MONTE-CARLO EVALUATION OF DIGITAL FILTERS FOR FIRE CONTROL SYSTEMS

Toshiaki Iida

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

MONTE-CARLO EVALUATION OF DIGITAL FILTERS FOR FIRE CONTROL SYSTEMS

by

Toshiaki Iida

December 1975

Thesis Advisor: D. E. Kirk

Approved for public release; distribution unlimited.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) MONTE-CARLO EVALUATION OF DIGITALFFILTERS FOR FIRE CONTROL SYSTEMS		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis December 1975
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Toshiaki Iida		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Montrey , California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Montrey , California 93940		12. REPORT DATE December 1975
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Montrey , California 93940		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)

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

Adaptive techniques are investigated for tracking maneuvering targets from noisy measurement of position coordinates. Two types of adaptive estimators are considered , a Q-generated estimator and a Residual-Testing estimator. Monte-Carlo

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE
(Page 1) S/N 0102-014-6601 |

simulation is used to compare the performance of these adaptive estimators to that obtained by using several constant-Q filters.

Monte-Carlo Evaluation of Digital Filters for Fire Control Systems

by

Toshiaki Iida
Lieutenant (junior grade),Japanese navy

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
DECEMBER 1975

ABSTRACT

Adaptive techniques are investigated for tracking maneuvering targets from noisy measurements of position coordinates . Two types of adaptive estimators are considered , a Q-generated estimator and a Residual-testing estimator . Monte-Carlo simulation is used to compare the performance of these adaptive estimators to that obtained by using several constant-Q filters .

TABLE OF CONTENTS

I. INTRODUCTION

A state estimator for a fire control system is designed to determine accurate estimates of target position , velocity and possibly acceleration for various target maneuvering levels .

Often , to simplify the state estimator , low-order linear models are used to approximate target motion . The characteristics of a Kalman filter based on these simplified models can be modified by making different selections for the assumed random forcing function covariance matrix , Q . Intuitively , "large" Q-matrices make the estimator more sensitive to maneuvers , but also more susceptible to the adverse effects of measurement noise . Thus , a tradeoff exists between having the capability of tracking maneuvering targets and degrading the filter's performance for non-maneuvering targets .

One approach for attempting to meet these conflicting objectives is to make the filter adaptive . The fundamental idea is to use low-Q levels when the target is not maneuvering and high-Q levels when the target maneuvers . There must also be decision logic to determine which model should be used at any given time .

In Chapter II , the theory of Kalman filters is discussed and various constant-Q models are presented . Chapter III presents Monte Carlo simulation results for the constant-Q state estimators . The Q-generated adaptive filter and the residual-testing adaptive filter are discussed in Chapters IV and V , respectively . Monte Carlo simulation results for each of these adaptive filters are presented and compared with the results obtained for the constant-Q filters .

II. PROBLEM DESCRIPTION

A. KALMAN FILTER THEORY

Sequential estimation is characterized by the serial recursive processing of observations taken in time sequence. The result of every processing cycle is the current best estimate of the vector being estimated. This estimate , therefore , includes the effects of all observation data up to and including the current observation. As a new observation is made, the current estimate is updated to reflect this most recent data. In such an estimation scheme the calculations are identical in nature from cycle to cycle so they are ideally suited for implementation on a digital computer.

The Kalman filter is a recursive filter of the type applicable to a digital fire control system in which discrete observations are available from the radar or other sensors . The filter offers the capability of not only generating estimates of the observed system's states ,but also of predicting future system(or plant) states.

The linear discrete model for which a Kalman filter is designed is characterized by the state and output equations.

$$\underline{X}(K+1) = \underline{\Phi}\,\underline{X}(K) + \underline{\Delta}\,\underline{U}(K) + \underline{\Gamma}\,\underline{W}(K) \tag{1}$$

$$\underline{Z}(K) = \underline{H}\,\underline{X}(K) + \underline{V}(K) \tag{2}$$

where

$\underline{X}(K)$ is the n-dimensional state vector at time $t=KT$

$\underline{U}(K)$ is the p-dimensional deterministic input vector at time $t=KT$

$\underline{Z}(K)$ is the m-dimensional vector of measurements or observations taken at time t=KT.

$\underline{W}(K)$ and $\underline{V}(K)$ are q-dimensional and m-dimensional noise processes, respectively, at time t=KT.

$\underline{\Phi}$ is the nXn state transition matrix , which is assumed to be known .

$\underline{H}$ is the mXn observation matrix which is assumed to be known.

$\underline{\Delta}$ and $\underline{\Gamma}$ are nXp and nXq matrices,respectively, which relate the deterministic and nondeterministic forcing terms to the state vector;they are assumed to be known.

T is the time period between measurements and K is a nonnegative integer.

The noise statistics are summarized below

$$E[\underline{v}(K)]=\underline{0} \quad , \; E[\underline{v}(K)\,\underline{v}^T(J)]=\underline{R}(K)\underline{\delta}(K,J) \tag{3}$$

$$E[\underline{W}(K)]=\underline{0} \quad , \; \underline{\Gamma}\,E[\underline{W}(K)\,\underline{W}^T(J)]\,\underline{\Gamma}^T=\underline{Q}(K)\underline{\delta}(K,J) \tag{4}$$

$$E[\underline{V}(K)\,\underline{W}^T(J)]=\underline{0} \text{ for all } K,J \tag{5}$$

$$\underline{\delta}(K,J)=\begin{cases}0 & K\neq J\\ 1 & K=J\end{cases} \tag{6}$$

where

E is the expectation operator

$\underline{\delta}(K,J)$ is the Kronecker delta function

$\underline{R}(K)$ is the mXm covariance of measurement noise

matrix

$\underline{Q}(K)$ is the nXn state excitation covariance matrix

It is assumed that the initial state is a random variable with known mean and covariance

$$E[\underline{X}(0)] = \overline{X}_o, E[(\underline{X}(0)-\overline{X}_o)(\underline{X}(0)-\overline{X}_o)] = \underline{P}_o \tag{7}$$

In addition, it is assumed that the measurement noise and initial state are uncorrelated

$$E[\underline{X}(0)\underline{V}^T(K)] = \underline{0} \text{ for all } K \tag{8}$$

and that the random forcing input and the initial state are uncorrelated

$$E[\underline{X}(0)\underline{W}^T(K)] = \underline{0} \text{ for all } K. \tag{9}$$

The Kalman filter equations are summarized below

$$\underline{G}(K) = \underline{P}(K|K-1)\underline{H}^T[\underline{H}\underline{P}(K|K-1)\underline{H}^T + \underline{R}(K)]^{-1} \tag{10}$$

$$\underline{P}(K|K) = [\underline{I} - \underline{G}(K)\underline{H}]\underline{P}(K|K-1) \tag{11}$$

$$\underline{P}(K+1|K) = \underline{\Phi}\underline{P}(K|K)\underline{\Phi}^T + \underline{Q}(K) \tag{12}$$

$$\underline{\hat{X}}(K|K) = \underline{\hat{X}}(K|K-1) + \underline{G}(K)[\underline{Z}(K) - \underline{H}\underline{\hat{X}}(K|K-1)] \tag{13}$$

$$\underline{\hat{X}}(K+1|K) = \underline{\Phi}\underline{X}(K|K) + \underline{\Delta}\underline{U}(K) \tag{14}$$

where the notation $(K|K-1)$ is defined as a condition at time $t=KT$ given information up to and including time $t=(K-1)T$. The matrices in these equations are

$\underline{G}(K)$: nXm gain matrix

$\underline{P}(K|K)$:nXn covariance matrix of estimation error

$\underline{I}$:nXn identity matrix

$\underline{P}(K+1|K)$:nXn prediction error covariance matrix

$\underline{X}(K|K)$:nX1 optimal (minimum variance) estimate of X(K)

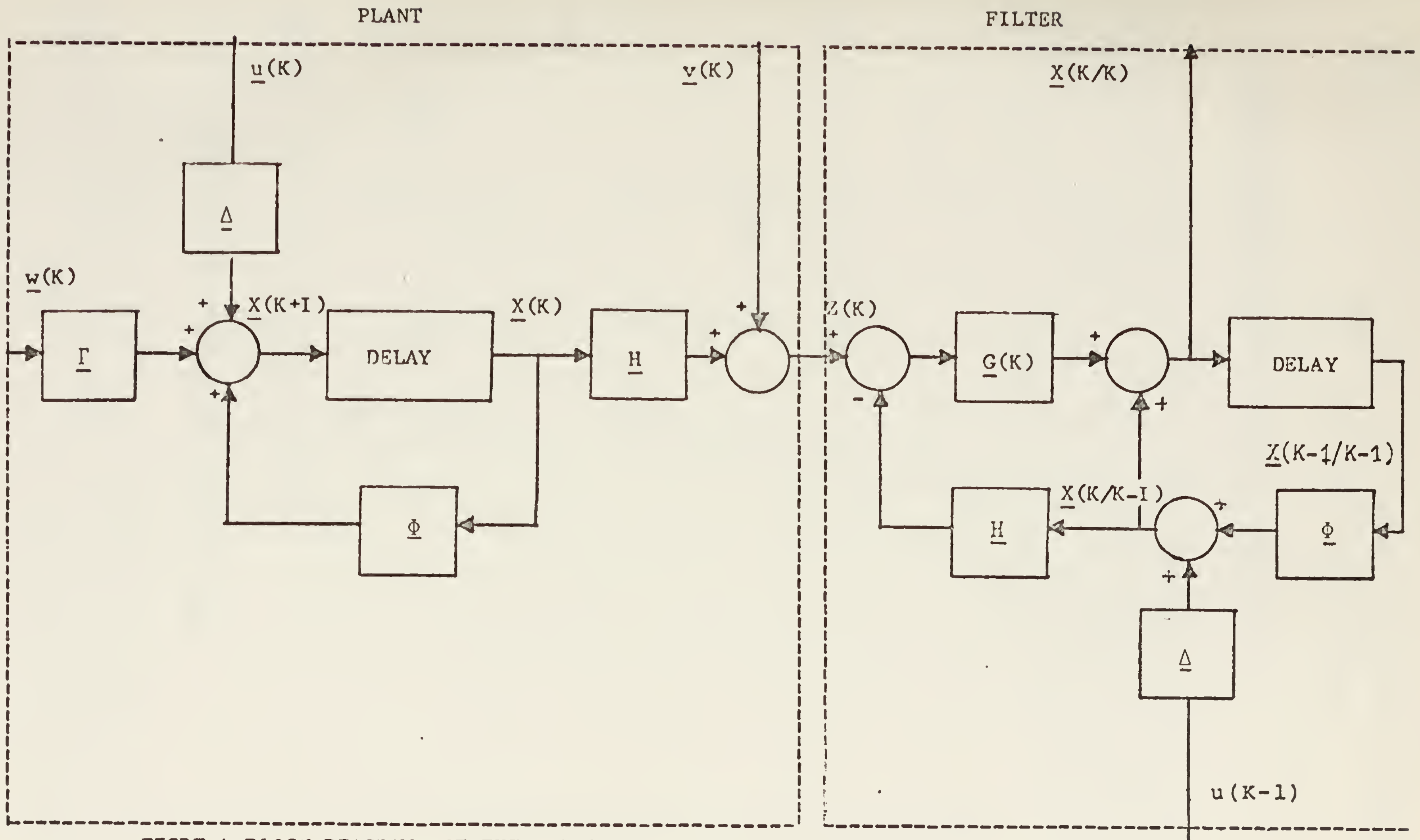
$\underline{Z}(K)$:mX1 observation vector

A block diagram of the discrete plant and Kalman filter is shown in Figure 1.

The Kalman filter takes advantage of all previous state measurements along with their respective error estimates ,and predicts ahead what the system states should be based on the state transition matrix $\underline{\Phi}$,and any known deterministic forcing input $\underline{u}$. When a new measurement becomes available , the filter takes the predicted state vector from the previous iteration , $\hat{\underline{X}}(K|K-1)$, and corrects it by some amount depending on the difference between the predicted measurement vector and the actual measurement vector . The amount of correction is a linear function of the difference and is determined by the gain matrix,$\underline{G}(K)$,which has been calculated using equations (10)-(12) so that the state estimates yield minimum variance estimates.

B. SYSTEM MODELS

In order to apply the Kalman filter to a given situation,a model for the plant must be assumed. In the case of a fire control system the type of model depends on the target. An example is the constant-velocity ($1/S^2$)



FIGRE 1 BLOCK DIAGRAM OF THE DISCRETE PLANT AND KALMAN FILTER

model which assumes that the target maintains constant velocity in all three coordinate directions.

1. THE CONSTANT-VELOCITY ($1/S^2$) MODEL

This model is based on the assumption that the target maintains constant velocity in all directions and that the motion is uncoupled . The state transition matrix and the matrix _ which relates the random forcing input to the state vector for one coordinate direction are

$$\underline{\Omega}(T) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} , \quad \underline{\Lambda}(T) = \begin{bmatrix} T^2/2 \\ T \end{bmatrix} \tag{15}$$

and for three coordinate directions

$$\underline{\Phi}(T) = \begin{bmatrix} \underline{\Omega}(T) & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Omega}(T) & \underline{0} \\ \underline{0} & \underline{0} & \underline{\Omega}(T) \end{bmatrix} \tag{16}$$

and

$$\Gamma(T) = \begin{bmatrix} \underline{\Lambda}(T) & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Lambda}(T) & \underline{0} \\ \underline{0} & \underline{0} & \underline{\Lambda}(T) \end{bmatrix} \tag{17}$$

where , T is the sampling period .

2. THE CONSTANT-ACCELERATION ($1/S^3$) MODEL

This model is based on the assumption that the target maintains constant acceleration in all directions and that the motion in uncoupled . The state transition matrix and

the random forcing input matrix for one direction are defined as

$$\underline{\Omega}(T) = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} , \quad \underline{\Lambda}(T) = \begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix} \tag{18}$$

For three coordinate directions

$$\underline{\Phi}(T) = \begin{bmatrix} \underline{\Omega}(T) & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Omega}(T) & \underline{0} \\ \underline{0} & \underline{0} & \underline{\Omega}(T) \end{bmatrix} \tag{19}$$

$$\underline{\Gamma}(T) = \begin{bmatrix} \underline{\Lambda}(T) & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Lambda}(T) & \underline{0} \\ \underline{0} & \underline{0} & \underline{\Lambda}(T) \end{bmatrix} \tag{20}$$

3. THE CORRELATED RANDOM ACCELERATION MODEL

One approach used to represent the effects of target motion involves introducing correlated random acceleration inputs to a $1/s^2$ model. The correlated random accelerations are generated by an input of white noise into a coloring filter having a transfer function of $\alpha/(S+\alpha)$. The value of can be selected to vary the time constant of the maneuver. This results in an overall transfer function for the model of target motion of

$$T(S) = \frac{\alpha}{S^2(S+\alpha)} \tag{21}$$

The state transition matrix and random forcing input matrix

for one direction are

$$\underline{\Omega}(T) = \begin{bmatrix} 1 & T & \frac{T}{\alpha} - \frac{1}{\alpha^2}(1-e^{-\alpha T}) \\ 0 & 1 & \frac{1}{\alpha}(1-e^{-\alpha T}) \\ 0 & 0 & e^{-\alpha T} \end{bmatrix} \tag{22}$$

and

$$\underline{\Lambda}(T) = \begin{bmatrix} \frac{T^2}{2} - \frac{T}{\alpha} + \frac{1}{\alpha^2}(1-e^{-\alpha T}) \\ T - \frac{1}{\alpha}(1-e^{-\alpha T}) \\ 1-e^{-\alpha T} \end{bmatrix} \tag{23}$$

For a three-dimensional coordinate system , the state transition matrix and random forcing input matrix are

$$\underline{\Phi}(T) = \begin{bmatrix} \underline{\Omega}(T) & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Omega}(T) & \underline{0} \\ \underline{0} & \underline{0} & \underline{\Omega}(T) \end{bmatrix} \tag{24}$$

and

$$\underline{\Gamma}(T) = \begin{bmatrix} \underline{\Lambda}(T) & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Lambda}(T) & \underline{0} \\ \underline{0} & \underline{0} & \underline{\Lambda}(T) \end{bmatrix} \tag{25}$$

Here the full-order transition matrix has dimension 9X9 and the random forcing distribution matrix $\underline{\Gamma}$ has dimension 9X3 .

III. MONTE CARLO SIMULATION FOR CONSTANT-Q MODEL

The Monte Carlo simulation program described in Appendix A was used to evaluate variations of the three models described in Chapter II for various values of $\underline{Q}$. All Monte Carlo simulations were done for only one-dimension and used one-hundred member ensembles .

A. SYSTEM MATRICES

The sampling period T is selected as 1.0 and this value is also used for the adaptive filters discussed subsequently . The transition matrices for each model are calculated from equations (15),(18) and (22) . To specify the measurement noise covariance matrix , R , the measurement noise standard deviation was selected as

$$\sigma = 5 \text{ m} \tag{26}$$

which makes

$$R(K) = 25 \tag{27}$$

In equation (4) , the state excitation matrix $\underline{Q}$ is defined as

$$\underline{Q} = \underline{\Gamma} \, E[\underline{W}\underline{W}^T] \, \Gamma^T \tag{28}$$

But here an additional restriction is imposed :$\underline{Q}$ is assumed to be a diagonal matrix , that is , $\underline{Q}$ is represented as

$$Q = \begin{bmatrix} q & 0 & 0 \\ 0 & q & 0 \\ 0 & 0 & q \end{bmatrix} \tag{29}$$

for the constant-acceleration model and the correlated-random-acceleration model .

The adaptive filter developed from the work of Jazwinski imposes this constraint . Thus ,the assumption that $\underline{Q}$ is diagonal is made in order to facilitate comparison among all of the adaptive filters that are considered here .

The value of q is changed and the performance of the filter is observed for various maneuvering tracks . These results are used to make comparisons with the Q-generated adaptive filter discussed subsequently and to design the Residual-testing adaptive filter discussed in Chapter V .

The Kalman filter requires that an apriori estimate of the system state vector at K=0 be made . The initial state estimate vector used in all simulation runs is

$$\underline{\hat{X}}(0|-1) = \begin{bmatrix} 50000 \\ -600 \\ 0 \end{bmatrix}$$

It is desired that the initial position measurements be used as the initial position state estimates in order to have the filter track as quickly as possible. This end is accomplished quite simply by making the diagonal elements of the initial covariance of estimation error matrix very large . The effect of doing this can be seen by examining the Kalman filter gain equation in the scalar case

$$G(K) = P(K|K-1)H^T[HP(K|K-1)H^T + R(K)]^{-1} \tag{30}$$

If $P(K|K-1)$ is made very large with respect to $R(K)$ for K=0, then we have ,for one coordinate direction

$$G(0) = \frac{P(0|-1)}{P(0|-1)+R} = 1 \tag{31}$$

which makes the first position estimate equal to the first

measurement-a logical result in the presence of a high degree of uncertainty. The initial covariance of estimation error matrix used for all simulations is

$$\underline{P}(0|-1) = \begin{bmatrix} 10^9 & & & & & \\ & 10^9 & & & 0 & \\ & & \cdot & & & \\ & & & \cdot & & \\ & & & & \cdot & \\ & 0 & & & 10^9 & \\ & & & & & 10^9 \end{bmatrix} \quad (32)$$

B. TRACK

In this thesis a simple track model is used. The total number of time points is 40 . From sampling point 1 to 10 and from 21 to 30 , the trajectory has constant velocity , from 11 to 20 and 31 to 40 the trajectory has constant acceleration . The amount of acceleration is changed tc simulate various maneuvering levels . A typical pattern for one coordinate direction is shown in Figure 2 .

In all cases ,the initial position is 60000 m , and the initial velocity is 600 m/sec . This simple track was used to facilitate the comparison of the various adaptive and nonadaptive filters .

C. RESULTS

Simulation results are plotted in Figures 3 and 4 . Only shown here are the models for which the maximum time average of position estimation error is less than 0.38 m and the maximum time average of velocity estimation error is less than 14 m/sec . Thus , only the estimators having the best performance are shown . As seen in Figure 3 , the $1/s^3$ model provides very good position estimates , but its

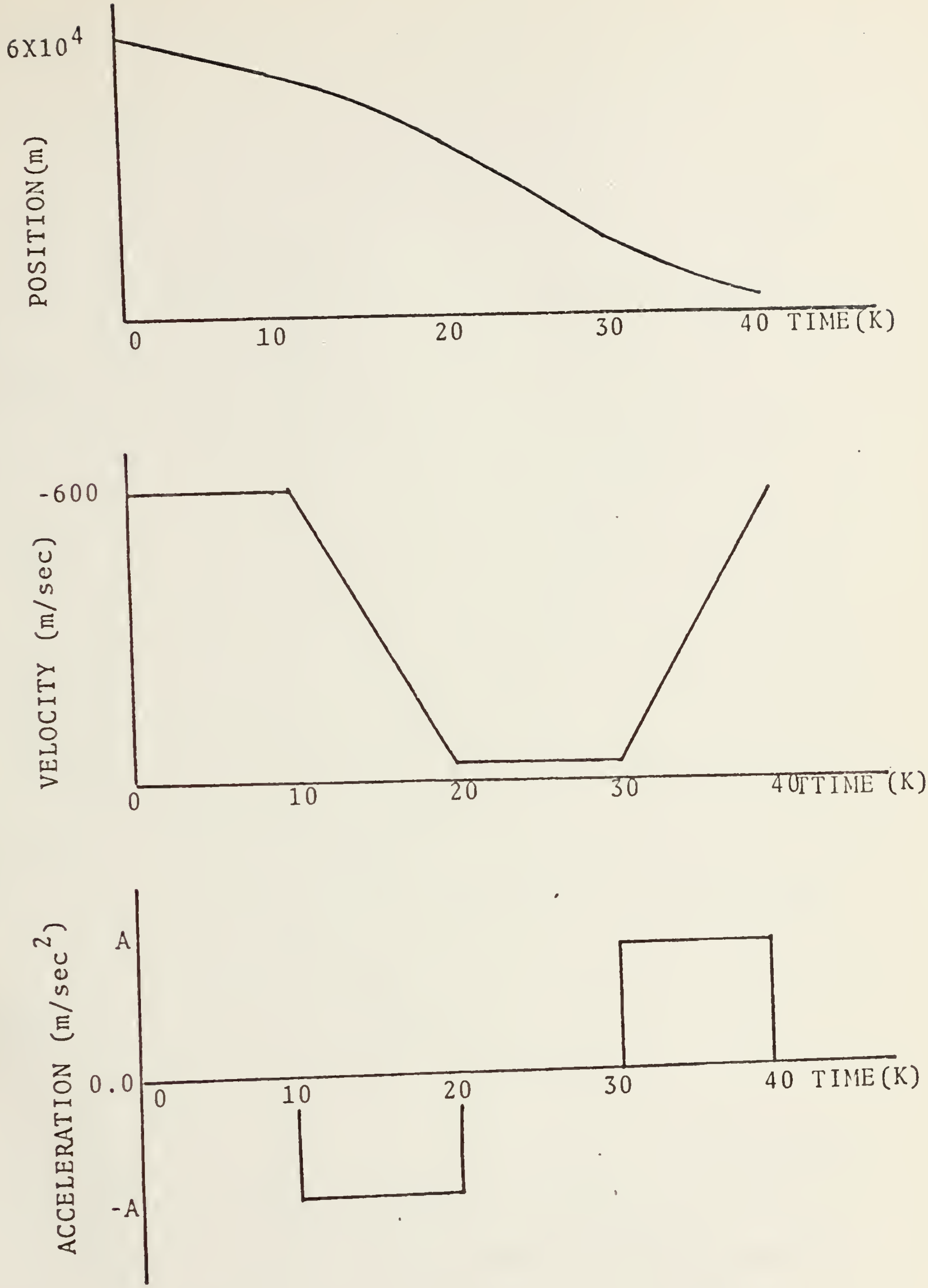


FIGURE 2 TRACK PATTERN USED FOR SIMULATION

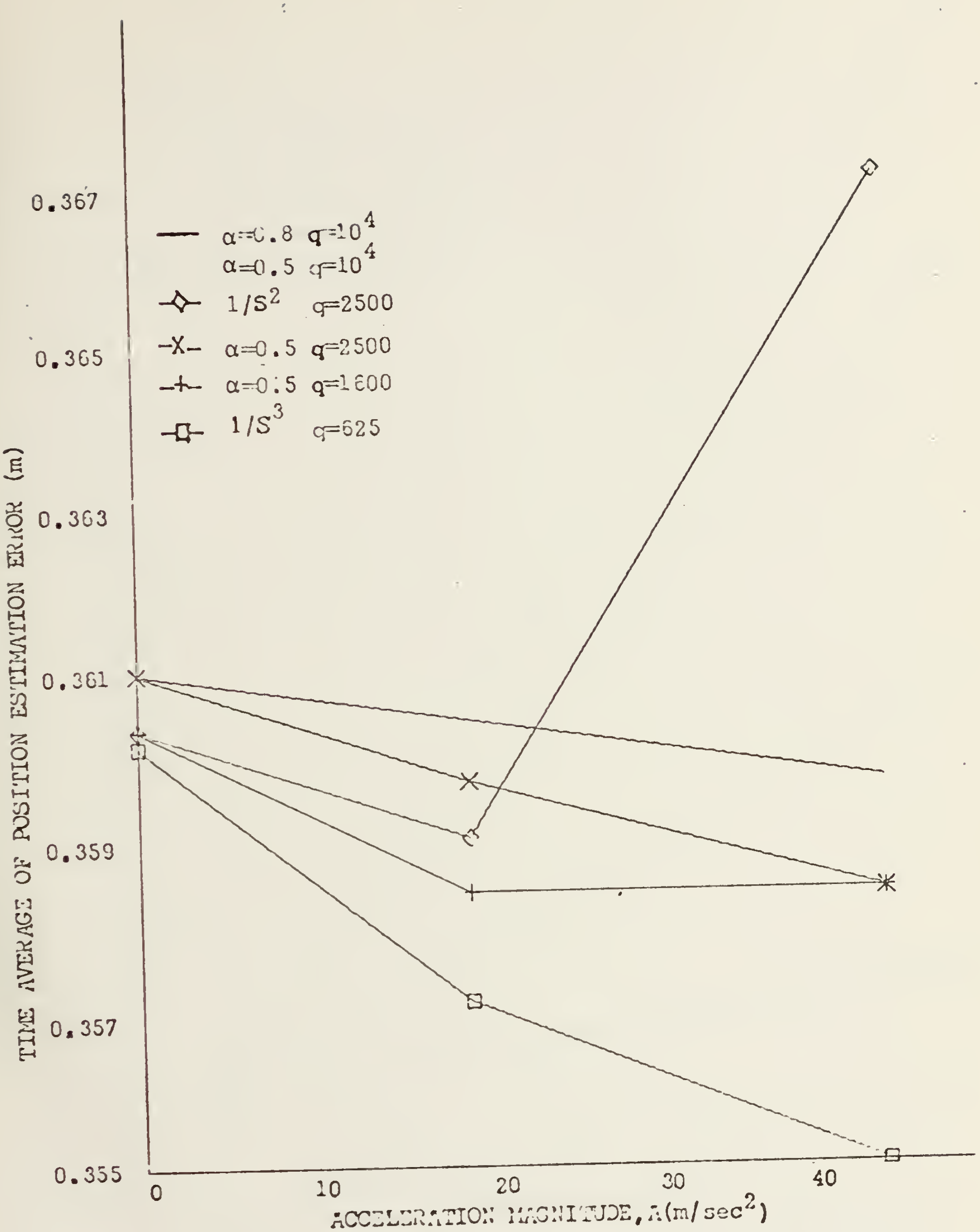


FIGURE 3 POSITION ESTIMATION ERROR FOR VARIOUS MODEL VS. MANEUVERING LEVEL FOR CONSTANT-Q FILTERS

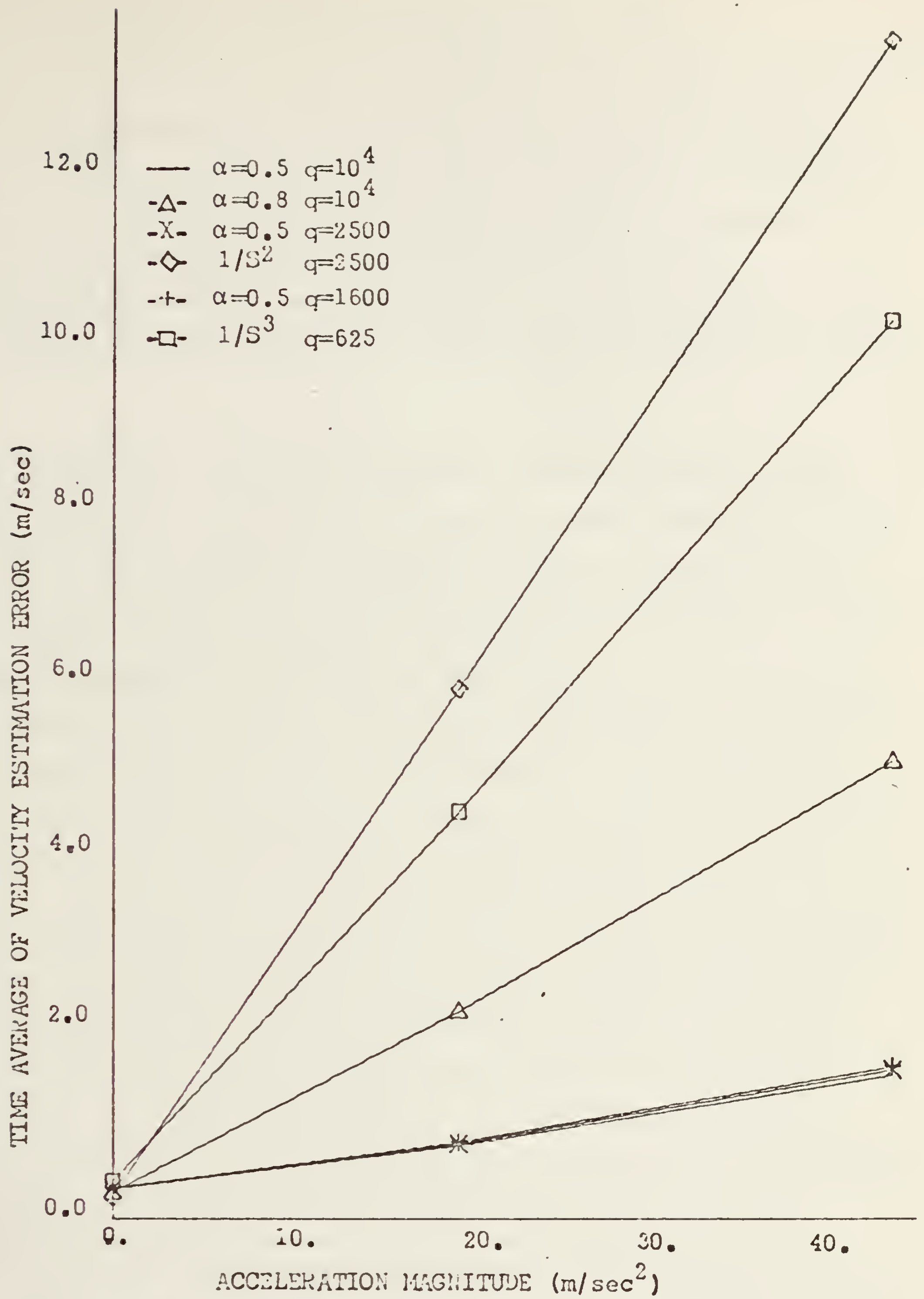


FIGURE 4 VELOCITY ESTIMATION ERROR FOR VARIOUS MODELS VS. MANEUVERING LEVEL

velocity estimation performance is poor. The correlated-constant-Q filters having q=10000 with α=0.5 and α=0.8 have the same position estimation error curve . The correlated models show relatively good performance for position estimation and α=0.5 shows excellent velocity estimation as well . The $1/s^3$ model provides accurate estimates for position and velocity for only the nonmaneuvering track , and it has poor performance for the maneuvering tracks even if the amount of maneuvering is very small.

Comparisons were made by using time averages of the absolute values of estimation errors. When looking at the estimation errors for each sampling time , It is observed that estimation errors for constant-velocity periods are not always smaller than estimation errors for constant-acceleration periods . This means that sometimes the filter provides better estimates for the maneuvering periods than for nonmaneuvering periods . The reason is that in the constant-Q model , the zero-Q filter has better performance for nonmaneuvering periods than non-zero-Q filters . Therefore , if we use a non-zero-Q filter for the track that has both nonmaneuvering and maneuvering periods , the filter provides estimates for nonmaneuvering periods that are worse than a filter with q=0 (provided that the gain schedule is prevented from reaching its steady-state value of 0) . From these facts , it is evident that we have the possibility of obtaining small estimation errors by changing $\underline{Q}$, or the gains , to match the corresponding maneuvers . The idea is to change the amount of $\underline{Q}$ as discussed in the Q-generated adaptive filter section , or to switch the gains directly for observed maneuver levels , as discussed in the section on the residual-testing adaptive filter .

D. CONCLUSIONS

It is important to consider whether position or velocity estimation errors are more important in comparing the filters . In this case , the differences in position estimation errors are less than 10 cm , but the differences in velocity estimation errors are greater than 3 m/sec . The velocity estimation error has more influence on prediction errors because velocity estimation errors are multiplied by elapsed time in the equation for position prediction estimation error . Therefore we can say that the filter which has the smallest velocity estimation error is the best filter in the context of fire control systems , if the position estimation error is small . From these considerations , the correlated model with α =0.5 is considered to have the best performance for the tracks and estimators considered .

IV. THE Q-GENERATED ADAPTIVE FILTER

A. DEVELOPMENT

There are several methods of determining random forcing input levels as a means of compensating for model inaccuracies . Discussed in this section is a method wherein the residuals themselves determine appropriate random input levels and adapt the gains accordingly [2] , [3] , [4] . It is assumed that the residual , defined as

$$r(K+1|K)=Z(K+1)-\underline{H}\hat{\underline{X}}(K+1|K) \qquad (33)$$

is a scalar and has the statistical property

$$E[r(K+1|K)]=0 \qquad (34)$$

The measurement Z(K) is known and is used to generate $\hat{\underline{X}}(K+1|K)$ using equation (13) and (14) . It is desired to use that value of the noise variance Q which produces the most probable predicted residual r(K+1|K) , as defined in equation (33). That is , the Q value which satisfies the relationship

$$q=\max f[r(K+1|K)] \qquad q>0 \qquad (35)$$

is to be found , where f is the probability density function of the residual . The restriction q>0 is consistent with the property of a variance.

The probability density in equation (35) is assumed to be zero-mean, Gaussian, with variance given by

$$\begin{aligned} Y(K+1|K) &= E[r^2(K+1|K)] \\ &= \underline{H}\underline{P}(K+1|K)\underline{H}^T+R \end{aligned} \qquad (36)$$

the maximizing q is determined by

$$q=\begin{cases} \dfrac{r^2(K+1|K)-E[r^2(K+1|K)|\underline{Q}=\underline{0}]}{\underline{H}\underline{\Gamma}\underline{\Gamma}^T\underline{H}^T} & \text{if positive} \\ 0 & \text{otherwise} \end{cases} \tag{37}$$

and

$$\underline{Q}=\underline{\Gamma}\, q\, \underline{\Gamma}^T \tag{38}$$

where

$$r^2(K+1|K)=E[r^2(K+1|K)] \tag{39}$$

if

$$r^2(K+1|K)>\underline{H}\underline{\Phi}\underline{P}(K|K)\underline{\Phi}^T\underline{H}^T+R$$

and

$$r^2(K+1|K)=0 \tag{40}$$

otherwise , since

$$E[r^2(K+1|K)]=\underline{H}\underline{\Phi}\underline{P}(K|K)\underline{\Phi}^T\underline{H}^T+q\underline{H}\underline{\Gamma}\underline{\Gamma}^T\underline{H}^T+R \tag{41}$$

and with abused notation

$$E[r^2(K+1|K)|\underline{Q}=\underline{0}]=\underline{H}\underline{\Phi}\underline{P}(K|K)\underline{\Phi}^T\underline{H}^T+R \tag{42}$$

then equation (37) is given from (39),(41) and (42) .

The linear filter with q that is estimated in (37) is the Q-generated adaptive filter for an uncorrelated and identically distributed noise input. This adaptive filter works in the following manner : as long as the square of the residual is smaller than the variance of the residual , the filter dces not generate a non-zero $\underline{Q}$, because the

residuals are small and consistent with small measurement noise levels . When the square of residual becomes larger than the variance of the residual , the filter is diverging and a non-zero $\underline{Q}$ is generated . This $\underline{Q}$ increases the value of $P(K+1|K)$, which increases the gains . Increasing the gains "opens " the filter to the incoming observation .

In Figure 5 the relationship among the residuals , the variance of estimation error and the generated q are plotted for 20 m/sec^2 acceleration with the α=0.5 correlated constant-Q filter . Between K=1 to K=10 and between K=21 to K=30 the velocity is constant ; during the intervals K=11 to 20 and K=31 to 40 the acceleration is constant .

B. SIMULATION RESULTS

For constant-Q estimators , the Q-matrix was specified by Equation (29) . For the Q-generated adaptive filter , Equation (38) gives a theoretical form of the Q-matrix , but there are other Q-matrix forms , also . Jazwinski [1] represented the Q-matrix as $\underline{Q}=q\underline{I}$, where q is defined by Equation (37) and $\underline{I}$ is the identity matrix . Another possible form is the diagonal matrix representation of Equation (38) .

In this Chapter , two form of Q-matrices are used to simulate the Q-generated adaptive filter . One is of the form $\underline{Q}=q\underline{I}$ and the other is the diagonal case of Equation (38) , that is , $\underline{Q}=\mathrm{diag}[\underline{\Gamma} q \underline{\Gamma}^T]$.

In the simulation studies the correlated filter with α=0.5 (which had the best performance as a constant-Q filter) is used . The results are compared with

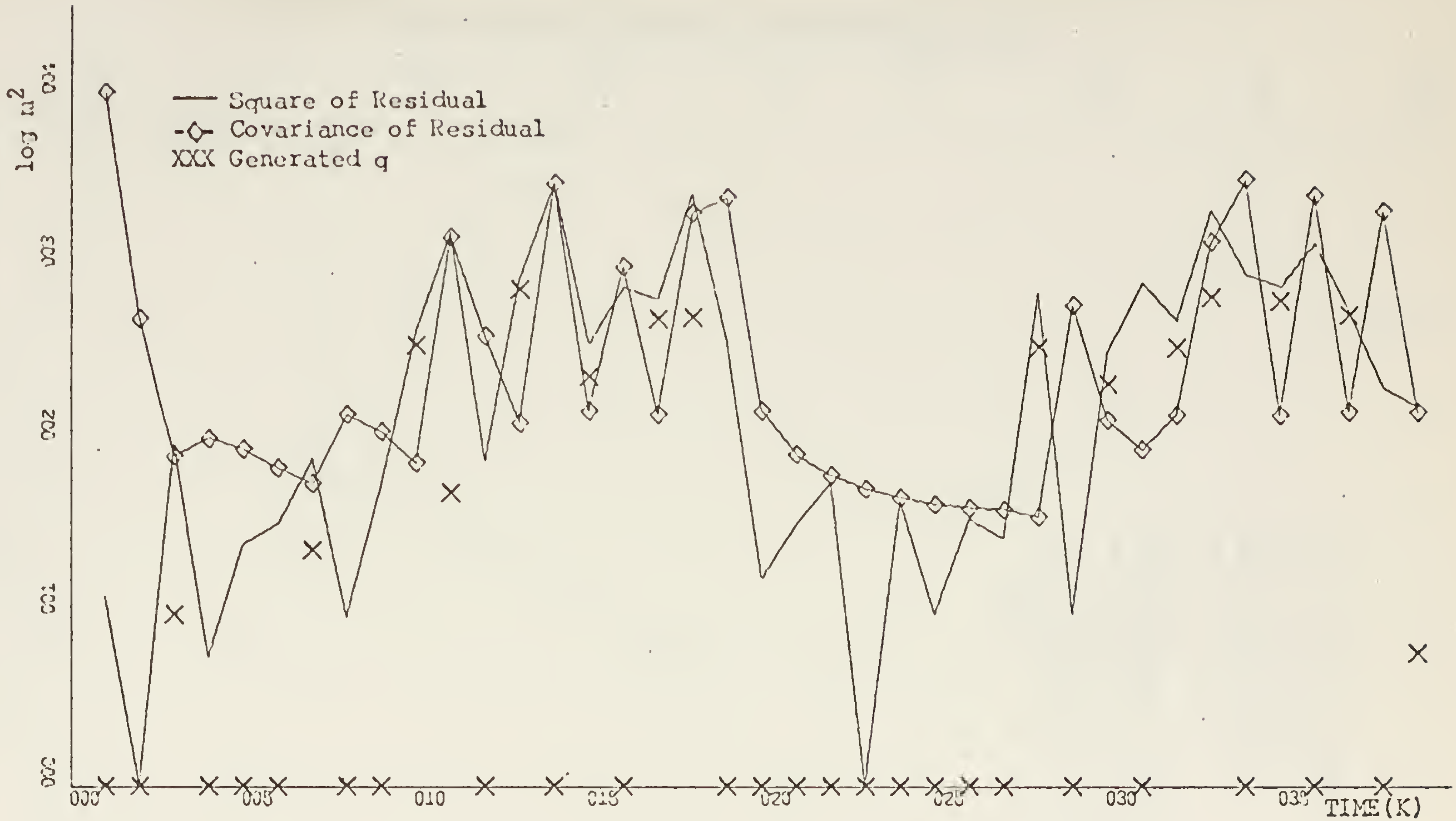


FIGURE 5 SQUARE OF RESIDUAL, COVARIANCE OF RESIDUAL AND THE RESULTING VALUE OF q GENERATED

X-SCALE=5.00E+00 UNITS INCH.
Y-SCALE=1.00E+00 UNITS INCH.

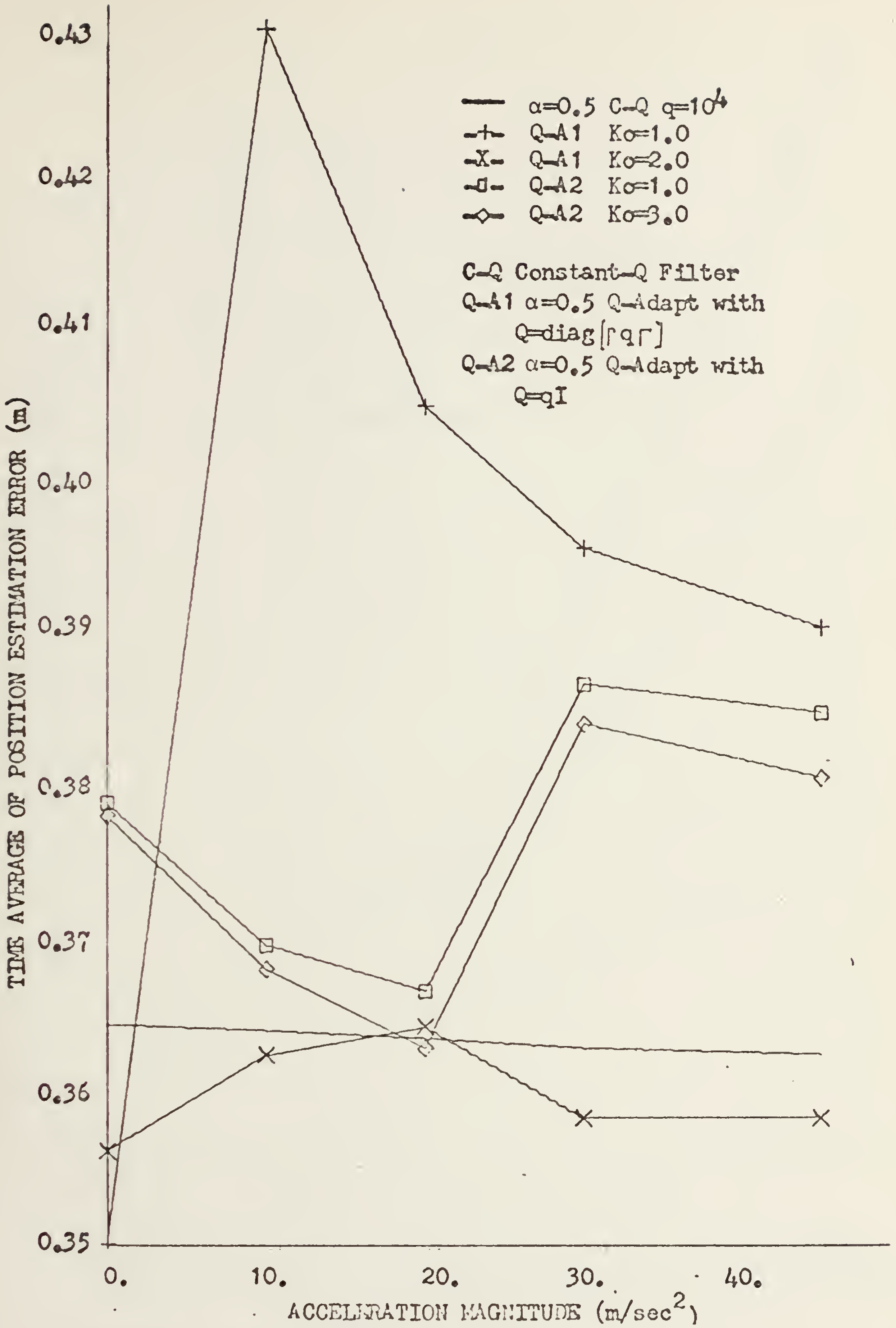


FIGURE 6 POSITION ESTIMATION ERROR FOR VARIOUS MODEL VS. MANEUVERING LEVEL FOR Q-GENERATED ADAPTIVE FILTERS

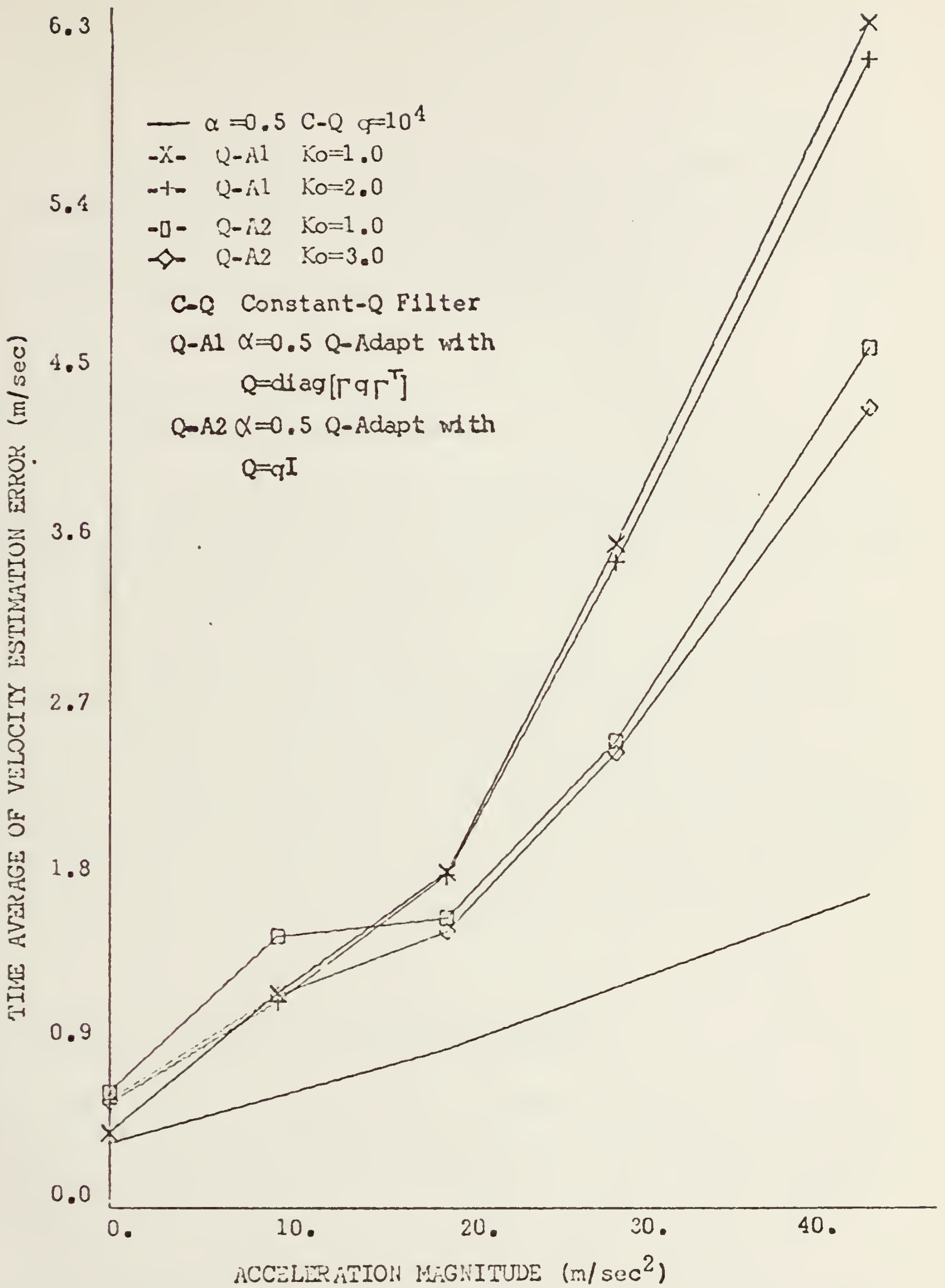


FIGURE 7 VELOCITY ESTIMATION ERRORS FOR VARIOUS MODELS VS. MANEUVERING LEVEL FOR Q-GENERATED ADAPTIVE FILTERS

constant-Q estimators , using the same track pattern , the same ensemble size and system matrices that were used for the constant-Q filter simulation in Chapter III .

Because the levels of q generated were smaller than expected and the adaptive filter did not perform as well as anticipated , the Q-generated adaptive filters were adjusted by multiplying Equation (37) by the constant Ko .

1. Simulation results with $\underline{Q}=q\underline{I}$

First , the simulation was performed to find a resonable value of Ko; it was found by trial and error that Ko=3.0 produced the best performance .

The simulation results for the Q-generated adaptive filter with $\underline{Q}=q\underline{I}$ are shown in Figures 6 and 7 . The adaptive filters provide position estimates that are comparable to , but somewhat worse than , those provided by the constant-Q filter with $\alpha=0.5$. The velocity estimation performance of the adaptive filters is significantly worse than that obtained using the constant-Q estimator .

2. Simulation results with $\underline{Q}=\mathrm{diag}[\underline{\Gamma}q\underline{\Gamma}^T]$.

An appropriate value of Ko for the correlated Q-generated adaptive filter with $\underline{Q}=\mathrm{diag}[\underline{\Gamma}q\underline{\Gamma}^T]$ was found to be Ko=2.0 . The simulation results for Ko=1.0 and Ko=2.0 are plotted in Figures 6 and 7 , and compared with the correlated constant-Q filter having $\alpha=0.5$ and the Q-generated adaptive filter with $\underline{Q}=q\underline{I}$. For Ko=1.0 , the Q-generated adaptive filter having $\underline{Q}=\mathrm{diag}[\underline{\Gamma}q\underline{\Gamma}^T]$ provides very poor performance compared with the $\alpha=0.5$ correlated constant-Q filter , especially for velocity estimates . The

adaptive filter with Ko=2.0 has good position estimates , but the velocity estimation error is essentially the same as with Ko=1.0 . Compared with the Q-generated adaptive filter having $\underline{Q}=q\underline{I}$ (α=0.5) , for Ko=2.0 , this filter has smaller position estimation errors , but greater velocity estimation errors .

C. CONCLUSIONS.

The Q-generated adaptive filters were simulated with two types of of Q-matrices . The choice of the form of the Q-matrix has great influence on the simulation results . As seen in Figure 6 and 7 , the adaptive filter with $\underline{Q}=q\underline{I}$, gives smaller estimation errors for velocity , but greater position estimation errors than for $\underline{Q}=\text{diag}\,[\underline{\Gamma}\,q\underline{\Gamma}^T]$. However , the maximum difference in position estimation was less than 5 cm , whereas the differences in velocity estimation error are more significant . From these facts , it is concluded that $\underline{Q}=q\underline{I}$ provides a better Q-generated adaptive filter than $\underline{Q}=\text{diag}\,[\underline{\Gamma}\,q\underline{\Gamma}^T]$.

Additional simulations were done for a $1/s^3$ model with the same Q-matrices that were used for the correlated Q-adaptive filter simulation . The $1/s^3$ Q-generated adaptive filter provided smaller position estimation errors with Ko=2.0 than the α=0.5 correlated Q-generated adaptive filter with $\underline{Q}=\text{diag}\,[\underline{\Gamma}\,q\underline{\Gamma}^T]$ for Ko=2.0 , but the velocity estimation errors were about seven times those of the correlated constant-Q filter with α =0.5 . In addition , there was not any significant improvement in velocity estimation errors obtained by adjusting Ko . Therefore , the $1/s^3$ Q-generated adaptive filter dose not compare favorably with the correlated constant-Q filter having α =0.5 , the best filter discussed previously .

The difficulty with the Q-generated adaptive filters considered is their velocity estimation performance . It was anticipated that the Q-generated adaptive filters would generate the smallest estimation errors . However , the best Q's for position estimation and velocity estimation are different . One reason for the mediocre performance may be that the Q is generated for only one residual measurement . Also , it is impossible to adjust the Q to yield optimal estimates for position , velocity , and acceleration at the same time . Another problem is computer time ; computer calculation time for the Q-generated adaptive filter is more than twice that of the constant-Q model , because of the on-line gain calculation . Thus , it is difficult to find a way of obtaining accurate estimates of position , velocity and acceleration simultaneously with computation time constraints . If there is no constraint on computation time and position estimation accuracy is paramount (e.g in a satellite tracking filter) , the adaptive-Q estimator might be profitably employed . However , in the context of fire control systems , other estimators considered in this investigation are superior .

V. THE RESIDUAL-TESTING ADAPTIVE FILTER

A. CONSTANT-Q FILTER PERFORMANCE.

If a constant-Q filter is used to track maneuvering targets , the performance of the filter will depend on the amount of maneuvering . To some extent the estimator can be "tuned" to tracks having similar maneuvering levels by adjusting $\underline{Q}$. However , if the tracks which the filter encounters have markedly different maneuvering levels than those for which it was designed , performance degradation results . Thus , it is virtually impossible to design a single constant-Q estimator that will perform well for tracks having a wide range of maneuvering levels . Since a constant-Q estimator can be designed to have good performance for a particular amount of maneuvering , one approach is to design several constant-Q filters and switch filters based on the size of the residual . This concept is illustrated in Figure 8.

To determine the performance of filters for various maneuvering levels , simulations were done for several constant-acceleration tracks , shown in Figure 9 , with the correlated filter having α=0.5 and the same system matrices that were used for the constant-Q filter simulations in Chapter III . The tracks used for the simulations all have the initial conditions of 60000 m for position , -600 m/sec for velocity , and the amount of acceleration B is constant during the run . The number of sampled points is twenty , which is long enough for the gains to have reached steady state . An ensemble size of one-hundred is used for the Monte Carlo simulations .

The results of the Monte Carlo simulations are plotted in Figures 10 and 11 . From Figure 10 , the minimizing q

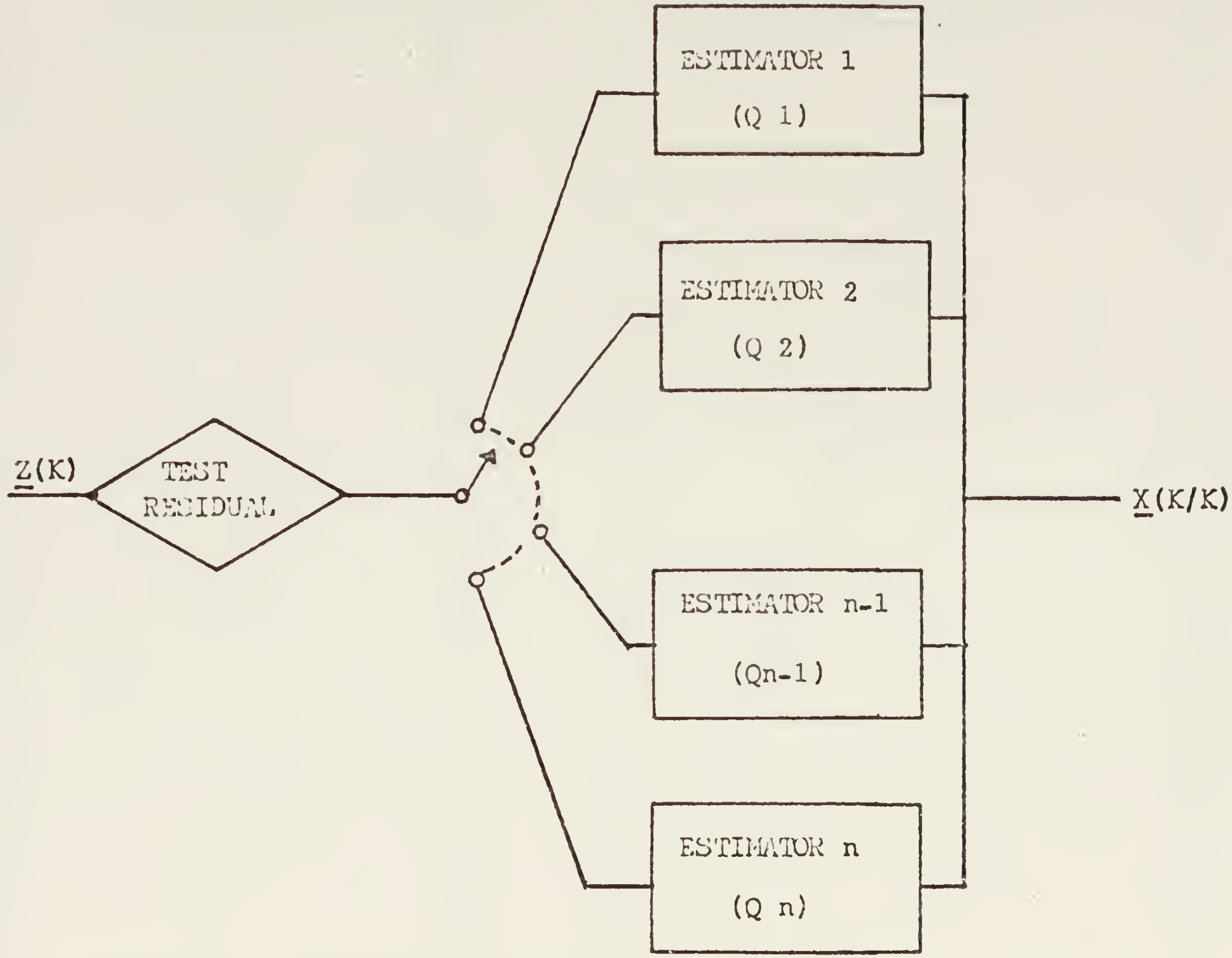


FIGURE 8 SIMPLE MODEL OF RESIDUAL-TESTING ADAPTIVE FILTER

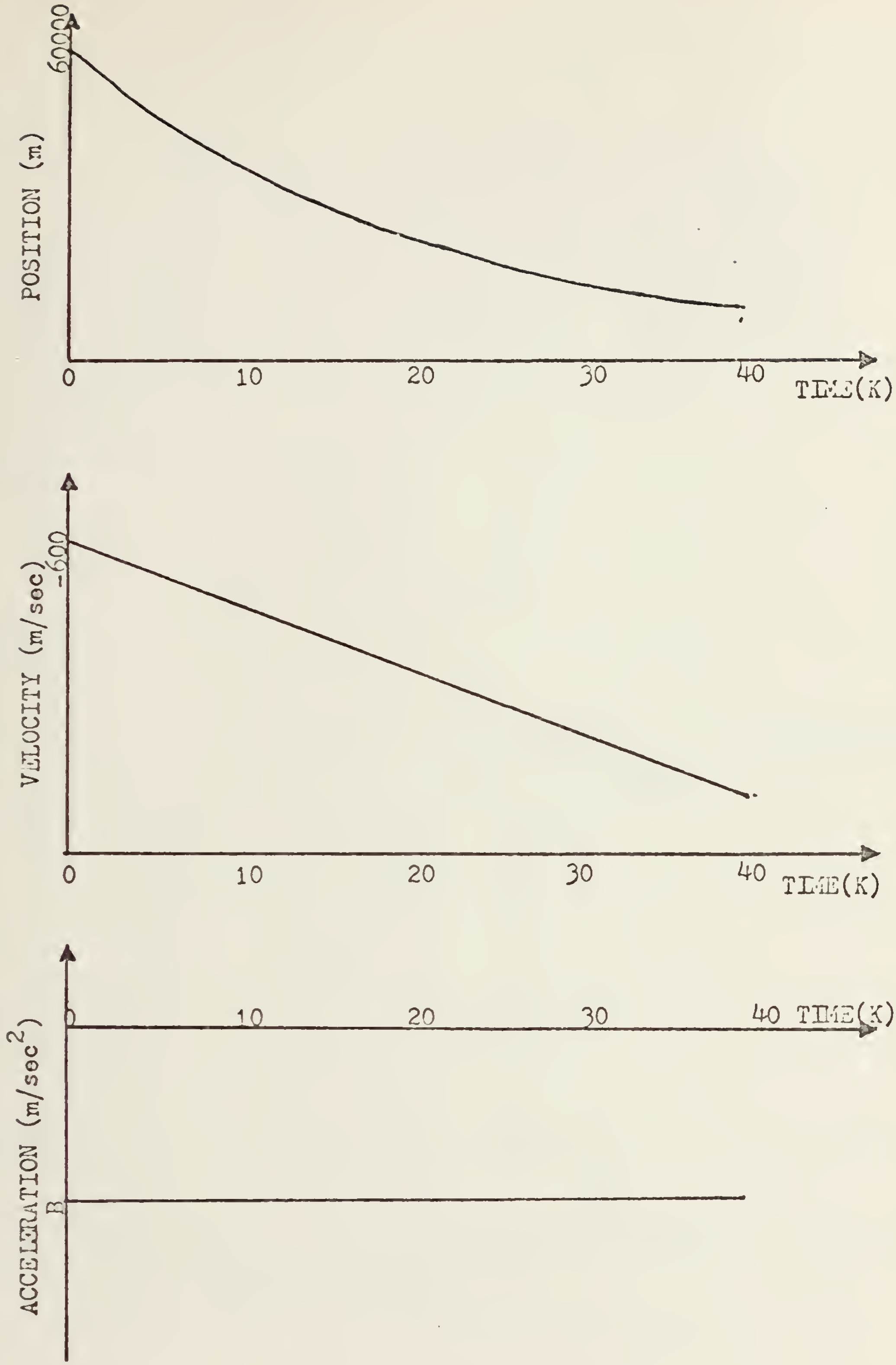


FIGURE 9 TRACK PATTERN FOR CONSTANT ACCELERATION SIMULATION

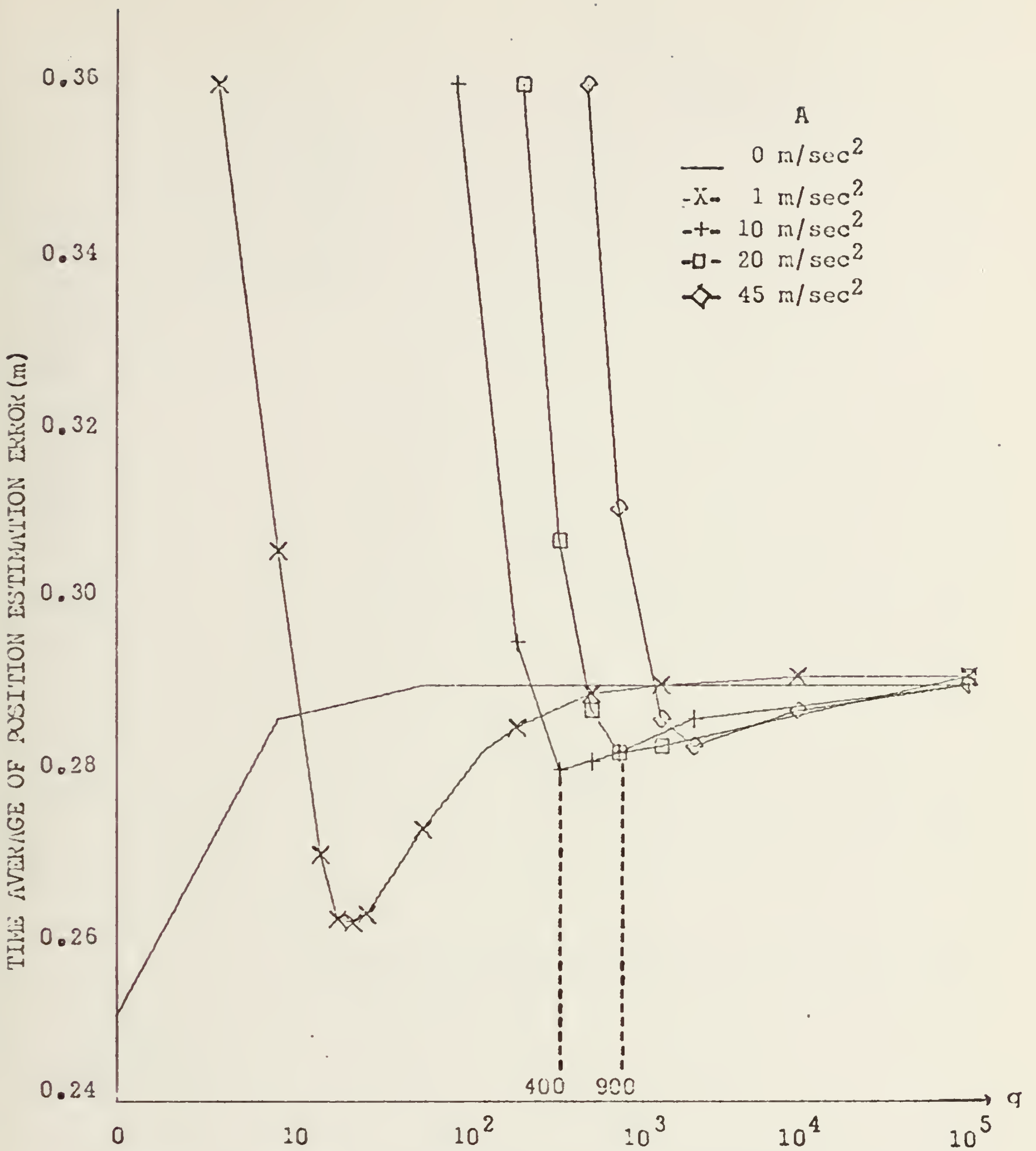


FIGURE 10 POSITION ESTIMATION ERROR DEPENDENCE ON q FOR CONSTANT MANEUVER-LEVEL, A.

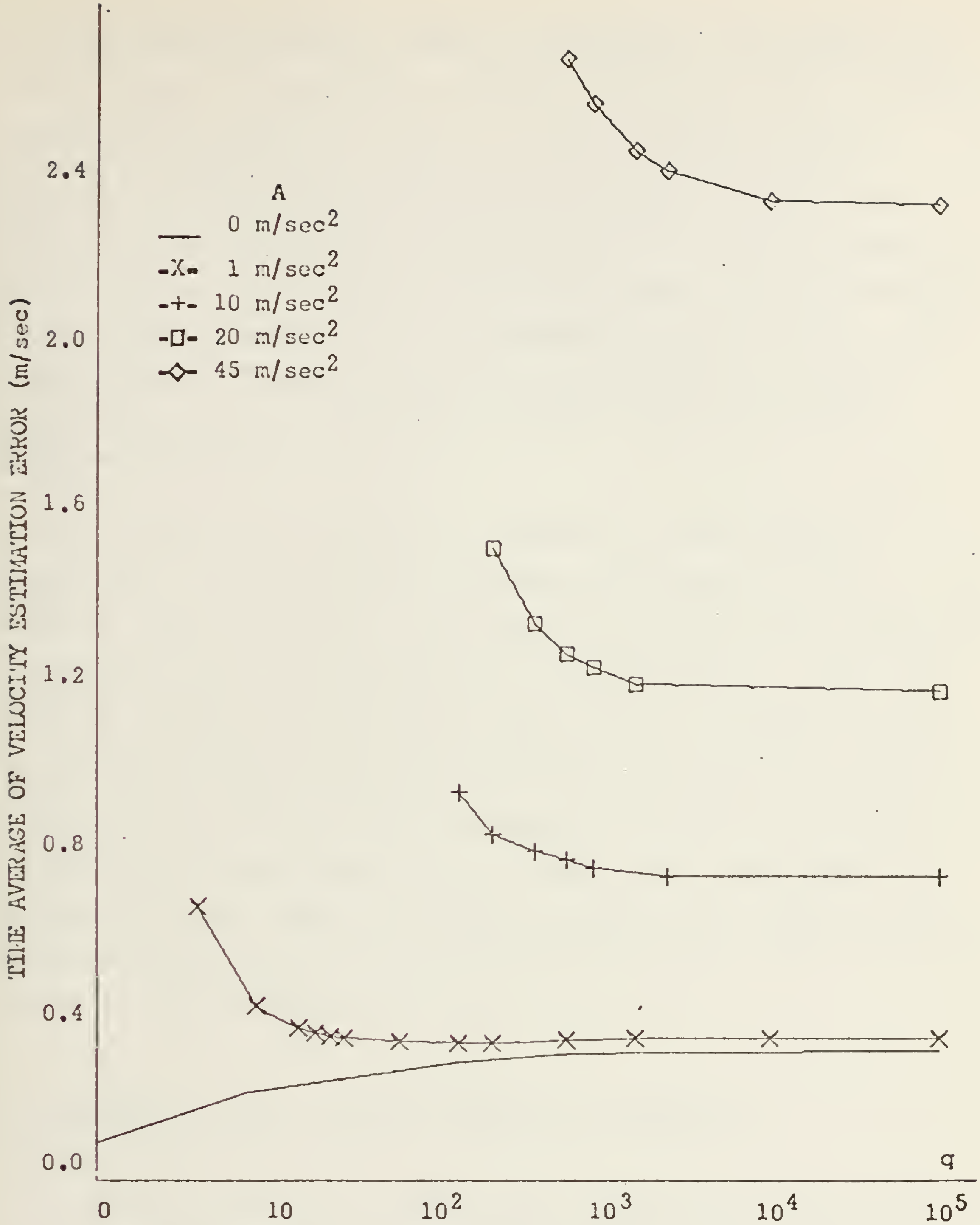


FIGURE 11 VELOCITY ESTIMATION ERROR DEPENDENCE ON q FOR CONSTANT MANEUVER-LEVEL, A.

for position estimation error is observed to be q=400 for the 10 m/sec^2 acceleration track , q=900 for the 20 m/sec^2 acceleration track and so on . For q larger than the minimizing value the position estimates are degraded slightly ; for q smaller than the minimizing value the position estimates degrade significantly . On the other hand , the velocity estimation errors monotonically decrease as q increases (except for a nonmaneuvering target). If the curves in Figure 11 were obtained for still larger values of q , it is anticipated that they would show a minimum eventually .

An additional factor to consider is the relative importance of position and velocity estimation errors . To predict future position , velocity errors are more important , provided the prediction time is significant . This is because position prediction errors tend to be proportional to velocity estimation errors multiplied by elapsed time . Thus , if a target is identified by some means as having an acceleration of 20 m/sec^2 , Figures 10 and 11 indicate q=1600 as a reasonable choice . Note that this value is slightly larger than the value of q=900 which yields the minimum time -average of position estimation error for this level of maneuvering .

B. DETECTION AND CLASSIFICATION OF RESIDUALS

The first step in designing a residual-testing adaptive filter is to provide some mechanism for determining the maneuvering level which corresponds to observed residual values . To determine a resonable residual-testing procedure ,it is necessary to investigate the characteristics of residuals .

1. Investigation of the characteristics of residuals .

As a mechanism to accomplish the detection and classification of residuals , it is resonable to use the time average of residual absolute values . Using the minimizing values of q for position estimation error , the time averages of residual absolute values were computed in the constant-acceleration track simulation described in Section A . The results are plotted in Figure 12 . Based on the curve in Figure 12 , a residual-testing adaptive filter (Filter-0) was synthesized and simulated . The filter was the correlated α =0.5 estimator , the system matrices and tracks used for the simulation were the same as for the constant-Q filter simulation in Chapter III . Figure 13 illustrates the use of the data in Figure 12 in designing an adaptive filter . The information in Figure 13 is used in the following manner : a calculated residual value at time K is used to enter the graph on the ordinate and the corresponding q value is read from the abscissa . For example , if the residual is 32.0 , the value of q used to determine the next gain value is q=900 . The level settings for the residuals and the specified q values shown in Figure 13 were obtained by subjective evaluation of the information in Figures 10-12 .

The simulation results for this filter were poor , but it was helpful in observing the characteristics of residuals . As seen in Figure 12 , the time average of the residual absolute value for a nonmaneuvering target is 6.15 . Thus , Filter-0 assumes that the track is in a nonmaneuvering period whenever the residual is less than 6.15 , and provides the gains which correspond to zero-Q to the system . However , frequently residuals were observed to be less than 6.15 , even if the track was maneuvering . It is apparent from Figure 10 and 11 , that the zero-Q

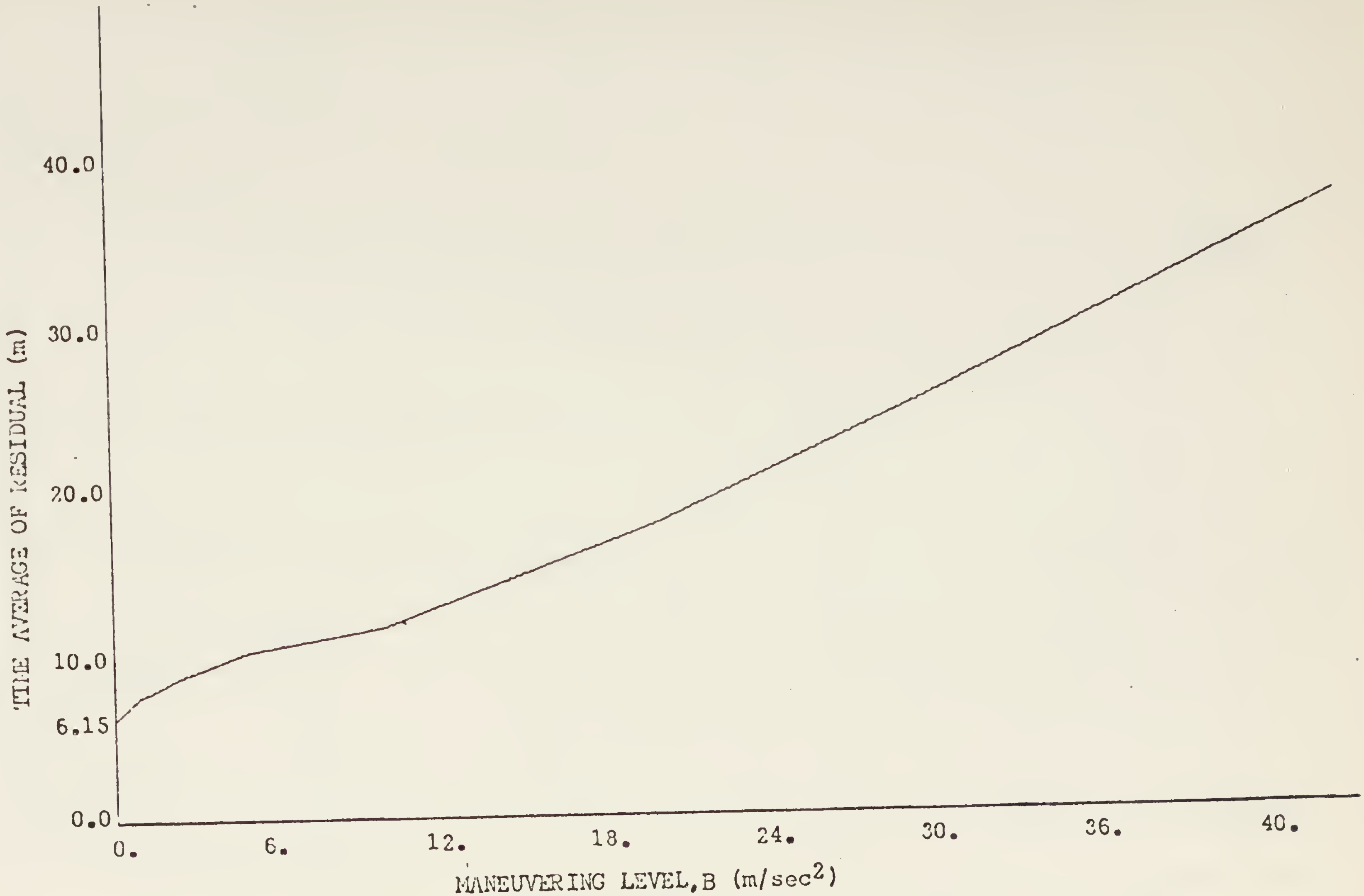


FIGURE 12 THE TIME AVERAGE OF RESIDUALS VS. MANEUVERING LEVEL

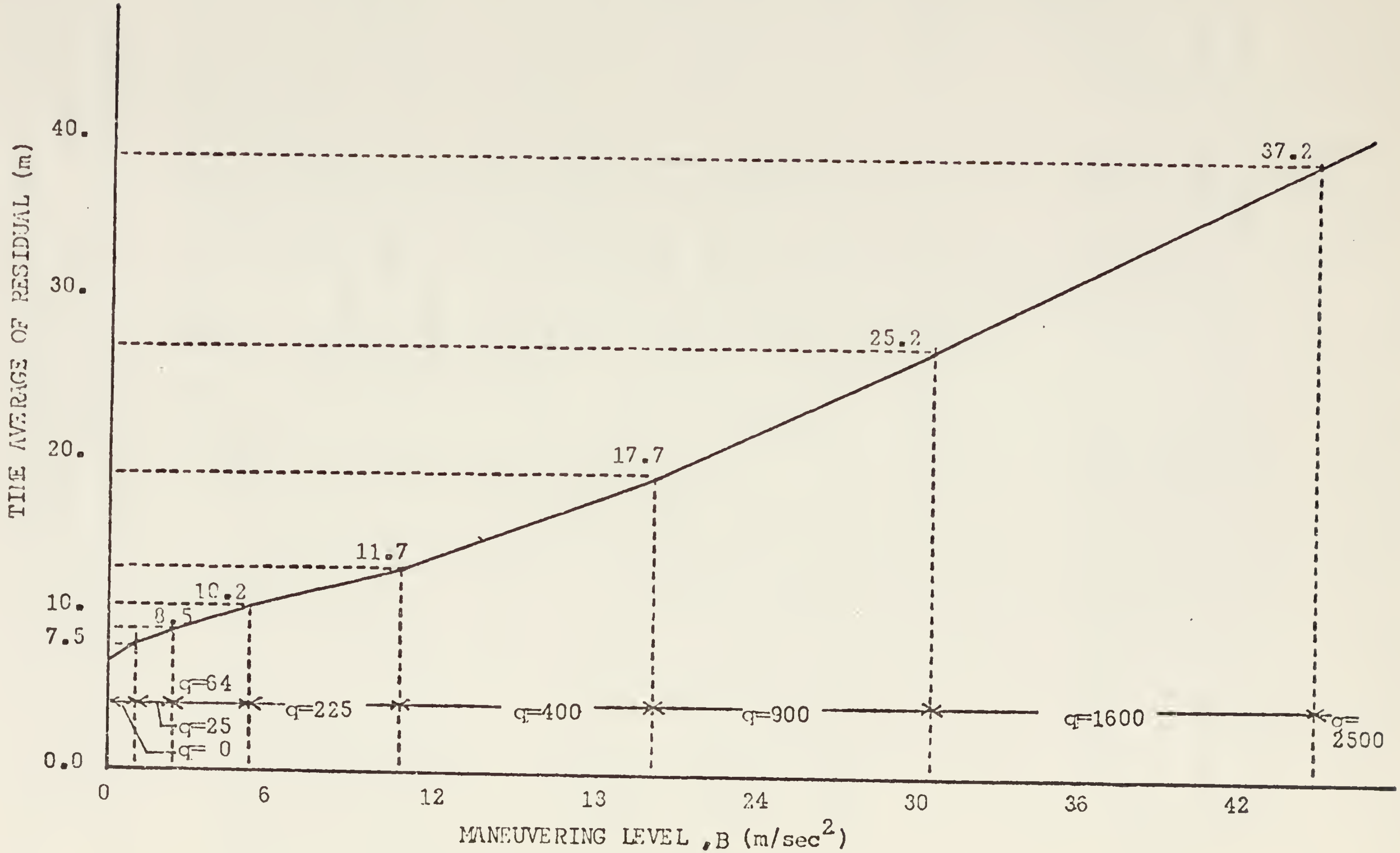


FIGURE 13 LEVEL SETTING FOR RESIDUAL AND q SELECTION FOR FILTER-0

gains generate large position and velocity estimation errors for maneuvering tracks ; this is the reason that Filter-0 provided such poor performance .

From the simulation for Filter-0 , it was found that the residuals cannot be detected and classified sufficiently well by using only the current value of the residual . Thus the residual testing mechanism tested in Filter-0 , required improvement . This point is addressed in the following section .

2. Residuals and acceleration estimation error.

In the previous section , it was found that the residuals are often less than the measurement noise level , even if the target is maneuvering , and that the time averages of residual absolute value do not provide enough information to adequately detect and classify the residuals . Thus , for the numerical values used here , the measurement noise does not influence the residuals as much as target acceleration does . Because of this observation , the acceleration estimation error for the constant-acceleration track was next considered .

For the constant-acceleration track simulations in Section A , it was observed that the acceleration estimation error for a constant-acceleration track reached constant values (the steady-state values of acceleration estimation error)as K becomes large . These steady-state acceleration values were computed in the simulation described in Section A , and are shown in Figure 14 .

The steady-state acceleration estimation error curve is almost parallel to the time average of residual absolute value curve , and the difference between these two curves is

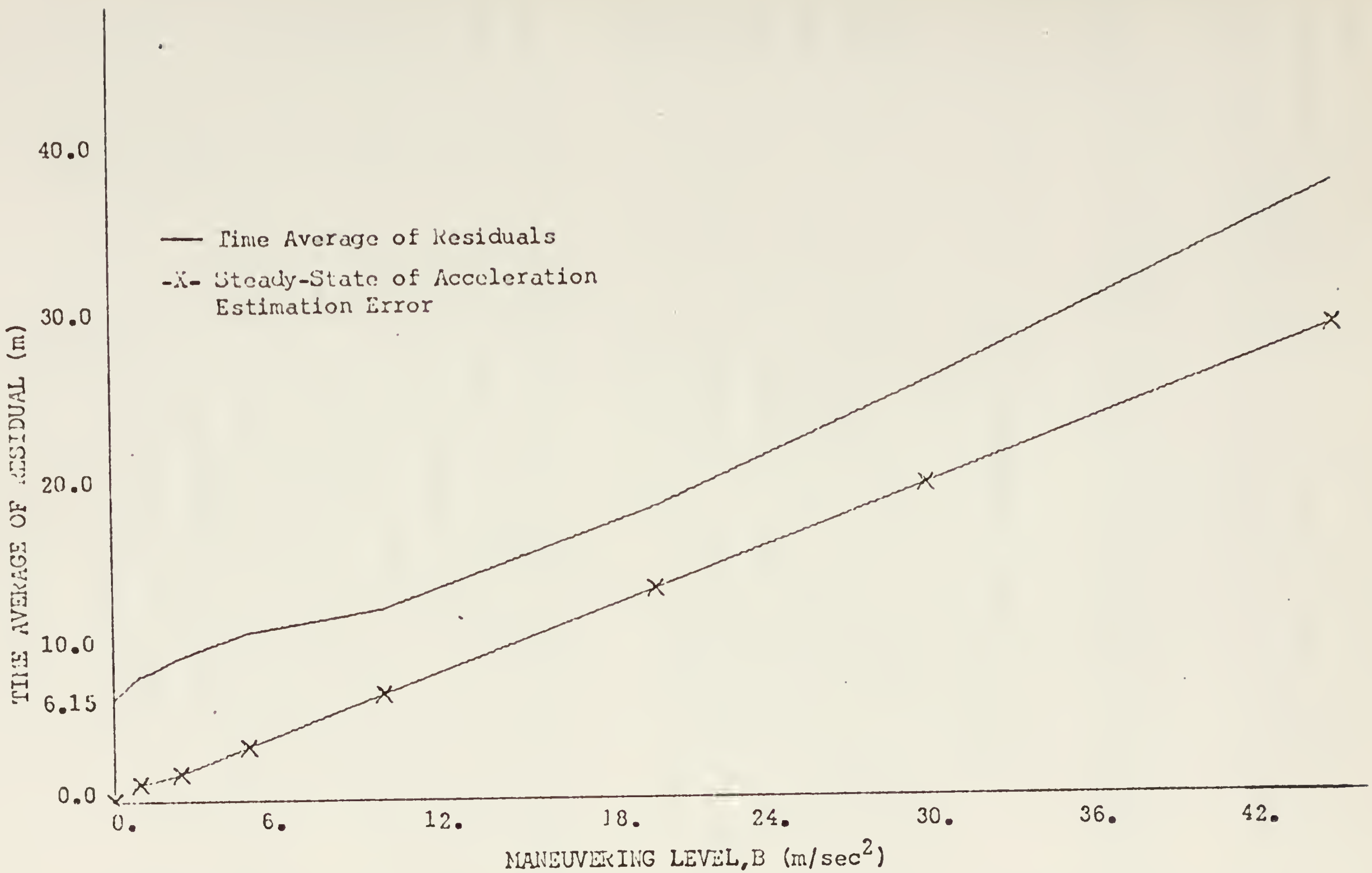


FIGURE 14 THE STEADY-STATE OF ACCELERATION ESTIMATION ERROR AND THE TIME AVERAGE OF RESIDUALS VS. MANEUVERING LEVEL

close to the value of the standard deviation of measurement noise level (5 m) . From this , it can be inferred that the steady-state acceleration estimation error can be obtained by subtracting the standard deviation of measurement ncise from the time-average of residuals .

Thus the steady-state values of acceleration estimation errors can be used to detect and classify the residuals .

3. A modification of the Residual-Testing Filter

In Sections B.1 and B.2 , two viewpoints concerning residuals were examined . The first approach assumes that the residuals are significantly influenced by measurement noise and attempts to classify the residuals by using the time average of their absolute values (see Figure 14) . A second possibility is ignore the effects of measurement ncise on residuals and attribute the residual values entirely to target acceleration .

One way to incorporate both of these observations in the filter is to utilize an adaptive switching scheme 5 which operates as follows : if two consecutive residuals are less than a selected threshold level , the gains are set to the zero-Q gains and it is assumed that the track is in a nonmaneuvering period ; otherwise , the residuals are equated to the steady-state values of acceleration estimation errors shown in Figure 14 (by assuming that the residual is negligibly influenced by the measurement noise) and the values of q are determined accordingly . The switching level threshold used for this scheme , 7.0 , is obtained from the time average of residuals for zero-acceleration in Figure 12 .

C. THE SELECTION OF Q.

The second step in designing the residual-testing adaptive filter is to select an appropriate value of Q for each acceleration level . To simplify the implementation of the filters , the steady-state gain values were used . Two sets of threshold selections were made by arbitrarily quantizing the steady-state acceleration dependence on q illustrated in Figure 14 .

D. FILTER A

Filter A uses the threshold levels given below . Two consecutive residuals having absolute values of less than 7.0 causes the zero-Q gains to be selected . Otherwise , the gains corresponding to the q values below are used .

Filter A

Acceleration (m/sec^2)	Residual value (m)	Selected q
0.0 - 1.0	0.00 - 1.00	0
1.0 - 2.5	1.00 - 1.67	64
2.5 - 5.0	1.67 - 3.25	144
5.0 - 10.0	3.25 - 6.41	400
10.0 - 20.0	6.41 - 12.75	625
20.0 - 30.0	12.75 - 9.08	1600
Above 30.0	Above 19.08	10000

All q values were selected to be slightly greater than the position estimation error minimizing q values .

E. FILTER B

Filter B has the following thresholds and corresponding values of q . Again 7.0 was used as the threshold for examining two consecutive residuals . Notice that the quantization levels for acceleration and residual values are the same as for Filter A .

Filter B

Acceleration (m/sec^2)	Residual value (m)	Selected q
0.0 - 1.0	0.00 - 1.00	0
1.0 - 2.5	1.00 - 1.67	64
2.5 - 5.0	1.67 - 3.25	625
5.0 - 10.0	3.25 - 6.41	1600
10.0 - 20.0	6.41 - 12.75	2500
20.0 - 30.0	12.75 - 19.08	10000
Above 30.0	Above 19.08	100000

Filter B utilizes generally higher q values than Filter A . As seen in Figures 10 and 11 , the estimation errors for position and velocity have very flat characteristics . Therefore , if q values are selected to be much greater than the minimizing q values for position estimation error , Filter B should generate larger position estimation errors , but smaller velocity estimation errors . Also , Filter B is expected to be more capable than Filter A of following large maneuvering levels .

F. PERFORMANCE RESULTS.

The simulation of the residual-testing adaptive filter was done with the same system matrices , the same ensemble size for Monte Carlo simulations and using the same track as for the constant-Q estimator simulation in Chapter III .

The performance results are shown in Figures 15 and 16 . Compared with the constant-Q filters , the residual-testing adaptive filters have smaller position and velocity estimation errors . The position estimation error for an acceleration of 10 m/sec^2 is excellent for both adaptive filters . Compared with the Q-generated adaptive filter , the residual-testing adaptive filters have relatively large position estimation errors for a nonmaneuvering track , but overall , they have smaller position estimation errors for the range of tracks tested . The velocity estimates are considerably better than those provided by the Q-generated adaptive filter and are very similar to those provided by the constant-Q estimator .

Comparing the two residual-testing filters , Filter A and Filter B have almost the same velocity estimation performance , and slightly different position estimation performance .

G. CONCLUSIONS

The residual-testing adaptive filters were developed from the considerations described in Section V.B . From simulation results , the assumptions used appear to be very reasonable because the residual-testing adaptive filters provide better performance than the constant-Q filters investigated .

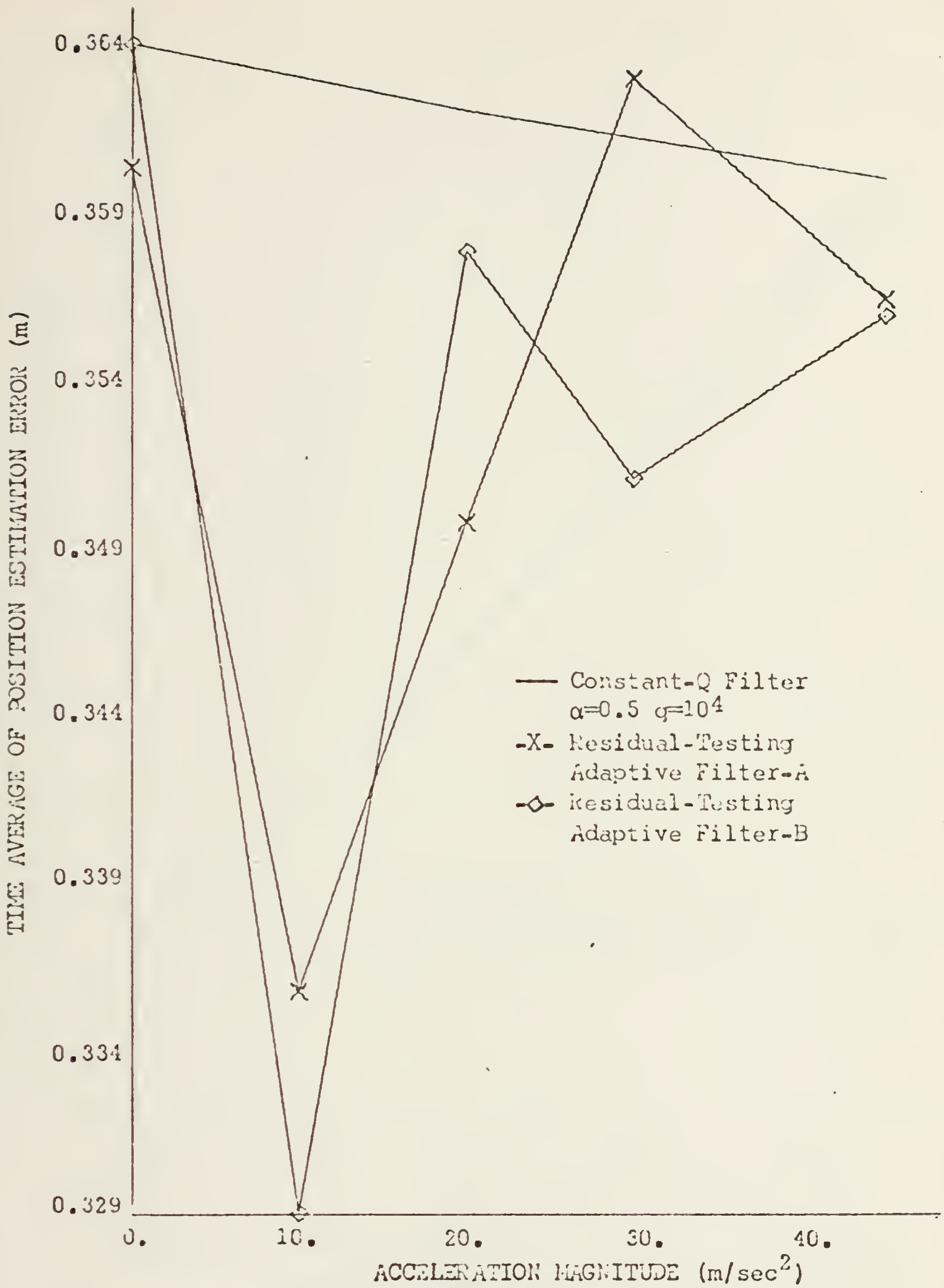


FIGURE 15 POSITION ESTIMATION ERRORS FOR TWO RESIDUAL-TESTING ADAPTIVE FILTER VS. MANEUVERING LEVEL

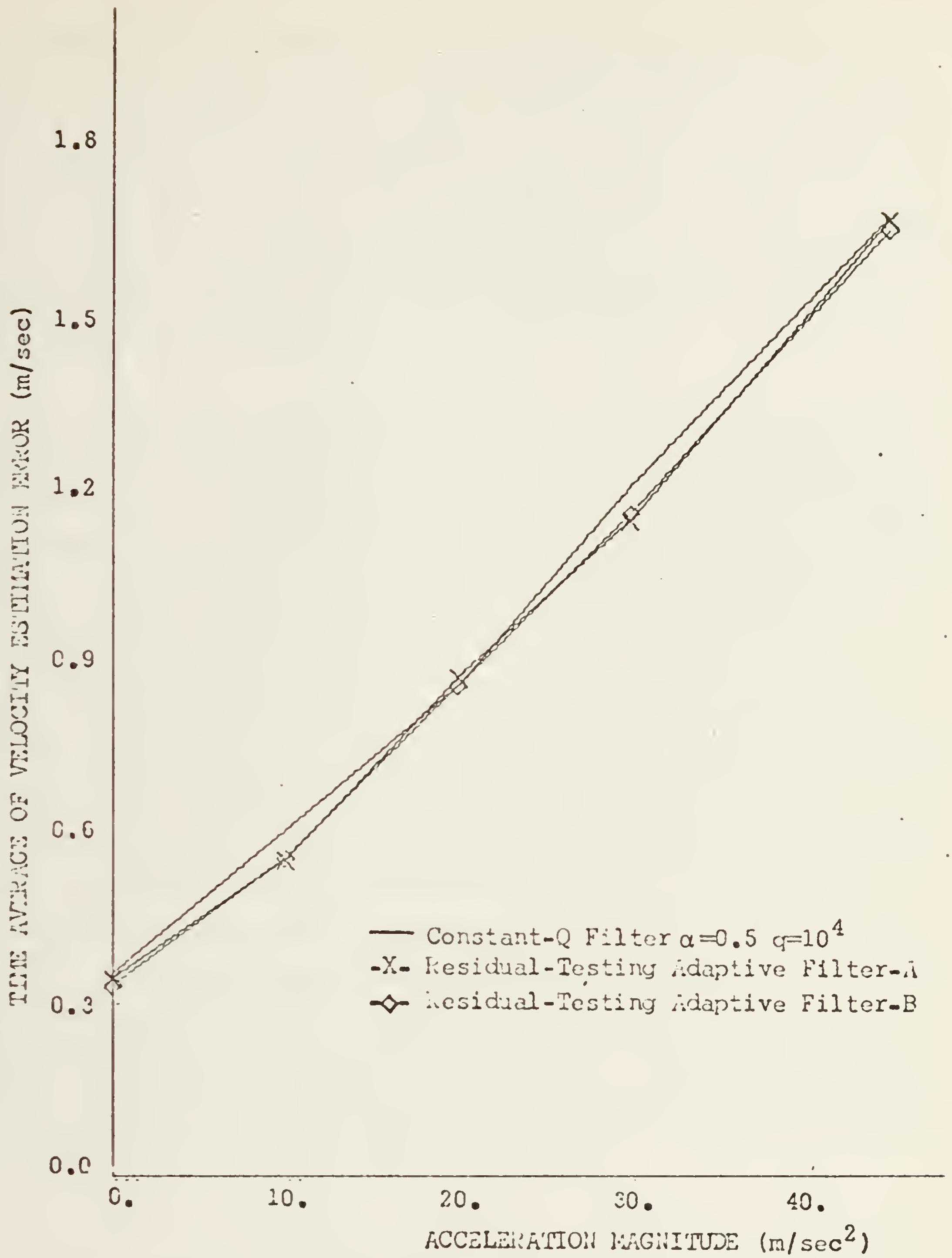


FIGURE 16 VELOCITY ESTIMATION ERRORS FOR TWO RESIDUAL-TESTING ADAPTIVE FILTER VS. MANEUVERING LEVEL

When the two residual-testing adaptive filters are compared with each other, the velocity estimates are very similar, even if the gains are different. This might be explained from the characteristics of the velocity estimation error curves shown in Figure 11. The curves are almost flat in the high-Q region, thus, differences in the gains are not expected to give significant differences in the simulation results.

For low-level maneuvering, it is difficult for the estimator to detect the maneuver because it is "hidden" to some extent in the measurement noise. Even for low-level maneuvers, however, the residuals will eventually tend to increase, thus signaling the presence of the maneuver. The statistical characteristics of these errors are completely different from those errors that are mainly caused by measurement noise. The adaptive-switching scheme detects nonmaneuvering periods very well. In Figure 17, a typical gain schedule is plotted. There are effects of measurement noise, but the detection of the nonmaneuvering period is clearly seen (K=1 to 10 and k=21 to 30 are nonmaneuvering periods). Notice that in the interval from K=1 to 5, the residual-testing adaptive filter is operated as a constant-Q filter. When Filters A and B were used without the adaptive switching feature, estimation errors for position and velocity were three times as large as when the adaptive switching scheme was added.

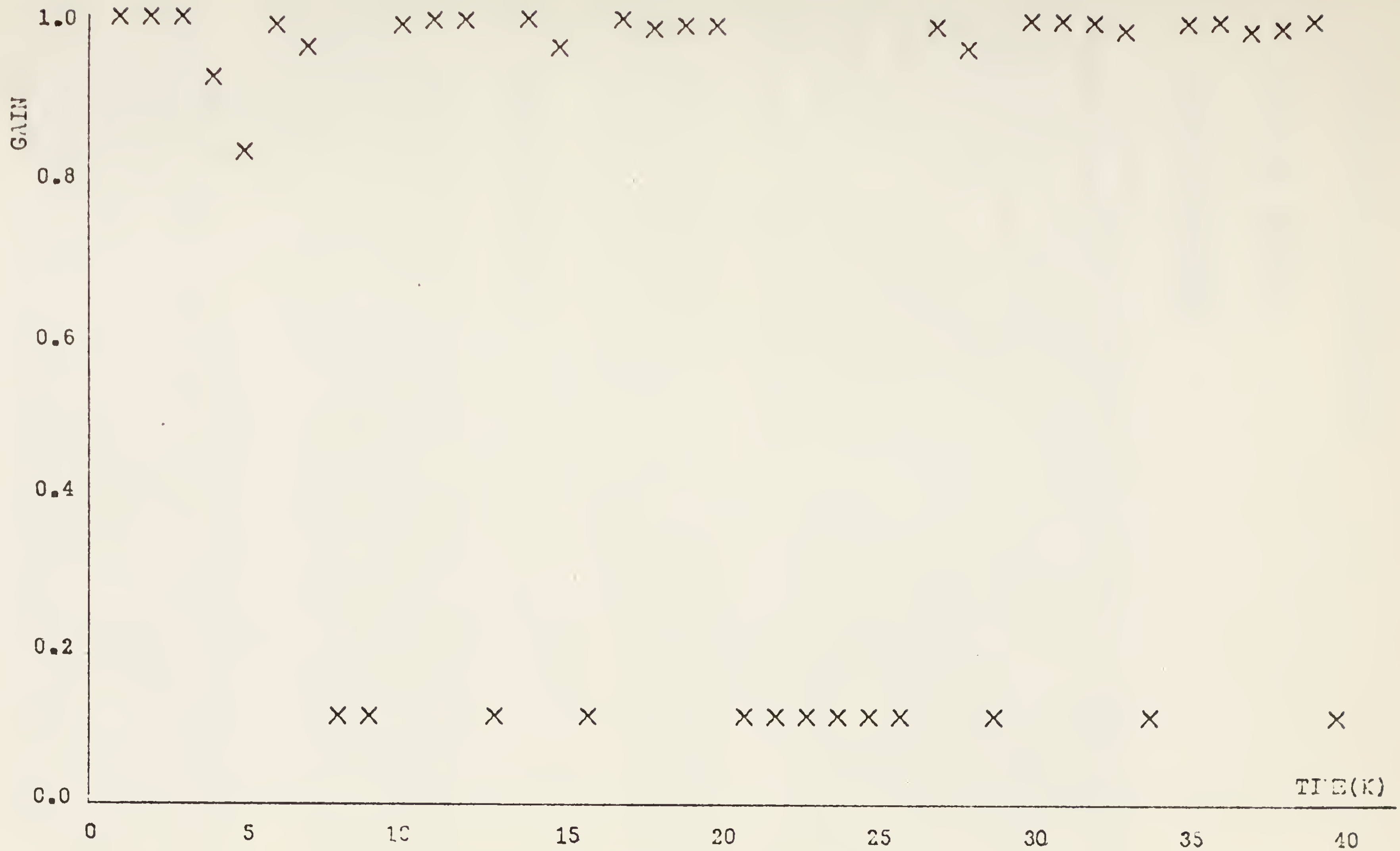


FIGURE 17 GAINS IN RESIDUAL-TESTING ADAPTIVE FILTER AND DETECTION OF NONMANEUVERING PERIOD

VI. SUMMARY AND CONCLUSIONS

A. THE TEST RESULTS

1. Constant-Q model

Simulation of the Constant-Q model was done to obtain knowledge about the performance of various models and to have data to compare with adaptive filters . However , the assumption that the Q-matrix is diagonal as expressed by Equation (29) may limit the conclusions that can be drawn . In the various models the correlated-constant-Q model with $\alpha=0.5$ had excellent performance for position and velocity estimation over the simulated maneuver range . The adaptive filters were compared with this filter to see if better performance could be achieved .

For the simulation studies , only one track pattern was used ; the amount of maneuvering was varied by adjusting the acceleration level A shown in Figure 2 .

2. Adaptive filters

The Q-generated adaptive filter , discussed in Chapter IV , provided generally poor performance over the simulated maneuvering range . The adjustment for generating q gave improvement for position estimates , but did not provide good velocity estimates .

The Residual-testing adaptive filter design was done in Chapter V . The design was based on the simulation results for the performance of the correlated constant-Q model with $\alpha=0.5$ against constant-acceleration tracks . The results were compared with the original correlated-constant-Q model ($\alpha=0.5$) . The residual-testing adaptive filter had

slightly better performance for position and velocity estimates than the correlated-constant-Q model .

B. SUGGESTIONS FOR FUTHER INVESTIGATION

The most significant assumption for the filters was the form of the Q-matrix . In Chapter III the Q-matrix was specified by Equation (29) . The same assumption was made for the Residual-testing adaptive filter . There was no strong reason for making this assumption , but this form of the Q-matrix gave monotonically decreasing velocity estimation errors as functions of q , over the simulated range of maneuver levels . It was desired to select the switching levels for residuals and gains to provide accurate estimates for position and velocity simultaneously . The properties and effects of the form of the Q-matrix should be investigated further , and the design procedure for the Residual-testing adaptive filter should be refined .

In the Residual-testing adaptive filter , the residual switching levels were determined by trial and error . To reduce uncertainty in the design , the characteristics of the residual should be analyzed more carefully .

In this thesis , adaptive filters were compared with constant-Q filters by using simplified tracks with medium-level maneuvering . It may be possible that the adaptive filters will perform better for particular tracks , for example , such as missile tracks , high-speed and high-level-maneuver attacking tracks .

For digital fire control systems , computation time is an important factor . For this reason , the Q-generated adaptive filter is not a good filter , especially when a

higher-order model is required , because the computation time requirement is extremly large compared with the constant-Q model .

C. CONCLUSIONS

The Residual-testing adaptive filter did not have remarkable improvement over the best correlated-constant-Q filter . The design of the Residual-testing adaptive filter is developed from a particular constant-Q filter and is based on the constant-acceleration track performance of this filter . In our case , as seen in Figures 9 and 10 , the correlated-constant-Q estimator with $\alpha=0.5$ had relatively the same performance for a wide range of of maneuvering levels for q greater than 900 . Even though the position estimation error has minimizing points , the biggest difference in position estimation error between the minimum point and the estimation error at $q=10^5$, was less than 2 cm . The velocity estimation error performance is monotonically decreasing with very small rates and has an almost flat characteristic over the simulated q range . Therefore , the Residual-testing filter could not have much improvement over the correlated-constant-Q filter . However , if the constant-Q estimator has sharper minimizing points for position and velocity estimation errors , and if it is possible to select q's that simultaneously provide small estimation errors for position , velocity and acceleration , the Residual-testing filter might give much improvement over the constant-Q filter .

APPENDIX A

THE MONTE-CARLO SIMULATION PROGRAM

A.1 Program description

An adition to the original Monte-Carlo simulation program that was developed by Prof . d . e . kirk of the Naval Postgraduate School is a new flag IQQ that is employed to implement the Residual-testing adaptive filter . Details of Subroutine QON and RETAD that perform the Q-generated adaptive scheme and the Residual-testing adaptive technique are described in Appendices B and C .

Input description and various options are explained by comments at the beginning of the program .

A.2 COMPUTER PROGRAM FOR MONTE-CARLO SIMULATION

```
C
C     THIS PROGRAM PERFORMS MONTE CARLO SIMULATION OF STATE ESTIMATORS
C     OF WHICH THE KALMAN FILTER IS ONE EXAMPLE. THERE ARE SEVERAL
C     OPTIONS AVAILABLE AS INDICATED IN THE DETAILLED COMMENTS BELOW.
C     IT SHOULD BE NOTED THAT ALL COMPUTATIONS OF GAINS USING THE
C     SUBROUTINE GAIN ARE PERFORMED IN DOUBLE PRECISION. THUS ALL
C     ARRAYS FOR USE IN "GAIN" MUST BE PREPARED ACCORDINGLY.
C
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),
     .XHATZ(4),
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND
      DIMENSION XP(80),YP(80)
      DIMENSION PKKM2(4,4)
      DIMENSION AVEE(4),AVEV(4),SUME(4),SUMV(4)
C
C
C     N=ORDER OF SYSTEM MODEL AND FILTER (DIMENSION OF X,XHAT)
C
C     M=NUMBER OF MEASUREMENTS (DIMENSION OF THE VECTOR Z)
C
C     IN=NUMBER OF INPUT RANDOM FORCING FCNS (=DIMENSION OF W)
C
C     NSAM=NUMBER OF TIME SAMPLES
C
C     NENS=NUMBER OF MEMBERS IN ENSEMBLE
C
      READ(5,100) N,M,IN,NSAM,NENS
  100 FORMAT(5(I10))
C
C
C     THE VALUE OF ND READ IN MUST EQUAL THE ROW (AND COLUMN) DIMENSION
C     SPECIFIED FOR THE SQUARE MATRIX "TEMP1", E.G. IF TEMP1(3,3) IS
C     SPECIFIED IN THE COMMON STATEMENT "ND" MUST BE EQUAL TO 3.
      READ(5,120) ND
  120 FORMAT(I2)
C
C
C
```

```
C       IG=-1 -- GAINS COMPUTED OFF-LINE AND READ IN                          MCSP0440
C            0 -- GAINS COMPUTED ONLY ONCE BEFORE STARTING MONTE CARLO        MCSP0450
C            1 -- GAINS COMPUTED FOR EACH MEMBER OF ENSEMBLE                  MCSP0460
C                                                                             MCSP0470
C       IFLR=0 -- R IS READ IN                                                MCSP0480
C       IFLR.NE.0 -- R IS COMPUTED ON-LINE AT EACH TIME SAMPLE                MCSP0490
C                                                                             MCSP0500
C       IEST=0 -- STANDARD KALMAN FILTER EQUATION IS USED                     MCSP0510
C       IEST.NE.0 -- STD. KALMAN FILTER EQ. NOT USED                          MCSP0520
C                                                                             MCSP0530
C       ITRK= 0 -- SEVERAL TRACKS GENERATED FROM STD LINEAR EQS               MCSP0540
C       ITRK= 1 -- ONLY ONE TRACK IS USED                                     MCSP0550
C        ITRK.NE.0OR1 -- SEVERAL TRACKS GENERATED BUT NOT FROM STD.           MCSP0560
C                        LINEAR DIFFERENCE EQUATIONS                          MCSP0570
C                                                                             MCSP0580
C       ISTAT=0 -- MEAN OF TRACK,MEAN & VARIANCE OF EST. ERROR COMPUTED       MCSP0590
C       ISTAT.NE.0 -- SAME AS ISTAT=0 BUT OFF-DIAGONAL TERMS IN COVARIANCEMCSP0600
C                                                                             MCSP0610
C                     MATRIX ARE ALSO COMPUTED                                MCSP0620
C                                                                             MCSP0630
        READ(5,101) IG,IFLR,IEST,ITRK,ISTAT,IQ,ITRO                           MCSP0640
    101 FORMAT(7(I10))                                                        MCSP0650
C                                                                             MCSP0660
C       IQ=-1 -- Q IS COMPUTED ON-LINE AT EACH SAMPLE BY "QON"                MCSP0670
C            0 -- Q MATRIX IS READ IN                                         MCSP0680
C            1 -- COVARIANCE OF W READ IN AND Q IS  COMPUTED ONCE BEFORE      MCSP0690
C                 MONTE CARLO BEGINS BY CALLING "QMAT"                        MCSP0700
C                                                                             MCSP0710
C       ITRO=0 -- ONLY USED IF ONE TRACK; ITRO=0 INDICATES THE                MCSP0720
C                 TRACK IS READ IN (S.P.), OTHERWISE THE TRACK                MCSP0730
C                 IS GENERATED BY A SINGLE CALL OF SUBROUTINE                 MCSP0740
C                 TRACK.                                                      MCSP0750
C                                                                             MCSP0760
C       IPRT=0 -- SOME OR ALL OUTPUT DATA IS PRINTED                          MCSP0770
C                                                                             MCSP0780
C       IPLT=0 -- SOME OR ALL OUTPUT DATA IS PLOTTED                          MCSP0790
        READ(5,110) IPRT,IPLT                                                 MCSP0800
    110 FORMAT(2(I5))                                                         MCSP0810
C                                                                             MCSP0820
C       IGPLT,ITHVPL,IMTPLT,ISMPLT,ISVPLT ARE PLOTTING INDICATORS             MCSP0830
C       IF THE INDICATOR IS EQUAL TO 1 PLOTS ARE GENERATED. THE INDICATORSMCSP0840
C       CORRESPOND TO THE FOLLOWING QUANTITIES                                MCSP0850
C       IGPLT(GAIN VS K),ITHVPL (THEORETICAL COVARIANCE MATRIX VS K),         MCSP0860
C       IMTPLT (SAMPLE MEAN OF TRACK=TRACK IF ONLY ONE TRACK),                MCSP0870
C       ISMPLT (SAMPLE MEAN OF EST. ERROR VS K), ISVPLT (SAMPLE VARIANCE      MCSP0880
C       OF EST. ERROR VS K)                                                   MCSP0890
        IF(IPLT.EQ.0) READ(5,111) IGPLT,ITHVPL,IMTPLT,ISMPLT,ISVPLT           MCSP0900
    111 FORMAT(5(I10))                                                        MCSP0910
```

```
C
C     IPHI=0 -- PHI MATRIX TO BE READ IN (IN DOUBLE PREC)
C     IPHI.NE.0 -- PHI MATRIX NOT TO BE READ
C
C     IH=0 -- H MATRIX TO BE READ IN (IN D.P.)
C
C     IR=0 -- R MATRIX IS READ IN
C
C     IPKKM1=0 -- THE STARTING VALUE FOR THE MATRIX P(K/K-1) IS TO
C                 BE READ IN (D.P.)
C
C     IGAM=0 -- GAMMA MATRIX TO BE READ IN (D.P.)
C
C     ISIGV=0 -- STD. DEVIATIONS OF MEASUREMENT NCISE TO BE READ IN (SP)
C
C     ISIGW=0 -- STD. DEVIATIONS OF RANDOM INPUTS TO BE READ IN (SP)
C
C     IXHZ=0 -- INITIAL VALUE OF THE PREDICTED VALUE XH(0/-1) TO
C               BE READ IN (S.P.)
C
C     IC=1 -- ONE INITIAL CONDITION VALUE USED FOR THE STATE
C             OTHERWISE THE VALUE OF X(0) IS GENERATED BY A RANDOM
C             NIMBER GENERATOR CALLED BY "XZERO"
C
C
C
C
C
      READ(5,103) IPHI,IH,IR,IPKKM1,IGAM,ISIGV,ISIGW,IXHZ,IC
  103 FORMAT(9I5)
C
C
C     IF THE IG=1  FOLLOWING FLAGS MUST BE SPECIFIED
C
C     IFLQ=1 -- SUBROUTINE QON IS CALLED FOR ON LINE Q CALCULATION
C               AND GAIN IS CALCULATED ON-LINE
C
C         0 -- QON IS NOT CALLED
C
C     IQQ=1 -- SUBROUTINE QON CALLED BUT GAIN IS NOT CALCULATED
C              ON-LINE
C              WHEN IQQ=1 , IFLQ  MUST BE 1 .
C
C
      READ(5,103) IFLQ ,IQQ
C
      CALL OVFLOW
      IW=6395217
```

```
      IV=1936748
      IXZ=135769
C
C
C     THE FOLLOWING SECTION PRINTS OUT A DESCRIPTION OF THE RUN AS
C     SPECIFIED BY THE USER'S FLAGS
C
C
      WRITE(6,250)
      WRITE(6,202)
  202 FORMAT(20X,'DESCRIPTION OF RUN',//)
      IF(IG.EQ.0) GO TO 300
      IF(IG.EQ.1) GO TO 301
      WRITE(6,240)
  240 FORMAT(10X,'GAINS COMPUTED OFF-LINE AND READ IN',/)
      GO TO 302
  300 WRITE(6,241)
  241 FORMAT(10X,'GAINS COMPUTED ONCE IN "GAIN" BEFORE STARTING MONTE CA
     .RLO',/)
      GO TO 302
  301 WRITE(6,242)
  242 FORMAT(10X,'GAINS COMPUTED FOR EACH MEMBER OF ENSEMBLE',/)
C
      IF(IFLQ.EQ.1.AND.IQQ.EQ.1) WRITE(6,666)
  666 FORMAT(10X,'GAINS SWITCHED BY SUBROUTINE QON',/)
  302 IF(IEST.EQ.0) GO TO 303
      WRITE(6,243)
  243 FORMAT(10X,'THE STANDARD LINEAR EQS. DO NOT CHARACTERIZE THE FILTE
     .R',/)
      GO TO 304
  303 WRITE(6,244)
  244 FORMAT(10X,'THE STD. KALMAN EQS. CHARACTERIZE THE LINEAR FILTER',/
     .)
  304 IF(ITRK.EQ.0) GO TO 305
      IF(ITRK.EQ.-1) GO TO 306
      IF(ITRO.EQ.0) GO TO 307
      WRITE(6,245)
  245 FORMAT(10X,'ONLY ONE TRACK IS USED AND IT IS GENERATED BY SUBROUTI
     .NE TRACK',/)
      GO TO 308
  307 WRITE(6,246)
  246 FORMAT(10X,'ONLY ONE TRACK IS USED AND IT IS READ IN',/)
      GO TO 308
  306 WRITE(6,247)
  247 FORMAT(10X,'SEVERAL TRACKS USED BUT NOT GENERATED FROM STD. LINEAR
     . DIFFERENCE EQS.',/)
      GO TO 308
  305 WRITE(6,248)
```

```
248 FORMAT(10X,'SEVERAL TRACKS GENERATED BY USING THE STD. LINEAR DIFFMCSP1880
   .ERENCE EQS.',/)                                                   MCSP1890
308 IF(ISTAT.EQ.0) GO TO 309                                          MCSP1900
    WRITE(6,249)                                                      MCSP1910
249 FORMAT(10X,'MEAN OF TRACK, MEAN OF EST. ERROR AND COVARIANCE OF ESMCSP1920
   .T. ERROR ARE COMPUTED',/)                                         MCSP1930
    GO TO 310                                                         MCSP1940
309 WRITE(6,251)                                                      MCSP1950
251 FORMAT(10X,'MEAN OF TRACK, MEAN AND VARRIANCES OF EST. ERROR ARE CMCSP1960
   .OMPUTED',/)                                                       MCSP1970
310 IF(IQ.EQ.0) GO TO 311                                             MCSP1980
    IF(IQ.EQ.1) GO TO 312                                             MCSP1990
    WRITE(6,252)                                                      MCSP2000
252 FORMAT(10X,'THE Q MATRIX IS COMPUTED ON-LINE AT EACH SAMPLE BY "QOMCSP2010
   .N"',/)                                                            MCSP2020
    GO TO 313                                                         MCSP2030
312 WRITE(6,253)                                                      MCSP2040
253 FORMAT(10X,'THE COVARIANCE OF W IS READ IN AND Q IS COMPUTED BY "QMCSP2050
   .MAT" BEFORE STARTING MONTE CARLO',/)                              MCSP2060
    GO TO 313                                                         MCSP2070
311 WRITE(6,254)                                                      MCSP2080
254 FORMAT(10X,'THE Q MATRIX IS READ IN',/)                           MCSP2090
313 IF(IFLR.EQ.0) GO TO 314                                           MCSP2100
    WRITE(6,255)                                                      MCSP2110
255 FORMAT(10X,'R IS COMPUTED ON-LINE AT EACH SAMPLE BY "RON"',/)     MCSP2120
    GO TO 315                                                         MCSP2130
314 WRITE(6,256)                                                      MCSP2140
256 FORMAT(10X,'R IS READ IN',/)                                      MCSP2150
315 WRITE(6,257)                                                      MCSP2160
257 FORMAT(////,20X,'INPUT DATA CALLED FOR',//)                       MCSP2170
    IF(IPHI.EQ.0) WRITE(6,258)                                        MCSP2180
258 FORMAT(10X,'PHI MATRIX',/)                                        MCSP2190
    IF(IH.EQ.0) WRITE(6,259)                                          MCSP2200
259 FORMAT(10X,'H MATRIX',/)                                          MCSP2210
    IF(IFLR.EQ.0) WRITE(6,260)                                        MCSP2220
260 FORMAT(10X,'R MATRIX',/)                                          MCSP2230
    IF(IQ.EQ.0) GO TO 316                                             MCSP2240
    IF(IQ.EQ.-1) GO TO 317                                            MCSP2250
    WRITE(6,261)                                                      MCSP2260
261 FORMAT(10X,'COVARIANCE OF W',/)                                   MCSP2270
    GO TO 317                                                         MCSP2280
316 WRITE(6,262)                                                      MCSP2290
262 FORMAT(10X,'Q MATRIX',/)                                          MCSP2300
317 IF(IGAM.EQ.0) WRITE(6,282)                                        MCSP2310
282 FORMAT(10X,'GAMMA MATRIX',/)                                      MCSP2320
    IF(ISIGV.EQ.0) WRITE(6,263)                                       MCSP2330
263 FORMAT(10X,'STANDARD DEVIATIONS OF MEASUREMENT NOISE',/)          MCSP2340
    IF(ISIGW.EQ.0) WRITE(6,264)                                       MCSP2350
```

```
  264 FORMAT(10X,'STANDARD DEVIATIONS OF INPUT FORCING W',/)                MCSP2360
      IF(IXHZ.EQ.0) WRITE(6,265)                                           MCSP2370
  265 FORMAT(10X,'XHAT(0/-1)',/)                                           MCSP2380
      IF(IC.NE.1) WRITE(6,268)                                             MCSP2390
  268 FORMAT(10X,'MEANS AND VARIANCES OF X(0)',/)                          MCSP2400
      IF(IPKKM1.EQ.0) WRITE(6,267)                                         MCSP2410
  267 FORMAT(10X,'P(0/-1)',/)                                              MCSP2420
C                                                                          MCSP2430
C                                                                          MCSP2440
C                                                                          MCSP2450
C                                                                          MCSP2460
C     THE FOLLOWING SECTION PRINTS OUT A DESCRIPTION OF THE OUTPUT         MCSP2470
C     DATA CALLED FOR                                                      MCSP2480
C                                                                          MCSP2490
C                                                                          MCSP2500
      WRITE(6,269)                                                         MCSP2510
  269 FORMAT(////,20X,'OUTPUT CALLED FOR',//)                              MCSP2520
      IF(IPRT.NE.0) GO TO 320                                              MCSP2530
      WRITE(6,270)                                                         MCSP2540
  270 FORMAT(10X,'PRINTED OUTPUT OF THE FOLLOWING DATA',/)                 MCSP2550
      IF(IG.EQ.0) WRITE(6,271)                                             MCSP2560
  271 FORMAT(15X,'GAIN MATRICES AND THEORETICAL COVARIANCE OF EST. ERRORMCSP2570
     . MATRICES',/)                                                        MCSP2580
      WRITE(6,272)                                                         MCSP2590
  272 FORMAT(15X,'SAMPLE MEANS OF TRACK AND ESTIMATION ERROR,SAMPLE VARIMCSP2600
     .ANCES OF EST. ERROR',/)                                              MCSP2610
      IF(ISTAT.NE.0) WRITE(6,273)                                          MCSP2620
  273 FORMAT(15X,'COVARIANCE OF ESTIMATION ERROR MATRIX',/)                MCSP2630
      GO TO 321                                                            MCSP2640
  320 WRITE(6,274)                                                         MCSP2650
  274 FORMAT(10X,'NO PRINTED OUTPUT CALLED FOR',//)                        MCSP2660
  321 IF(IPLT.NE.0) GO TO 322                                              MCSP2670
      WRITE(6,275)                                                         MCSP2680
  275 FORMAT(10X,'THE FOLLOWING PLOTS ARE CALLED FOR',/)                   MCSP2690
      IF(IGPLT.EQ.1) WRITE(6,276)                                          MCSP2700
  276 FORMAT(15X,'G(K) VS. K',/)                                           MCSP2710
      IF(ITHVPL.EQ.1) WRITE(6,277)                                         MCSP2720
  277 FORMAT(15X,'P(K/K) THEORETICAL VS. K',/)                             MCSP2730
      IF(IMTPLT.EQ.1) WRITE(6,278)                                         MCSP2740
  278 FORMAT(15X,'MEAN OF TRACK VS. K',/)                                  MCSP2750
      IF(ISMPLT.EQ.1) WRITE(6,279)                                         MCSP2760
  279 FORMAT(15X,'SAMPLE MEANS OF ESTIMATION ERROR VS. K',/)               MCSP2770
      IF(ISVPLT.EQ.1) WRITE(6,280)                                         MCSP2780
  280 FORMAT(15X,'SAMPLE VARIANCES OF ESTIMATION ERROR VS. K',/)           MCSP2790
      GO TO 323                                                            MCSP2800
  322 WRITE(6,281)                                                         MCSP2810
  281 FORMAT(10X,'NO PLOTS CALLED FOR',/)                                  MCSP2820
  323 CONTINUE                                                             MCSP2830
```

```
      WRITE(5,250)                                                      MCSP2840
      WRITE(6,201)                                                      MCSP2850
  201 FORMAT(20X,'INPUT DATA',///)                                      MCSP2860
      WRITE(6,200) N,M,IN,NSAM,NENS                                     MCSP2870
  200 FORMAT(4X,'N=',I2,4X,'M=',I2,4X'IN=',I2,4X'NSAM=',I3,4X,'NENS=',I5MCSP2880
     .,//)                                                              MCSP2890
C                                                                       MCSP2900
C                                                                       MCSP2910
C     THE FOLLOWING SECTION READS THE SPECIFIED INPUT MATRICES          MCSP2920
C                                                                       MCSP2930
      IF(IPHI.NE.0) GO TO 61                                            MCSP2940
      CALL MREAD(PHI,N,N)                                               MCSP2950
      DO 60 I=1,N                                                       MCSP2960
      DO 60 J=1,N                                                       MCSP2970
   60 PHIS(I,J)=PHI(I,J)                                                MCSP2980
      WRITE(6,204)                                                      MCSP2990
  204 FORMAT(//,10X,'THE PHI MATRIX IS',/)                              MCSP3000
      CALL MWRITE(PHI,N,N)                                              MCSP3010
C                                                                       MCSP3020
C                                                                       MCSP3030
   61 IF(IH.NE.0) GO TO 63                                              MCSP3040
      CALL MREAD(H,M,N)                                                 MCSP3050
      DO 62 I=1,M                                                       MCSP3060
      DO 62 J=1,N                                                       MCSP3070
   62 HS(I,J)=H(I,J)                                                    MCSP3080
      WRITE(6,205)                                                      MCSP3090
  205 FORMAT(//,10X,'THE H MATRIX IS',/)                                MCSP3100
      CALL MWRITE(H,M,N)                                                MCSP3110
C                                                                       MCSP3120
C                                                                       MCSP3130
   63 IF(IR.NE.0) GO TO 66                                              MCSP3140
      CALL MREAD(R,M,M)                                                 MCSP3150
      WRITE(6,206)                                                      MCSP3160
  206 FORMAT(//,10X,'THE R MATRIX IS',/)                                MCSP3170
      CALL MWRITE(R,M,M)                                                MCSP3180
C                                                                       MCSP3190
C                                                                       MCSP3200
   66 IF(IQ.NE.1) GO TO 59                                              MCSP3210
      CALL MREAD(COVW,IN,IN)                                            MCSP3220
      WRITE(6,207)                                                      MCSP3230
  207 FORMAT(//,10X,'THE COVARIANCE OF W MATRIX IS',/)                  MCSP3240
      CALL MWRITE(COVW,IN,IN)                                           MCSP3250
      GO TO 68                                                          MCSP3260
   59 IF(IQ.NE.0) GO TO 68                                              MCSP3270
      CALL MREAD(Q,N,N)                                                 MCSP3280
      WRITE(6,203)                                                      MCSP3290
  203 FORMAT(//,10X,'THE Q MATRIX IS',/)                                MCSP3300
      CALL MWRITE(Q,N,N)                                                MCSP3310
```

```
C                                                                        MCSP3320
C                                                                        MCSP3330
   68 IF(IGAM.NE.0) GO TO 65                                             MCSP3340
      CALL MREAD(GAMMA,N,IN)                                             MCSP3350
      DO 64 I=1,N                                                        MCSP3360
      DO 64 J=1,IN                                                       MCSP3370
   64 GAMMAS(I,J)=GAMMA(I,J)                                             MCSP3380
      WRITE(6,208)                                                       MCSP3390
  208 FORMAT(//,10X,'THE GAMMA MATRIX IS',/)                             MCSP3400
      CALL MWRITE(GAMMA,N,IN)                                            MCSP3410
C                                                                        MCSP3420
C                                                                        MCSP3430
   65 IF(IPKKM1.NE.0) GO TO 67                                           MCSP3440
      CALL MREAD(PKKM1,N,N)                                              MCSP3450
      WRITE(6,209)                                                       MCSP3460
  209 FORMAT(//,10X,'THE MATRIX P(0/-1) IS',/)                           MCSP3470
      CALL MWRITE(PKKM1,N,N)                                             MCSP3480
      DO 1500 I=1,N                                                      MCSP3490
      DO 1500 J=1,N                                                      MCSP3500
 1500 PKKM2(I,J)=PKKM1(I,J)                                              MCSP3510
C                                                                        MCSP3520
C                                                                        MCSP3530
   67 IF(ISIGV.NE.0) GO TO 69                                            MCSP3540
      CALL VREAD(SIGV,M)                                                 MCSP3550
      WRITE(6,210)                                                       MCSP3560
  210 FORMAT(//,10X,'THE STD. DEVIATIONS OF MEASUREMENT NOISE ARE',/)    MCSP3570
      CALL VWRITE(SIGV,M)                                                MCSP3580
C                                                                        MCSP3590
C                                                                        MCSP3600
   69 IF(ISIGW.NE.0) GO TO 71                                            MCSP3610
      CALL VREAD(SIGW,IN)                                                MCSP3620
      WRITE(6,211)                                                       MCSP3630
  211 FORMAT(//,10X,'THE STD. DEVIATIONS OF INPUT FORCING W ARE',/)      MCSP3640
      CALL VWRITE(SIGW,IN)                                               MCSP3650
C                                                                        MCSP3660
C                                                                        MCSP3670
   71 IF(IXHZ.NE.0) GO TO 73                                             MCSP3680
      CALL VREAD(XHATZ,N)                                                MCSP3690
      WRITE(6,212)                                                       MCSP3700
  212 FORMAT(//,10X,'THE VECTOR XHAT(0/-1) IS',/)                        MCSP3710
      CALL VWRITE(XHATZ,N)                                               MCSP3720
C                                                                        MCSP3730
C                                                                        MCSP3740
   73 IF(IC.EQ.1) GO TO 75                                               MCSP3750
C     IC.NE.1 MEANS THAT MEANS AND STD. DEVIATIONS OF THE INITIAL STATE  MCSP3760
C     VALUE MUST BE READ IN. OTHERWISE NOT READ IN.                      MCSP3770
      CALL VREAD(XZMEAN,N)                                               MCSP3780
      WRITE(6,213)                                                       MCSP3790
```

```
  213 FORMAT(//,10X,'THE MEAN OF THE VECTOR X(0) IS',/)                       MCSP3800
      CALL VWRITE(XZMEAN,N)                                                   MCSP3810
C                                                                             MCSP3820
C                                                                             MCSP3830
      CALL VREAD(SIGXZ,N)                                                     MCSP3840
      WRITE(6,214)                                                            MCSP3850
  214 FORMAT(//,10X,'THE STANDARD DEVIATIONS OF THE VECTOR X(0) ARE',/)       MCSP3860
      CALL VWRITE(SIGXZ,N)                                                    MCSP3870
      GO TO 90                                                                MCSP3880
C                                                                             MCSP3890
C                                                                             MCSP3900
   75 READ(5,105) (XS(I,1),I=1,N)                                             MCSP3910
C     INITIAL CONDITION HAS BEEN READ                                         MCSP3920
      WRITE(6,219)                                                            MCSP3930
  219 FORMAT(//,10X,'THE INITIAL STATE IS',/)                                 MCSP3940
      WRITE(6,221) (XS(I,1),I=1,N)                                            MCSP3950
      IF(ITRK.NE.1) GO TO 90                                                  MCSP3960
      IF(ITRO.NE.0) GO TO 85                                                  MCSP3970
      DO 81 K=2,NSAM                                                          MCSP3980
   81 READ(5,105) (XS(I,K),I=1,N)                                             MCSP3990
  105 FORMAT(4F20.0)                                                          MCSP4000
      GO TO 86                                                                MCSP4010
   85 CALL TRACK                                                              MCSP4020
C     IF TRACK CALLED HERE IT SHOULD BE WRITTEN TO GENERATE AND               MCSP4030
C     STORE THE TRACK IN XS(N,K) FOR K=2(NOTE),NSAM                           MCSP4040
C                                                                             MCSP4050
   86 WRITE(6,220)                                                            MCSP4060
  220 FORMAT(//,10X,'THE FIRST AND LAST POINTS ON THE SINGLE TRACK TO BEMCSP4070
     . USED ARE',/)                                                           MCSP4080
      WRITE(6,221) (XS(I,1),I=1,N)                                            MCSP4090
      WRITE(6,221) (XS(I,NSAM),I=1,N)                                         MCSP4100
  221 FORMAT(9(2X,1PE12.5),/)                                                 MCSP4110
   90 CONTINUE                                                                MCSP4120
C                                                                             MCSP4130
C                                                                             MCSP4140
C                                                                             MCSP4150
C                                                                             MCSP4160
C     THE FOLLOWING SECTION PREPARES FOR THE MONTE CARLO LOOP                 MCSP4170
C                                                                             MCSP4180
C     FORM NXN IDENTITY MATRIX IN DOUBLE PRECISION                            MCSP4190
      DO 30 I=1,N                                                             MCSP4200
      DO 30 J=1,N                                                             MCSP4210
      EI(I,J)=0.D0                                                            MCSP4220
   30 IF(I.EQ.J) EI(I,J)=1.D0                                                 MCSP4230
C     GIVEN THE MATRIX GAMMA AND THE COVARIANCE OF W COMPUTE Q                MCSP4240
C     USING DOUBLE PRECISION ARITHMETIC                                       MCSP4250
      IF(IQ.NE.1) GO TO 4                                                     MCSP4260
      CALL QMAT                                                               MCSP4270
```

```
      WRITE(6,203)                                                      MCSP4280
      CALL MWRITE(Q,N,N)                                                MCSP4290
    4 IF(IG.NE.-1) GO TO 10                                             MCSP4300
      DO 40 K=1,NSAM                                                    MCSP4310
      DO 40 I=1,N                                                       MCSP4320
   40 READ(5,105) (GKS(I,J,K),J=1,M)                                    MCSP4330
      GO TO 20                                                          MCSP4340
   10 IF(IQQ.EQ.1) GO TO 990                                            MCSP4350
      IF (IG.NE.0) GO TO 20                                             MCSP4360
  990 DO 11 K=1,NSAM                                                    MCSP4370
      IF (IQQ.EQ.1.AND.K.GT.5) GO TO 20                                 MCSP4380
      CALL GAIN(IG,IQQ)                                                 MCSP4390
      DO 11 I=1,N                                                       MCSP4400
      DO 12 L=1,N                                                       MCSP4410
   12 PKKS(I,L,K)=PKK(I,L)                                              MCSP4420
      DO 11 J=1,M                                                       MCSP4430
   11 GKS(I,J,K)=G(I,J)                                                 MCSP4440
   20 CONTINUE                                                          MCSP4450
C     IF GAINS WERE TO BE READ IN (IG=-1) OR COMPUTED ONLY              MCSP4460
C     ONCE(IG=0), THIS HAS NOW BEEN DONE                                MCSP4470
C                                                                       MCSP4480
C                                                                       MCSP4490
C                                                                       MCSP4500
C                                                                       MCSP4510
C     SET UP ARRAYS FOR COMPUTING STATISTICS                            MCSP4520
      DO 1 K=1,NSAM                                                     MCSP4530
      DO 1 J=1,N                                                        MCSP4540
      XM(J,K)=0.                                                        MCSP4550
      ERR(J,K)=0.                                                       MCSP4560
      DO 1 L=1,N                                                        MCSP4570
    1 VAR(J,L,K)=0.                                                     MCSP4580
C                                                                       MCSP4590
C                                                                       MCSP4600
C     BEGIN MAIN ITERATION LOOP HERE                                    MCSP4610
      DO 1000 ITER=1,NENS                                               MCSP4620
      IF(IC.EQ.1) GO TO 14                                              MCSP4630
      CALL XZERO                                                        MCSP4640
   14 DO 15 I=1,N                                                       MCSP4650
   15 XHKKM1(I)=XHATZ(I)                                                MCSP4660
      DO 1512 I=1,N                                                     MCSP4670
      DO 1512 J=1,N                                                     MCSP4680
 1512 PKKM1(I,J)=PKKM2(I,J)                                             MCSP4690
      A3=0.0                                                            MCSP4700
      DO 1000 K=1,NSAM                                                  MCSP4710
C     FORM NOISY MEASUREMENT FROM TRUE STATE VALUE                      MCSP4720
C                                                                       MCSP4730
      DO 9 I=1,N                                                        MCSP4740
    9 X(I)=XS(I,K)                                                      MCSP4750
```

```
      CALL MEAS                                                         MCSP4760
C     GAIN IS NOT TO BE COMPUTED ON-LINE IF IG.NE.1                     MCSP4770
      IF (IG.NE.1) GO TO 70                                             MCSP4780
      IF(IGQ.EQ.1.AND.K.LE.5) GO TO 70                                  MCSP4790
      IF(IFLQ.EQ.1.AND.K.EQ.1) GO TO 999                                MCSP4800
C                                                                       MCSP4810
C     Q IS TO BE COMPUTED ON-LINE IF IFLQ.NE.0                          MCSP4820
C                                                                       MCSP4830
      IF(ICQ.EQ.1) GO TO 994                                            MCSP4840
      CALL PEST(A1,A2,A3)                                               MCSP4850
      IF (IFLQ.NE.0) CALL QON(A1,A2,A3)                                 MCSP4860
C                                                                       MCSP4870
C     R TO BE COMPUTED ON-LINE IF IFLR.NE.0                             MCSP4880
C                                                                       MCSP4890
      IF(IFLR.NE.0) CALL RON                                            MCSP4900
      GC TO 999                                                         MCSP4910
C                                                                       MCSP4920
C     RESIDUAL TESTING ADAPTIVE FILTER PERFORMED IN "RETAD"             MCSP4930
C                                                                       MCSP4940
  994 CALL RETAD                                                        MCSP4950
      GC TO 991                                                         MCSP4960
C                                                                       MCSP4970
C                                                                       MCSP4980
  999 CALL GAIN(IG,IQQ)                                                 MCSP4990
      DC 996 I=1,N                                                      MCSP5000
      DC 996 L=1,N                                                      MCSP5010
  996 PKKS(I,L,K)=PKK(I,L)                                              MCSP5020
  991 DC 3 I=1,N                                                        MCSP5030
      DC 3 J=1,M                                                        MCSP5040
    3 GKS(I,J,K)=G(I,J)                                                 MCSP5050
C                                                                       MCSP5060
C     UPDATE THE STATE ESTIMATE                                         MCSP5070
C                                                                       MCSP5080
   70 CALL ESTIM                                                        MCSP5090
C                                                                       MCSP5100
C     UPDATE RUNNING SUMS USED IN COMPUTING STATISTICS                  MCSP5110
      CALL STAT                                                         MCSP5120
      IF(K.EQ.NSAM) GO TO 1000                                          MCSP5130
C                                                                       MCSP5140
C     UPDATE TRACK BY COMPUTING X(K+1)                                  MCSP5150
      IF(ITRK.NE.1) CALL TRACK                                          MCSP5160
 1000 CCNTINUE                                                          MCSP5170
C                                                                       MCSP5180
C     DIVIDE RUNNING SUMS COMPUTED BY SUBROUTINE STAT BY ENSEMBLE        MCSP5190
C     SIZE TO COMPUTE STATISTICS                                        MCSP5200
      ENS=NENS                                                          MCSP5210
      DC 2 K=1,NSAM                                                     MCSP5220
      DO 2 J=1,N                                                        MCSP5230
```

```
      IF(ITRK.EQ.1) GO TO 6                                            MCSP5240
      XM(J,K)=XM(J,K)/ENS                                              MCSP5250
    6 ERR(J,K)=ERR(J,K)/ENS                                            MCSP5260
    2 VAR(J,J,K)=VAR(J,J,K)/ENS-ERR(J,K)**2                            MCSP5270
      IF(ISTAT.EQ.0) GO TO 80                                          MCSP5280
C                                                                      MCSP5290
C     COMPUTE OFF-DIAGONAL TERMS IN COVARIANCE OF ESTIMATION           MCSP5300
C     ERROR MATRIX IF ISTAT.NE.0                                       MCSP5310
      DO 5 K=1,NSAM                                                    MCSP5320
      DO 5 L=2,N                                                       MCSP5330
      LM1=L-1                                                          MCSP5340
      DO 5 J=1,LM1                                                     MCSP5350
    5 VAR(L,J,K)=VAR(L,J,K)/ENS-ERR(L,K)*ERR(J,K)                      MCSP5360
   80 CONTINUE                                                         MCSP5370
C                                                                      MCSP5380
C                                                                      MCSP5390
      IF(IPRT.NE.0) GO TO 600                                          MCSP5400
      WRITE(6,230)                                                     MCSP5410
  230 FORMAT('1',20X,'OUTPUT DATA',///)                                MCSP5420
C     WRITE GAINS, THEORETICAL COVARIANCES OF ESTIMATION ERROR         MCSP5430
C     IF ONE SET OF GAINS HAS BEEN USED.                               MCSP5440
      WRITE(6,222)                                                     MCSP5450
  222 FORMAT(10X,'THE GAIN MATRICES ARE',/)                            MCSP5460
      DO 425 K=1,NSAM                                                  MCSP5470
      WRITE(6,223) K                                                   MCSP5480
  223 FORMAT(5X,'K=',I3,/,10X,'G(K)=',/)                               MCSP5490
      DO 425 I=1,N                                                     MCSP5500
  425 WRITE(6,221) (GKS(I,J,K),J=1,M)                                  MCSP5510
      IF(IG.NE.0) GO TO 450                                            MCSP5520
      WRITE(6,224)                                                     MCSP5530
  224 FORMAT(1X,//,10X,'THE THEORETICAL COVARIANCE MATRIX IS',/)       MCSP5540
      DO 426 K=1,NSAM                                                  MCSP5550
      WRITE(6,225) K                                                   MCSP5560
  225 FORMAT(5X,'K=',I3,/,10X,'P(K/K)=',/)                             MCSP5570
      DO 426 I=1,N                                                     MCSP5580
  426 WRITE(6,221) (PKKS(I,J,K) ,J=1,N)                                MCSP5590
C                                                                      MCSP5600
C                                                                      MCSP5610
  450 WRITE(6,250)                                                     MCSP5620
      WRITE(6,226)                                                     MCSP5630
      WRITE(6,227)                                                     MCSP5640
  226 FORMAT(T5,'TIME',T16,'VECTOR COM-',T34,'SAMPLE MEAN',            MCSP5650
     .T51,'SAMPLE MEAN OF',T71,'SAMPLE VARIANCE OF')                   MCSP5660
  227 FORMAT(T5,'INDEX',T16,'PONENT INDEX',T34,'OF TRACK',             MCSP5670
     .T51,'ESTIMATION ERROR',T71,'ESTIMATION ERROR')                   MCSP5680
  228 FORMAT(6X,I3,13X,I1,10X,1PE14.7,2(6X,1PE14.7))                   MCSP5690
  229 FORMAT(//)                                                       MCSP5700
      DO 696 I=1,N                                                     MCSP5710
```

```
      SUME(I)=0.0                                                       MCSP5720
      SUMV(I)=0.0                                                       MCSP5730
      AVEE(I)=0.0                                                       MCSP5740
      AVEV(I)=0.0                                                       MCSP5750
  696 CONTINUE                                                          MCSP5760
      DO 451 K=1,NSAM                                                   MCSP5770
      WRITE(6,229)                                                      MCSP5780
      DO 451 I=1,N                                                      MCSP5790
      SUME(I)=SUME(I)+ABS(ERR(I,K))                                     MCSP5800
      SUMV(I)=SUMV(I)+VAR(I,I,K)                                        MCSP5810
  451 WRITE(6,228) K,I,XM(I,K),ERR(I,K),VAR(I,I,K)                      MCSP5820
      DO 697 I=1,N                                                      MCSP5830
      AVEE(I)=SUME(I)/NSAM                                              MCSP5840
      AVEV(I)=SUMV(I)/NSAM                                              MCSP5850
      WRITE(6,470) AVEE(I),AVEV(I)                                      MCSP5860
  697 CONTINUE                                                          MCSP5870
  470 FORMAT(//53X,E14.7,6X,E14.7)                                      MCSP5880
      WRITE(6,250)                                                      MCSP5890
      IF(ISTAT.EQ.0) GO TO 600                                          MCSP5900
  250 FORMAT('1')                                                       MCSP5910
      WRITE(6,290)                                                      MCSP5920
  290 FORMAT(10X,'THE SAMPLE COVARIANCE OF EST. ERROR MATRIX IS',//)    MCSP5930
      DO 452 K=1,NSAM                                                   MCSP5940
      WRITE(6,231) K                                                    MCSP5950
      DO 452 I=1,N                                                      MCSP5960
  452 WRITE(6,221) ( VAR(I,L,K),L=1,I)                                  MCSP5970
  231 FORMAT(//,2X,'K=',I3,/)                                           MCSP5980
      WRITE(6,250)                                                      MCSP5990
  600 IF(IPLT.NE.0) GO TO 650                                           MCSP6000
      DO 603 K=1,NSAM                                                   MCSP6010
  603 XP(K)=K                                                           MCSP6020
      IF(IGPLT.NE.1) GO TO 610                                          MCSP6030
      DO 601 I=1,N                                                      MCSP6040
      DO 601 J=1,M                                                      MCSP6050
      DO 602 K=1,NSAM                                                   MCSP6060
  602 YP(K)=GKS(I,J,K)                                                  MCSP6070
      WRITE(6,250)                                                      MCSP6080
      CALL PLOTP(XP,YP,NSAM,0)                                          MCSP6090
  601 WRITE(6,232) I,J                                                  MCSP6100
  232 FORMAT(12X,'G(',I1,',',I1,')  VS. K')                             MCSP6110
  610 IF(ITHVPL.NE.1) GO TO 620                                         MCSP6120
      DO 611 I=1,N                                                      MCSP6130
      DO 612 K=1,NSAM                                                   MCSP6140
  612 YP(K)=PKKS(I,I,K)                                                 MCSP6150
      WRITE(6,250)                                                      MCSP6160
      CALL PLOTP(XP,YP,NSAM,0)                                          MCSP6170
  611 WRITE(6,233) I,I                                                  MCSP6180
  233 FORMAT(12X,'PKK(',I1,',',I1,')  VS. K')                           MCSP6190
```

```
620 IF(IMTPLT.NE.1) GO TO 630                                          MCSP6200
    DO 621 I=1,N                                                       MCSP6210
    DO 622 K=1,NSAM                                                    MCSP6220
622 YP(K)=XM(I,K)                                                      MCSP6230
    WRITE(6,250)                                                       MCSP6240
    CALL PLOTP(XP,YP,NSAM,0)                                           MCSP6250
621 WRITE(6,234) I                                                     MCSP6260
234 FORMAT(12X,'MEAN OF X(',I1,')  VS. K')                             MCSP6270
630 IF(ISMPLT.NE.1) GO TO 640                                          MCSP6280
    DO 631 I=1,N                                                       MCSP6290
    DO 632 K=1,NSAM                                                    MCSP6300
632 YP(K)=ERR(I,K)                                                     MCSP6310
    WRITE(6,250)                                                       MCSP6320
    CALL PLOTP(XP,YP,NSAM,0)                                           MCSP6330
631 WRITE(6,235) I,I                                                   MCSP6340
235 FORMAT(12X,'XHATKK(',I1,') -X(',I1,')  VS. K')                     MCSP6350
640 IF(ISVPLT.NE.1) GO TO 650                                          MCSP6360
    DO 641 I=1,N                                                       MCSP6370
    DO 642 K=1,NSAM                                                    MCSP6380
642 YP(K)=VAR(I,I,K)                                                   MCSP6390
    WRITE(6,250)                                                       MCSP6400
    CALL PLOTP(XP,YP,NSAM,0)                                           MCSP6410
641 WRITE(6,236) I                                                     MCSP6420
236 FORMAT(12X,'ERROR VARIANCE(',I1,') VS.. K')                        MCSP6430
    WRITE(6,250)                                                       MCSP6440
650 CONTINUE                                                           MCSP6450
    STOP                                                               MCSP6460
    END                                                                MCSP6470


      SUBROUTINE ESTIM                                                 MCSP0010
C     THIS SUBROUTINE UPDATES THE STATE ESTIMATE. IN THE DEFAULT       MCSP0020
C     CONDITION (IEST.EQ.0) THE STANDARD EQUATIONS                     MCSP0030
C                                                                      MCSP0040
C     XHAT(K/)=XHAT(K/K-1)+G(K)*(Z(K)-H(K)*XHAT(K/K-1))                MCSP0050
C                                                                      MCSP0060
C     XHAT(K+1/K)=PHI*XHAT(K/K)                                        MCSP0070
C                                                                      MCSP0080
C     ARE EVALUATED                                                    MCSP0090
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI      MCSP0100
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),      MCSP0110
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0120
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),         MCSP0130
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),     MCSP0140
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKM1(4),VTMP(4),Z(4),V(4),SIGV(4),   MCSP0150
```

```
     .XHATZ(4),                                                          MCSP0160
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND           MCSP0170
C     TAKE THE APPROPRIATE GAIN AND STORE IN THE ARRAY GK               MCSP0180
      DO 1 I=1,N                                                         MCSP0190
      DO 1 J=1,M                                                         MCSP0200
    1 GK(I,J)=GKS(I,J,K)                                                 MCSP0210
      IF(IEST.NE.0) GO TO 100                                            MCSP0220
      CALL VPROD(HS,XHKKM1,M,N,VTMP)                                     MCSP0230
      CALL VSUB(Z,VTMP,M,VTMP)                                           MCSP0240
      CALL VPROD(GK,VTMP,N,M,VTMP)                                       MCSP0250
      CALL VADD(XHKKM1,VTMP,N,XHKK)                                      MCSP0260
C                                                                        MCSP0270
C     XHAT(K/K) HAS BEEN COMPUTED AND STORED IN THE ARRAY XHKK          MCSP0280
      CALL VPROD(PHIS,XHKK,N,N,XHKKM1)                                   MCSP0290
C                                                                        MCSP0300
C     XHAT(K+1/K) HAS BEEN COMPUTED AND STORED IN THE ARRAY XHKKM1      MCSP0310
      RETURN                                                             MCSP0320
  100 CONTINUE                                                           MCSP0330
C     IF STANDARD EQUATIONS ARE NOT TO BE USED, THE APPROPRIATE         MCSP0340
C     EQUATIONS MUST BE INSERTED HERE BY THE USER.                      MCSP0350
      RETURN                                                             MCSP0360
      END                                                                MCSP0370


      SUBROUTINE GAIN(IG,IQQ)                                            MCSP0010
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI       MCSP0020
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),       MCSP0030
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0040
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),          MCSP0050
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),      MCSP0060
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),   MCSP0070
     .XHATZ(4),                                                          MCSP0080
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND           MCSP0090
C                                                                        MCSP0100
C     G(K) = P(K/K-1)*HT*(H*P(K/K-1)*HT + R)                            MCSP0110
      CALL TRANS(H,M,N,TEMP2)                                            MCSP0120
      CALL PROD(PKKM1,TEMP2,N,N,M,TEMP)                                  MCSP0130
      CALL PROD(H,TEMP,M,N,M,TEMP1)                                      MCSP0140
      CALL ADD(TEMP1,R,M,M,TEMP1)                                        MCSP0150
      IF(M.EQ.1) GO TO 1                                                 MCSP0160
      MD=ND                                                              MCSP0170
      CALL GAUSS3(M,EPS,TEMP1,TEMP2,KER,MD)                              MCSP0180
      CALL PROD(TEMP,TEMP2,N,M,M,G)                                      MCSP0190
C                                                                        MCSP0200
C     NOTE HERE  PKK(I,J) = P(K/K)  WHERE                               MCSP0210
```

```
C     P(K/K) = (I-G(K)*H)*P(K/K-1)                                     MCSP0220
    2 CALL PROD(G,H,N,M,N,TEMP)                                        MCSP0230
      CALL SUB(EI,TEMP,N,N,TEMP2)                                      MCSP0240
      CALL PRCD(TEMP2,PKKM1,N,N,N,PKK)                                 MCSP0250
      IF(IQQ.EQ.1) GO TO 30                                            MCSP0260
      IF (IG.EQ.1) RETURN                                              MCSP0270
   30 CALL TRANS(PHI,N,N,TEMP2)                                        MCSP0280
      CALL PROD(PKK,TEMP2,N,N,N,TEMP)                                  MCSP0290
      CALL PROD(PHI,TEMP,N,N,N,TEMP1)                                  MCSP0300
      CALL ADD(TEMP1,Q,N,N,PKKM1)                                      MCSP0310
      RETURN                                                           MCSP0320
    1 DC 3 I=1,N                                                       MCSP0330
    3 G(I,1)=TEMP(I,1)/TEMP1(1,1)                                      MCSP0340
      GO TO 2                                                          MCSP0350
      END                                                              MCSP0360


      SUBROUTINE MEAS                                                  MCSP0010
C     THIS SUBROUTINE STARTS WITH THE TRUE STATE VALUE XS              MCSP0020
C     AND ADDS ZERO-MEAN WHITE GAUSSIAN NOISE TO H*XS TO               MCSP0030
C     GENERATE A NOISY VERCTOR OF MEASUREMENTS Z.                      MCSP0040
C                                                                      MCSP0050
C                                                                      MCSP0060
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI      MCSP0070
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),      MCSP0080
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4.4),PHI(4,4),MCSP0090
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),         MCSP0100
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),     MCSP0110
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),  MCSP0120
     .XHATZ(4),                                                        MCSP0130
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND          MCSP0140
      CALL SNORM(IV,V,M)                                               MCSP0150
      DO 1 I=1,M                                                       MCSP0160
    1 V(I)=SIGV(I)*V(I)                                                MCSP0170
      Z(1)=XS(1,K)+V(1)                                                MCSP0180
      RETURN                                                           MCSP0190
      END                                                              MCSP0200


      SUBROUTINE PEST(A1,A2,A3)                                        MCSP0010
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI      MCSP0020
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),      MCSP0030
```

```
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0040
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),          MCSP0050
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),      MCSP0060
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),   MCSP0070
     .XHATZ(4),                                                         MCSP0080
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND           MCSP0090
C                                                                       MCSP0100
C     THIS SUBROUTINE CALCULATE THE COVARIANCE OF RESIDUAL              MCSP0110
C                                                                       MCSP0120
C     WITH ASSUMING Q=0  ,FOR SUBROUTINE  QON                           MCSP0130
C                                                                       MCSP0140
C                                                                       MCSP0150
      CALL PROD (H,PHI,M,N,N,TEMP)                                      MCSP0160
      CALL TRANS (TEMP,M,N,TEMP1)                                       MCSP0170
      CALL PRCD (PKK,TEMP1,N,N,M,TEMP1)                                 MCSP0180
      CALL PROD(TEMP,TEMP1,M,N,M,TEMP)                                  MCSP0190
      CALL ADD (TEMP,R,M,M,TEMP)                                        MCSP0200
      A1=TEMP(1,1)                                                      MCSP0210
      RETURN                                                            MCSP0220
      END                                                               MCSP0230


      SUBROUTINE QMAT                                                   MCSP0010
C                                                                       MCSP0020
C     THIS SUBROUTINE COMPUTES THE MATRIX Q FROM THE EQUATION           MCSP0030
C                                                                       MCSP0040
C        Q=GAMMA* E(W*WT) * GAMMAT                                      MCSP0050
C                                                                       MCSP0060
C     DOUBLE PRECISION ARITHMETIC IS USED                               MCSP0070
C                                                                       MCSP0080
C                                                                       MCSP0090
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI       MCSP0100
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),       MCSP0110
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0120
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),          MCSP0130
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),      MCSP0140
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),   MCSP0150
     .XHATZ(4),                                                         MCSP0160
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND           MCSP0170
C                                                                       MCSP0180
C                                                                       MCSP0190
      CALL PROD(GAMMA,COVW,N,IN,IN,TEMP)                                MCSP0200
      CALL TRANS(GAMMA,N,IN,TEMP1)                                      MCSP0210
      CALL PRCD(TEMP,TEMP1,N,IN,N,Q)                                    MCSP0220
      SUBROUTINE QON(A1,A2,A3)                                          MCSP0230
```

```
C
C     IF Q IS TO BE COMPUTED ON-LINE (IFLQ.NE.0) IT IS DONE
C     IN THIS SUBROUTINE
C
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),
     .XHATZ(4),
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND
C
C     THE APPROPRIATE STATEMENTS FOR COMPUTING Q ON-LINE MUST
C     BE INSERTED HERE BY THE USER
C
      RETURN
      END
      RETURN
      END
```

```
      SLBROUTINE RETAD
C
C     "RETAD" CALLED IF IFLQ AND IQQ EQ TO 1
C
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),
     .XHATZ(4),
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND
C
C     GAIN IS COMPUTED IN THIS PROGRAM BY RESIDUAL TESTING
C
C     ACAPTIVE SCHEME
      RETURN
      END
```

```
      SUBROUTINE RON                                                    MCSP0010
C                                                                       MCSP0020
C     IF R IS TO BE COMPUTED ON-LINE (IFLR.NE.0) IT IS DONE             MCSP0030
C     IN THIS SUBROUTINE                                                MCSP0040
C                                                                       MCSP0050
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI       MCSP0060
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),       MCSP0070
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0080
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),          MCSP0090
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),      MCSP0100
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),   MCSP0110
     .XHATZ(4),                                                         MCSP0120
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND           MCSP0130
C                                                                       MCSP0140
C     THE APPROPRIATE STATEMENTS FOR COMPUTING R ON-LINE MUST           MCSP0150
C     BE INSERTED HERE BY THE USER                                      MCSP0160
C                                                                       MCSP0170
      RETURN                                                            MCSP0180
      END                                                               MCSP0190
```

```
      SUBROUTINE STAT                                                   MCSP0010
C     THIS SUBROUTINE  COMPUTES RUNNING SUMS USED IN DETERMINING THE    MCSP0020
C     SAMPLE STATISTICS OF TRACK AND ESTIMATION ERRORS. IN THE DEFAULT  MCSP0030
C     OPTION (ISTAT.EQ.0) THE STATISTICS TO BE COMPUTED ARE MEAN OF     MCSP0040
C     TRACK, MEAN OF ESTIMATION ERROR AND VARIANCE OF ESTIMATION        MCSP0050
C     ERROR. IF(ISTAT.NE.0) THE OFF-DIAGONAL TERMS IN THE COVARIANCE OF MCSP0060
C     ESTIMATION ERROR MATRIX ARE ALSO COMPUTED.                        MCSP0070
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI       MCSP0080
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),       MCSP0090
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0100
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),          MCSP0110
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),      MCSP0120
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),   MCSP0130
     .XHATZ(4),                                                         MCSP0140
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND           MCSP0150
      DIMENSION EXH(3)                                                  MCSP0160
      IF(ITRK.NE.1) GO TO 2                                             MCSP0170
      IF(ITER.NE.1) GO TO 3                                             MCSP0180
      DO 4 J=1,N                                                        MCSP0190
    4 XM(J,K)=XS(J,K)                                                   MCSP0200
      GO TO 3                                                           MCSP0210
    2 CONTINUE                                                          MCSP0220
      DO 6 J=1,N                                                        MCSP0230
    6 XM(J,K)=XM(J,K)+XS(J,K)                                           MCSP0240
```

```
    3 CCNTINUE                                                            MCSP0250
      DO 1 J=1,N                                                          MCSP0260
      EXH(J)=XHKK(J)-XS(J,K)                                              MCSP0270
      ERR(J,K) =ERR(J,K)+EXH(J)                                           MCSP0280
    1 VAR(J,J,K)=VAR(J,J,K)+EXH(J)**2                                     MCSP0290
      IF(ISTAT.EQ.0) RETURN                                               MCSP0300
      DO 5 L=2,N                                                          MCSP0310
      LM1=L-1                                                             MCSP0320
      DO 5 J=1,LM1                                                        MCSP0330
    5 VAR(L,J,K)=VAR(L,J,K)+EXH(L)*EXH(J)                                 MCSP0340
      RETURN                                                              MCSP0350
      END                                                                 MCSP0360
```

```
      SUBROUTINE TRACK                                                    MCSP0010
C     IF TRACK IS TO BE GENERATED ON-LINE IT IS DONE IN THIS SUBROUTINE  MCSP0020
C     IN THE DEFAULT OPTION (ITRK.EQ.0) THE TRACK IS GENERATED           MCSP0030
C     FROM THE STANDARD LINEAR DIFFERENCE EQUATION                       MCSP0040
C                                                                         MCSP0050
C     X(K+1)=PHI*X(K)+GAMMA*W(K)                                          MCSP0060
C                                                                         MCSP0070
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI         MCSP0080
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),         MCSP0090
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0100
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),            MCSP0110
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),        MCSP0120
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),     MCSP0130
     .XHATZ(4),                                                           MCSP0140
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND             MCSP0150
      DIMENSION W(3)                                                      MCSP0160
C     ITRK NE.0 OR 1 -- SEVERAL TRACKS GENERATED,BUT NOT FROM STD.       MCSP0170
C                       LINEAR EQS.                                      MCSP0180
C         = 0 -- SEVERAL TRACKS GENERATED FROM STD LINEAR EQS            MCSP0190
C         = 1 -- ONLY ONE TRACK IS USED                                  MCSP0200
      IF(ITRK.NE.0) GO TO 100                                             MCSP0210
      CALL SNORM(IW,W,IN)                                                 MCSP0220
C     CONVERT EACH N(0,1) R.V. TO N(0,SIGW(I)) R.V.                      MCSP0230
      DO 1 I=1,IN                                                         MCSP0240
    1 W(I)=SIGW(I)*W(I)                                                   MCSP0250
      DO 3 I=1,N                                                          MCSP0260
    3 X(I)=XS(I,K)                                                        MCSP0270
      CALL VPROD(GAMMAS,W,N,IN,W)                                         MCSP0280
      CALL VPROD(PHIS,X,N,N,VTMP)                                         MCSP0290
      CALL VADD(VTMP,W,N,VTMP)                                            MCSP0300
      DO 2 I=1,N                                                          MCSP0310
```

```
      2 XS(I,K+1)=VTMP(I)                                                     MCSP0320
C       NEW VALUE OF X HAS BEEN COMPUTED AND STORED IN THE ARRAY XS           MCSP0330
        RETURN                                                                MCSP0340
    100 IF(ITRK.NE.1) GO TO 200                                               MCSP0350
C        IF(ITRK.EQ.1) THE USER MUST INSERT HERE THE STATEMENTS REQUIRED      MCSP0360
C       TO GENERATE A SINGLE TRAJECTORY AND STORE IT IN THE ARRAY             MCSP0370
C       XS(I,K),I=1,N,K=2,NSAM (NOTE THAT IF A SINGLE TRAJECTORY IS TO BE    MCSP0380
C       GENERATED, THE INITIAL CONDITION HAS BEEN READ IN AND STORED          MCSP0390
        DO 40 K=2,NSAM                                                        MCSP0400
        IF (K.GT.10) GO TO 10                                                 MCSP0410
        XS(3,K)=0.0                                                           MCSP0420
        XS(2,K)=-600.0                                                        MCSP0430
        XS(1,K)=60000.0-600.0*(K-1)                                           MCSP0440
        GO TO 40                                                              MCSP0450
     10 IF(K.GT.20) GO TO 20                                                  MCSP0460
        XS(3,K)=-20.0                                                         MCSP0470
        XS(2,K)=-600.0+XS(3,K)*(K-10)                                         MCSP0480
        XS(1,K)=XS(1,K-1)+XS(2,K)                                             MCSP0490
        GO TO 40                                                              MCSP0500
     20 IF(K.GT.30) GO TO 30                                                  MCSP0510
        XS(3,K)=0.0                                                           MCSP0520
        XS(2,K)=XS(2,20)                                                      MCSP0530
        XS(1,K)=XS(1,K-1)+XS(2,K)                                             MCSP0540
        GO TO 40                                                              MCSP0550
     30 XS(3,K)=20.0                                                          MCSP0560
        XS(2,K)=XS(2,30)+XS(3,K)*(K-30)                                       MCSP0570
        XS(1,K)=XS(1,K-1)+XS(2,K)                                             MCSP0580
     40 CONTINUE                                                              MCSP0590
        RETURN                                                                MCSP0600
    200 CONTINUE                                                              MCSP0610
C       IF THIS POINT IS REACHED, ITRK NCT EQUAL 0 CR 1 INDICATING THAT       MCSP0620
C       SEVERAL TRACKS ARE TO BE GENERATED, BUT NOT BY USING THE STD.         MCSP0630
C       LINEAR DIFFERENCE EQS..THE USER MUST SUPPLY THE APPROPRIATE           MCSP0640
C       STATEMENTS HERE.                                                      MCSP0650
        RETURN                                                                MCSP0660
        END                                                                   MCSP0670


        SUBROUTINE XZERO                                                      MCSP0010
C       THIS SUBROUTINE GENERATES THE INITIAL STATE VALUE FROM A NORMAL       MCSP0020
C       RANDOM NUMBER GENERATOR. IT IS ASSUMED THAT THE INITIAL STATE         MCSP0030
C       HAS COMPONENTS THAT ARE INDEPENDENT                                   MCSP0040
        REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI           MCSP0050
        COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),           MCSP0060
       .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),MCSP0070
```

```
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),               MCSP0080
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),           MCSP0090
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),        MCSP0100
     .XHATZ(4),                                                              MCSP0110
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND                MCSP0120
      CALL SNORM(IXZ,X,N)                                                    MCSP0130
      DO 1 I=1,N                                                             MCSP0140
    1 XS(I,1)=SIGXZ(I)*X(I)+XZMEAN(I)                                        MCSP0150
      RETURN                                                                 MCSP0160
      END                                                                    MCSP0170


      SUBROUTINE ADD (A,B,N,M,C)                                             MCSP0010
C     THIS SUBROUTINE ADDS THE NXM MATRICES A AND B, STORING THE             MCSP0020
C     RESULT IN C                                                            MCSP0030
      REAL*8 A,B,C                                                           MCSP0040
      DIMENSION A(4,4),B(4,4),C(4,4)                                         MCSP0050
      DO 152 I=1,N                                                           MCSP0060
      DO 152 J=1,M                                                           MCSP0070
  152 C(I,J) = A(I,J) + B(I,J)                                               MCSP0080
      RETURN                                                                 MCSP0090
      END                                                                    MCSP0100


      SUBROUTINE MREAD(A,N,M)                                                MCSP0010
C     THIS SUBROUTINE READS AN NXM MATRIX A ACCORDING TO THE FORMAT          MCSP0020
C     8D10.5. THE ENTRIES IN THE FIRST ROW OF A ARE READ FIRST, THEN         MCSP0030
C     THE ENTRIES IN THE SECOND ROW, AND SO ON.                              MCSP0040
      REAL*8 A                                                               MCSP0050
      DIMENSION A(4,4)                                                       MCSP0060
      DO 10 I=1,N                                                            MCSP0070
   10 READ(5,20) (A(I,J),J=1,M)                                              MCSP0080
   20 FORMAT(8F10.0)                                                         MCSP0090
      RETURN                                                                 MCSP0100
      END                                                                    MCSP0110


      SUBROUTINE MWRITE(A,N,M)                                               MCSP0010
C     THIS SUBROUTINE WRITES THE ENTRIES OF THE NXM MATRIX A                 MCSP0020
```

```
      REAL*8 A                                                          MCSP0030
      DIMENSION A(4,4)                                                  MCSP0040
      DO 10 I=1,N                                                       MCSP0050
   10 WRITE(6,20) (A(I,J),J=1,M)                                        MCSP0060
   20 FORMAT(9(2X,1PE12.5))                                             MCSP0070
      RETURN                                                            MCSP0080
      END                                                               MCSP0090


      SUBROUTINE PROD (A,B,N,M,L,C)                                     MCSP0010
C     THIS SUBROUTINE COMPUTES THE MATRIX PRODUCT AB AND STORES THE     MCSP0020
C     RESULT IN C                                                       MCSP0030
C     A = NXM,    B = MXL,    C = NXL                                   MCSP0040
      REAL*8 A,B,C,T                                                    MCSP0050
      DIMENSION A(4,4),B(4,4),C(4,4),T(4,4)                             MCSP0060
      DO 1 I=1,N                                                        MCSP0070
      DO 1 J=1,L                                                        MCSP0080
    1 T(I,J)=0.0                                                        MCSP0090
      DO 151 I=1,N                                                      MCSP0100
      DO 151 J=1,L                                                      MCSP0110
      DO 151 K = 1,M                                                    MCSP0120
  151 T(I,J) = T(I,J) + A(I,K)*B(K,J)                                   MCSP0130
      DO 2 I=1,N                                                        MCSP0140
      DO 2 J=1,L                                                        MCSP0150
    2 C(I,J)=T(I,J)                                                     MCSP0160
      RETURN                                                            MCSP0170
      END                                                               MCSP0180


      SUBROUTINE SUB (A,B,N,M,C)                                        MCSP0010
C     THIS SUBROUTINE SUBTRACTS THE NXM MATRIX B FROM THE NXM MATRIX    MCSP0020
C     A AND STORES THE RESULT IN C                                      MCSP0030
      REAL*8 A,B,C                                                      MCSP0040
      DIMENSION A(4,4),B(4,4),C(4,4)                                    MCSP0050
      DO 152 I=1,N                                                      MCSP0060
      DO 152 J=1,M                                                      MCSP0070
  152 C(I,J) = A(I,J) - B(I,J)                                          MCSP0080
      RETURN                                                            MCSP0090
      END                                                               MCSP0100
```

```
      SUBROUTINE TRANS(A,N,M,C)                                       MCSP0010
C     THIS SUBROUTINE FORMS THE MATRIX TRANSPOSE CF A STORING THE     MCSP0020
C     RESULT IN C                                                     MCSP0030
C     A = NXM,    C = MXN                                             MCSP0040
      REAL*8 A,C                                                      MCSP0050
      DIMENSION A(4,4),C(4,4)                                         MCSP0060
      DO 153 I = 1,N                                                  MCSP0070
      DO 153 J=1,M                                                    MCSP0080
  153 C(J,I) = A(I,J)                                                 MCSP0090
      RETURN                                                          MCSP0100
      END                                                             MCSP0110
```

```
      SUBROUTINE VADD(X,Y,N,Z)                                        MCSP0010
C     THIS SUBROUTINE COMPUTES THE SUM OF THE N-VECTORS X AND         MCSP0020
C     Y AND STORES THE RESULT IN THE N-VECTOR Z                       MCSP0030
C                                                                     MCSP0040
      REAL*4 X(4),Y(4),Z(4)                                           MCSP0050
      DO 1 I=1,N                                                      MCSP0060
    1 Z(I)=X(I)+Y(I)                                                  MCSP0070
      RETURN                                                          MCSP0080
      END                                                             MCSP0090
```

```
      SUBROUTINE VPROD(A,X,M,N,Y)                                     MCSP0010
C     THIS SUBROUTINE COMPUTES THE PRODUCT OF THE MXN MATRIX          MCSP0020
C     A AND THE N-VECTOR X AND STORES THE RESULT IN THE               MCSP0030
C     M-VECTOR Y                                                      MCSP0040
C                                                                     MCSP0050
C                                                                     MCSP0060
      REAL*4 A(4,4),X(4),Y(4),T(4)                                    MCSP0070
      DO 1 I=1,M                                                      MCSP0080
      T(I)=0.D0                                                       MCSP0090
      DC 1 J=1,N                                                      MCSP0100
    1 T(I)=T(I)+A(I,J)*X(J)                                           MCSP0110
      DC 2 I=1,M                                                      MCSP0120
    2 Y(I)=T(I)                                                       MCSP0130
      RETURN                                                          MCSP0140
      END                                                             MCSP0150
```

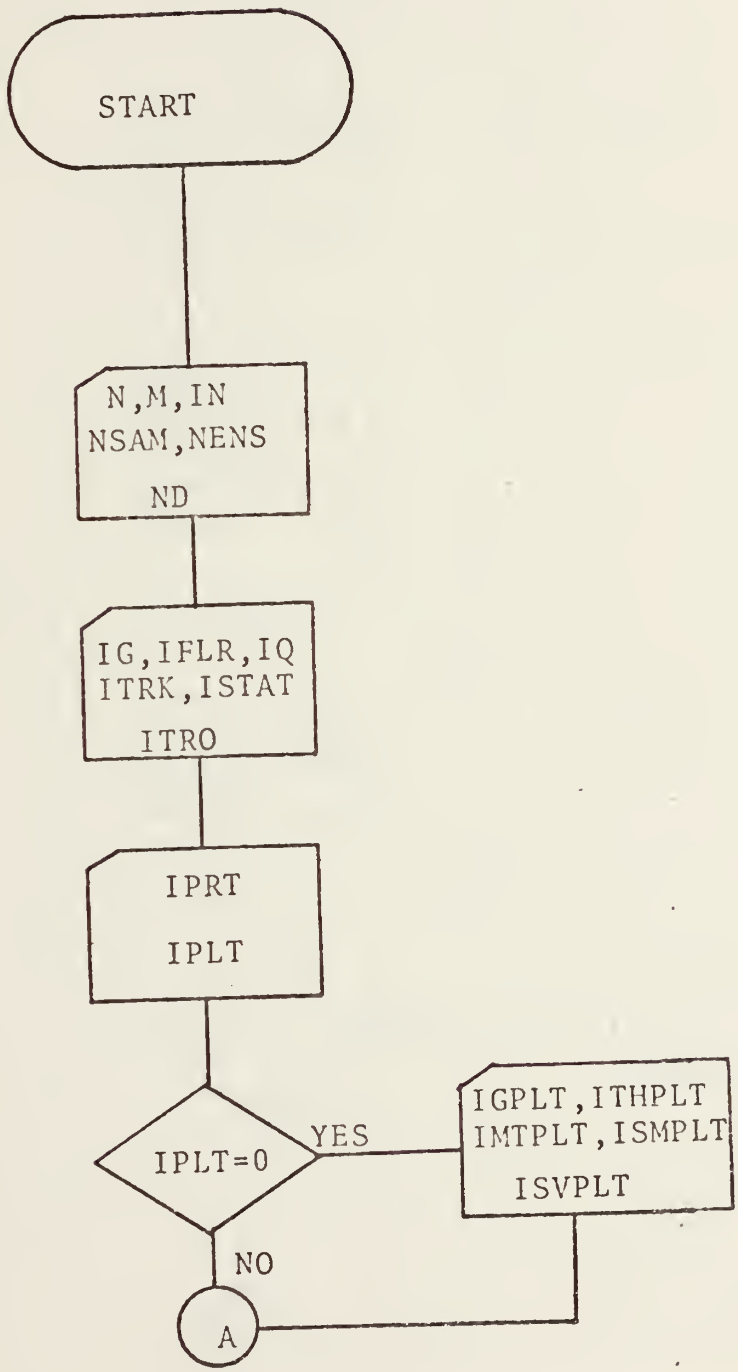
```
      SUBROUTINE VREAD(V,N)                                            MCSP0010
C                                                                      MCSP0020
C     THIS SUBROUTINE READS THE N-DIMENSIONAL S.P. VECTOR V            MCSP0030
C                                                                      MCSP0040
      DIMENSION V(4)                                                   MCSP0050
      READ(5,10) (V(I),I=1,N)                                          MCSP0060
   10 FORMAT(8F10.0)                                                   MCSP0070
      RETURN                                                           MCSP0080
      END                                                              MCSP0090
```

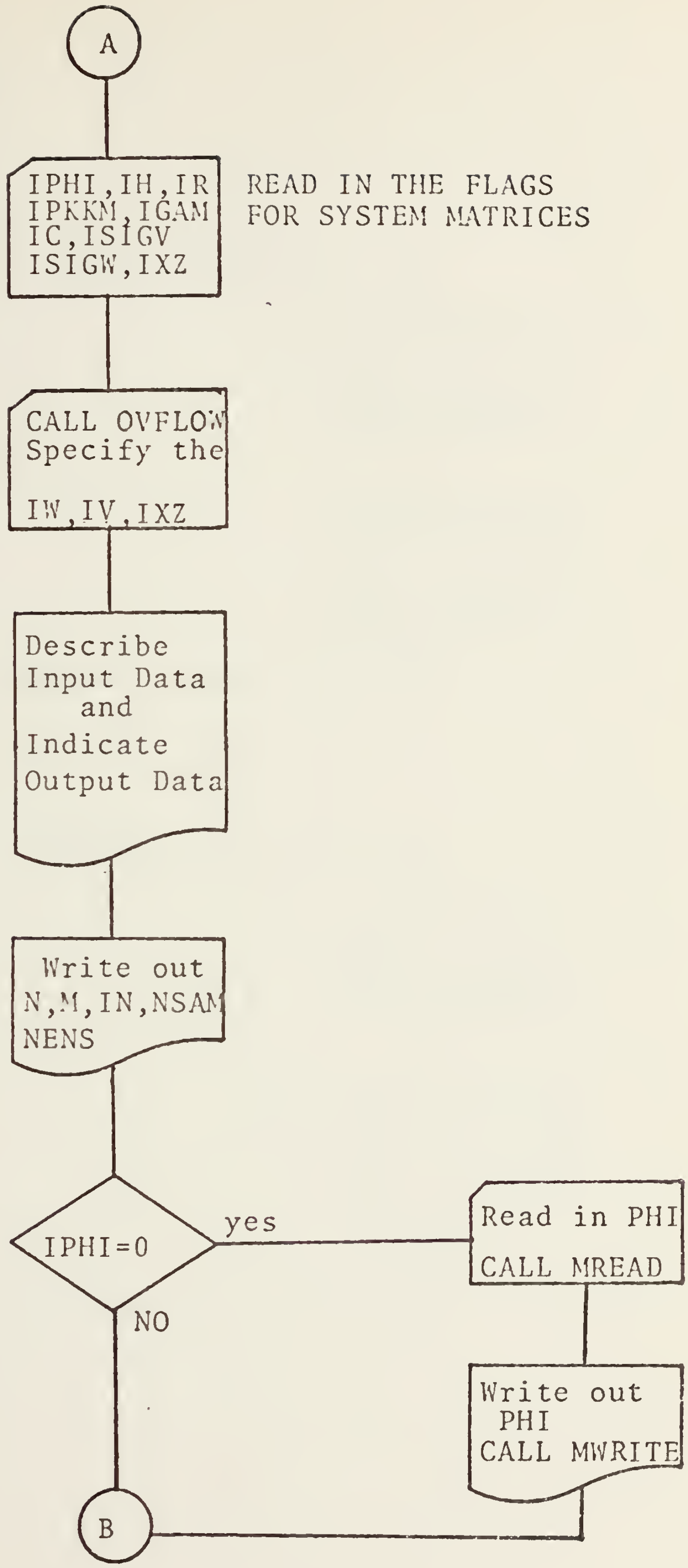
```
      SUBROUTINE VSUB(X,Y,N,Z)                                         MCSP0010
C     THIS SUBROUTINE COMPUTES THE DIFFERENCE X-Y OF THE TWO           MCSP0020
C     N-VECTORS X & Y AND STORES THE RESULT IN THE N-VECTOR Z          MCSP0030
C                                                                      MCSP0040
      REAL*4 X(4),Y(4),Z(4)                                            MCSP0050
      DO 1 I=1,N                                                       MCSP0060
    1 Z(I)=X(I)-Y(I)                                                   MCSP0070
      RETURN                                                           MCSP0080
      END                                                              MCSP0090
```

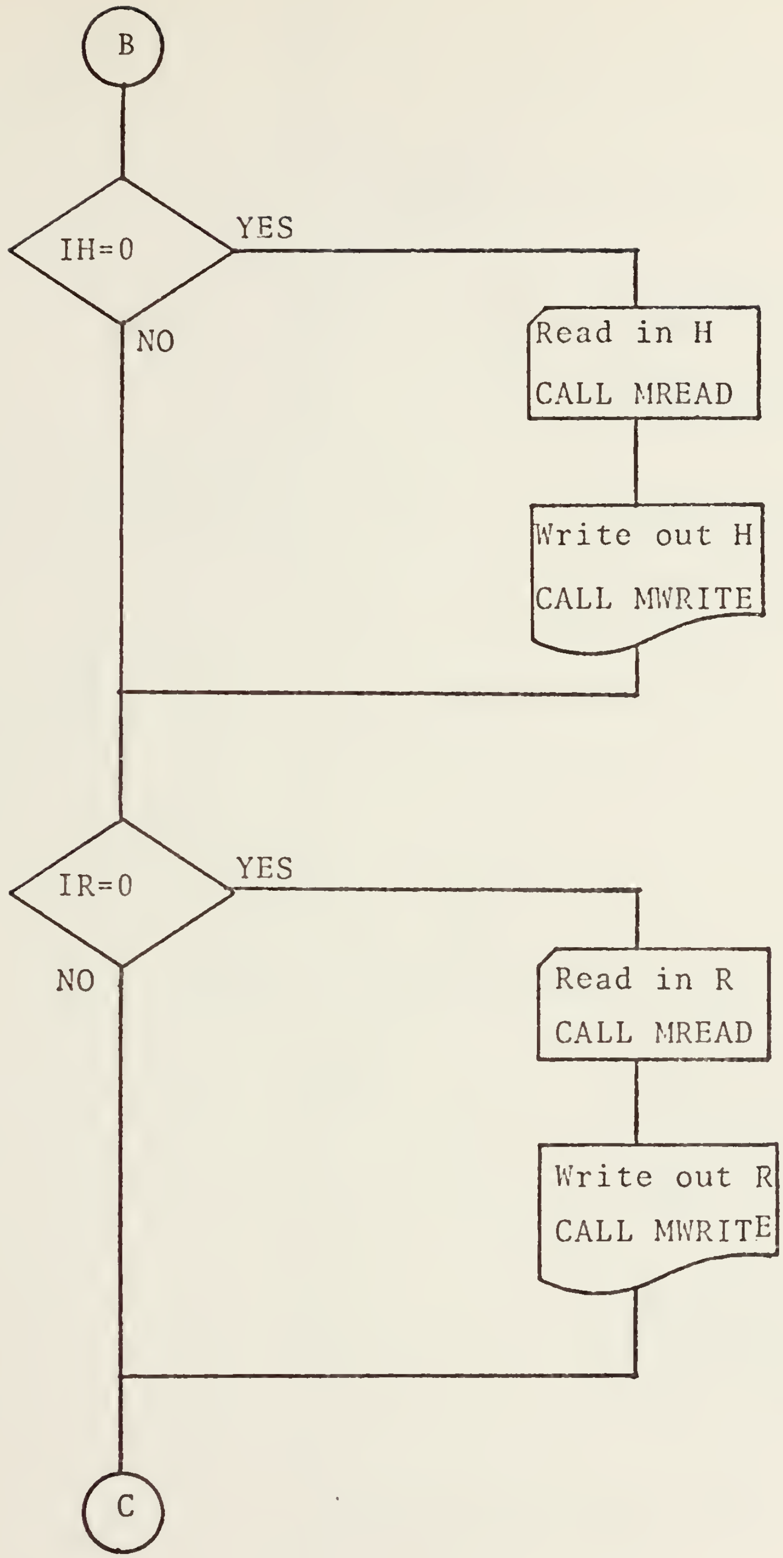
```
      SUBROUTINE VWRITE(V,N)                                           MCSP0010
C                                                                      MCSP0020
C     THIS SUBROUTINE WRITES THE N-DIMENSIONAL S.P. VECTOR V           MCSP0030
C                                                                      MCSP0040
      DIMENSION V(4)                                                   MCSP0050
      WRITE(6,10) (V(I),I=1,N)                                         MCSP0060
   10 FORMAT(9(2X,1PE12.5))                                            MCSP0070
      RETURN                                                           MCSP0080
      END                                                              MCSP0090
```

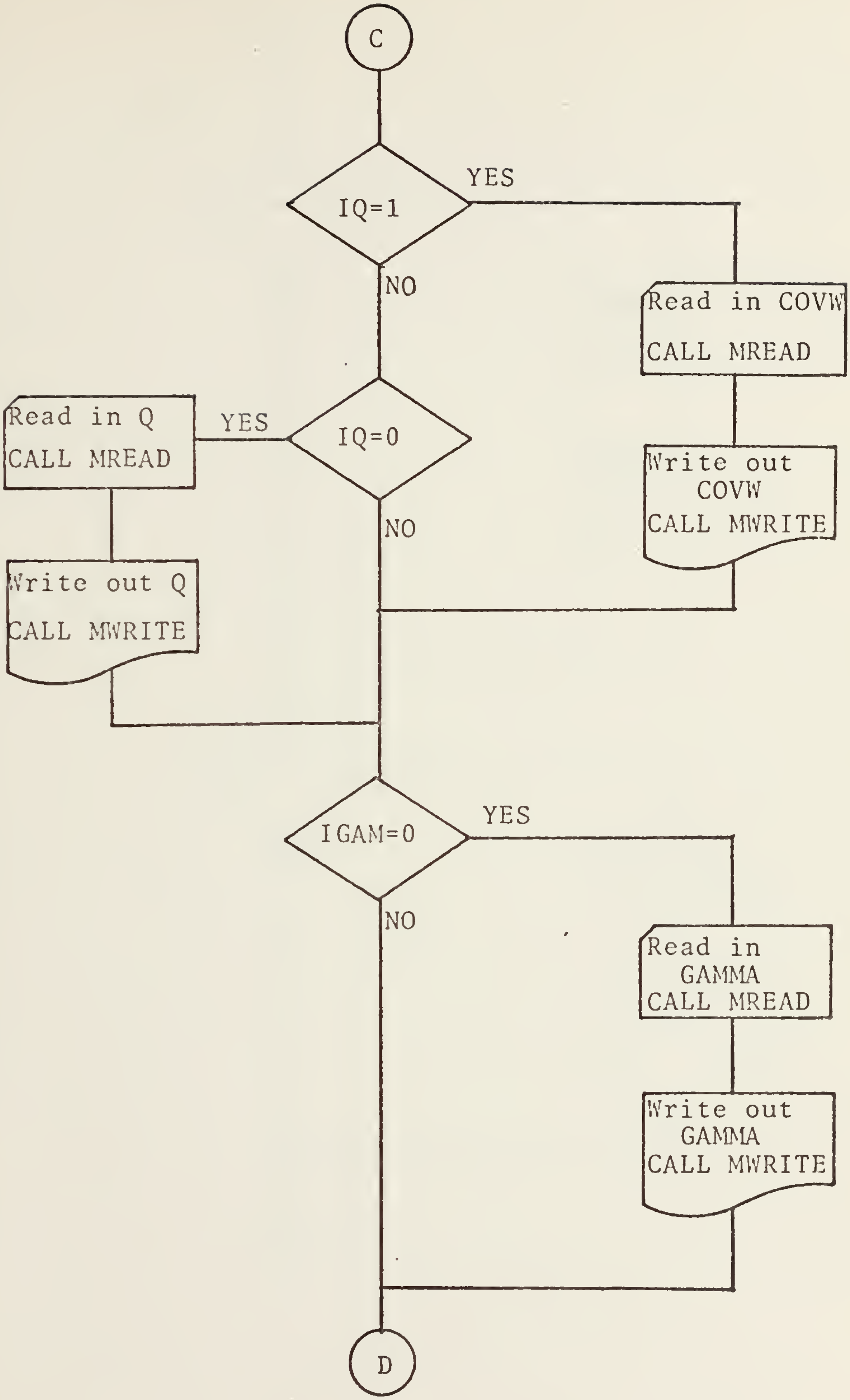
A.3 Flowchart of the Monte Carlo Simulation Program

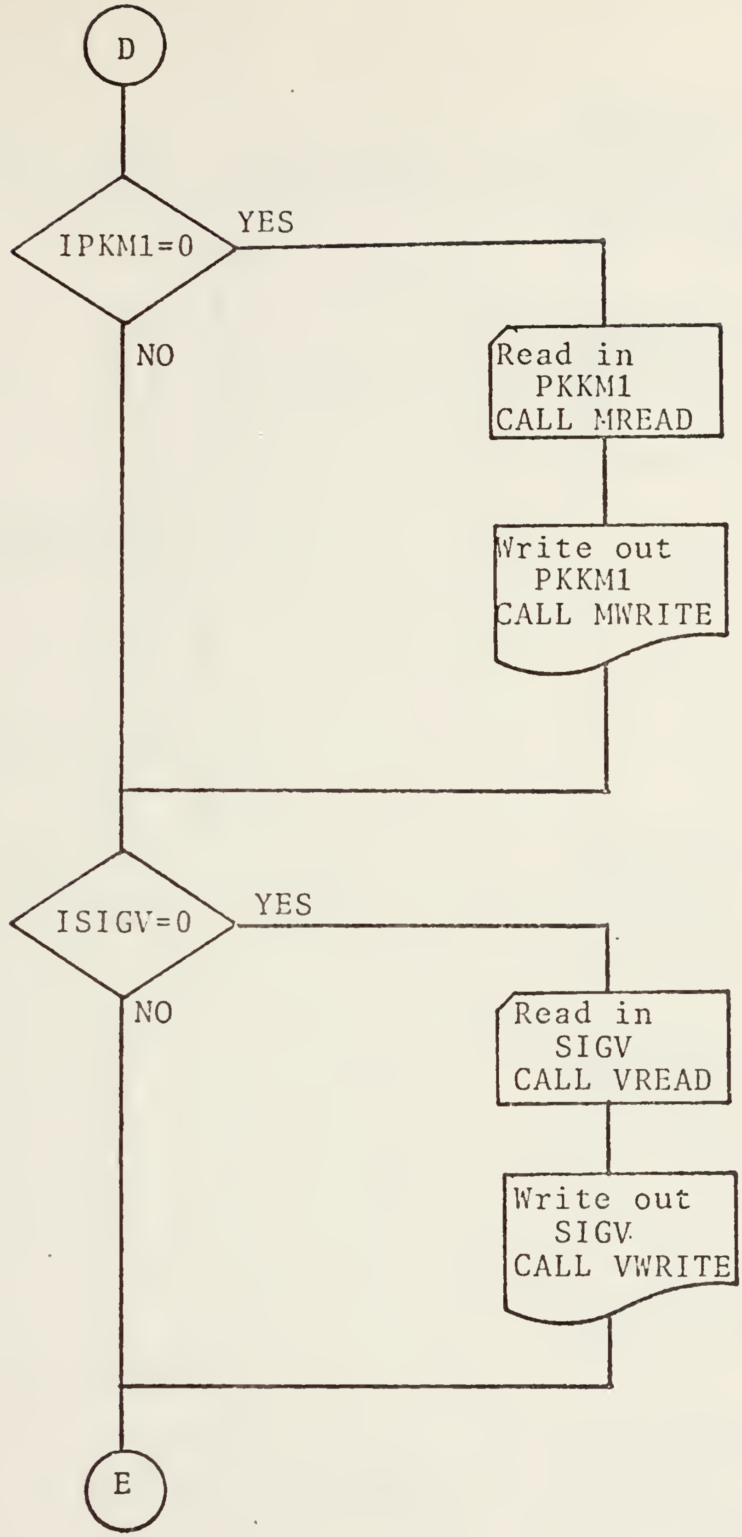
1. Main Flowchart

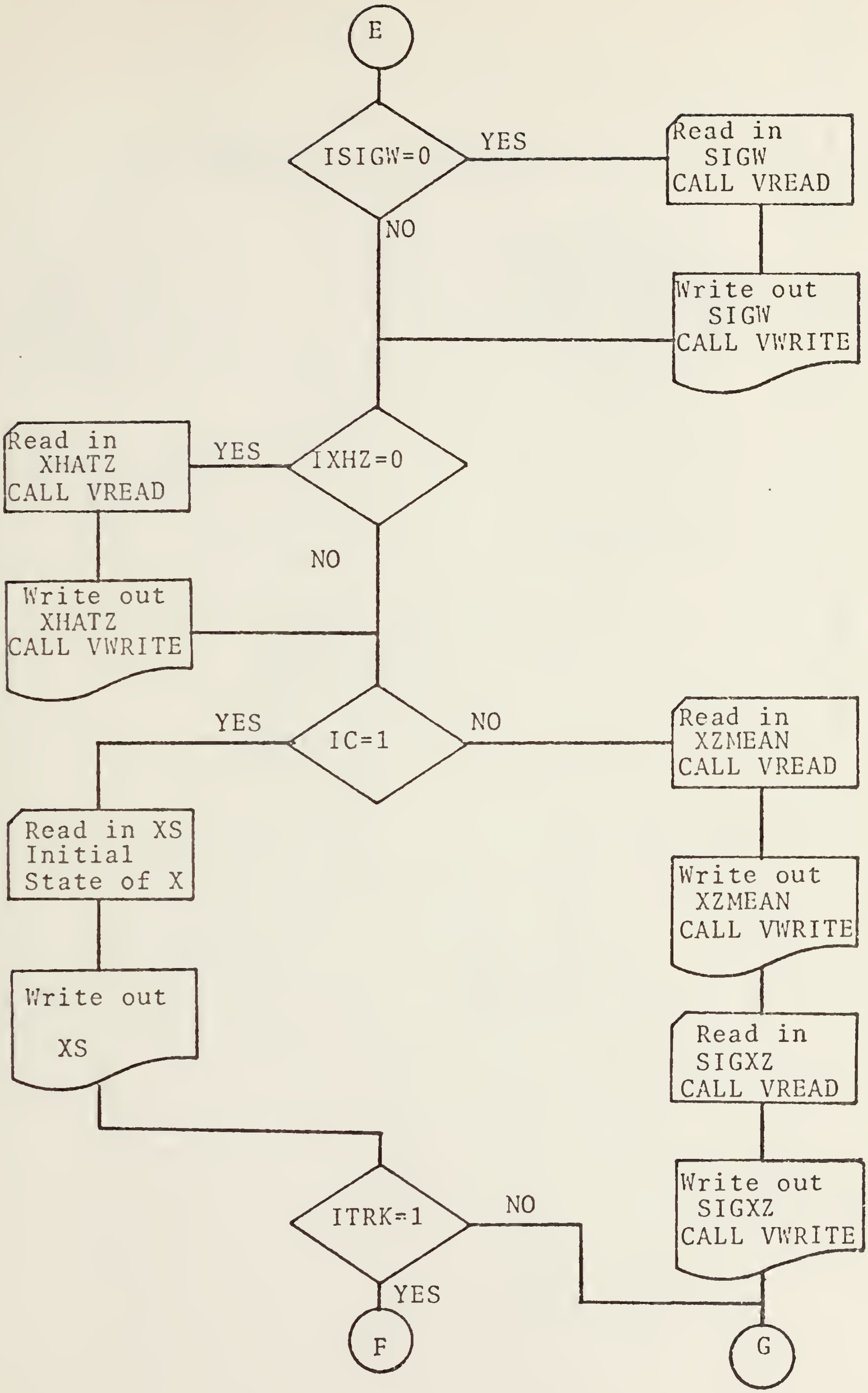


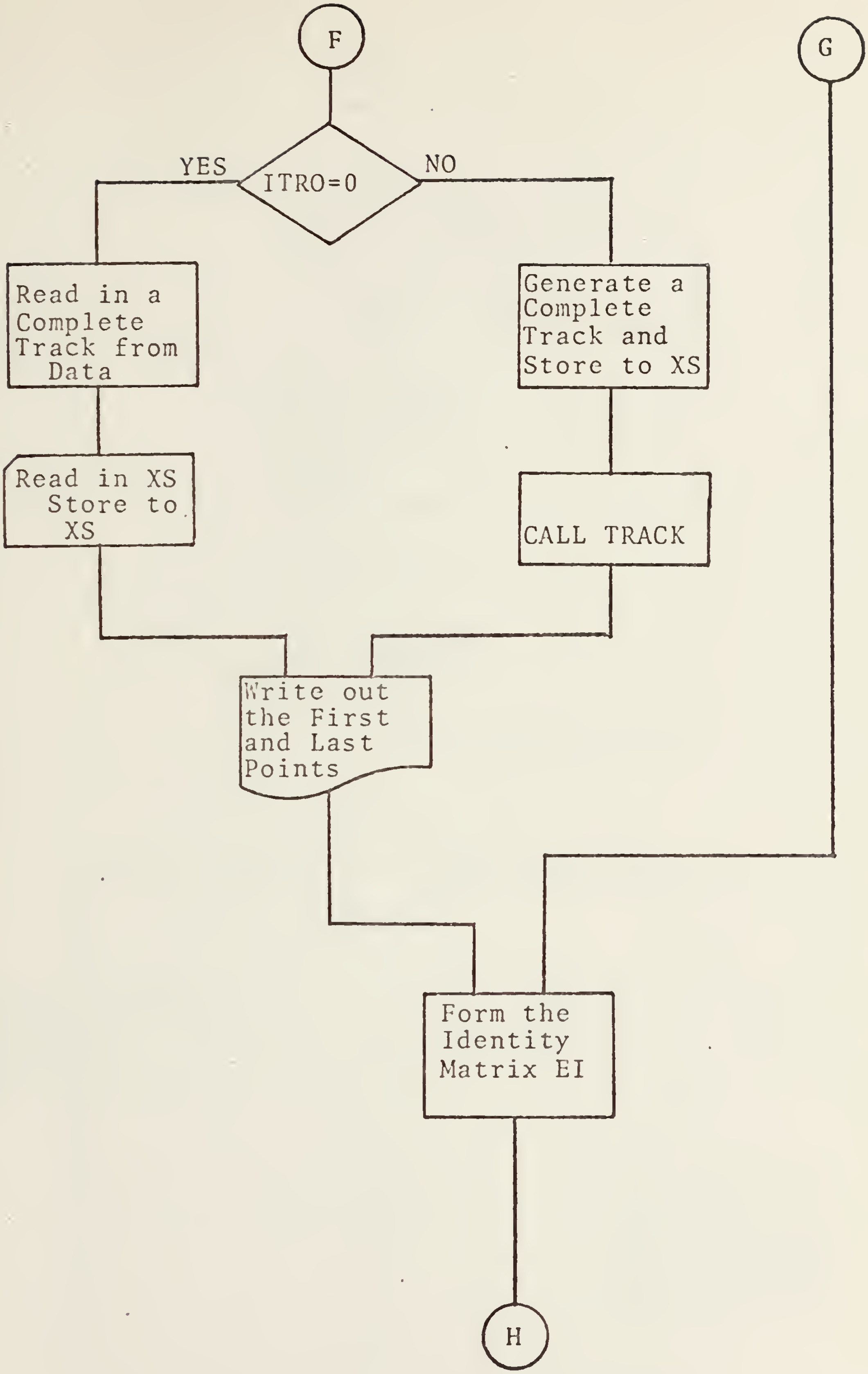

A
IPHI,IH,IR
IPKKM,IGAM
IC,ISIGV
ISIGW,IXZ
READ IN THE FLAGS
FOR SYSTEM MATRICES
CALL OVFLOW
Specify the
IW,IV,IXZ
Describe
Input Data
and
Indicate
Output Data
Write out
N,M,IN,NSAM
NENS
IPHI=0
yes
NO
Read in PHI
CALL MREAD
Write out
PHI
CALL MWRITE
B

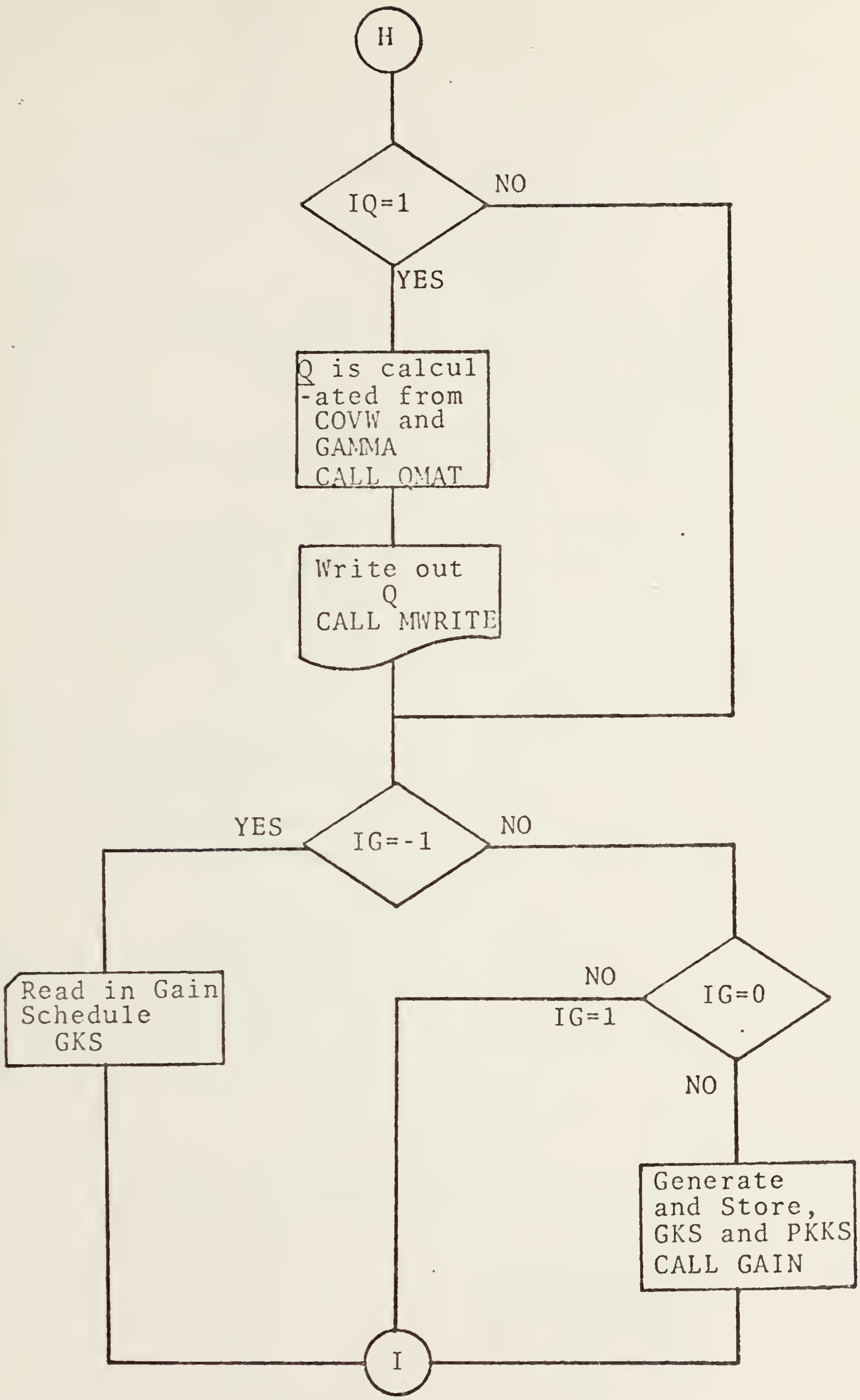

B
IH=0
YES
NO
Read in H
CALL MREAD
Write out H
CALL MWRITE
IR=0
YES
NO
Read in R
CALL MREAD
Write out R
CALL MWRITE
C

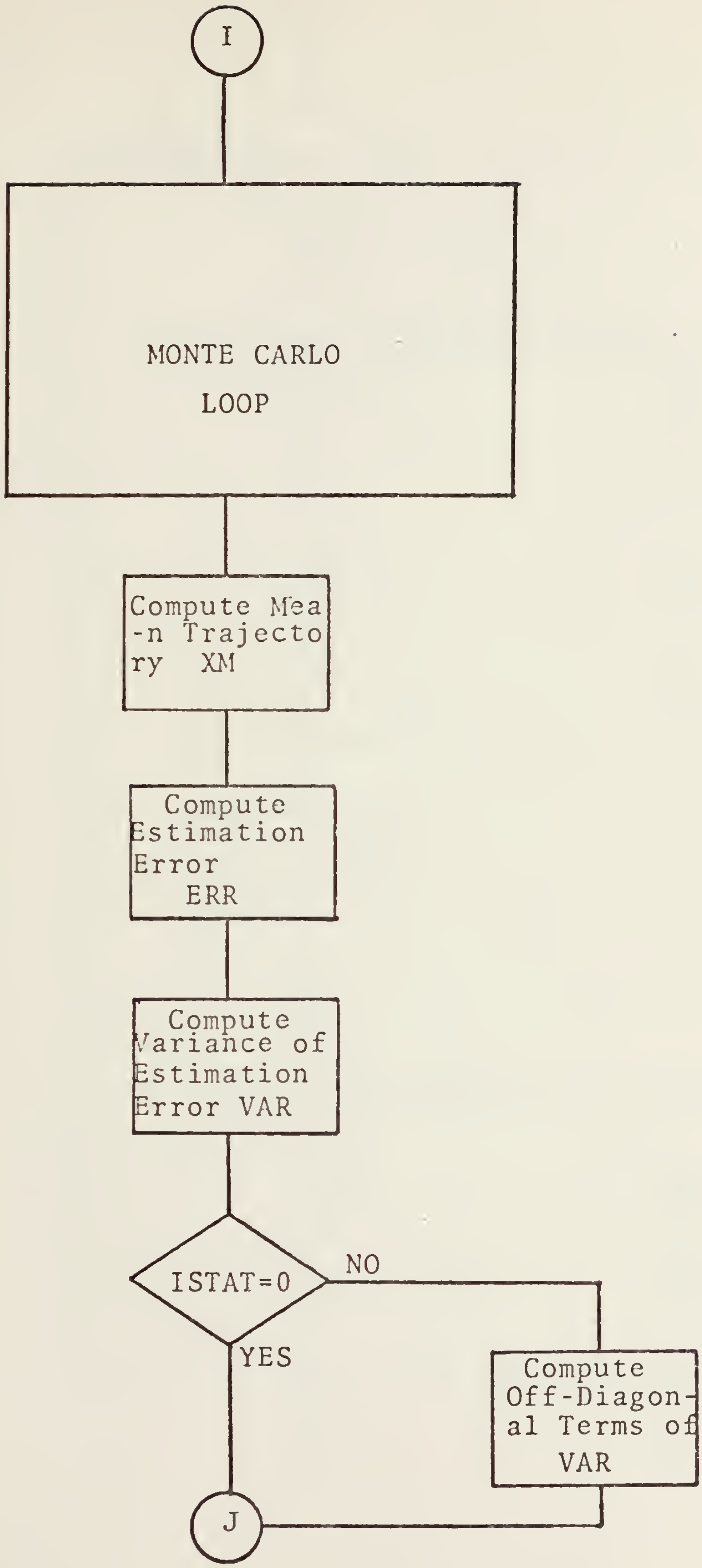

C
IQ=1
YES
NO
Read in COVW
CALL MREAD
Read in Q
CALL MREAD
YES
IQ=0
Write out
COVW
CALL MWRITE
NO
Write out Q
CALL MWRITE
IGAM=0
YES
NO
Read in
GAMMA
CALL MREAD
Write out
GAMMA
CALL MWRITE
D

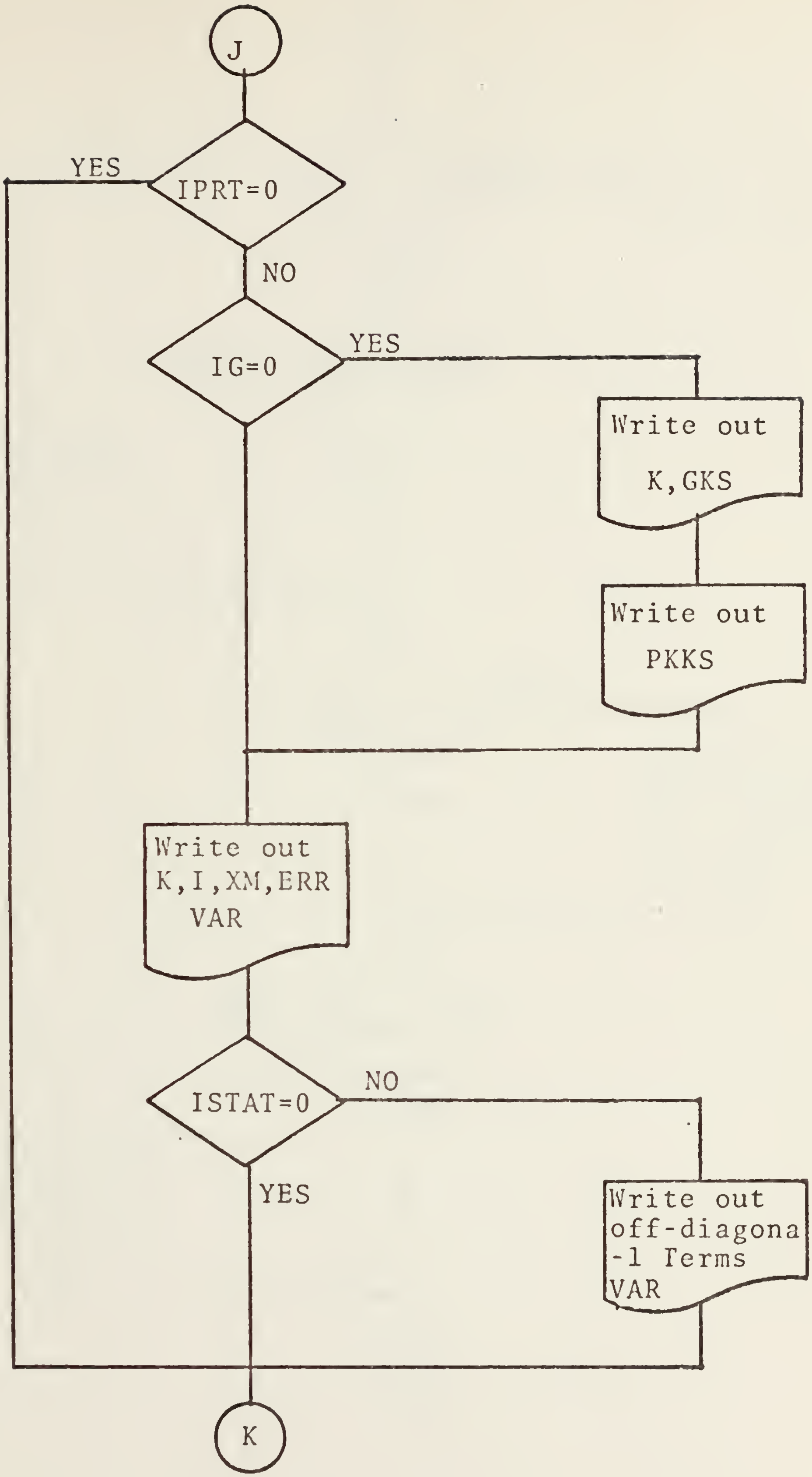

D
IPKM1=0
YES
NO
Read in
PKKM1
CALL MREAD
Write out
PKKM1
CALL MWRITE
ISIGV=0
YES
NO
Read in
SIGV
CALL VREAD
Write out
SIGV
CALL VWRITE
E

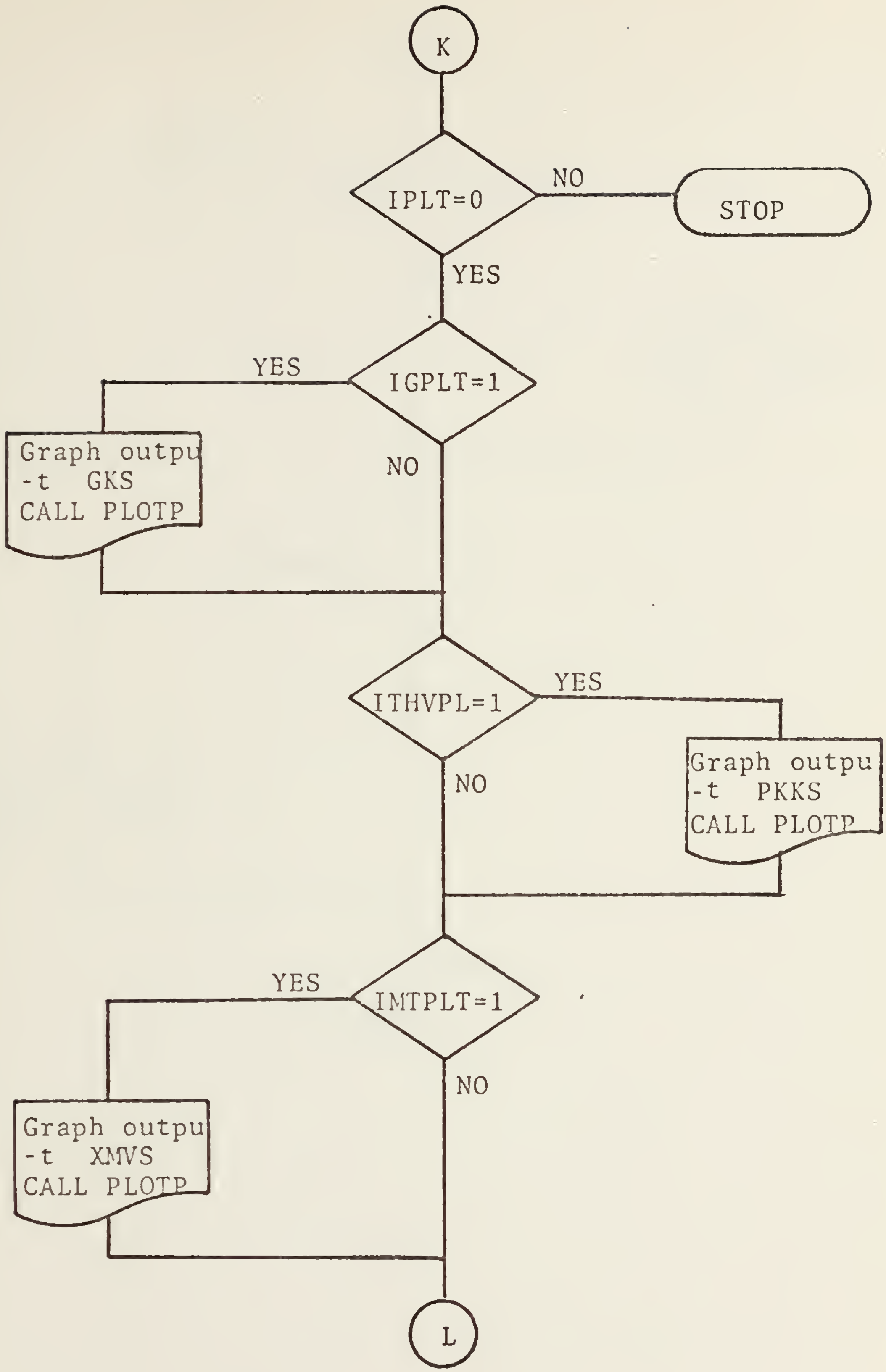

E
ISIGW=0
YES
NO
Read in
SIGW
CALL VREAD
Write out
SIGW
CALL VWRITE
Read in
XHATZ
CALL VREAD
YES
IXHZ=0
NO
Write out
XHATZ
CALL VWRITE
YES
IC=1
NO
Read in
XZMEAN
CALL VREAD
Read in XS
Initial
State of X
Write out
XZMEAN
CALL VWRITE
Write out
XS
Read in
SIGXZ
CALL VREAD
ITRK=1
NO
Write out
SIGXZ
CALL VWRITE
YES
F
G

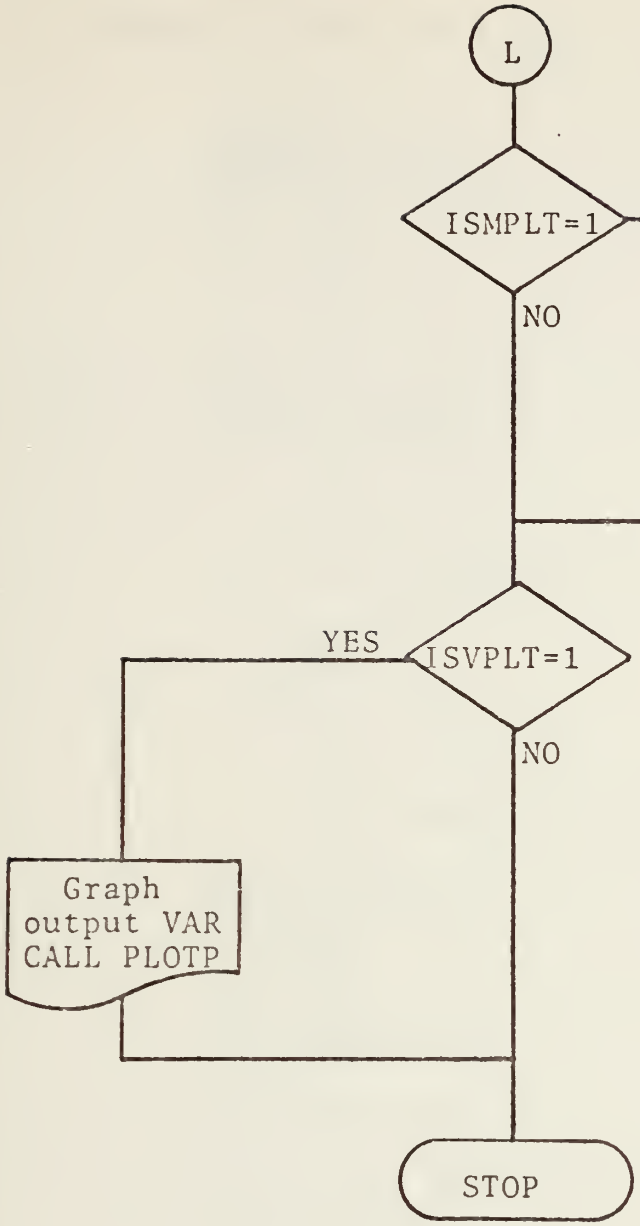

F
G
ITRO=0
YES
NO
Read in a Complete Track from Data
Generate a Complete Track and Store to XS
Read in XS Store to XS
CALL TRACK
Write out the First and Last Points
Form the Identity Matrix EI
H


H
IQ=1
NO
YES
Q is calcul
-ated from
COVW and
GAMMA
CALL QMAT
Write out
Q
CALL MWRITE
YES
IG=-1
NO
Read in Gain
Schedule
GKS
NO
IG=1
IG=0
NO
Generate
and Store,
GKS and PKKS
CALL GAIN
I

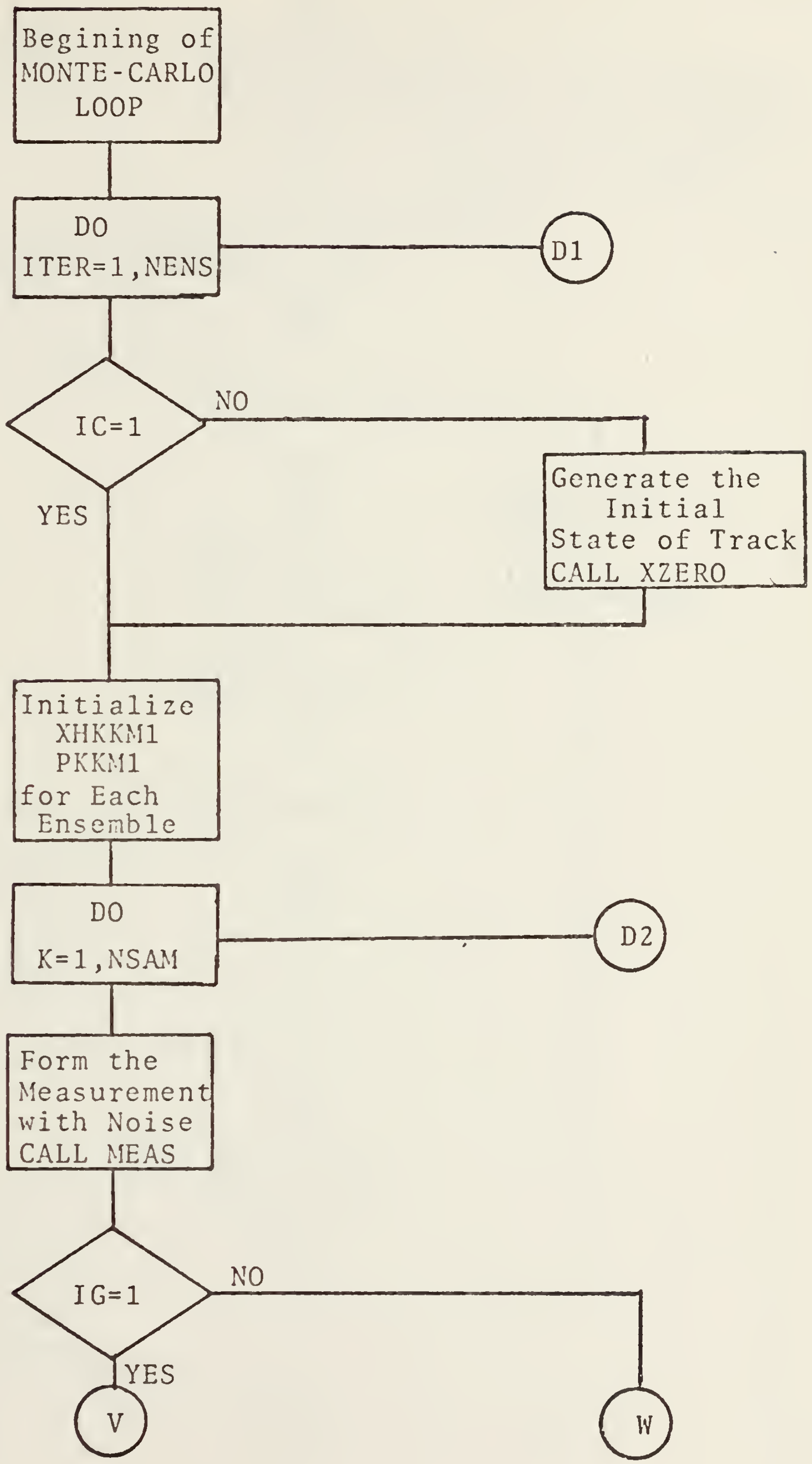

I
MONTE CARLO
LOOP
Compute Mea
-n Trajecto
ry XM
Compute
Estimation
Error
ERR
Compute
Variance of
Estimation
Error VAR
ISTAT=0
NO
YES
Compute
Off-Diagon-
al Terms of
VAR
J

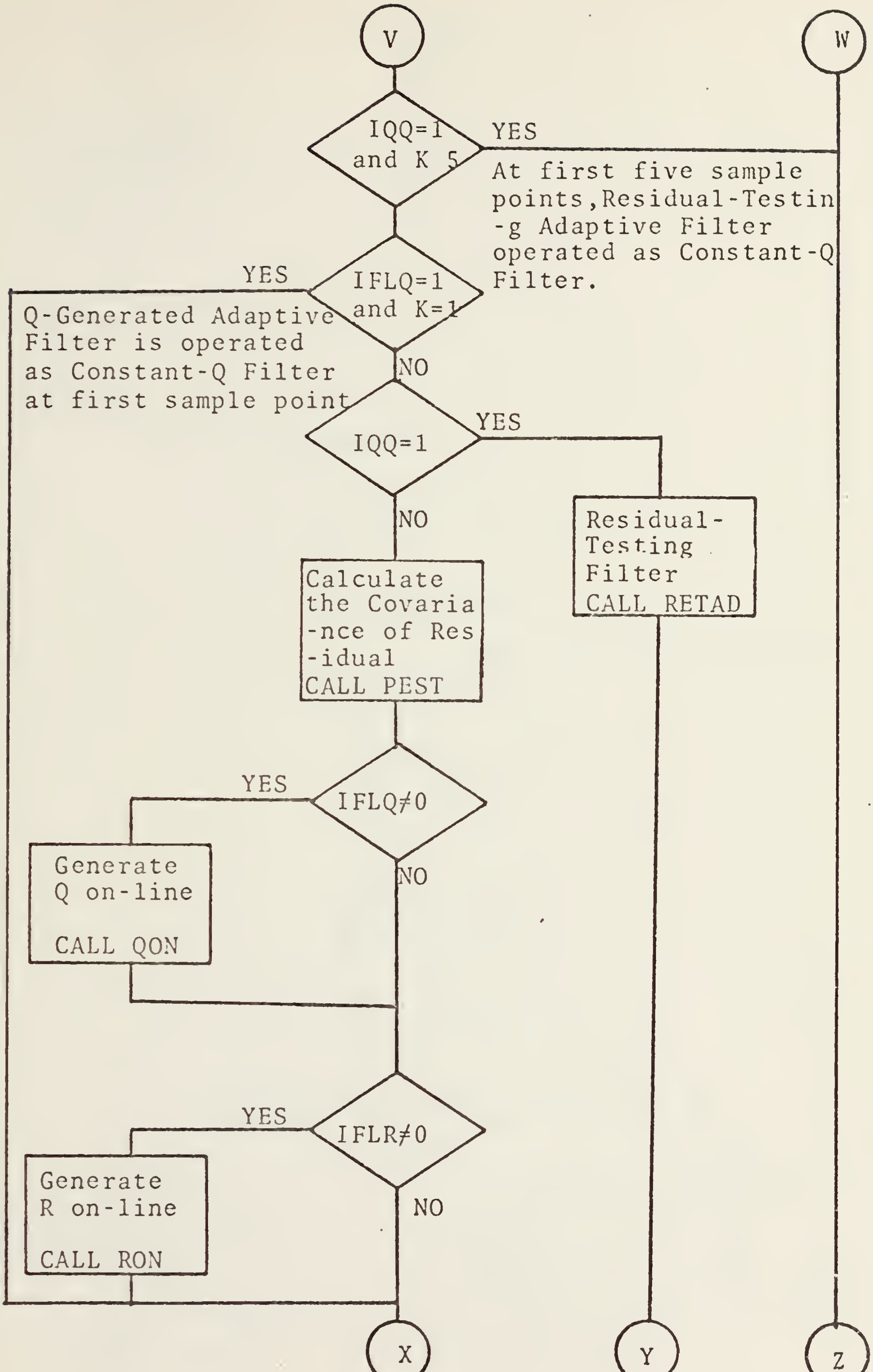

J
IPRT=0
YES
NO
IG=0
YES
Write out
K,GKS
Write out
PKKS
Write out
K,I,XM,ERR
VAR
ISTAT=0
NO
YES
Write out
off-diagona
-1 Terms
VAR
K

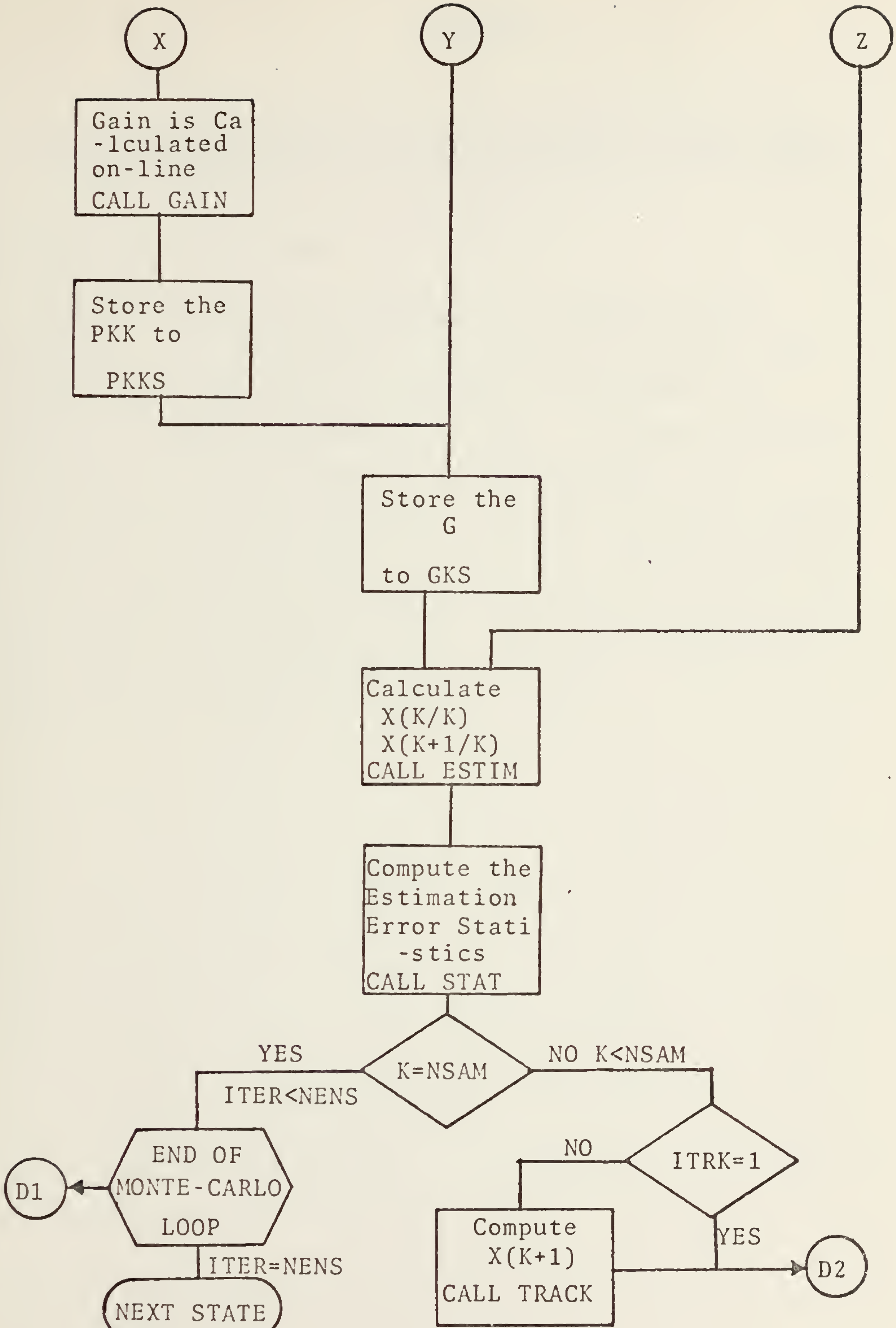

K
IPLT=0
NO
STOP
YES
YES
IGPLT=1
Graph outpu
-t GKS
CALL PLOTP
NO
ITHVPL=1
YES
NO
Graph outpu
-t PKKS
CALL PLOTP
YES
IMTPLT=1
NO
Graph outpu
-t XMVS
CALL PLOTP
L


L
ISMPLT=1
YES
NO
Graph
output ERR
CALL PLOTP
YES
ISVPLT=1
NO
Graph
output VAR
CALL PLOTP
STOP

2. Flowchart of Monte Carlo Loop




V
W
IQQ=1
and K 5
YES
At first five sample
points,Residual-Testin
-g Adaptive Filter
operated as Constant-Q
Filter.
YES
IFLQ=1
and K=1
Q-Generated Adaptive
Filter is operated
as Constant-Q Filter
at first sample point
NO
IQQ=1
YES
NO
Residual-
Testing
Filter
CALL RETAD
Calculate
the Covaria
-nce of Res
-idual
CALL PEST
YES
IFLQ≠0
Generate
Q on-line
CALL QON
NO
YES
IFLR≠0
Generate
R on-line
CALL RON
NO
X
Y
Z


X
Y
Z
Gain is Ca
-lculated
on-line
CALL GAIN
Store the
PKK to
PKKS
Store the
G
to GKS
Calculate
X(K/K)
X(K+1/K)
CALL ESTIM
Compute the
Estimation
Error Stati
-stics
CALL STAT
K=NSAM
YES
ITER<NENS
NO K<NSAM
ITRK=1
NO
YES
END OF
MONTE-CARLO
LOOP
D1
ITER=NENS
NEXT STATE
Compute
X(K+1)
CALL TRACK
D2

APPENDIX B

COMPUTER PROGRAM FOR Q-GENERATED ADAPTIVE FILTER (QON)

B.1 Program description

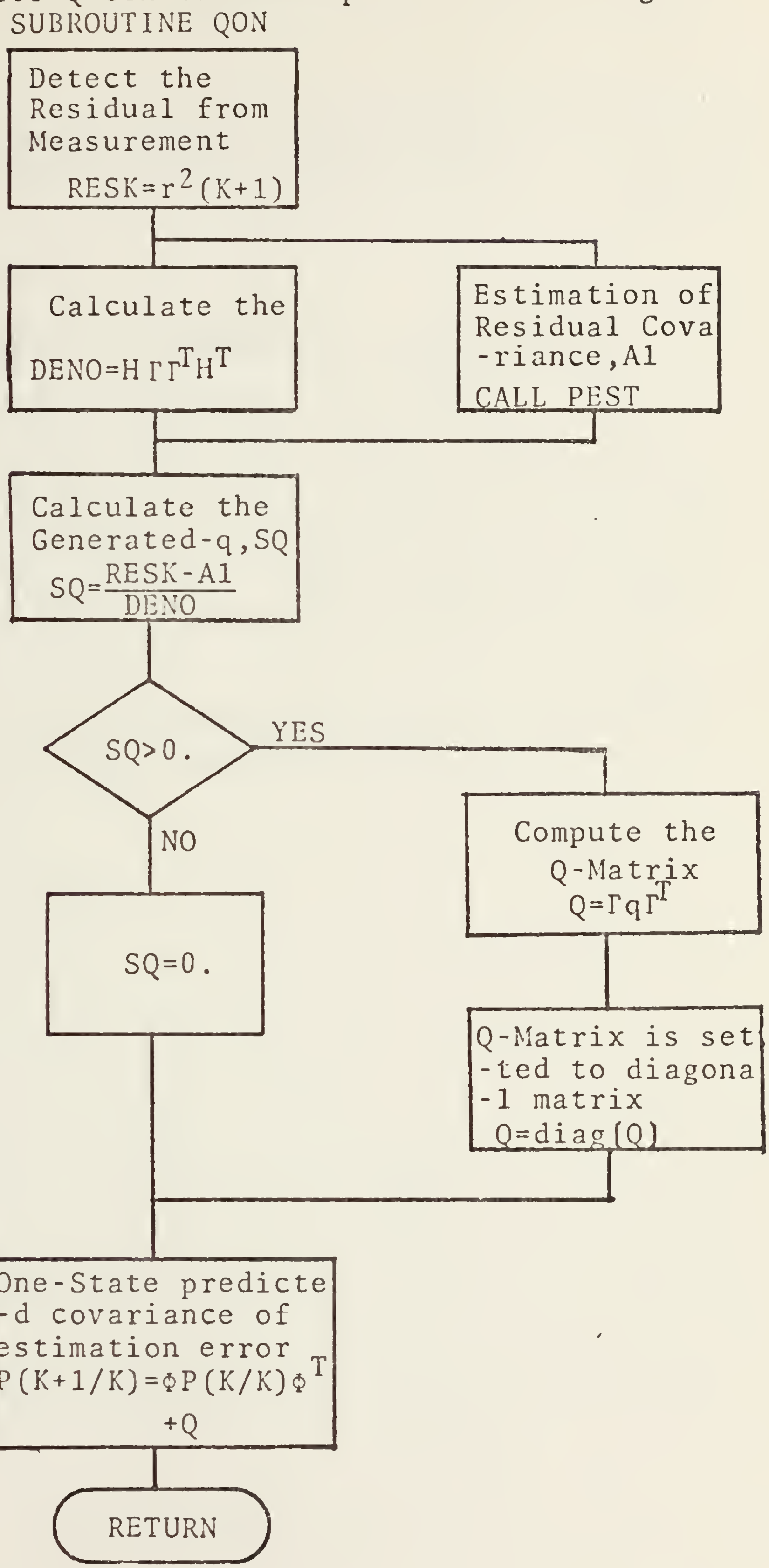
Subroutine QON is programmed to accomplish the generation of the Q-matrix . In addition to this , the program that calculates the predicted covariance of estimation error is included in "QON" . To have the gain calculation on-line under the adaptive scheme , the one-state predicted covariance of estimation error must be calculated before the gain calculation . Subroutine GAIN has the capability of calculating the one-state predicted covariance of estimation error , but this calculation is done after the gain calculation . To avoid complexity , the same program was added to subroutine QON .

B.2 COMPUTER PROGRAM FOR Q-GENERATED ADAPTIVE FILTER

```
      SUBROUTINE QON(A1,A2,A3)
C
C     IF Q IS TO BE COMPUTED ON-LINE (IFLQ.NE.0) IT IS DONE
C     IN THIS SUBROUTINE
C
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),
     .XHATZ(4),
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND
C
C     THE APPROPRIATE STATEMENTS FOR COMPUTING Q ON-LINE MUST
C     BE INSERTED HERE BY THE USER
C
C
C     FOLOWING PROGRAM ACOMPLISH THE Q-GENERATED ADAPTIVE FILTER
C
C     DETECTING THE RESIDUAL
C
      RESK=Z(1)-XHKKM1(1)
      RESK=RESK**2
C
C     CALCULATE THE DENOMINATOR (H GAM GAMT HT)
C
      CALL PROD (H,GAMMA,M,N,IN,TEMP)
      CALL TRANS (TEMP,M,IN,TEMP1)
      CALL PROD (TEMP,TEMP1,M,IN,M,TEMP2)
C
      DENO=TEMP2(1,1)
C
C     CALCULATE THE GENERATED - SQ AND INVESTIGATE
C
      SQHAT=(RESK-A1)/DENO
      IF (SQHAT.LE.0.0) GO TO 5
      SQHAT=5.0*SQHAT
      TEMP1(1,1)=SQHAT
      GO TO 10
    5 TEMP1(1,1)=0.0
C
C     FORM THE Q-MATRIX
```

```
C                                                                       MCSP0440
   10 CALL TRANS(GAMMA,N,IN,TEMP2)                                      MCSP0450
      CALL PROD(GAMMA,TEMP1,N,IN,IN,TEMP1)                              MCSP0460
      CALL PROD(TEMP1,TEMP2,N,IN,N,Q)                                   MCSP0470
C                                                                       MCSP0480
C     SET THE  Q-MATRIX TO DIAGONAL                                     MCSP0490
C                                                                       MCSP0500
      DC 15 I=1,N                                                       MCSP0510
      DO 15 J=1,N                                                       MCSP0520
      IF (I.NE.J) Q(I,J)=0.0                                            MCSP0530
   15 CCNTINUE                                                          MCSP0540
C                                                                       MCSP0550
C     CCMPUTE THE PKKM1 FOR GAIN CALCULATION                            MCSP0560
C                                                                       MCSP0570
C     NOTE HERE  PKKM1(I,J) = P(K/K-1)  WHERE                           MCSP0580
C     P(K/K-1)= PHI*P(K-1/K-1)*PHIT + Q                                 MCSP0590
      CALL TRANS(PHI,N,N,TEMP2)                                         MCSP0600
      CALL PROD(PKK,TEMP2,N,N,N,TEMP)                                   MCSP0610
      CALL PROD(PHI,TEMP,N,N,N,TEMP1)                                   MCSP0620
      CALL ADD(TEMP1,Q,N,N,PKKM1)                                       MCSP0630
      RETURN                                                            MCSP0640
      END                                                               MCSP0650
```

B.3 Flowchart for Q-Generated Adaptive Filter Program



APPENDIX C

RESIDUAL-TESTING ADAPTIVE FILTER (RETAD)

C.1 Program description

Subroutine RETAD is programmed to accomplish the Residual-testing adaptive estimator technique . This program can be separated into three parts : compute the residual from the noisy measurement , apply the Switch-on adaptive scheme , and classify the residual and assign the gains . The difference between Filter A and Filter B is only in classifying the residuals and assigning the gains .

This subroutine "retad" is called by the flag IQQ=1 , when the other flags , IFLQ and IG are equal to 1 .

C.2-1 RESIDUAL-TESTING ADAPTIVE FILTER PROGRAM FCR FILTER-1

```
C
C
C     RESIDUAL-TESTING ADAPTIVE FILTER-1
C
      SUBROUTINE RETAD
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERR(4,60),
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),
     .XHATZ(4),
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,IEST,ND
C
C     DETECT THE RESIDUAL
C
      RESK=ABS(Z(1)-XHKKM1(1))
C
C     SWITCH ON ADAPTIVE SCHEME IS DOING  HERE
C
C     SWITCH LEVEL FOR THE SWITCH-ON ADAPTIVE SCHEME SETTED AS 7.0
      IF(RESK.GT.7.00) GO TO 31
      IF(LL.EQ.0) GC TO 35
C
C     ZERC-Q STEADY STATE GAIN
      G(1,1)=0.112731
      G(2,1)=0.0050795
      G(3,1)=2.46745E-10
      GC TO 100
   35 LL=1
      GC TO 32
   31 LL=0
C     IN FOLLCWING PART , RESIDUAL IS TESTED .
C
   32 IF(RESK.GT.1.00) GO TO 8
C
C     ZERO-Q STEADY STATE GAINS
      G(1,1)=0.112731
      G(2,1)=0.0050795
      G(3,1)=2.46745E-10
      GC TO 100
    8 IF(RESK.GT.1.67) GO TO 9
C
```

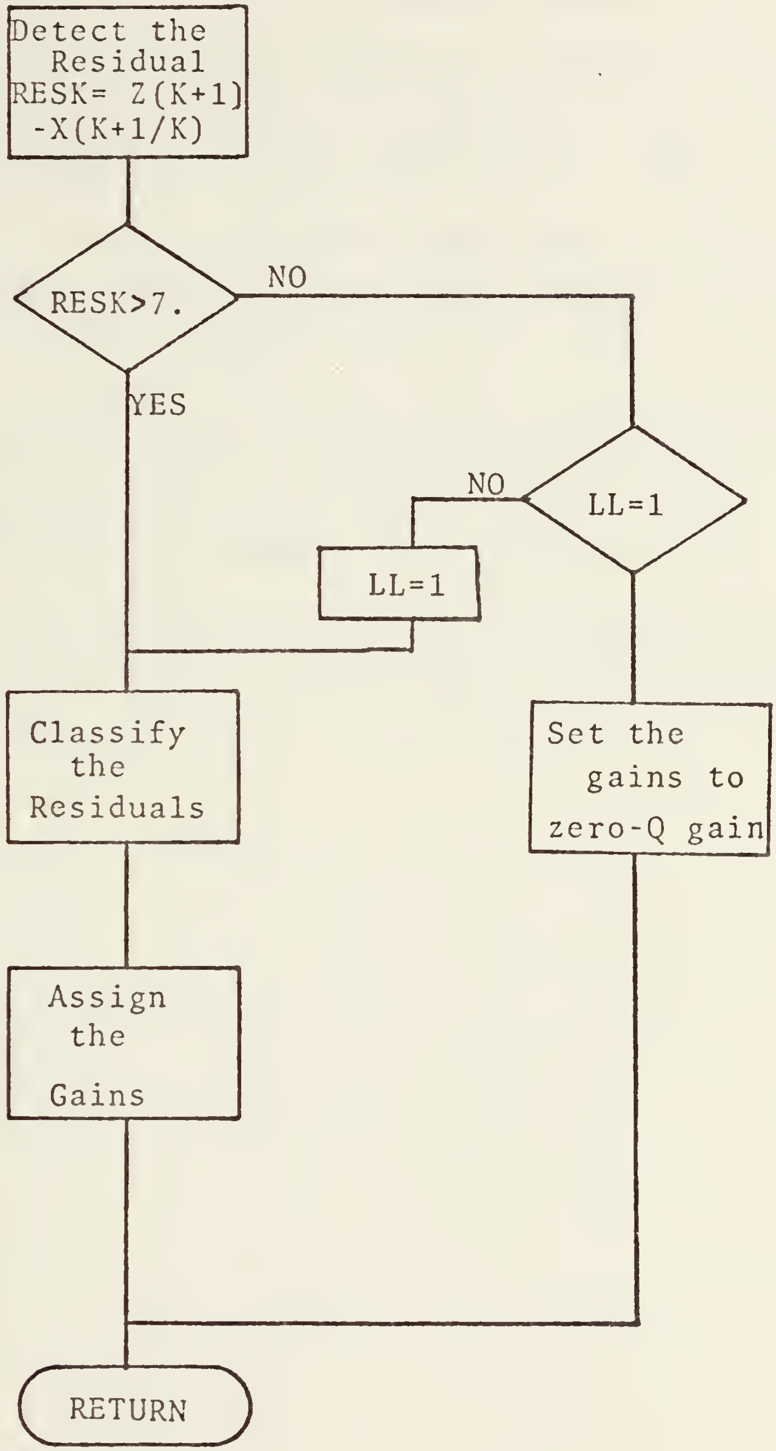
```
C      STEADY STATE GAINS FOR Q=64
       G(1,1)=0.928215
       G(2,1)=0.700369
       G(3,1)=0.129066
       GO TO 100
    9  IF(RESK.GT.3.25) GO TO 10
C
C      STEADY STATE GAINS FOR Q=144
       G(1,1)=0.961589
       G(2,1)=0.758942
       G(3,1)=0.146380
       GO TO 100
   10  IF(RESK.GT.6.41) GO TO 20
C
C      STEADY STATE GAINS FOR Q=400
       G(1,1)=0.984379
       G(2,1)=0.800035
       G(3,1)=0.158887
       GC TO 100
   20  IF(RESK.GT.12.75) GO TO 30
C
C      STEADY STATE GAINS FOR Q=625
       G(1,1)=0.989719
       G(2,1)=0.809785
       G(3,1)=0.161896
       GC TO 100
   30  IF(RESK.GT.19.08) GO TO 40
C
C      STEADY STATE GAINS FOR Q=1600
       G(1,1)=0.995854
       G(2,1)=0.821043
       G(3,1)=0.165389
       GO TO 100
C
C      STEADY STATE GAINS FOR Q=10000
   40  G(1,1)=0.999325
       G(2,1)=0.827438
       G(3,1)=0.167382
C
  100  RETURN
       END
```

C.2-2 RESIDUAL-TESTING ADAPTIVE FILTER PROGRAM FOR FILTER-2

```
C
C
C     RESIDUAL-TESTING ADAPTIVE FILTER-2
C
      SUBROUTINE RETAD
      REAL*8 GAMMA,COVW,R,PHI,H,TEMP,TEMP1,TEMP2,PKKM1,G,PKK,Q,EI
      COMMON EI(4,4),Q(4,4),G(4,4),PKK(4,4),GAMMA(4,4),COVW(4,4),
     .TEMP(4,4),TEMP1(4,4),TEMP2(4,4),H(4,4),PKKM1(4,4),R(4,4),PHI(4,4),
     .VAR(4,4,60),GKS(4,4,60),PKKS(4,4,60),XM(4,60),ERP(4,60),
     .GAMMAS(4,4),PHIS(4,4),XS(4,60),HS(4,4),GK(4,4),SIGW(4),X(4),
     .SIGXZ(4),XZMEAN(4),XHKK(4),XHKKM1(4),VTMP(4),Z(4),V(4),SIGV(4),
     .XHATZ(4),
     .N,NSAM,IQ,M,ITER,ITRK,IN,ISTAT,K,ITRO,IXZ,IV,IW,TEST,ND
C
C     DETECT THE RESIDUAL
C
      RESK=ABS(Z(1)-XHKKM1(1))
C
C     SWITCH ON ADAPTIVE SCHEME IS DOING  HERE
C
C     SWITCH LEVEL FOR THE SWITCH-ON ADAPTIVE SCHEME SETTED AS 7.0
      IF(RESK.GT.7.00) GO TO 31
      IF(LL.EQ.0) GO TO 35
C
C     ZERO-Q STEADY STATE GAIN
      G(1,1)=0.112731
      G(2,1)=0.0050795
      G(3,1)=2.46745E-10
      GO TO 100
   35 LL=1
      GO TO 32
   31 LL=0
 C
C     IN FOLLOWING PART , RESIDUAL IS TESTED .
C
   32 IF(RESK.GT.1.00) GO TO 8
C
C     ZERO-Q STEADY-STATE GAINS
      G(1,1)=0.112731
      G(2,1)=0.0050795
      G(3,1)=2.46745E-10
      GO TO 100
    8 IF (RESK.GT.1.67) GO TO 9
```

```
C                                                                     MCSP0440
C     STEADY STATE GAINS FOR Q=64                                     MCSP0450
      G(1,1)=0.928215                                                 MCSP0460
      G(2,1)=0.700369                                                 MCSP0470
      G(3,1)=0.129066                                                 MCSP0480
      GO TO 100                                                       MCSP0490
    9 IF (RESK.GT.3.25) GO TO 10                                      MCSP0500
C                                                                     MCSP0510
C     STEADY STATE GAINS FOR Q=625                                    MCSP0520
      G(1,1)=0.989719                                                 MCSP0530
      G(2,1)=0.809785                                                 MCSP0540
      G(3,1)=0.161896                                                 MCSP0550
      GO TO 100                                                       MCSP0560
   10 IF(RESK.GT.6.415) GO TO 20                                      MCSP0570
C                                                                     MCSP0580
C     STEADY STATE GAINS FOR Q=1600                                   MCSP0590
      G(1,1)=0.995854                                                 MCSP0600
      G(2,1)=0.821043                                                 MCSP0610
      G(3,1)=0.165389                                                 MCSP0620
      GO TO 100                                                       MCSP0630
   20 IF(RESK.GT.12.750) GO TO 30                                     MCSP0640
C                                                                     MCSP0650
C     STEADY STATE GAINS FOR Q=2500                                   MCSP0660
      G(1,1)=0.997326                                                 MCSP0670
      G(2,1)=0.823754                                                 MCSP0680
      G(3,1)=0.166233                                                 MCSP0690
      GO TO 100                                                       MCSP0700
   30 IF (RESK.GT.19.0800) GO TO 40                                   MCSP0710
C                                                                     MCSP0720
C     STEADY STATE GAINS FOR Q=10000                                  MCSP0730
      G(1,1)=0.999325                                                 MCSP0740
      G(2,1)=0.827438                                                 MCSP0750
      G(3,1)=0.167382                                                 MCSP0760
      GO TO 100                                                       MCSP0770
C     STEADY STATE GAINS FOR Q=100000                                 MCSP0780
   40 G(1,1)=0.999932                                                 MCSP0790
      G(2,1)=0.828560                                                 MCSP0800
      G(3,1)=0.167732                                                 MCSP0810
C                                                                     MCSP0820
  100 RETURN                                                          MCSP0830
      END                                                             MCSP0840
```

C.3 Flowchart for Residual-Testing Adaptive Filter Program ,SUBROUTINE RETAD



REFERENCES

1. Andrew H.Jazwinski., Stochastic Processes and Filtering Theory , p.311-331, Academic Press , Inc.1970

2. Andrew H.Jazwinski and A.E.Bailie, Adaptive Filtering , Analytical Mechanics Associates , Inc., Seabrook Maryland , Intrerim Report 67-6 , 1967 .

3. Andrew H.Jazwinski , Adaptive Filtering , Proc . IFAC Symp . Multivariable Control Systems , dusseldorf , Germany , October 1968 , 2 , 1-15 .

4. Andrew H.Jazwinski , Adaptive Filtering in Satellite Orbit Estimation , Second Asilomar Conference on Circuits and Systems , Pacific Grove , California , October 1968 (presented paper) .

5. John M.Parr., An Estimator for an Anti-Aircraft Gun Fire Control system , MSEE Thesis ,Naval postgraduate School , Monterey , California , 1974 .

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chirman, Code 52 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
4.	AssocProfessor D. E. Kirk, Code 52 Ki Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
5.	LTJG Iida TOSHIAKI General Education Sec., Maritime Staff Office 9-7-45, Akasaka, Minato-ku Tokyo, JAPAN	2
6.	Education Department of Postgraduate Course Defense Academy 1-10-20 Hashirimizu Yokosuka-city Kanagawa, JAPAN	1