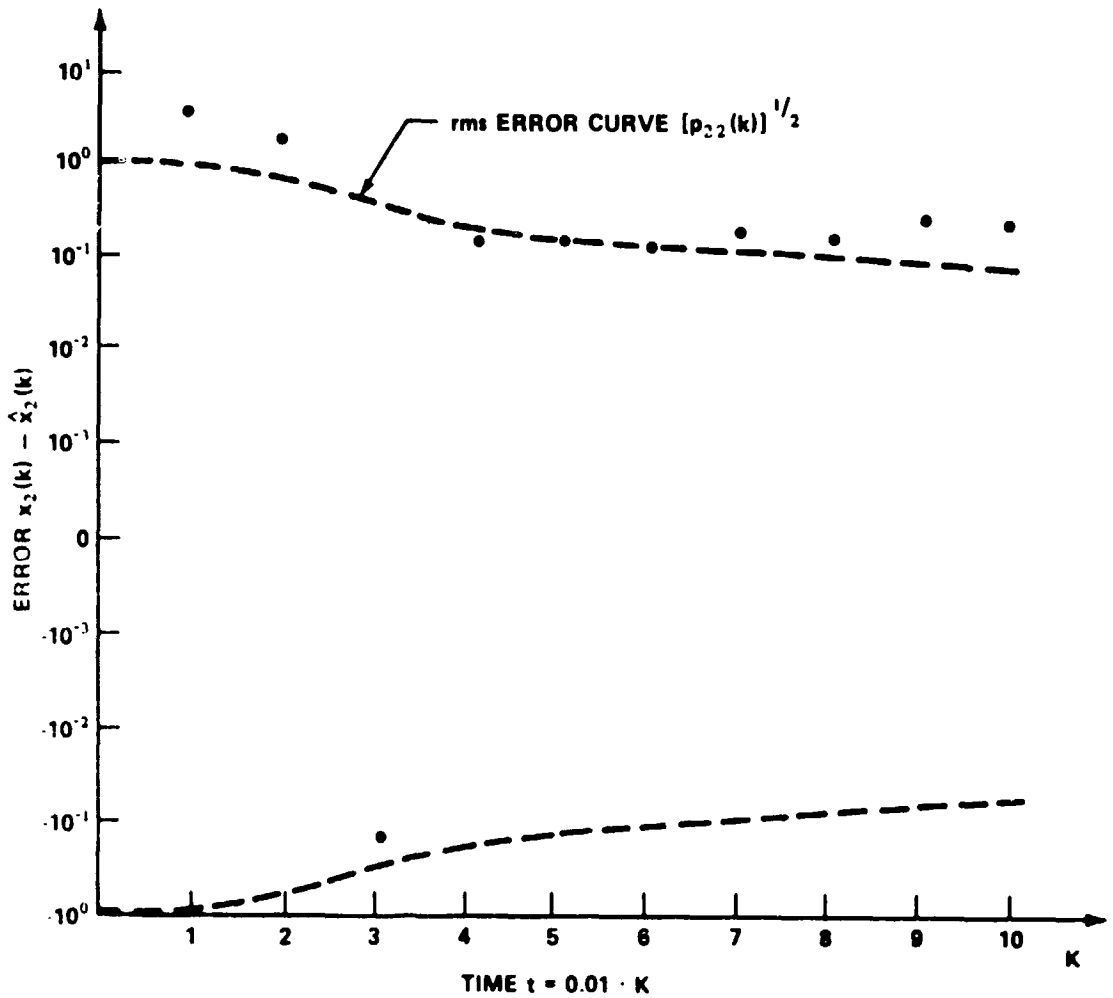

Figure 18. Estimation error $x_2(k) - \hat{x}_2(k)$ (with $x_2(0) = 5$, $\hat{x}_2(0) = 0$).

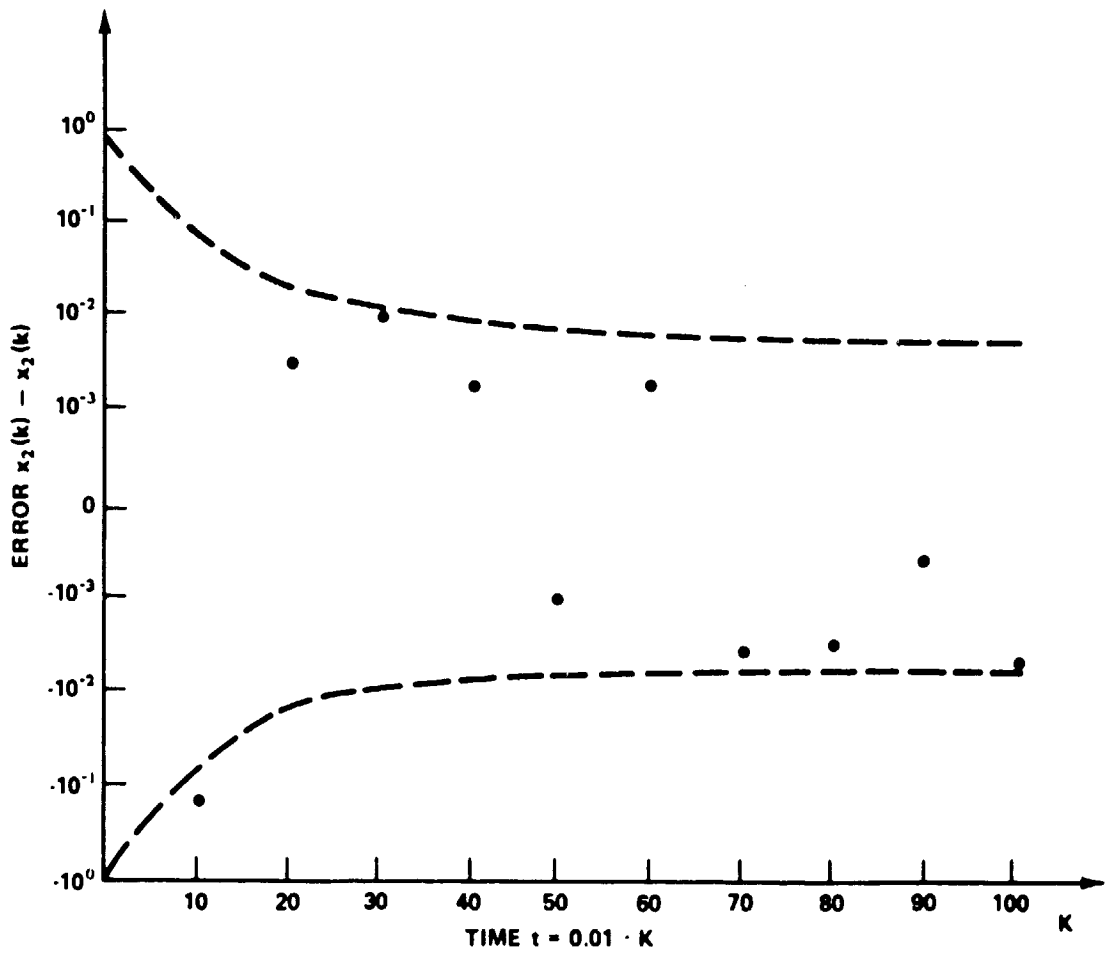

Figure 19. rms error curve and estimation error of $x_2(k) - \hat{x}_2(k)$ (with $x_2(0) = 1$, $\hat{x}_2(0) = 0$, and 100 iterations).

VIII. CONCLUSIONS

There are six BASIC programs presented in this paper. The first four programs (Programs I through IV) were developed for the construction of simulation programs of the attitude estimation problem. The main results of this research are Programs V and VI — simulation of attitude estimation of Space Telescope by statistical filters (including decomposed linear recursive filter and Kalman filter). Figures 8 through 10 and 13 through 19 show the simulation results. The advantages of the decomposed linear recursive filter are its accuracy in estimation and its speed in response time. Similar results were obtained by using Kalman filter; however, for higher order systems, the decomposed linear recursive filter has computational advantages (i.e., fewer integration errors and roundoff errors) over the Kalman filter.

REFERENCES

1. Kou, S. R.: Attitude Estimation of Earth Orbiting Satellites by Decomposed Linear Recursive Filters. NASA TM X-64943, May 1975.

2. Hamming, R. W.: Numerical Methods for Scientists and Engineers. McGraw-Hill, New York, 1962, pp. 34 and 389.

3. IBM System/360 Scientific Subroutine Package — Version III Programmer's Manual. 1970, pp. 119, 158 and 333.

4. Kou, S. R.: Optimal Control of Discrete-time Linear Dynamic Systems. Research Report at Systems Dynamics Laboratory, MSFC, NASA, April 1976.

BIBLIOGRAPHY

Hewlett-Packard 9830A Calculator, Operating and Programming Manual. 1973.

Scheid, F.: Numerical Analysis. Schanm's Outline Series, McGraw-Hill, New York, 1968.

APPROVAL

COMPUTER SIMULATION RESULTS OF ATTITUDE ESTIMATION OF EARTH ORBITING SATELLITES

By S. R. Kou

The information in this report has been reviewed for security classification. Review of any information concerning Department of Defense or Atomic Energy Commission programs has been made by the MSFC Security Classification Officer. This report, in its entirety, has been determined to be unclassified.

This document has also been reviewed and approved for technical accuracy.

James C. Blair, Chief
Control, Systems Division

J. A. Lovingood
Director, Systems Dynamics Laboratory

☆ U.S. GOVERNMENT PRINTING OFFICE 1976-641-255/578 REGION NO. 4