PULSE DOPPLER AMBIGUITY RESOLUTION

Fred J. Taylor

Texas University at El Paso

Prepared for:

Army Research Office

1974

PULSE DOPPLER AMBIGUITY RESOLUTION

Prepared by

[1974]

Fred J. Taylor
Department of Electrical Engineering
University of Texas at El Paso

Prepared for

Instrumentation Directorate
White Sands Missile Range
New Mexico

Under

DAHC
(ARO) [redacted]04-72-A-0001
TASK - 7[illegible] - 3

TABLE OF CONTENTS

NARRATIVE	PAGE
ANALYSIS OF EXISTING SYSTEM	1
KALMAN FILTER. . .AN APPROACH	4
ADAPTIVE KALMAN FILTERING	8
STEADY STATE KALMAN FILTERING	13
CONCLUSIONS	16

FIGURES

APPENDIX
- A. "PULSE DOPPLER RESOLUTION"
- B. EXPERIMENTS
- C. AGC
- D. SOURCE PROGRAM
- E. SOURCE PROGRAM
- F. "ON THE COMPUTATION OF KALMAN GAINS"
- G. "TRANSIENT RESPONSE ANALYSIS ON MINI COMPUTERS"

PULSE DOPPLER AMBIGUITY RESOLUTION

The author of this note has been involved with the analysis of existing and future pulse doppler ambiguity resolution methods. The existing pulse doppler ambiguity resolution technique is supplied by R.C.A. and a future algorithm is presented by this author.

R.C.A.

The R.C.A. algorithm is a least squares range error minimization. Given, the state differential model of range and range-rate information

$$\overset{o}{x}(t) = Ax + B\overset{o}{r} + w \quad , x(0) = x_0 \tag{1}$$

where $(^o)$ denotes d/dt

$$x(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} \quad , w(t) = \begin{pmatrix} w_1(t) \\ w_2(t) \end{pmatrix}$$

$x_1(t)$ = range yds

$x_2(t)$ = range-rate yds/sec

$r(t)$ = observed range yds

$\overset{o}{r}(t)$ = observed range-rate yds/sec

$w_1(t)$ = range measurement error

$w_2(t)$ = range-rate measurement error

where

$$A = \begin{bmatrix} 0 & \delta \\ 0 & 0 \end{bmatrix} \quad , \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{2}$$

and δ = yds/sec/ spectral line conversion factor.

Let the system observations be given by

$$y = Cx + w_3(t) \tag{3}$$

where $C = (1\ 0)$ (4)

and $w_3(t)$ = observation measurement error.

R.C.A. uses the following criterion for optimality. They choose to minimize J, where

$$J = \int_0^T || r(t) - x_1(t) ||^2 \, dt = \int_0^T || \varepsilon_1(t) ||^2 \, dt \tag{5}$$

The conditions for optimality are well known and are given by the maximum principle of Pontryagin. It is interesting to note that R.C.A. chooses to minimize range error only. They do not attempt to minimize the range-rate error $\varepsilon_2(t)$. Secondly, the normed error $||\varepsilon_1(t)||^2$ is implicitly weighted by 1 "one". This may not seem critical, but it means that all errors are considered equally independent of system signal to noise ratios. Heuristically, if the range measurement errors are stochastically small, then any difference between r(t) and its estimate $x_1(t)$ is significant. However, if the range measurement noise is stochastically large, then apparent differences between r(t) and $x_1(t)$ are no longer significant. In this case, only long term error trends should be considered significant.

In summary, the author feels that there are some philosophical disadvantages associated with R.C.A. methods. They totally ignore range-rate errors in their optimality criterion and they do not weight the apparent error $r(t) - x_1(t)$ by a measure of system

signal to noise. These two features make their method especially sensitive to scintillations. This will be demonstrated in a simulation found at the end of this report.

The computational method used to solve the required necessary linear differential equations which define the optimal solution is the method of invariant imbedding. This method is generally used on nonlinear mixed two-point boundary value problems. Needless to say, using the invariant imbedding method is a tremendous overkill when applied to their two dimensional mixed two-point boundary value differential equation.

R.C.A. also uses an unusual technique to measure the accuracy of their algorithm. They compute the variance of the apparent error $r(t) - x_1(t)$ (This is called traditionally an innovations process). If the error implies an error of greater than 1/2 spectral line the estimation of $\overset{o}{r}(t)$, namely $x_2(t)$ is considered to be bogus and the estimation process is aborted. This is meaningful under ideal low noise conditions only. As mentioned before, R.C.A. does not use information about system measurement noise (signal to noise). Therefore, in a low noise case the variance of the error $r(t) - x_1(t)$ does indeed represent estimation error based on the R.C.A.'s estimation algorithm. However, if the system measurement noise is stochastically large, the estimations of $\overset{o}{r}(t)$, namely $x_2(t)$, could be statistically good but the error variance measure of $r(t) - x_1(t)$ could be large due to system noise and not estimation errors. This phenomenon has caused the system to "flag," an error in the $\overset{o}{r}(t)$ estimation when indeed there was no such error.

In 1972 the author of this report developed a new algorithm using invariant imbedding techniques. The objective function to be minimized was J, where

$$J = \int_0^T || r(t) - x_1(t)||^2_{W_1(t)} + || \overset{o}{r}(t) - x_2(t) - \delta x_1(t)||^2_{W_2(t)} \, dt$$

$$= \int_0^T || \varepsilon_1(t)||^2_{W_1(t)} + || \varepsilon_2(t)||^2_{W_2(t)} \, dt \qquad (6)$$

Here, both range and range-rate errors are minimized. In addition, these errors are weighted by $W_1(t)$ and $W_2(t)$, respectively. The weight $W_1(t)$ and $W_2(t)$ are chosen to be the inverse of the assumed a priori error variances associated with $\varepsilon_1(t)$ and $\varepsilon_2(t)$. The advantages of this technique can be found in a simulation found in the paper included in appendix A.

A new algorithm has been developed which holds promise as an r dot extraction system. It has been found to be the most accurate and versatile algorithm tested to date. Its origin is rooted in the theory of minimal variance filter (Kalman filter).

Briefly, the minimal variance filter is known to be "the" optimal estimation algorithm for linear systems being corrupted by white noise with known covariances. The Kalman filter has been successfully applied to a myriad of linear system problems. The classic difficulties associated with Kalman filters will be noted at the end of this section. The derivation of the algorithm is derived

as follows:

given equation (1), namely for x(t): n dimensional

$$\overset{o}{x}(t) = Ax(t) + B(t)\overset{o}{r}(t) + w(t) \tag{1}$$

with x(0) given by

$$x_1(0) = r(0)$$

$$x_2(0) = 0$$

and equation 3 for y(t) a scalar

$$Y(t) = Cx(t) + v(t) \tag{3}$$

Let the a priori noise statistics be given by

$$\mathcal{E}\,(w(t)) = \phi$$

$$\mathcal{E}\,(w_3(t)) = \phi$$

$$\text{cov}\ (w(t)\ w^T(\tau)) = V_w(t)\ \delta(t - \tau)$$

$$\text{cov}\ (x(0)) = V_x(0)$$

$$\text{cov}\ (w(t)\ w_3{}^T(\tau)) = 0$$

$$\text{cov}\ (w_3(t)\ w_3{}^T(\tau)) = V_v\delta(t - \tau)$$

Define the estimation error to be

$$\tilde{x}(t) = x(t) - \hat{x}(t)$$

There exists a n x n dimensional positive definite symmetric matrix error covariance matrix, say $V_{\tilde{x}}(t)$, such that

$$\text{cov}\ (\tilde{x}(t)\ \hat{x}^T(t)) = \text{cov}\ ((x(t) - \hat{x}(t))(x(t) - \hat{x}^T(t) = V_{\tilde{x}}(t) \tag{9}$$

and

$$\overset{o}{V}_{\tilde{x}}(t) = AV_{\tilde{x}}(t) + V_{\tilde{x}}(t)\, A^T - V_{\tilde{x}}(t)\, C^T V_v^{-1}(t)\, CV_{\tilde{x}}(t) + V_w(t) \tag{10}$$

with

$$V_{\tilde{x}}(0) = V_x(0) \tag{11}$$

With respect to the state ambiguity problem, $V_{\tilde{x}}(t)$ becomes, for $n = 2$

$$\overset{o}{V}_{\tilde{x}}(t)_{11} = 2\delta V_{\tilde{x}}(t)_{12} + V_{w_1} - V_{\tilde{x}}^2(t)_{11}\, V_v^{-1}$$

$$\overset{o}{V}_{\tilde{x}}(t)_{12} = V_{\tilde{x}}(t)_{22}\, \delta - V_{\tilde{x}}(t)_{12}\, V_{\tilde{x}}(t)_{11}\, V_v(t)^{-1}$$

$$\overset{o}{V}_{\tilde{x}}(t)_{22} = V_{w_2}(t) - V_{\tilde{x}}^2(t)_{12}\, V_v^{-1}$$

with $V_{\tilde{x}}(0)_{11} = V_x(0)_{11}$, $V_{\tilde{x}}(0)_{22} = V_x(0)_{22}$, $V_{\tilde{x}}(0)_{12} = V_{\tilde{x}}(0)_{12} = 0$ (13)

The general estimation of x(t) in the presence of noise w(t) and v(t) is given by, say $\overset{o}{\hat{x}}(t)$.

$$\overset{o}{\hat{x}}(t) = A\hat{x}(t) + B\overset{o}{r}(t) + K(t)[y(t) - C\hat{x}(t)] \tag{14}$$

where $\hat{x}(0) = \mathcal{E}(x(0))$ (15)

where $\mathcal{E}$ denotes expected value

where K(t) is referred to as the Kalman gain and satisfies

$$K(t) = V_{\tilde{x}}(t)\, C^T\, V_v^{-1}(t)$$

For the problem under consideration

$$K(t) = \begin{bmatrix} V_{\tilde{x}}(t)_{11} & V_v^{-1} \\ V_{\tilde{x}}(t)_{12} & V_v^{-1} \end{bmatrix} \tag{16}$$

In general, K(t) is precomputed and stored off-line (see appendix F). The resulting estimation differential equation is

$$\overset{o}{\hat{x}}_1(t) = \delta x_2(t) + K_1(t)\ (r(t) - \hat{x}_1(t)) + \overset{o}{r}(t) \qquad (17)$$

$$\overset{o}{\hat{x}}_2(t) = K_2(t)\ (r(t) - \hat{x}_1(t))$$

This algorithm has been under test and has performed extremely well. The results of this test are compared side-by-side with the existing R.C.A. algorithm. It always produced a superior answer to that obtained using R.C.A.'s method. In many cases the improvement was dramatic. The results of the simulations can be found in appendix B.

As previously noted there are some potential difficulties associated with Kalman filtering. They are in defining (usually assuming) the initial covariances $V_w(t)$, $V_v(t)$, and $V_x(0)$. These quantities are generally assumed to be known. However, if the assumption differs considerably from the actual noise covariances, poor estimation and even divergence can result. Therefore, it is important that a reasonably accurate estimate of these noise covariances be made if satisfactory performance is to be insured. The author of this report has several suggestions which, when implemented, should provide W.S.M.R. with a very powerful and sophisticated r dot estimation system.

Suggestion 1

The actual error covariance process, say $V_A(t)$

$$V_A(t) = E \begin{bmatrix} (r_0(t) - \hat{x}_1(t))^2 & (r_0(t) - \hat{x}_1(t))(\dot{r}_0(t) - \hat{x}_2(t) \\ (\dot{r}_0(t) - \hat{x}_2(t)(r_0(t) - \hat{x}_1(t) & (\dot{r}_0(t) - \hat{x}_2(t))^2 \end{bmatrix}$$

can be computed directly using observed data and the estimation vector $\hat{x}(t)$. The actual covariance $V_A(t)$ can be compared with the theoretical error covariance $V_{\tilde{x}}(t)$. The resultant error will then be used to reprogram the a priori assumption on $V_w(t)$ and $V_v(t)$.

Suggestion 2

The MPS-36 offers a unique feature which can serve to rescale the a priori covariance matrices. That feature is the system A.G.C. The output of the A.G.C. is a measure of system signal-to-noise ration. This real-time measure of noise can be used to rescale the covariance data and thereby optimize the algorithm. Typical A.G.C. noise measure data is presented and discussed in appendix C. A new algorithm, which shall be called an adaptive Kalman/A.G.C. algorithm shall now be developed.

Adaptive Filtering

It was assumed throughout this work that over an averaging interval, say 5 seconds, the noise statistics are stationary. A filter whose a priori statistics are assumed to be constant over an averaging interval shall be called a constant covariance filter. The noise covariances associated with the processes x_1 and and x_2, or equivalently the output process y should be scaled proportional to the apparent A.G.C. signal to noise ratio. It shall be assumed that over

an averaging interval, the estimation subsystem's noise covariance possesses the same qualitative time varying properties of the A.G.C. noise meteric. Consider the following simulated result which examplifies the properties of the adaptive Kalman/A.G.C. approach to pulse-doppler parameter estimation.

Example: Kalman Filter

The developed Kalman Filter fine line estimation algorithm was implemented on a PDP-11/45 at The University of Texas at El Paso. A source listing can be found in appendix D. Several numerical experiments were performed on the algorithm. All tests involve 5 second averaging intervals, and an initial target range, velocity, and acceleration of 100,000 yds, -1000 yds/sec, and 20 yds/sec^2 respectively.

The first test investigates the algorithms sensitivity to the choice of input covariances V_{w_1} and V_{w_2} for a given signal/noise ratio. The experimental results can be found in figure K 1. It can be noted that of the parameteric values tested, all choices tested successfully in that the error at 5 sec was considerably less than $\pm.5$ spectral lines. However, qualitatively there were differences. It can be seen that as the a priori estimate of V_w, and V_{w_2} decreases, the estimation time constants also increase. That is, as the a priori assumption on input noise decreases (ex: $V_{w_1} = .001$. $V_{w_2} = .001$), the Kalman filter is very reluctant to change its previous estimate since it is assumed that the estimate was obtained in a good signal/to noise environment. Contrapositively, if the input noise covariances are assumed to be

large (ex: $V_{w_1} = .1$, $V_{w_2} = .1$) the Kalman filter will readily change its previous estimate since it consider the previous estimate statistically uncertain due to increased signal/noise ratio. To use the Kalman filter optimally, the covariance weights should be determined experimentally or through simulation. A trade off is sought between fast response and therefore possible inaccuracies and a slow response which may present incomplete estimate at the end of an estimation period.

The second experiment found in figure K 2, considers a parameterization of V_{w_1} for the indicated signal to noise ratio. Here, V_{w_1} is the noise corrupting the range constraint equation $\overset{o}{x}_1(t)$ (see eq. 1). Due to the advantageous large signal/noise ratio, all parameterizations were successfull and performed essentially the same. Therefore, it can be assumed that the algorithm will work well over a wide range of assumptions in the presence of good range data.

The third experiment, found in figure K 3, considers a parameterization of V_{w_2} for the indicated signal/noise ratio. It can be noted that the algorithm is sensitive to the choice of V_{w_2} when $V_{w_2} = 1$, the Kalman filter error gains are large. Therefore, the estimate reacts rapidly to correct any apparent error. When $V_{w_2} = .01$, the Kalman filter error gains are small. This means the algorithm can not react rapidly to apparent error. This explains the large error between .2 and .6 seconds. Here it took approximately .8 seconds to purge the initial estimation error from the estimation processes.

Convergence is slow but methodical. The performance for $V_{w_2} = .1$ is found to be a compromise between the two extremes.

The experiments considered use a standard Kalman filter approach to the fine line tracking problem in the presence of short term stationary noise. The following experiments consider a radical time varying of noise behavior. This hypothesis represents simulated scintillation.

The fourth experiment found in figure K-4 involves such test. Here a noise burst, over the interval 1 to 2.5 seconds, was numerically generated to decrease by 10 dB, the nominal range and range-rate signal to noise ratio (assumed to be 20dB). As a reference experiment, $V_{w_1} = V_{w_2} = .01$, and $V_v = 1$, for all time was assumed. It can be noted that during periods of high scintillation, the reference algorithm behaved radically. The errors generated during this period remained with the estimation process in the future. Recall a decrease in V_w produces a decrease in the Kalman error gain. Thus, a A. G. C. noise meteric feedback can be used to reduce the Kalman gain during periods of high scintillation. This reduced gain will forbid the system to rapidly track apparent errors which are known to come from a noisy environment. It can be noted that with the proper scaling of V_w, significant improvements in the estimation processes can be obtained.

The fifth experiment found in Figure K 5 involves such a scintillation test. Here a noise burst, over the interval 1 to 2.5 seconds, was numerically generated to decrease, by 10 dB, the nominal range and range-rate signal to noise ratio (assumed to be 20dB). As a reference

experiment, $V_{w_1} = V_{w_2} = .01$ and $V_v = 1$, for all time was assumed. It can be noted that during periods of high scintillation, the reference algorithms behaved radically. The error and error rates that were resident in the algorithm, when the noise figure returned to a nominal value (ie: at t = 2.5 seconds) were such that the future error performance was poor. It shall be shown that the Kalman gains are inversely proportional to V_v. (This is implicitly the same as reducing the value of V_w.) Therefore, increasing the value of the observation (output) covariance V_v in harmony with an A.G.C. noise meteric, to say 4 to 10, will selectively reduce the error gains in the presence of noise burst. This has the property of stabilizing the estimate to be a minor variation in the last estimate generated under good signal/noise ratio. It can be noted that significant fine line estimation performance can be expected using an adaptive Kalman/A.G.C. philosophy.

It is of interest to more closely examine the quantitative properties of the Kalman gains under varied parameteric conditions. The trajectories of the Kalman gains $K_1(t)$ and $K_2(t)$ (see eq. 13 and 16) can then be found in figures K1-1, K1-2, K2-1, and K2-2. Consider first K1-1. It can be noted the Kalman gain K1(t) is inversely related the differences between V_w and V_v.

This property can be witnessed again in figure K1-2. A scintillation burst is assumed to occur from 1 to 2.5 seconds. Here, rescaling V_v will cause a decrease in the error feedback gain K 1

during periods of high scintillation. It is interesting to note that the trajectory time constants are small compared to the averaging interval. This feature will be developed in the next section. The arguments associated with the Kalman $K_2(t)$ are identical to the used with the Kalman gain $K_1(t)$.

Steady-State Kalman Filter

Since it can be noted that the Kalman gain trajectories tend to their steady-state value rapidly (with respect to the averaging interval) replacing the time varying Kalman gains with their steady state value would appear to be justifiable. In addition, if the numerical integration of equation 13 can be bypassed. Thus a potential problem as numerical divergence of the generating differential equation when large value of V_w or V_v are used, is avoided.

The steady state covariance matrix $V_{\tilde{x}}(t)$ positive definite symmetric error must satisfy

$$\frac{d\, V_{\tilde{x}}(t)}{dt} = V_{\tilde{x}}(t) = 0 \tag{18}$$

therefore

$$\begin{aligned} 0 &= 2\,\delta\, V_{\tilde{x}_{12}} + V_{w_1} - V_{\tilde{x}_{11}}^2\, V_v^{-1} \\ 0 &= \delta\, V_{\tilde{x}_{22}} - V_{\tilde{x}_{12}}\, V_{\tilde{x}_{11}}\, V_v^{-1} \\ 0 &= V_{w_2} - V_{\tilde{x}_{12}}^2\, V_v^{-1} \end{aligned} \tag{19}$$

which simplifies to the following algebraic relationship

$$V_{\tilde{x}_{12}} = (V_{w_2} V_v)^{1/2}$$

$$V_{\tilde{x}_{11}} = (V_v (2 \delta V_{\tilde{x}_{12}} + V_{w_1}))^{1/2} \qquad (20)$$

$$V_{\tilde{x}_{22}} = V_{\tilde{x}_{12}} V_{\tilde{x}_{11}} \quad V_v \delta$$

A source listing of a steady state Kalman gain algorithm can be found in appendix E. There are several distinct advantages associated with a steady state algorithm. First, it is trivial to implement since it only requires the algebraic computation of the gains found in equation (20). These gains are formally substituted into the estimation differential equation (14). This equation is solved numerically using any method applicable to constant coefficient state differential systems (see appendix G). Secondly, a problem common to all numerical integration methods is that they may diverge (ie: induce a floating point error) for certain parameteric events. This is particularly true when dealing with the error covariance differential equation found in Kalman filtering. The algebraic equation found in equation (20) will produce an approximation error covariance which cannot become numerically unstable. A numerical experiment used to demonstrate the effects of an adaptive Kalman/A.G.C. philosophy is treated using the steady-state algorithm. Again, the a priori noise covariance shall be chosen to be in harmony with the A.G.C. error meteric. The result of this experiment can be found in figure KSS-1

-2. The qualitative results obtained are similar to those obtained using an adaptive Kalman/A.G.C. algorithm. Due to it being purely algebraic, cannot diverge if its parameters are finite.

To show the utility of such a technique numerical experiments were performed. The initial target information was identical to that found in the Kalman filter test. However, the time varying Kalman gains shall now be replaced by a steady-state approximation.

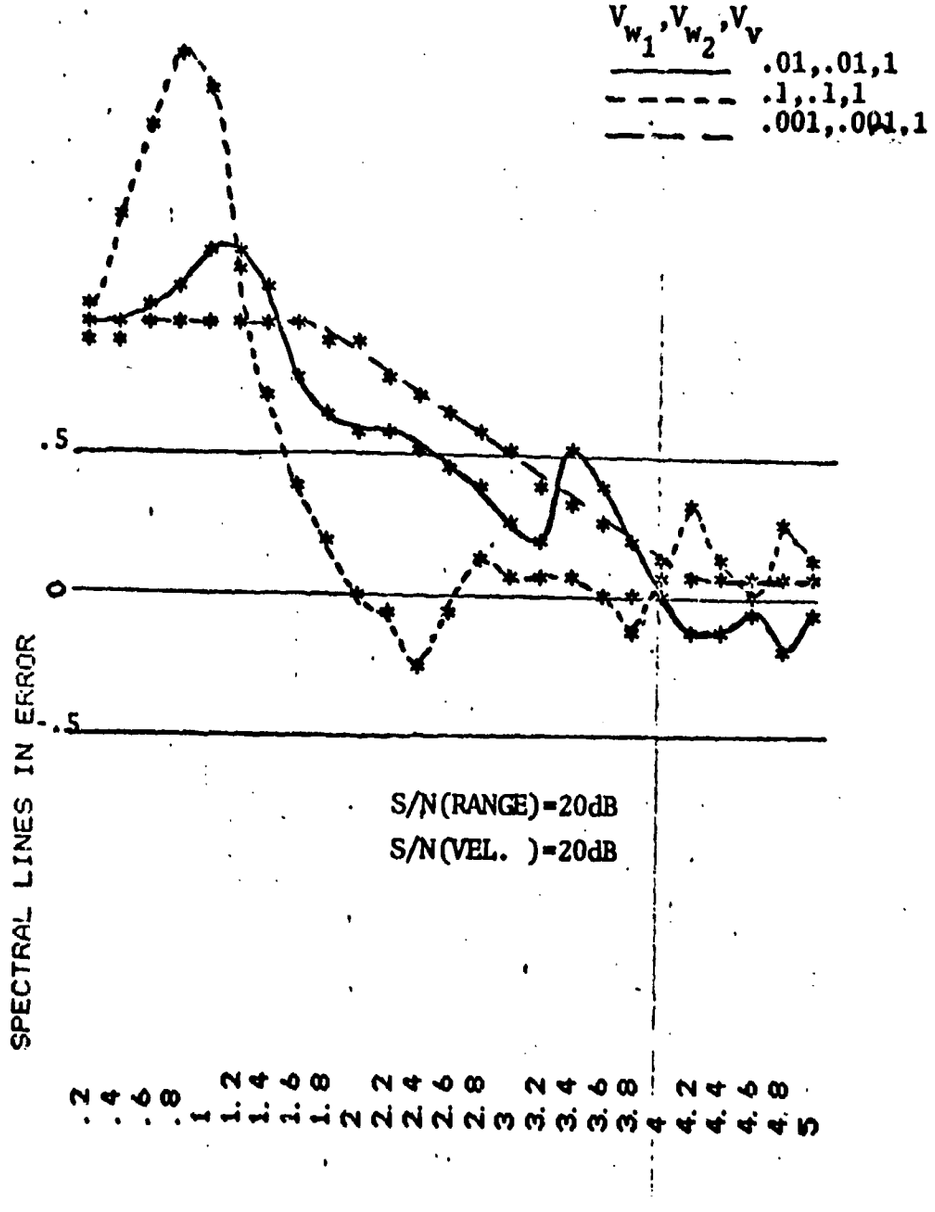
The first experiment, described in figure KSS-1, shows that the steady-state algorithm error estimate does converge to an acceptable error value over a wide range of V_w. The adaptive A.G.C. philosophy was also integrated in a steady-state Kalman filter configuration. The results of this experiment can be found in figure KSS-2. A reference test using a 10dB decrease of a nominal signal to noise ratio, namely 30dB, was assigned the parameteric value of $V_{w_1} = V_{w_2} = .1$ and $V_v = 1$. It can be seen that erratic estimation behavior occurs in the presence of strong scintillation. As the a priori output covariance parameter V_v is increased in harmony with increasing noise, the steady-state Kalman error gain decreases. The reduction in gain results in reduced fluxuations in the fine line estimate during a high noise condition. The adaptive steady-state Kalman gain algorithm is an improved estimate.

Conclusions

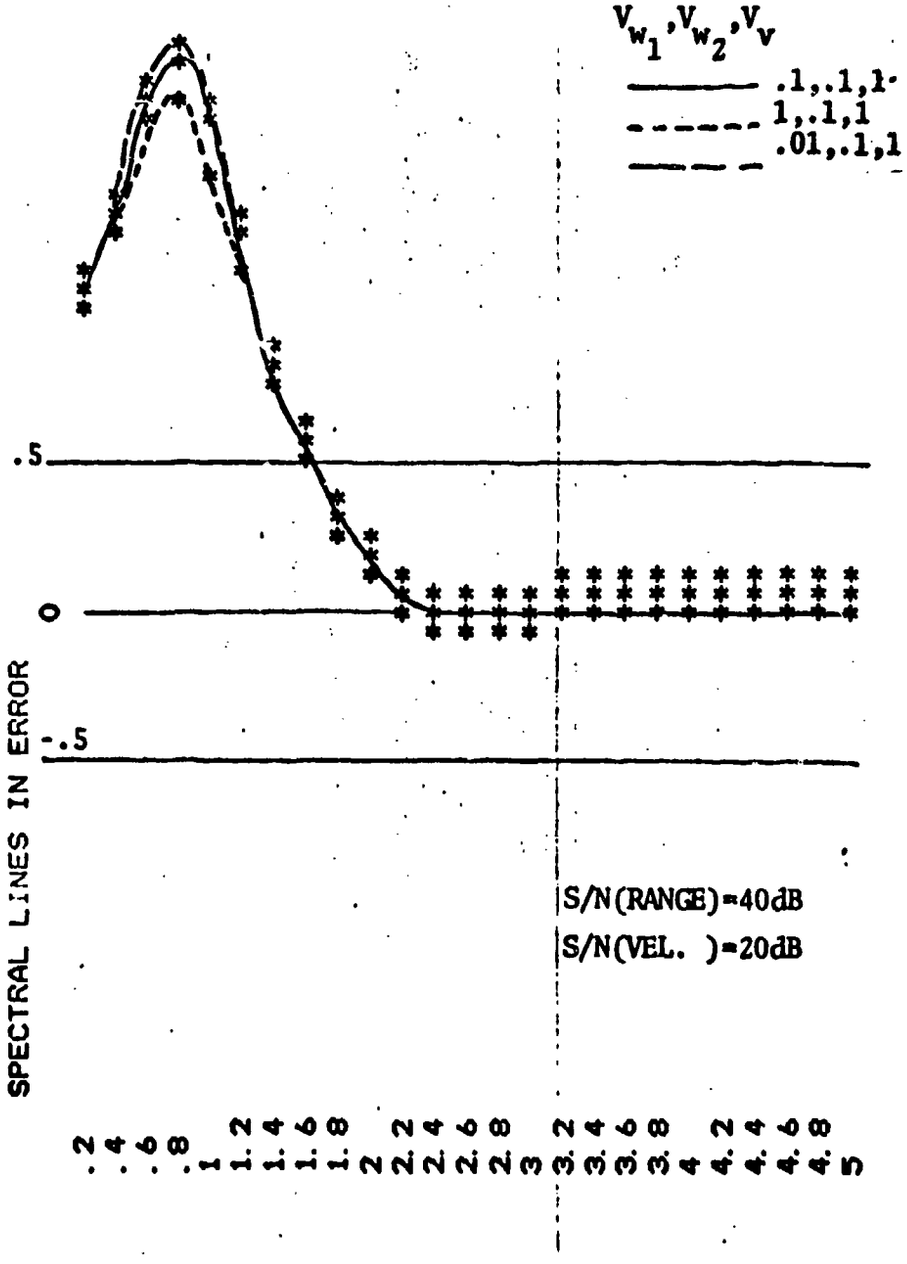
The existing R.C.A. ambiguity resolution method has been tested and found to perform satisfactorily when the received data is scintillation free. These noise bursts produce extremely large estimation errors and poor convergence properties, this is due to their algorithm minimizing only range error, with a fixed time-invariant weight (namely the number 1).

The new algorithm using Kalman filter produces superior results. It embodies the simultaneous minimization of both range and range-rate errors and uses optimal time varying weights, namely covariance information. It is forcefully felt that using Suggestion 2 as the modifier, the Kalman filter algorithm would give W.S.M.R. a reliable and flexible MPS-36 based r dot extraction system. In this configuration, system performance would be limited primarily by hardware limitations intrinsic to the system.

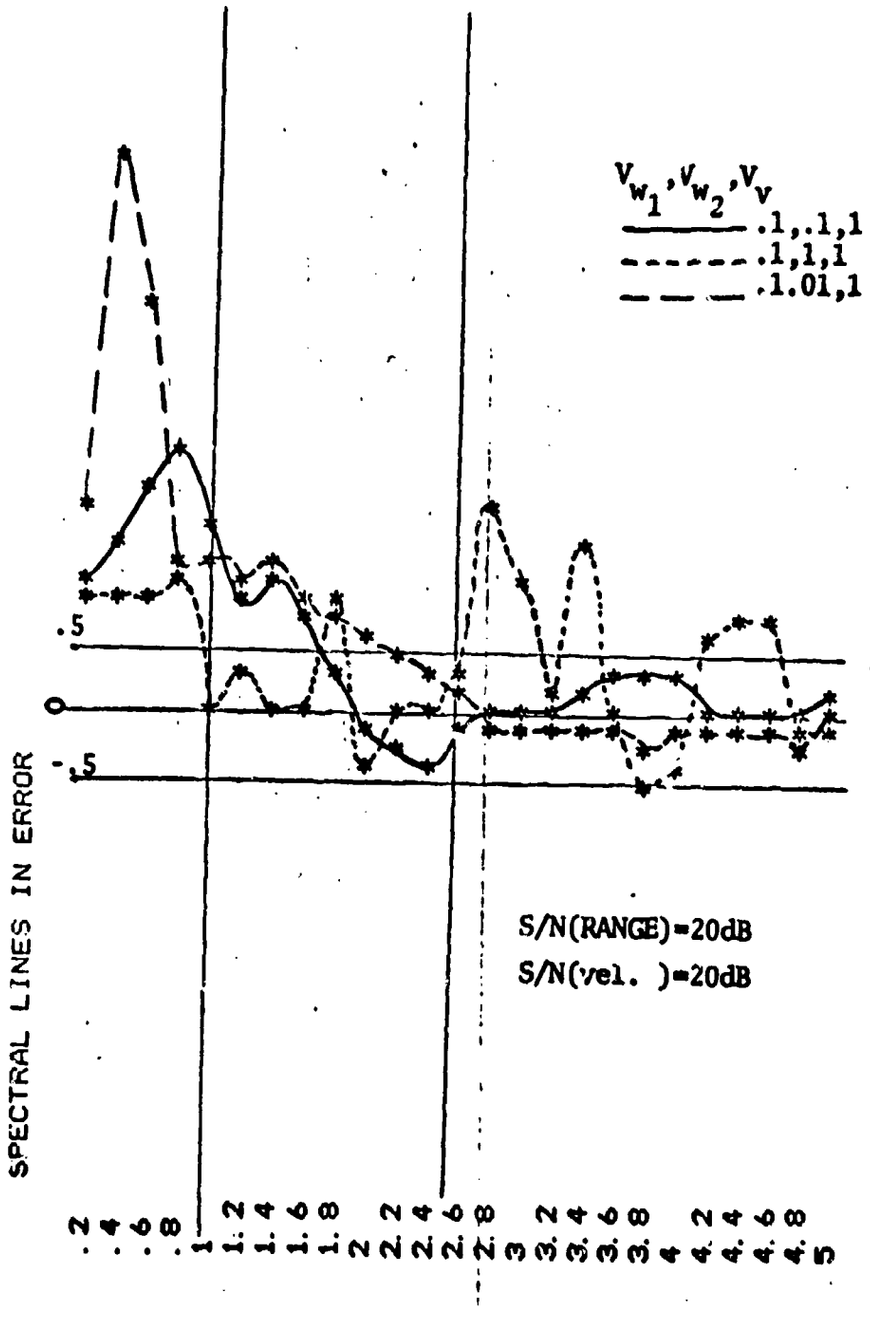
FIGURES



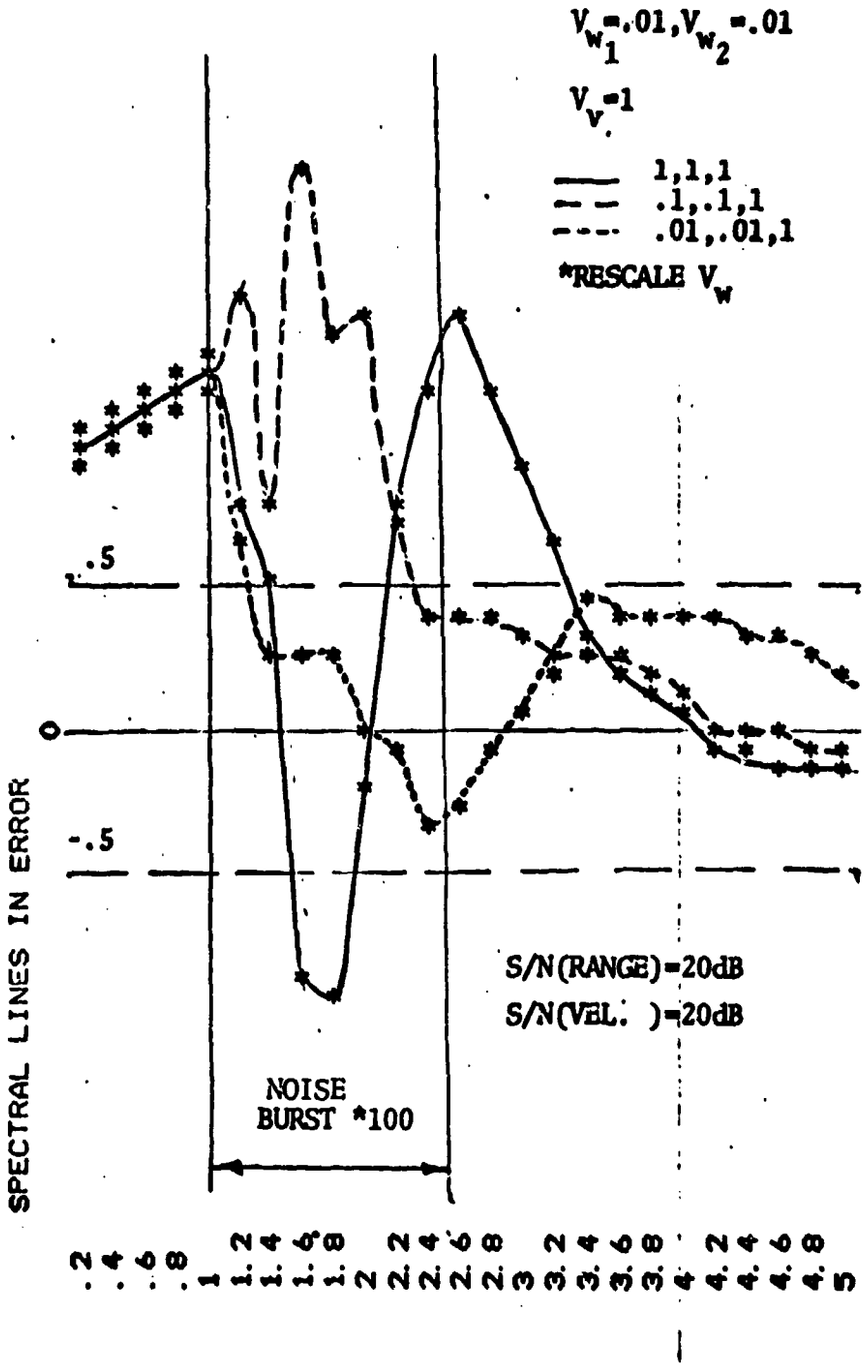
KALMAN FILTER



KALMAN FILTER

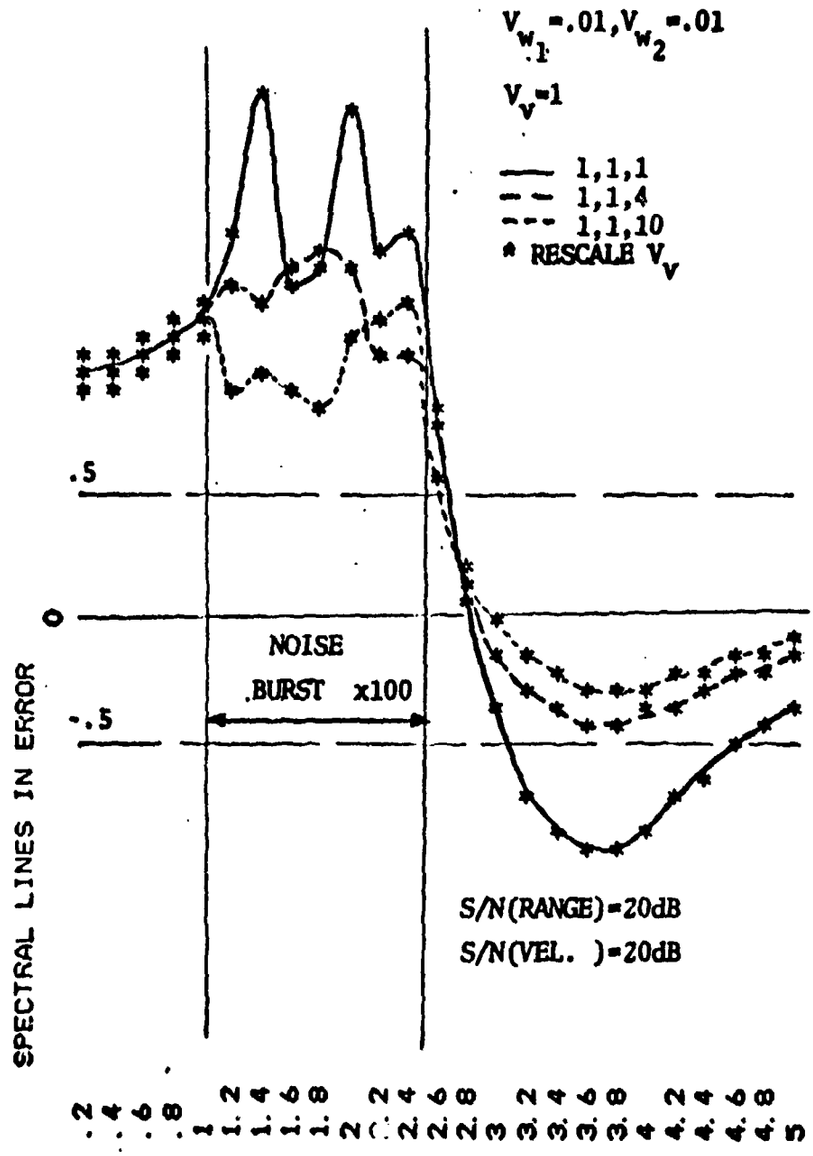


KALMAN FILTER

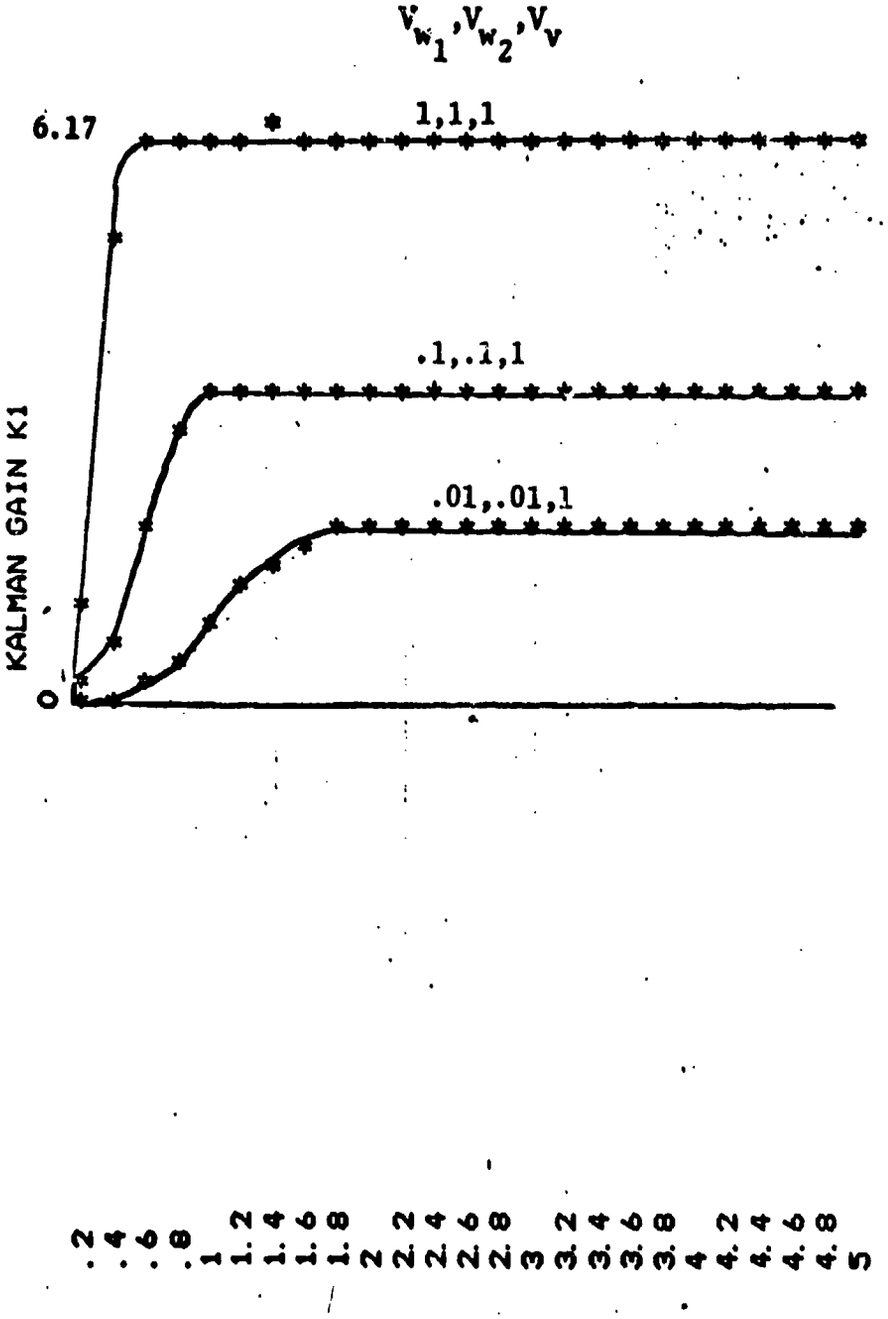


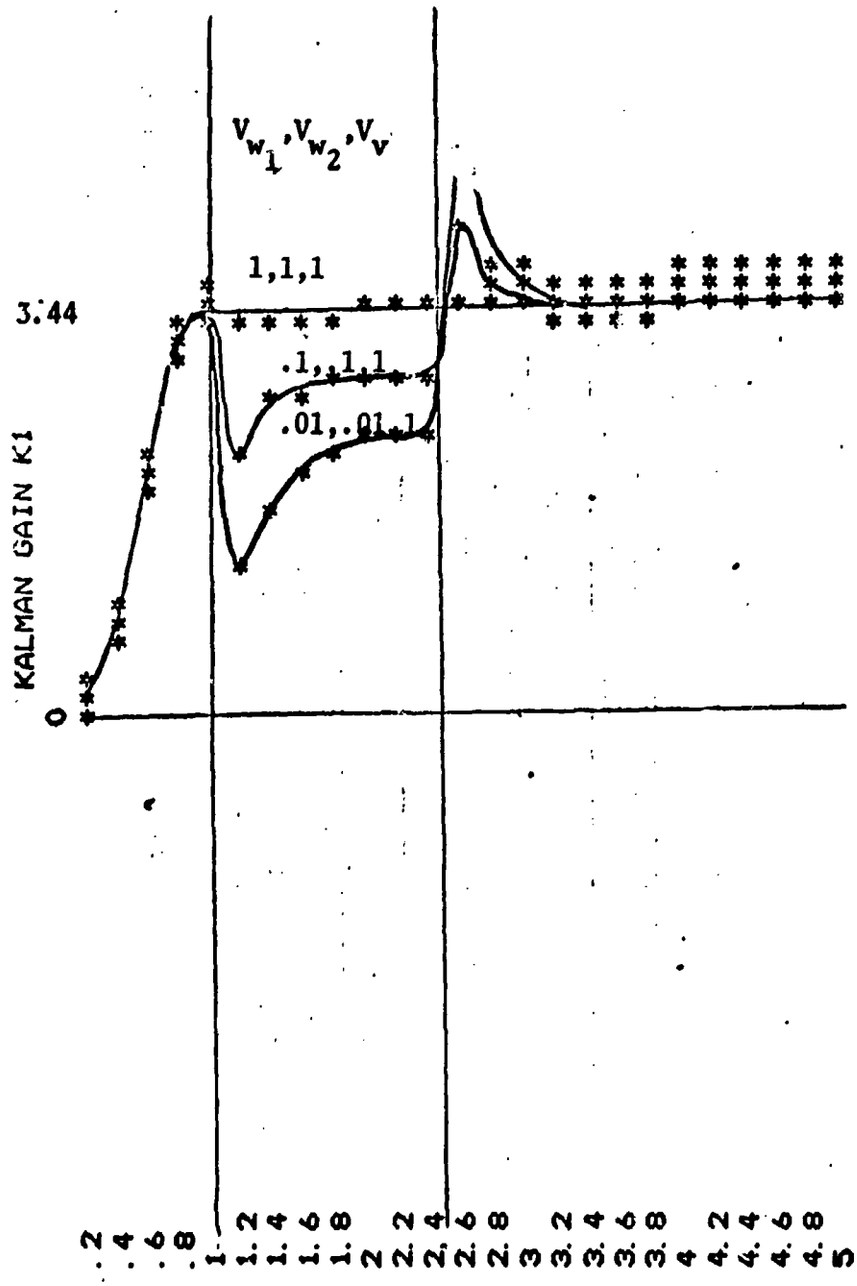
ADAPTIVE KALMAN FILTER

(INPUT COVARIANCE)

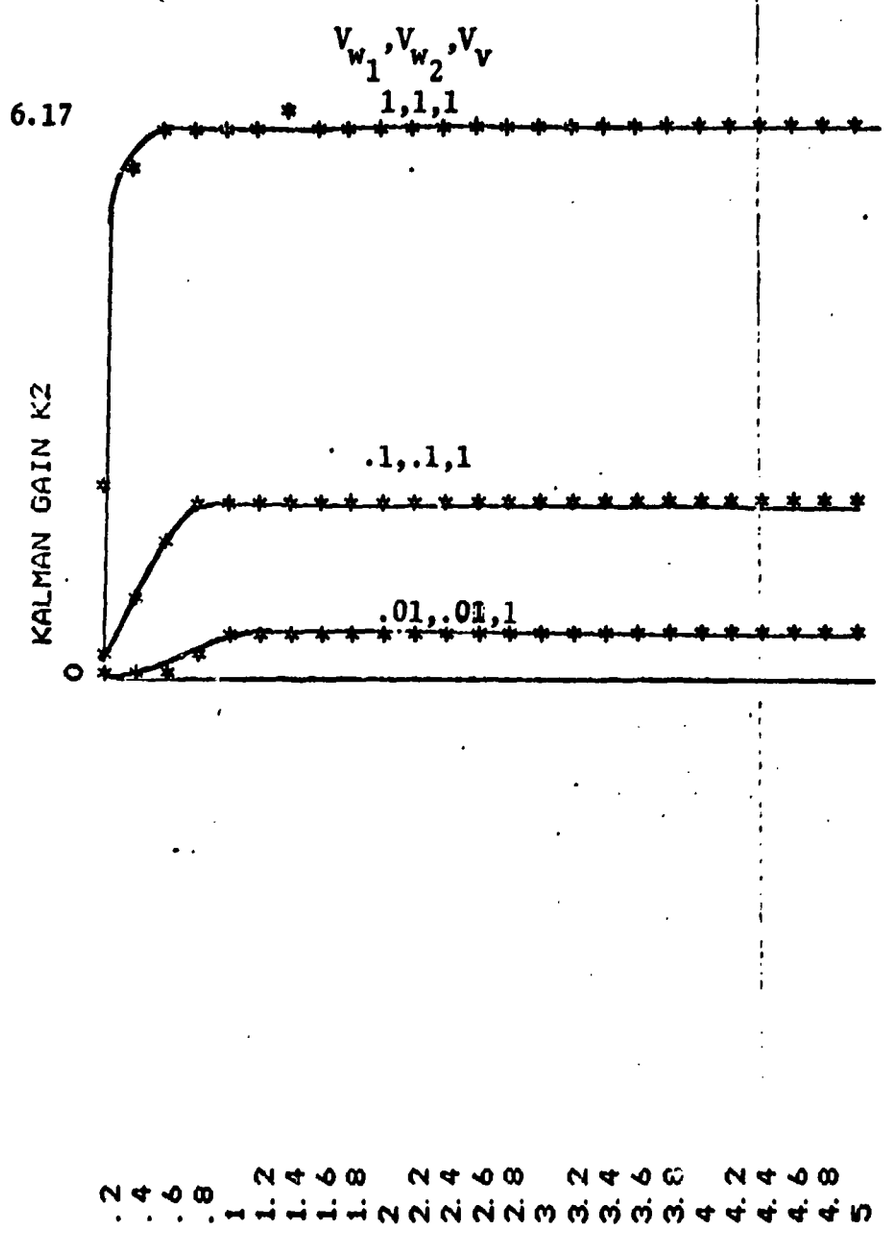


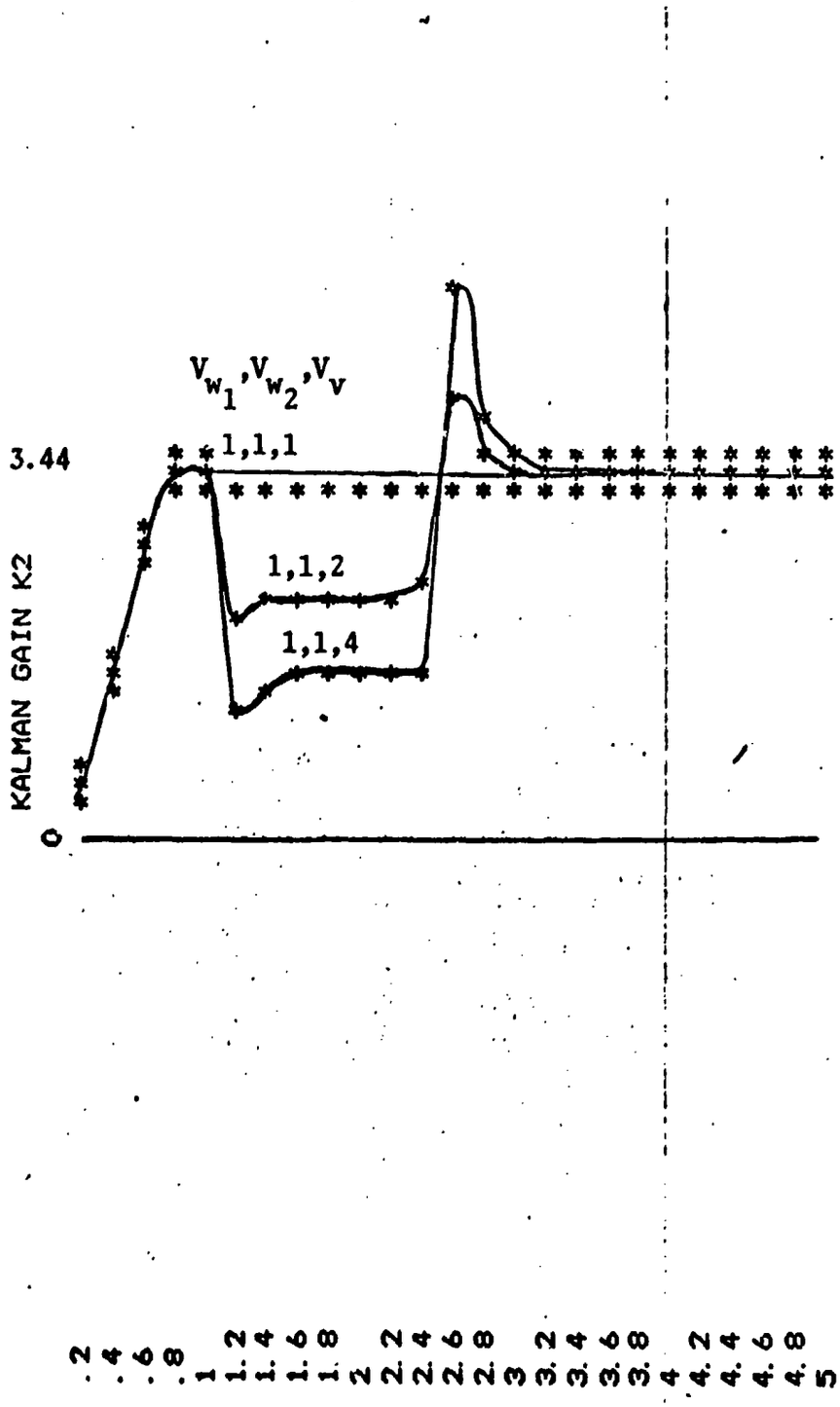
ADAPTIVE KALMAN FILTER
(OUTPUT COVARIANCE)

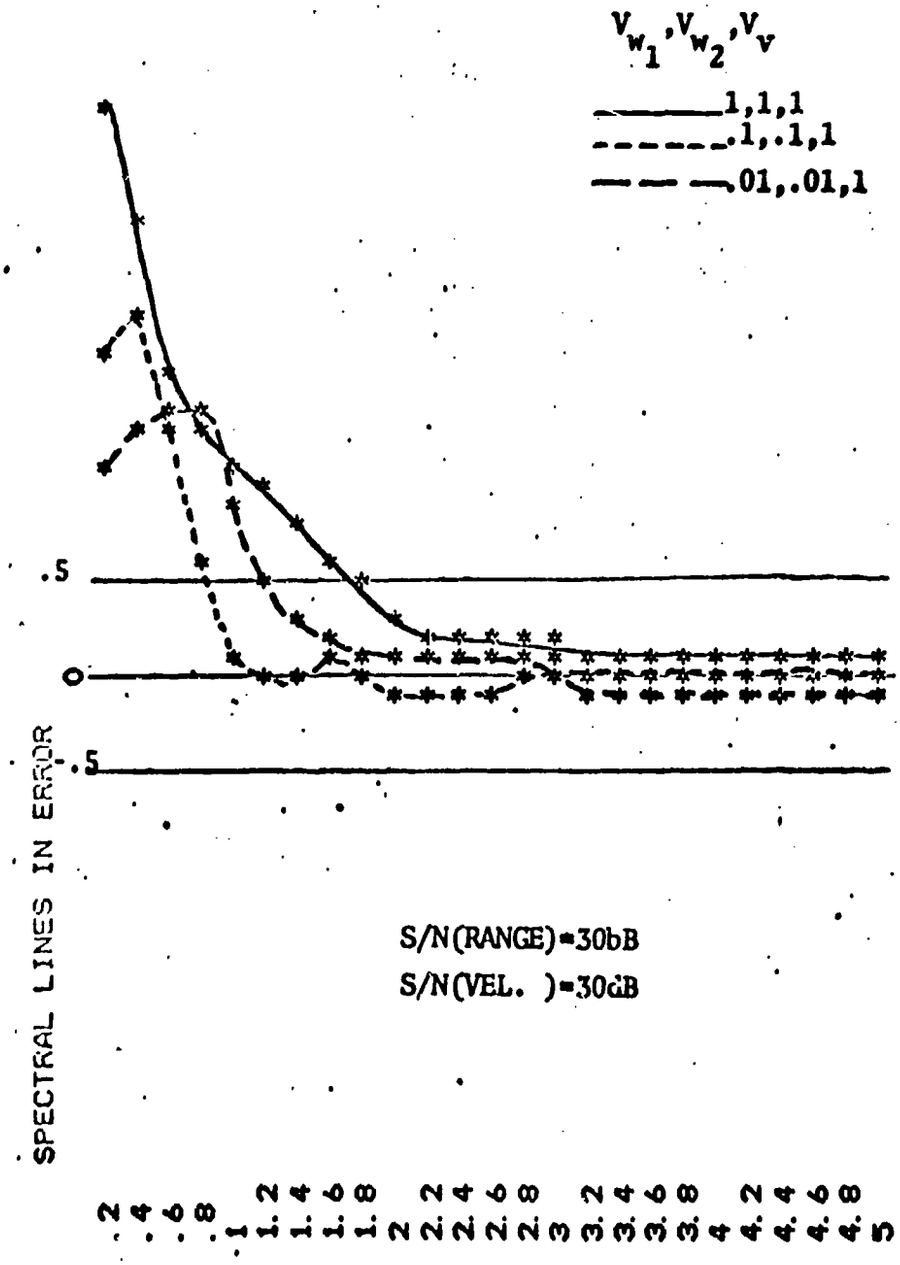

V_{w_1}, V_{w_2}, V_v
6.17
1,1,1
.1,.1,1
.01,.01,1
KALMAN GAIN K1
0
.2 .4 .6 .8 1 1.2 1.4 1.6 1.8 2 2.2 2.4 2.6 2.8 3 3.2 3.4 3.6 3.8 4 4.2 4.4 4.6 4.8 5


V_{w_1}, V_{w_2}, V_v
1,1,1
3.44
.1,.1,1
.01,.01,1
KALMAN GAIN K1
0
.2
.4
.6
.8
1.
1.2
1.4
1.6
1.8
2.
2.2
2.4
2.6
2.8
3.
3.2
3.4
3.6
3.8
4.
4.2
4.4
4.6
4.8
5.

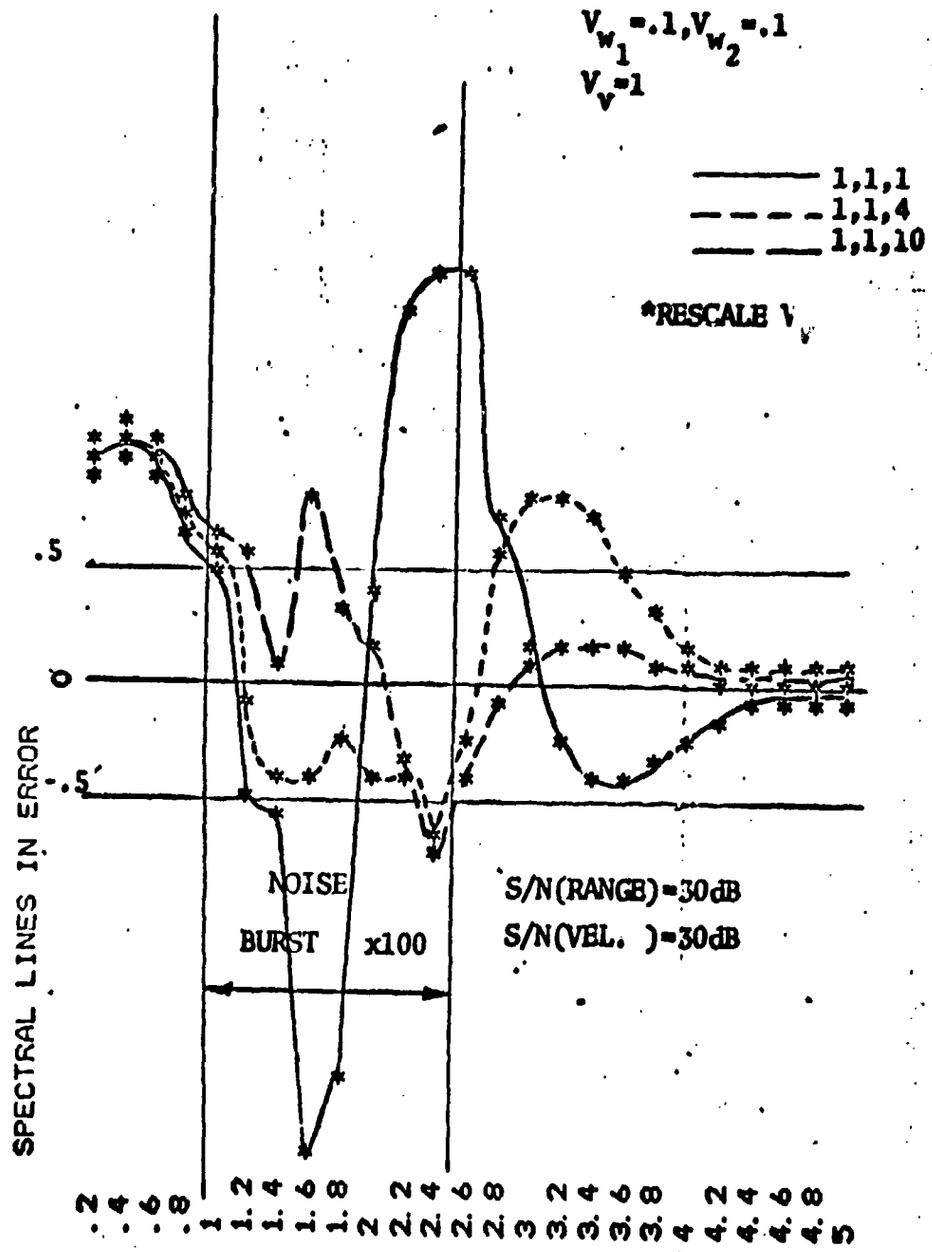
K1-2


V_{w_1}, V_{w_2}, V_v
6.17
1,1,1
.1,.1,1
.01,.01,1
KALMAN GAIN K2
0
.2 .4 .6 .8 1 1.2 1.4 1.6 1.8 2 2.2 2.4 2.6 2.8 3 3.2 3.4 3.6 3.8 4 4.2 4.4 4.6 4.8 5


V_{w_1}, V_{w_2}, V_v
3.44
1,1,1
1,1,2
1,1,4
KALMAN GAIN K2
0
.2
.4
.6
.8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6
3.8
4
4.2
4.4
4.6
4.8
5



STEADY STATE KALMAN FILTER



ADAPTIVE STEADY STATE KALMAN FILTER
(OUTPUT COVARIANCE)

APPENDICIES

APPENDIX A

Pulse Doppler Ambiguity Resolution

FRED J. TAYLOR
The University of Texas at El Paso
El Paso, Tex. 79968

Abstract

A mathematical algorithm using the method of invariant imbedding is developed which generates a best $\mathcal{L}^2$ estimate of range and range rate from pulse Doppler data.

A pulse Doppler system has the ability to track moving targets in relatively stationary clutter in the presence of high energy noise. The spectral representation of a pulse train with a given PRF and pulsewidth τ is given by the $\sin(X)/X$ function whose first zero crossings occurs at $1/\tau$ and the spacial frequency between adjacent spectral lines is 1/PRF. Thus at band (5650 MHz $\triangleq f_0$) a $\Delta f = 1$ Hz implies a resolution of 0.029 yd/s and a PRF = 640 yields a spectral spacing of 18.56 yd/s between lines. If a pulsewidth of 1μs is considered, then there exists over 1500 spectral lines in the interval $[f_0, f_0 + 1/\tau]$. Thus a pulse Doppler system is cursed by an abundance of ambiguous spectral data about any arbitrary spectral line.

One ambiguity resolution technique that has been implemented uses the method of invariant imbedding [1]. The algorithm developed was a direct adaptation of a set of notes published by Bellman and Kalaba. Herein, the optimal estimate was one that gave the "best" $\mathcal{L}^2$ fit to the observed range data. Since range rate is often considered to be a more accurate data source than range data, any optimal estimate should be optimally fitted to range-rate data also.

Consider then the following problem in terms of the following state variables:

$x_1(t)$ = actual range
$x_2(t)$ = number of spectral lines (real number)
$x_1^0(t)$ = observed range.

Observation dynamics:

$$\text{range } x_1^0(t) = x_1(t) - \epsilon_1(t), \quad \epsilon_1(t) = \text{measurement error} \tag{1}$$

$$\text{range rate } \dot{x}_1^0(t) = \dot{x}_1(t) - x_2(t)\delta - \epsilon_2(t), \quad \epsilon_2(t) = \text{measurement error} \tag{2}$$

where $\delta = 18.56$ if PRF = 640.

If an estimate x_2 (number of spectral lines in error from a coarse track spectral line) is assumed to be constant over an observation interval, then require

$$\dot{x}_2 = 0. \tag{3}$$

If a time-varying estimate of $x_2(t)$ is desired, then $x_2(t)$ may be approximated by a power series in t whose coefficients are chosen optimally. However, the additional dimensionality requirements imposed on the solution process makes this approach unattractive in general.

Defining

$$\hat{x} = [\hat{x}_1, \hat{x}_2]^T \text{ (estimation vector)} \tag{4}$$

$$\hat{\epsilon} = [\hat{\epsilon}_1, \hat{\epsilon}_2]^T \text{ (estimated error vector)}, \tag{5}$$

then

$$\hat{x}_1 = x_1^0 + \hat{\epsilon}_1 \text{ (range estimate)} \tag{6}$$

Manuscript received February 28, 1972.

This work was supported in part under U.R.I. Grant 083-50-790-04.

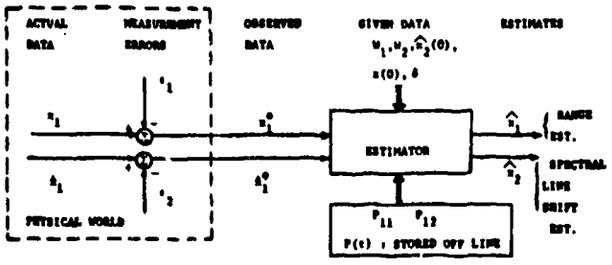


Fig. 1. System model.

and

$$\dot{\hat{x}} = \begin{bmatrix} \dot{\hat{x}}_1 \\ \dot{\hat{x}}_2 \end{bmatrix} = \begin{bmatrix} \dot{x}_1{}^0 + \hat{x}_2\delta - \hat{\epsilon}_2 \\ 0 \end{bmatrix} \begin{matrix} \text{(range-rate estimate)} \\ \text{(spectral error estimate).} \end{matrix} \tag{7}$$

Let it be required that an objective function J be minimized:

$$J = \frac{1}{2}\int_0^T \|\hat{\epsilon}\|_W{}^2\, dt, \quad W > 0, \text{ diagonal.}$$

The necessary conditions for producing an optimal estimate $\hat{x}(t)$ are given by the Maximum Principle [2], and they define the following two-point boundary value problem.

State equations:

$$\dot{\hat{x}} = f(x, \lambda) = \begin{bmatrix} \dot{x}_1{}^0 + \hat{x}_2\delta - \lambda/W^2 \\ 0 \end{bmatrix}. \tag{8}$$

Costate equation:

$$\lambda = g(x, \lambda) = \begin{bmatrix} W_1(x_1{}^0 - \hat{x}_1) \\ -\lambda_1\delta \end{bmatrix}, \quad \begin{matrix} \lambda(0) = 0 \\ \lambda(T) = 0. \end{matrix} \tag{9}$$

One technique used to solve two-point boundary value problems is the method of invariant imbedding [3]. The imbedding equation is known to be

$$\frac{\partial r(C, T)}{\partial T} + \frac{\partial r(C, T)}{\partial C}\, g(r(C, T), C) = f(r(C, T), C) \tag{10}$$

where $\lambda(T) = C$ (arbitrary class of functions) and r has the assumed form $r(C, T) = \hat{x}(T) + P(T)C$; $P = P^T$. Substituting r into (10) yields the *fixed-point* boundary value problem (thus bypassing the difficult two-point boundary value problem), in terms of z (estimated range error):

$$\dot{z} \triangleq (\dot{\hat{x}}_1 - \dot{x}^0) = P_{11}W_1(\hat{x}_1 - x_1{}^0) + \hat{x}_2\delta = P_{11}W_1 z + \hat{x}_2\delta \tag{11}$$

$$\dot{\hat{x}}_2 = P_{21}W_1(\hat{x}_1 - x_1{}^0) = P_{21}W_1 z, \quad z(0), \hat{x}_2(0) \text{ arbitrary} \tag{12}$$

and

$$\dot{P} = \left[\begin{array}{c|c} \dot{P}_{11} = 2P_{21}\delta + P_{11}{}^2W_1 - 1/W_2 & \dot{P}_{12} = P_{22}\delta - P_{11}P_{12}W_1 \\ \hline \dot{P}_{21} = \dot{P}_{12} & \dot{P}_{22} = P_{12}{}^2W_1 \end{array}\right], \quad \begin{matrix} P(0) \\ \text{arbitrary.} \end{matrix} \tag{13}$$

Equation (13) is solved off-line and stored to facilitate a real-time estimation $\hat{x}$ (see Fig. 1). Since all the initial conditions are arbitrary it is assumed that they would be determined experimentally under actual mission conditions. Finally, it is known from linear estimation theory that the optimal choice of W is the inverse of the measurement error covariance matrix which should be used, when available, to admit a minimal variance estimate of x.

Experimental Results

Given are the following observations:

$$x_1{}^0(t) = x_1(t) - \epsilon_1(t)$$

$$\dot{x}_1{}^0(t) = \dot{x}_1(t) - x_2(t)\delta - \epsilon_2(t)$$

where

$\delta = 18.56$, PRF = 640, $t \in [0, 5]$, and $T = 5$ seconds.

The noise sources $\epsilon_1(t)$ and $\epsilon_2(t)$ are assumed to be independent and normally distributed, $N_1(0, \sigma_1{}^2)$ and $N_2(0, \sigma_2{}^2)$, respectively. An initial spectral velocity error of 5 lines is assumed (i.e., $x_2(0) = 5 \rightarrow 92.8$ yd/s error). The actual velocity $\dot{x}_1(t)$ is to be 5000 yd/s for all time, and the actual range is described by $x_1(t) = x_1(0) + \dot{x}_1(t)t \triangleq 10\,000 + 5000t$ yards. The measurement noise error variances are to be (0, 0) (no noise case), (10^2, 10^2), and (10^4, 10^4), respectively. Thus the ideal estimation vector $\hat{x}$, if all measurement error could be suppressed, would be $\hat{x}_1(t) =$

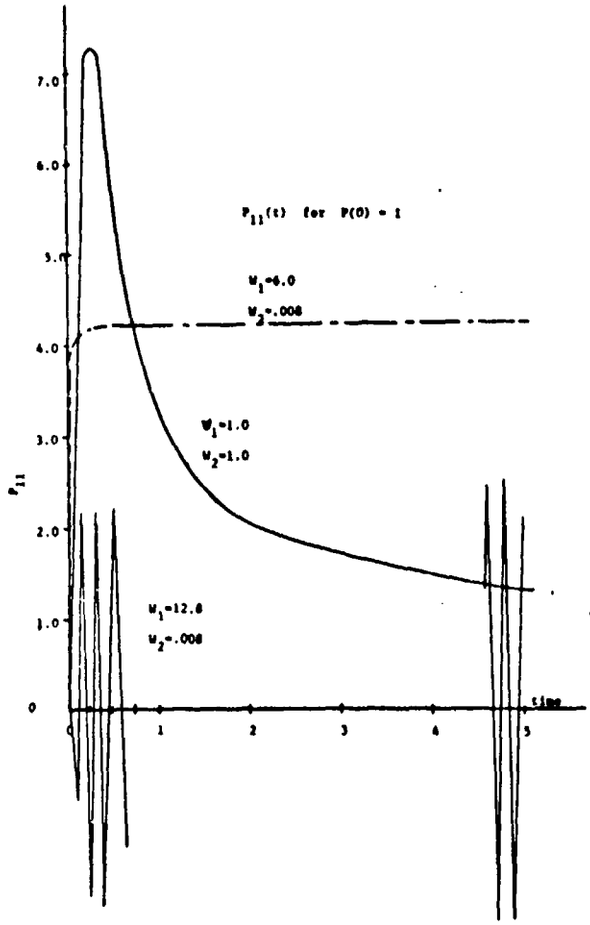


Fig. 2. Optimal gain P_{11}.

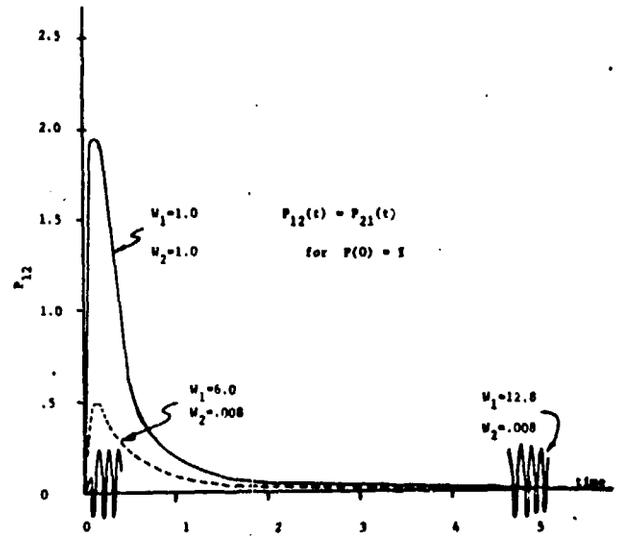
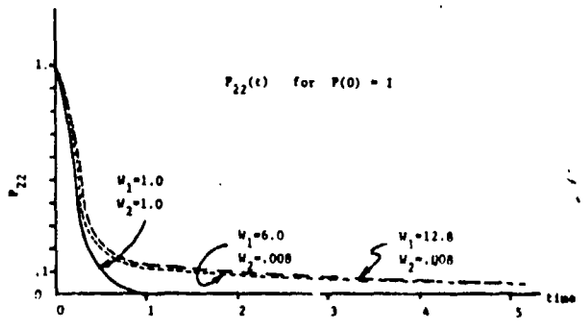


Fig. 3. Optimal gain P_{12}.

Fig. 4. Optimal gain P_{22}.



$x_1{}^0(t)$ and $\hat{x}_2(t) = 0$. The choice of W will determine the relative importance associated with minimizing ϵ_1 (range error) and ϵ_2 (range-rate error) which are W_1 and W_2 dependent, respectively. Thus a range-dependent estimate would imply $W_1 \gg W_2$, whereas the proposed optimal estimate requires minimizing both errors with respect to the measurement covariance matrix, or in this embodiment, it will be assumed that $W_1 = W_2$, since $\sigma_1{}^2 = \sigma_2{}^2$, is optimal.

It can be noted from Figs. 2, 3, and 4 that the optimal filter matrix $P(t)$ is sensitive to the choice of W. This

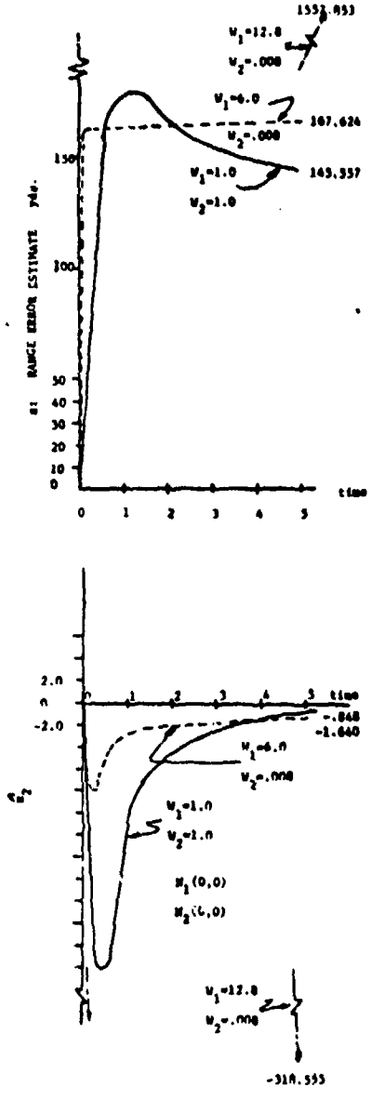


Fig. 5. Optimal estimates as a function of W and noise sources N_1 and N_2, [$N_1(0, 0)$ and $N_2(0, 0)$].

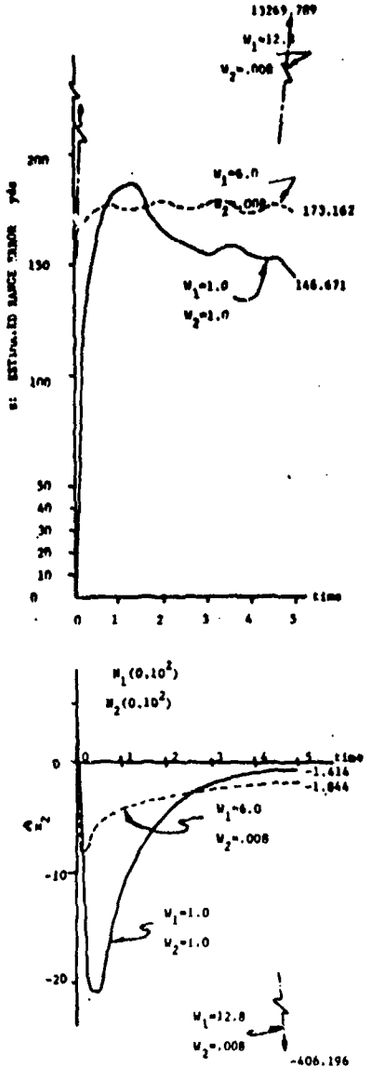


Fig. 6. Optimal estimates as a function of W and noise sources N_1 and N_2 [$N_1(0, 10^2)$ and $N_2(0, 10^2)$].

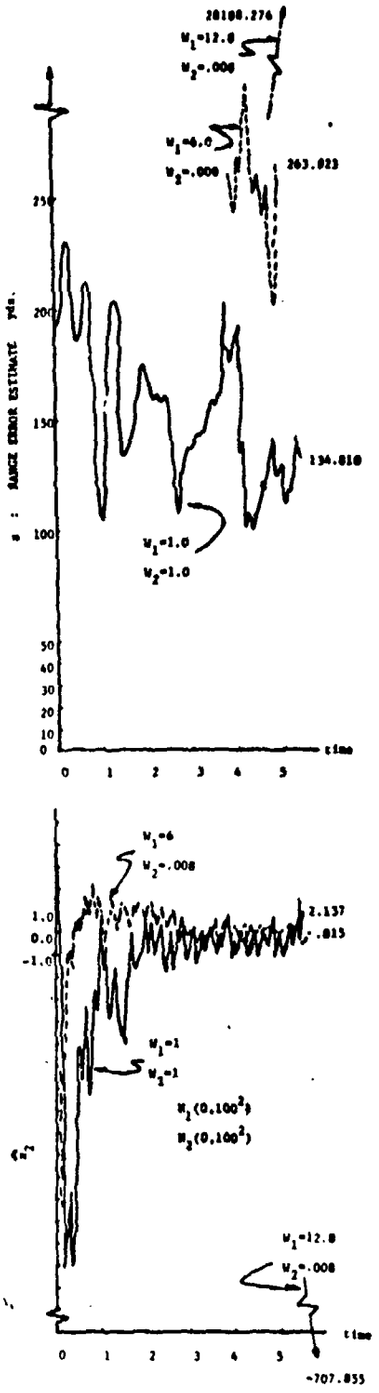


Fig. 7. Optimal estimates as a function of W and noise sources N_1 and N_2 [$N_1(0, 100^2)$ and $N_2(0, 100^2)$].

sensitivity is reflected in the estimator's unstable behavior as W_1 becomes much greater than W_2. As indicated in Figs. 5, 6, and 7, the estimation tends towards the ideal estimate $z(t)$ [i.e., $\hat{x}(t) - x^0(t)$] $= 0$ and $\hat{x}_2 = 0$ as the noise is reduced and as $W_1 \to W_2 \to 1$. It was also noted that the estimate $\hat{x}$ improves as $\hat{x}_2(0) \to 0$. In all cases, when $W_1 = W_2 = 1$, the spectral error indicator $\hat{x}_2$ possessed very acceptable values and was superior to those found when $W_1 = 12.8$ and $W_2 = 0.008$. Similarly, the best range error performance occurred when $W_1 = W_2$, which agrees with

50a

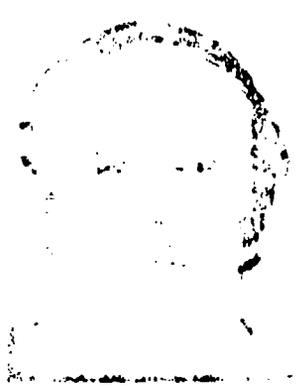
the original hypotheses that an estimate based on both range and range-rate data weighted by an inverse convariance relationship, will be superior to a weighted range estimate only. A highly accurate range-rate estimate, in terms of resolving pulse Doppler ambiguous data, results. The achieved range estimate may be used to augment more classical pulse delay ranging methods.

References

[1] Palye, "An application of invariant imbedding smoothing to real time pulse Doppler ambiguity resolution," RCA Rept., Moorestown, N.J.

[2] Athans and Falb, *Optimal Control*. New York: McGraw-Hill, 1966.

[3] Sage, *Optimums Systems Control*. Englewood Cliffs, N.J.: Prentice-Hall, 1968.



Fred J. Taylor was born in Wisconsin Rapids, Wisc., on April 28, 1940. He received the B.S.E.E. degree from the Milwaukee School of Engineering, Milwaukee, Wisc., in 1965, and the M.S.E.E. and Ph.D. degrees from the University of Colorado, in 1966 and 1969, respectively.

He was a member of the technical staff of Texas Instruments Inc. from 1969 to 1970, and was awarded several patents during this time. He was a Visiting Industrial Professor at Southern Methodist University in 1970. Since 1970 he has been with the University of Texas at El Paso.

APPENDIX B

AMBIGUITY RESOLUTION EXPERIMENT

EXPERIMENT NO.	INITIAL COVARIANCE		SIGNAL/ NOISE dB		TARGET			COVARIANCE			5 SECOND TEST XD/SEC		SCINTILLATION COMMENT
	$V_{x_1}(0)$	$V_{x_2}(0)$	R	$\dot{R}$	R_o x10^3	$\dot{R}_o$ x10^2	$\ddot{R}_o$ x10	V_{w_1}	V_{w_2}	V_v	RCA	KALMAN	
1	Ø	0	25	25	100	-10	0	1	1	1	4.203	.1910	
2	1	1	25	25	100	-10	0	1	1	1	4.203	.1916	*
3	Ø	0	25	25	100	-10	0	5	5	1	4.203	-.1419	
4	Ø	0	25	25	100	-10	0	10	10	1	4.203	-.0520	
5	0	0	25	25	100	-10	0	25	25	1	4.203	-.3546	
6	0	0	25	25	100	-10	0	10	10	10	4.203	.1910	
7	0	0	25	25	100	-10	0	10	10	1	4.203	1.5707	
8	0	0	25	25	100	-10	0	10	10	1	-1.770	-.0520	*i*
9	0	0	25	25	100	-10	0	10	10	1	- .1707	-.0520	*ii*
10	0	0	10	10	1	-10	20	1	1	1	10.8407	-8.2778	
11	1	1	10	10	1	-10	20	1	1	1	10.8407	-8.2274	*
12	0	0	20	20	1	-10	20	1	1	1	3.472	-.0817	
13	0	0	20	20	1	-10	20	.5	.5	1	3.472	-.1246	**
14	0	0	20	20	1	-10	20	2	2	1	3.472	-.0238	***
15	0	0	20	20	1	-10	100	1	1	1	3.454	-.0599	
16	0	0	20	20	1	-10	100	1	1	1	5.412	-.0483	*iii*
17	0	0	20	20	1	-10	100	1	1	1	16.2347	.15618	*iv*

SCINTILLATION CODE

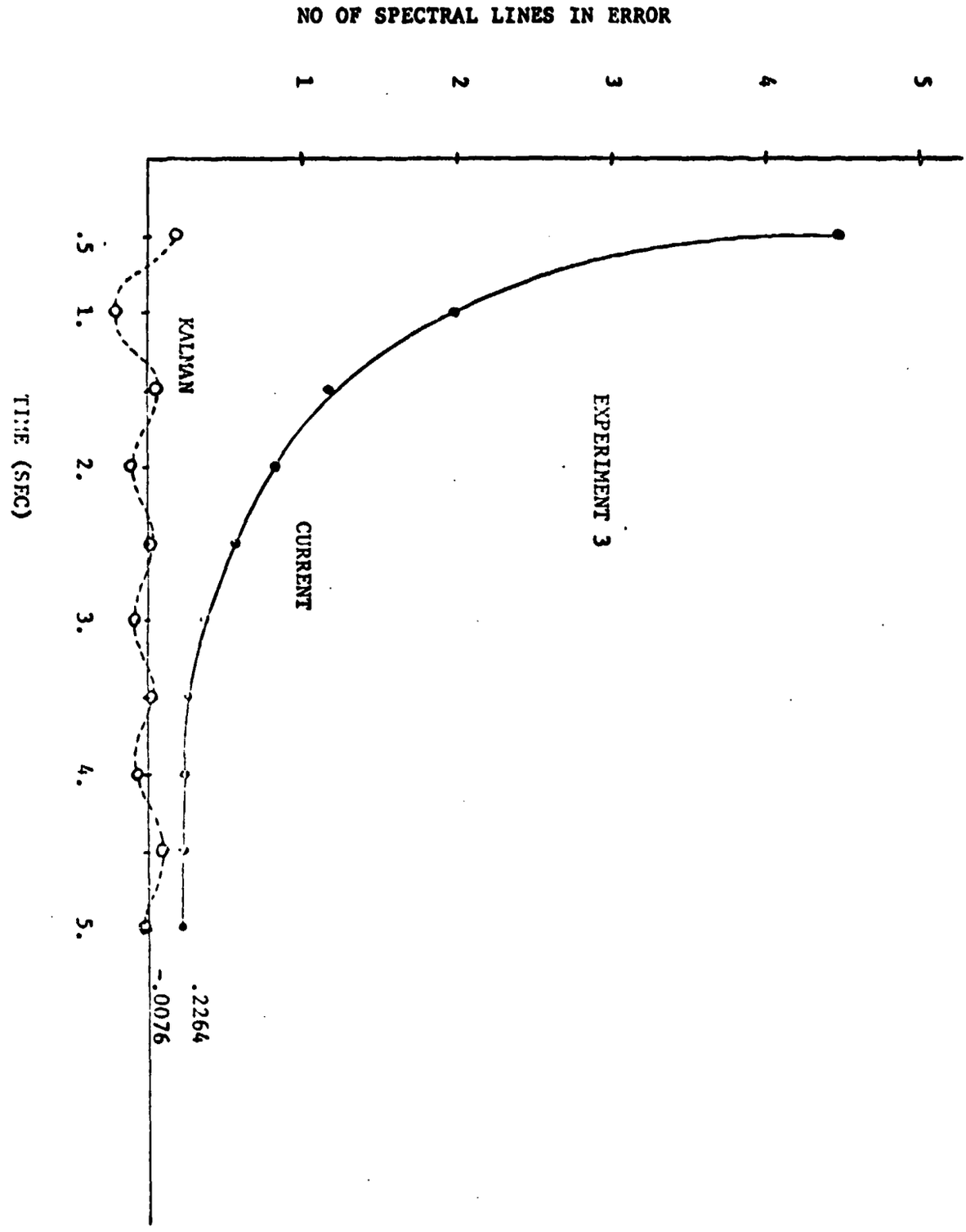
CODE	NOISE BURST (RANGE)	TIME SEC.	NOISE BURST (RDOT	TIME SEC.
i	*10	.5-1	*10	1-2
ii	*25	1-2	*25	1-2
iii	*10	2-3	*20	2-3
iv	*50	2-3	*50	2-3

R observed = R actual *(1 + N(0, K * σ^2))
RDOT observed = RDOT actual * (1 + N(0, K * σ^2))

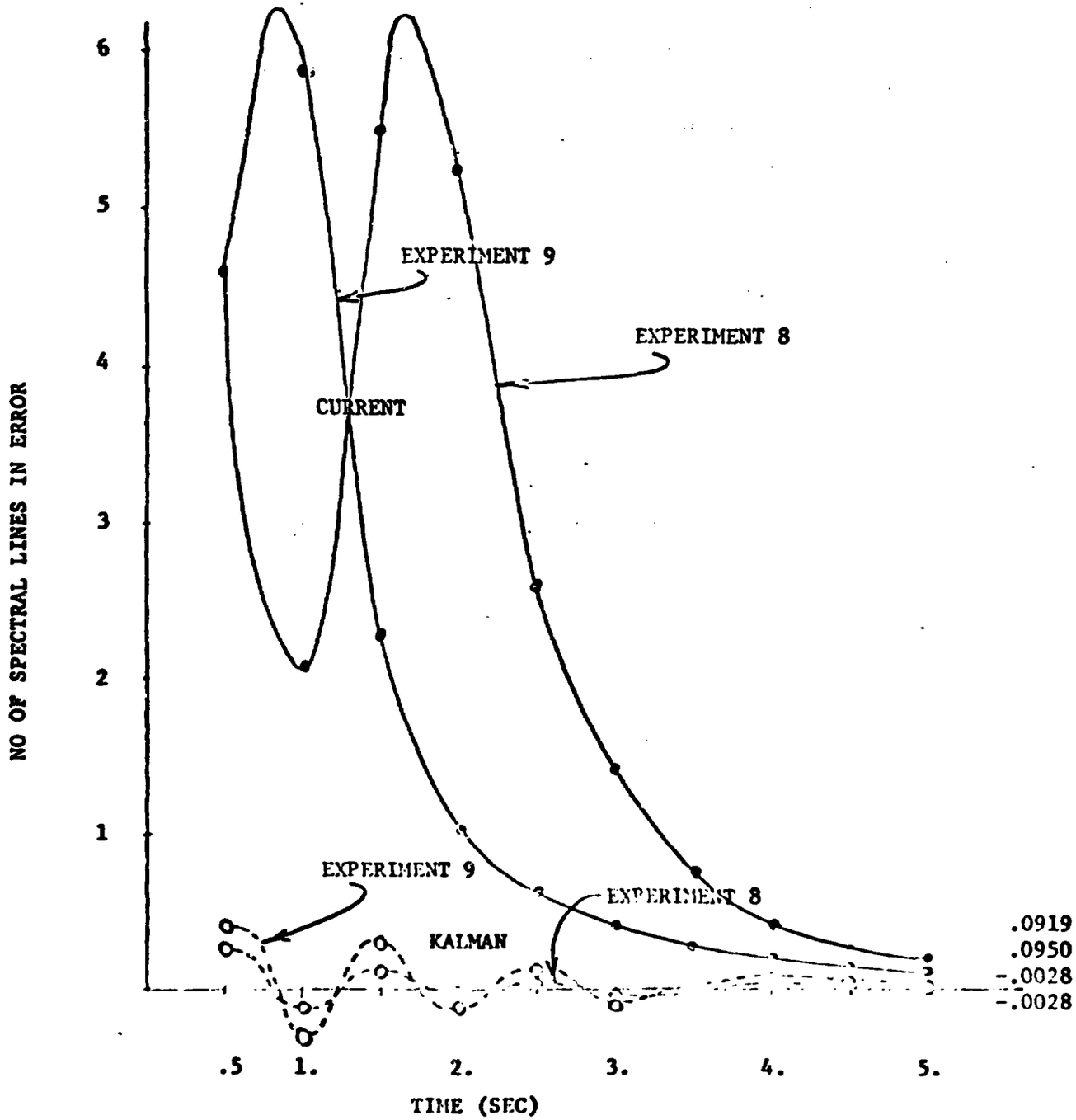
N implies normally distributed random variable, mean zero, variance σ^2 times any burst factor.

Comments:

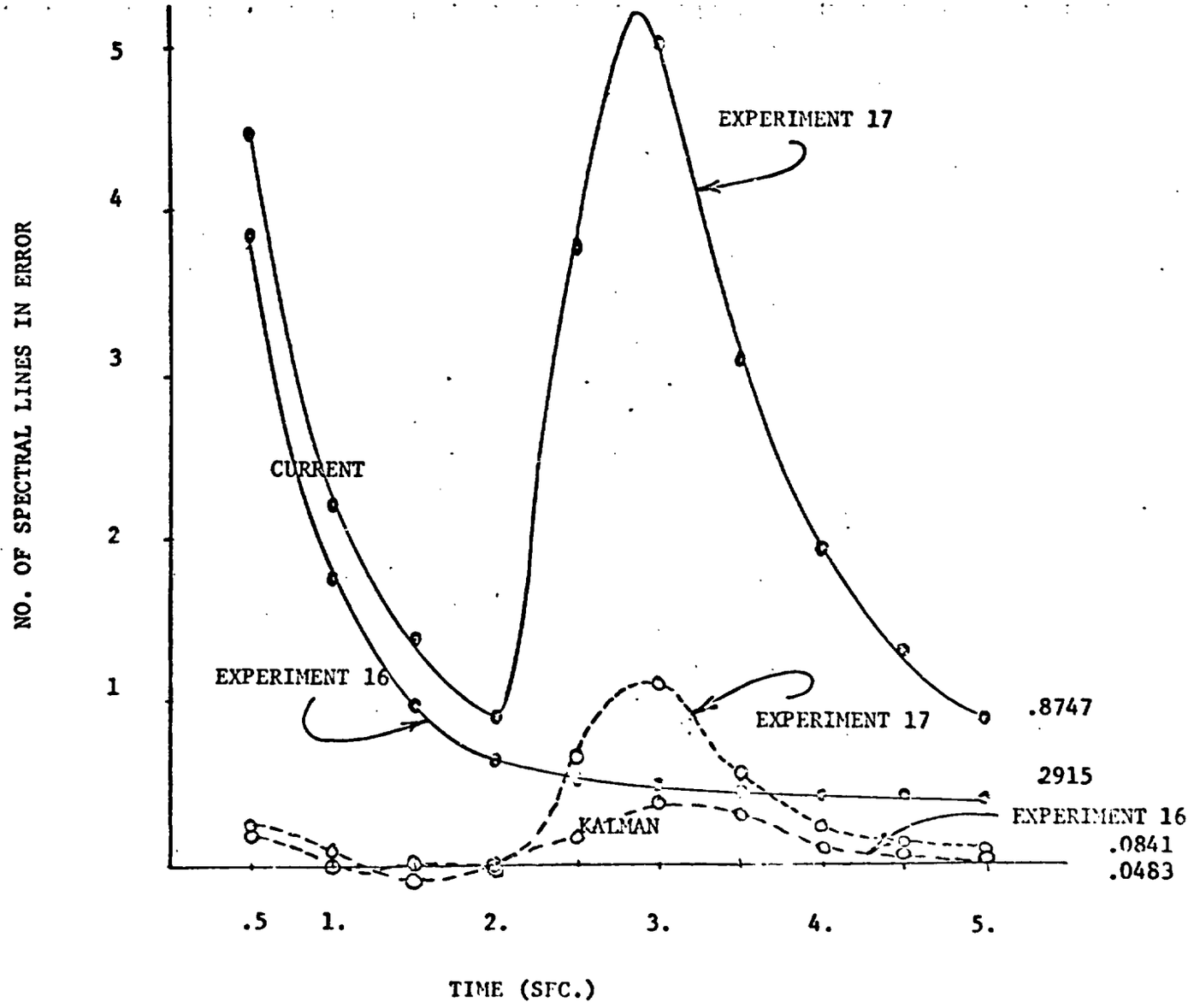
*, ** and *** Algorithm has been determined insensitive to initial covariances.
Algorithm has been shown to be sensitive to apriori covariance assumptions.


NO OF SPECTRAL LINES IN ERROR
1
2
3
4
5
EXPERIMENT 3
CURRENT
KALMAN
.5
1.
2.
3.
4.
5.
TIME (SEC)
.2264
-.0076

APPENDIX B


NO OF SPECTRAL LINES IN ERROR
6
5
4
3
2
1
EXPERIMENT 9
EXPERIMENT 8
CURRENT
EXPERIMENT 9
EXPERIMENT 8
KALMAN
.0919
.0950
-.0028
-.0028
.5
1.
2.
3.
4.
5.
TIME (SEC)

APPENDIX B


NO. OF SPECTRAL LINES IN ERROR
5
4
3
2
1
EXPERIMENT 17
CURRENT
EXPERIMENT 16
EXPERIMENT 17
KALMAN
.8747
2915
EXPERIMENT 16
.0841
.0483
.5
1.
2.
3.
4.
5.
TIME (SEC.)

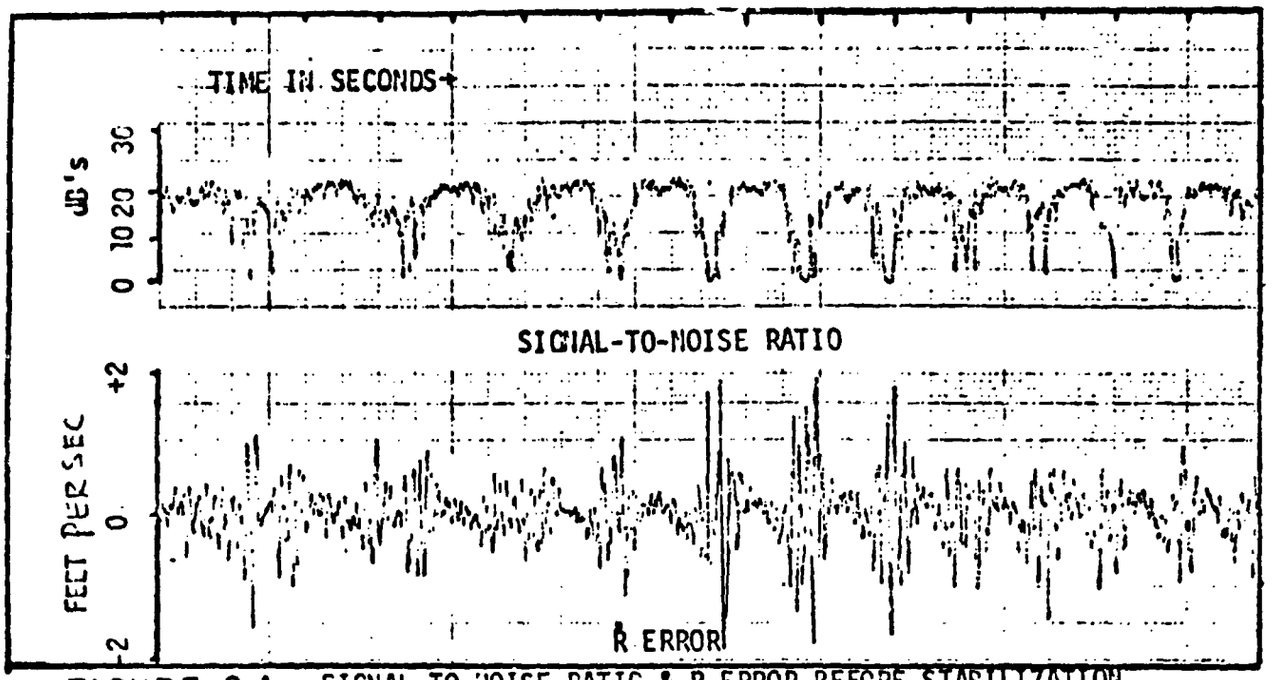


FIGURE 2 A - SIGNAL-TO-NOISE RATIO & R ERROR BEFORE STABILIZATION

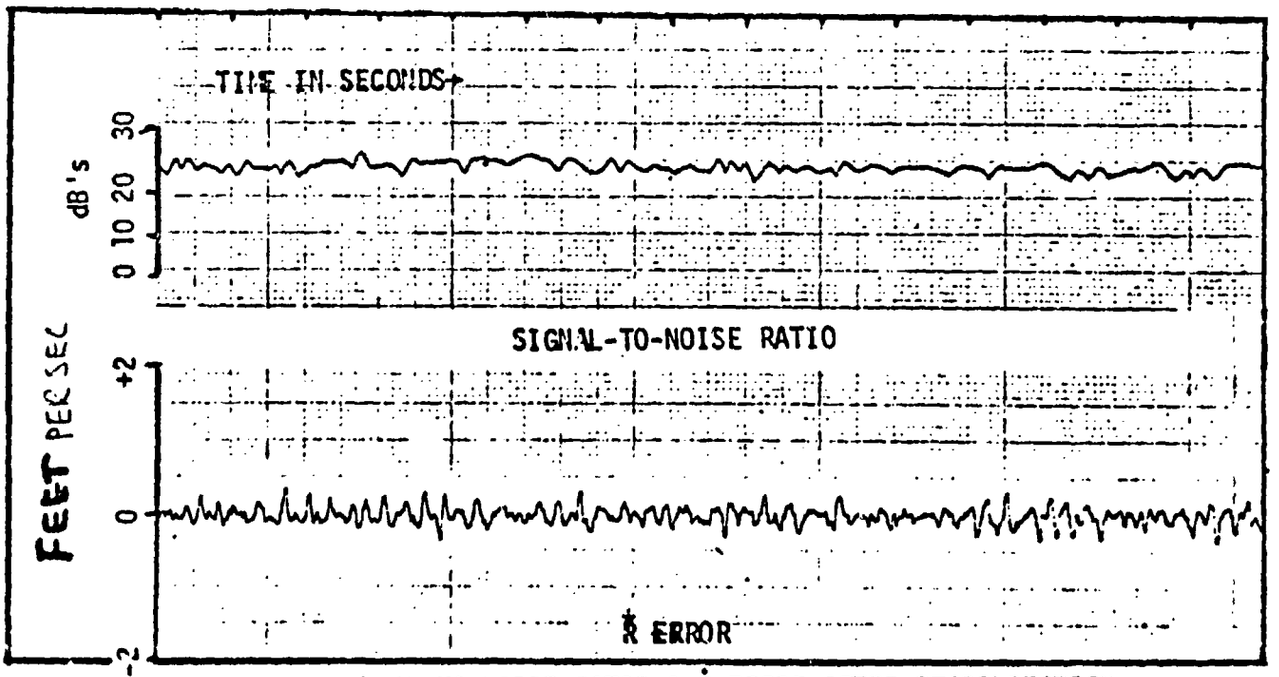


FIGURE 2 B - SIGNAL-TO-NOISE RATIO & R ERROR AFTER STABILIZATION

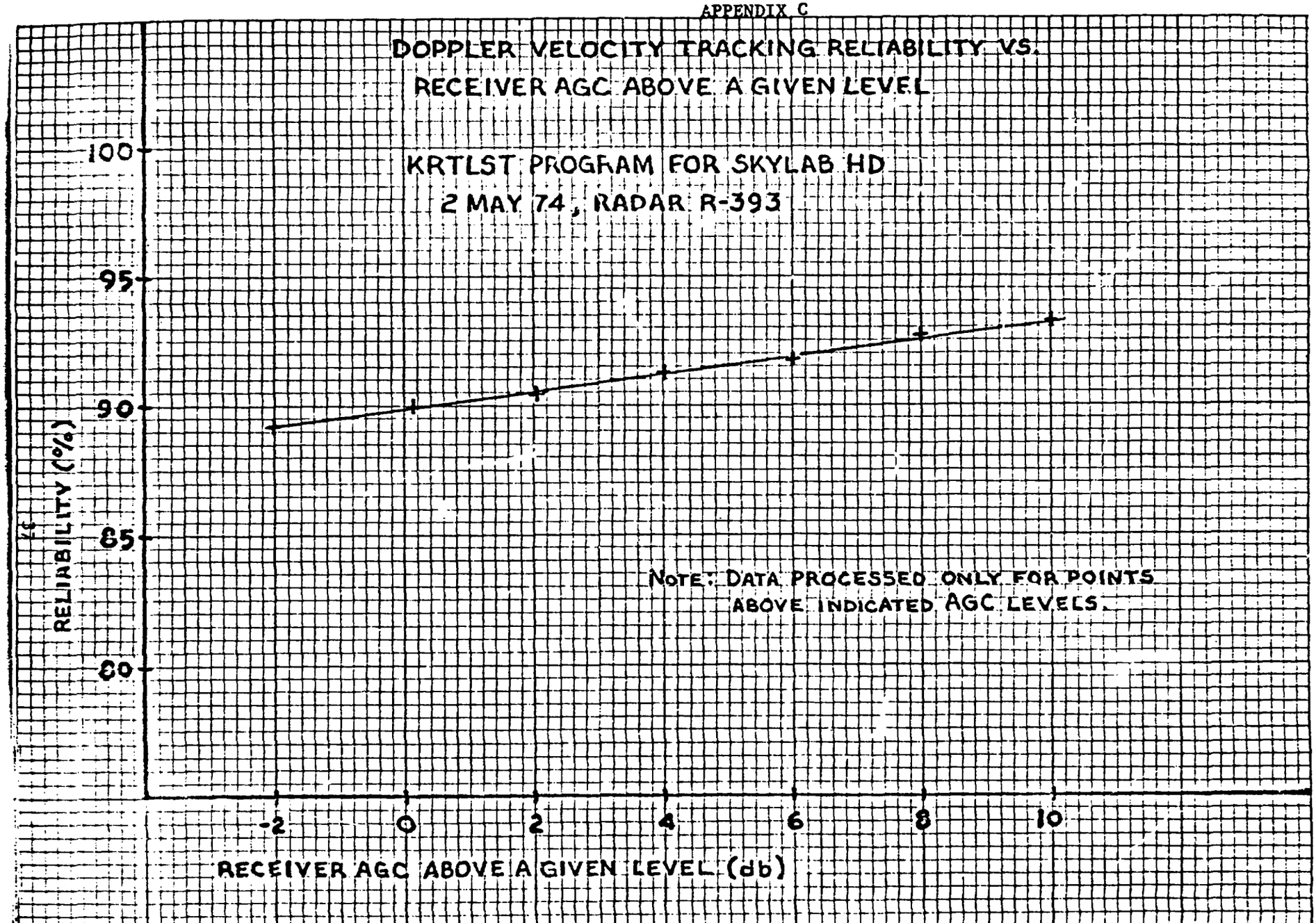

DOPPLER VELOCITY TRACKING RELIABILITY VS.
RECEIVER AGC ABOVE A GIVEN LEVEL
KRTLST PROGRAM FOR SKYLAB HD
2 MAY 74, RADAR R-393
100
95
90
85
80
RELIABILITY (%)
NOTE: DATA PROCESSED ONLY FOR POINTS ABOVE INDICATED AGC LEVELS.
-2
0
2
4
6
8
10
RECEIVER AGC ABOVE A GIVEN LEVEL (db)

FIGURE 3

```
1 OPEN"LP:" FOR OUTPUT AS FILE 1
10 DIM G(4,3),X(101),Z(101)
25 DIM Q(4,50),S(4)
30 LET A=0:LET B=0:LET C=0
40 LET I1=0:LET D=18.56:LET T=0:LET H=.05
50       PRINT"MISSION TIME";
55       PRINT #1,"MISSION TIME";
60       INPUT NO
65       PRINT #1,NO
100      PRINT"BURST TIMES AND SCALE FACTOR";
105      PRINT #1,"BURST TIMES AND SCALE FACTOR";
110      INPUT TO;T1;S
115      PRINT #1,TO;T1;S
150      PRINT"INITIAL RANGE. . RANGE RATE. . . ACCELERATION";
155      PRINT #1,"INITIAL RANGE . RATE RATE . ACCELERATION";
160      INPUTSO;S1;S2
165      PRINT #1,SO;S1;S2
200 LET X=1:LET I=0
220 LET RO=SO:LET R1=0
250      PRINT"SIGNAL TO NOISE. . RANGE. . RANGE RATE";
255      PRINT #1,"SIGNAL TO NOISE RATIO . RANGE . RANGE RATE"
260      INPUT V1;V2
265      PRINT #1,V1;V2
280 LET N5=0:LET N6=0:LET N7=0:LET N8=0:LET N9=0
300 LET V1=10^(V1/10):LET V2=10^(V2/10)
310 FOR S5=1 TO 3
320 LET Z=SO:GO TO 1000
330 PRINT #1,"RESCALE ECHO";S/(V1^2);1/(V1^2);"::";S/(V2^2);1/(V2^2)
350 PRINT"VW1. . . . VW2. . . . VOBS"
355      PRINT #1,"VW1 . VW2 . VOBS";
360 INPUT W8,W9,W7
365      PRINT #1,W8;W9;W7
400      PRINT"RESCALE";
405      PRINT #1,"RESCALE";
420 INPUT S8,S9,S7
425      PRINT #1,S8,S9,S7
500 PRINT '**************"
600 PRINT #1:PRINT #1:PRINT #1
700 PRINT #1,"TIME       RANGE ERROR       LINE ERROR       GAIN K1       GAIN K2"
720 PRINT #1
800 FOR I=1 TO NO/.05
900      REM TARGET NOMINAL
910 LET RO=SO+T*S1+T*T*S2/2
920 LET R1=S1+T*S2
1000     REM TARGET
1020 LET N1=0:LET N2=0
1040 FOR K=1 TO 12
1060 LET N1=N1+RND(I):LET N2=N2+RND(I)
1080 NEXT K
1100     REM NOISE GEN
1200 LET N1=N1-6:LET N2=N2-6
1210 LET N5=N5+1:LET N6=N6+N1:LET N7=N7+N2:LET N8=N8+N1^2:LET N9=N9+N2^2
1220 LET W=1:LET W1=W7:LET W2=W8:LET W3=W9
1240 IF T<TO THEN 1300
1260 IF T>T1 THEN 1300
1270 LET R2=RO
1280 LET W=S:LET W1=S7*W7:LET W2=S8*W8:LET W3=S9*W9
1300 LET RO=RO*(1+W*N1/(V1^2))
1320 LET R1=R1*(1+W*N2/(V2^2))
1340 IF I>0 THEN 2000
1360 GO TO 330
2000     REM KALMAN GAINS
2020 LET K1=0:LET K2=0:LET K3=0
2040 GOSUB 8000
2060 LET G(1,1)=F1:LET G(1,2)=F2:LET G(1,3)=F3
```

APPENDIX D

SOURCE:KALMAN FILTER

I

```
2100 GOSUB 8000
2120 LET G(2,1)=F1:LET G(2,2)=F2:LET G(2,3)=F3
2140 LET K1=H*F1/2:LET K2=H*F2/2:LET K3=H*F3/2
2160 GOSUB 8000
2180 LET G(3,1)=F1:LET G(3,2)=F2:LET G(3,3)=F3
2200 LET K1=H*F1:LET K2=H*F2:LET K3=H*F3
2220 GOSUB 8000
2240 LET G(4,1)=F1:LET G(4,2)=F2:LET G(4,3)=F3                    II
2260 LET A=A+H*(G(1,1)+2*G(2,1)+2*G(3,1)+G(4,1))/6
2280 LET B=B+H*(G(1,2)+2*G(2,2)+2*G(3,2)+G(4,2))/6
2300 LET C=C+H*(G(1,3)+2*G(2,3)+2*G(3,3)+G(4,3))/6
2560 IF ABS(A)>1E20 OR ABS(B)>1E20 OR ABS(C)>1E20 THEN 7000
3000    REM STATE ESTIMATIONS
3020 LET K1=0:LET K2=0
3040 GOSUB 6000
3060 LET G(1,1)=F1:LET G(1,2)=F2
3080 LET K1=H*F1/2:LET K2=H*F2/2
3100 GOSUB 6000
3120 LET G(2,1)=F1:LET G(2,2)=F2
3140 LET K1=H*F1/2:LET K2=H*F2/2
3460 GOSUB 6000
3480 LET G(3,1)=F1:LET G(3,2)=F2
3500 LET K1=H*F1:LET K2=H*F2
3520 GOSUB 6000
3540 LET G(4,1)=F1:LET G(4,2)=F2
4000 LET Z=Z+H*(G(1,1)+G(2,1)*2+G(3,1)*2+G(4,1))/6
4020 LET X=X+H*(G(1,2)+G(2,2)*2+G(3,2)*2+G(4,2))/6
4040 IF ABS(Z)>1E20 OR ABS(X)>1E20 THEN 7000
4500 LET T=T+.05
4600 LET T5=I-INT(I/4)*4
4620 IF T5<>0 THEN 4800
4700 PRINT X,A/W1,B/W1
4710 PRINT #1,I*H       ,Z-R2         ,X          ,A/W1          ,B/W1
4715 LET S4=INT(I/4)
4720            LET Q(S5,S4)=X
4800 NEXT I
4810 LET T=0:LET A=0:LET B=0:LET C=0:LET X=0:LET Z=S0
4820LET X=1 :LET I=0
4830 NEXT S5
4840 LET S6=0
4850 FOR S1=1 TO 3
4860 FOR S2=1 TO 25
4870 IF S6>=ABS(Q(S1,S2)) THEN 4890
4880 LET S6=ABS(Q(S1 S2))
4890 NEXT S2
4900 NEXT S1
4910 FOR S1=1 TO 3
4920 FOR S2=1 TO 25
4930 LET Q(S1,S2)=Q(S1,S2)/S6  !NORMALIZED SOECTRAL LINES
4940 NEXT S2
4950 NEXT S1
4970 STOP
4980 PRINT #1:LET N6=N6/N5:LET N7=N7/N5:LET N8=N8/N5-N6:LET N9=N9/N5-N7
4990 PRINT #1,"MEAN AND VARIANCE...RANGE";N6;N8;"RANGE RATE";N7;N9
5015 PRINT #1:PRINT #1
5020 PRINT #1,"      SPECTRAL LINES IN ERROR"
5025 PRINT #1,TAB(30);"0"
5030 FOR S=1 TO 25
5040 MAT S=ZER
5050 LET I=-2
5060 FOR M=1 TO 3
5080 FOR J=1 TO 3
5090 IF J=S(1) THEN 5200
5100 IF J=S(2) THEN 5200
5120 IF Q(J,S)<=1 THEN 5200
```

```
5200 NEXT J
5210 LET S(M)=I0:LET I=-2
5220 NEXT M
5400 FOR J=1 TO 3
5410 LET S(J)=30*Q(S(J),S)+30
5420 NEXT J
5500 PRINT #1,S#.2;TAB(S(3));"*";TAB(S(2));"*";TAB(S(1));"*"
5600 NEXT S
5700 GO TO 100
6000    REM STATE FUNCTIONS
6020 LET F1=R1+A*(R0-(Z+K1))/W1+D*(X+K2)
6040 LET F2=B*(R0-(Z+K1))/W1
6100    RETURN
7000 PRINT"ERROR DETECTED"
7005    PRINT #1,"ERROR DETECTED"
7020 PRINTI;Z;X;A;B;C;W2;W3;W1
7025    PRINT #1,Z;X;A;B;C;W2;W3;W1
7040 GO TO 30
8000 REM KALMAN FUNCTIONS
8010 LET F1=2*(B+K2)*D-((A-K1)^2)/W1+W2
8020 LET F2=(C+K3)*D-(A+K1)*(B+K2)/W1
8030 LET F3=W3-((B+K2)^2)/W1
8040 RETURN
9900 CLOSE 1
9999 END
```

III

```
BURST TIMES AND SCALE FACTOR1 OPEN"KB:" FOR OUTPUT AS FILE 1
10 DIM G(4,3),X(101),Z(101)
25 DIM Q(4,50),S(4)
30 LET A=0:LET B=0:LET C=0
40 LET I1=0:LET D=18.56:LET T=0:LET H=.05
50       PRINT"MISSION TIME";
55       PRINT #1,"MISSION TIME";                          SOURCE: STEADY STATE
60       INPUT NO                                                  KALMAN FILTER
65       PRINT #1,NO
100      PRINT"BURST TIMES AND SCALE FACTOR";                        I
105      PRINT #1,"BURST TIMES AND SCALE FACTOR";
110      INPUT TO;T1;S
115      PRINT #1,TO;T1;S
150      PRINT"INITIAL RANGE..RANGE RATE...ACCELERATION";
155      PRINT #1,"INITIAL RANGE . RATE RATE . ACCELERATION";
160      INPUTSO;S1;S2
165      PRINT #1,SO;S1;S2
200 LET X=1:LET I=0
220 LET RO=SO:LET R1=0
250      PRINT"SIGNAL TO NOISE..RANGE..RANGE RATE";
255      PRINT #1,"SIGNAL TO NOISE RATIO . RANGE . RANGE RATE"
260      INPUT V1;V2
265      PRINT #1,V1;V2
270 LET N5=0:LET N6=0:LET N7=0:LET N8=0:LET N9=0
300 LET V1=10^(V1/10):LET V2=10^(V2/10)
310 FOR S5=1 TO 3
320 LET Z=SO:GO TO 1000
330 PRINT #1,"RESCALE ECHO";S/(V1^2);1/(V1^2);"::";S/(V2^2);1/(V2^2)
350 PRINT"VW1....VW2....VOBS"
355      PRINT #1,"VW1 . VW2 . VOBS";
360 INPUT W8,W9,W7
365      PRINT #1,W8;W9;W7
400      PRINT"RESCALE";
405      PRINT #1,"RESCALE";
420 INPUT S8,S9,S7
425      PRINT #1,S8,S9,S7
500 PRINT "**************"
600 PRINT #1:PRINT #1:PRINT #1
700 PRINT #1,"TIME       RANGE ERROR       LINE ERROR       GAIN K1       GAIN K2"
720 PRINT #1
800 FOR I=1 TO NO/.05
900      REM TARGET NOMINAL
910 LET RO=SO+T*S1+T*T*S2/2
920 LET R1=S1+T*S2
1000     REM TARGET
1020 LET N1=0:LET N2=0
1040 FOR K=1 TO 12
1060 LET N1=N1+RND(I):LET N2=N2+RND(I)
1080 NEXT K
1100     REM NOISE GEN
1200 LET N1=N1-6:LET N2=N2-6
1210 LET N5=N5+1:LET N6=N6+N1:LET N7=N7+N2:LET N8=N8+N1^2:LET N9=N9+N2^2
1220 LET W=1:LET W1=W7:LET W2=W8:LET W3=W9
1240 IF T<TO THEN 1300
1260 IF T>T1 THEN 1300
1270 LET R2=RO
1280 LET W=S:LET W1=S7*W7:LET W2=S8*W8:LET W3=S9*W9
1300 LET RO=RO*(1+W*N1/(V1^2))
1320 LET R1=R1*(1+W*N2/(V2^2))
1340 IF I>0 THEN 2000
```

```
2000     REM KALMAN GAINS
2020 LET B=SQR(W1*W3)
2040 LET A=SQR(W1*(2*D*B+W2))
2060 LET C=A*B/(D*W1)
2560 IF ABS(A)>1E20 OR ABS(B)>1E20 OR ABS(C)>1E20 THEN 7000
3000     REM STATE ESTIMATIONS
3020 LET K1=0:LET K2=0
3040 GOSUB 6000
3060 LET G(1,1)=F1:LET G(1,2)=F2
3080 LET K1=H*F1/2:LET K2=H*F2/2                                        II
3100 GOSUB 6000
3120 LET G(2,1)=F1:LET G(2,2)=F2
3140 LET K1=H*F1/2:LET K2=H*F2/2
3460 GOSUB 6000
3480 LET G(3,1)=F1:LET G(3,2)=F2
3500 LET K1=H*F1:LET K2=H*F2
3520 GOSUB 6000
3540 LET G(4,1)=F1:LET G(4,2)=F2
4000 LET Z=Z+H*(G(1,1)+G(2,1)*2+G(3,1)*2+G(4,1))/6
4020 LET X=X+H*(G(1,2)+G(2,2)*2+G(3,2)*2+G(4,2))/6
4040 IF ABS(Z)>1E20 OR ABS(X)>1E20 THEN 7000
4500 LET T=T+.05
4600 LET T5=I-INT(I/4)*4
4620 IF T5<>0 THEN 4800
4700 PRINT I*H;Z;X;A;B;C;W2;W3;W1
4710 PRINT#1,I*H       ,Z-R2          ,X             ,A*W1          ,B*W1
4720 LET S4=INT(I/4):LET Q(S5,S4)=X
4800 NEXT I
4810 LET T=0:LET A=0:LET B=0:LET C=0:LFT X=0:LET Z=S0
4820LET X=1 :LET I=0
4830 NEXT S5
4840 LET S6=0
4850 FOR S1=1 TO 3
4860 FOR S2=1 TO 25
4870 IF S6>=ABS(Q(S1,S2)) THEN 4890
4880 LET S6=ABS(Q(S1,S2))
4890 NEXT S2
4900 NEXT S1
4910 FOR S1=1 TO 3
4920 FOR S2=1 TO 25
4930 LET Q(S1,S2)=Q(S1,S2)/S6  !NORMALIZED SOECTRAL LINES
4940 NEXT S2
4950 NEXT S1
4970 STOP
4980 PRINT #1:LET N6=N6/N5:LET N7=N7/N5:LET N8=N8/N5-N6:LET N9=N9/N5-N7
4990 PRINT #1,"MEAN * VARIANCE. . RANGE";N6;N8;"RANGE RATE. . ";N7;N9
5000 PRINT #1
5010 PRINT #1
5020 PRINT #1,"      SPECTRAL LINES IN ERROR"
5025 PRINT #1,TAB(30);"0"
5030 FOR S=1 TO 25
5040 MAT S=ZER
5050 LET I=-2
5060 FOR M=1 TO 3
5080 FOR J=1 TO 3
5090 IF J=S(1) THEN 5200
5100 IF J=S(2) THEN 5200
5110 IF J=S(3) THEN 5200
5120 IF Q(J,S)<=I THEN 5200
5140 LET I=Q(J,S): LET I0=J
5200 NEXT J
5210 LET S(M)=I0:LET I=-2
5220 NEXT M
5400 FOR J=1 TO 3
5410 LET S(J)=30*Q(S(J),S)+30
```

```
5500 PRINT #1, S$, 2; TAB(S(3)); "*"; TAB(S(2)); "*"; TAB(S(1)); "*"
5600 NEXT S
5700 GO TO 100
6000    REM STATE FUNCTIONS
6020 LET F1=R1+A*(R0-(Z+K1))/W1+D*(X+K2)
6040 LET F2=B*(R0-(Z+K1))/W1
6100    RETURN
7000 PRINT"ERROR DETECTED"
7005    PRINT #1, "ERROR DETECTED"
7020 PRINTI; Z; X; A; B; C; W2; W3; W1                                  III
7025    PRINT #1, Z; X; A; B; C; W2; W3; W1
7040 GO TO 30
9900 CLOSE 1
9999 END
```

APPENDIX E

Comput. & Elect. Engng, Vol. 2, pp. 105–115. Pergamon Press, 1975. Printed in Great Britain.

ON THE COMPUTATION OF KALMAN GAINS

FRED J. TAYLOR

Department of Electrical Engineering, University of Texas at El Paso, El Paso, Texas 79968, U.S.A.

(*Received* 13 *May* 1974)

Abstract—Modern filtering methods often require high order matrix differential equations to be solved. Standard numerical methods are traditionally slow and prone to be unstable. A numerical approach to the problem of computing the Kalman gain matrix is developed which is both numerically efficient and stable. If a piecewise approximation of the Kalman gain matrix is desired, efficiencies of many orders of magnitude can be realized.

INTRODUCTION

Contemporary systems literature is rich in the study of minimal variance filtering theory as applied to linear constant coefficient differential dynamic system corrupted by stationary white noise[1–3].

The high level of activity in this area has produced numerous papers on the utility, as well as the dangers, of filtering. However, in the embodiment of literature, little attention has been given to the numerical problems associated with computing the required time varying matrix gains (Kalman gains). Such problems are usually treated through neo-classic Runge–Kutta, Milne, Adams, etc. methods. They are often slow. This can sometimes be forgiven if a non-real-time filtering is sought. Besides being slow, they are inherently unstable. This cannot be tolerated in most applications. Therefore, a computationally fast and stable algorithm shall be sought.

Problem

Let a n dimensional message model be given by

$$\dot{x}(t) = Fx(t) + Gw(t) \tag{1}$$

with r observations given by

$$z(t) = Hx(t) + v(t). \tag{2}$$

Let the white noise processes be defined as usual.

$$\begin{aligned} E(x(0)) &= x_0;\ E(w(t)) = E(v(t)) = 0 \\ \text{var}(x(0)) &= V_x(0);\ \text{cov}(x(0)w^T(t)) = 0 \\ \text{cov}(w(t)w^T(\tau)) &= V_w\delta(t-\tau) \\ \text{cov}(v(t)v^T(\tau)) &= V_v\delta(t-\tau) \\ \text{cov}(w(t)v^T(\tau)) &= \text{cov}(v(t)w^T(\tau)) = 0. \end{aligned} \tag{3}$$

The minimal variance estimate of $x(t)$, say $\hat{x}(t)$, is known to satisfy [3]

$$\dot{\hat{x}}(t) = F\hat{x}(t) + K(t)[z(t) - H\hat{x}(t)] \tag{4}$$

$$K(t) = V_{\hat{x}}(t)H^T V_v^{-1} \tag{5}$$

where $\tilde{x}(t)$ is the extimation error (i.e. $\tilde{x}(t) = x(t) - \hat{x}(t)$) and $V_{\hat{x}}(t)$ is the error covariance matrix satisfying

$$\dot{V}_{\hat{x}}(t) = FV_{\hat{x}}(t) + V_{\hat{x}}(t)F^T - V_{\hat{x}}(t)H^T V_v^{-1} HV_{\hat{x}}(t) + GV_w G^T \tag{6}$$

$$V_{\hat{x}}(0) = V_x(0).$$

The heart of the filter is quantifying the error covariance matrix $V_{\hat{x}}(t)$, for $t > 0$. This is in general a non-trivial numerical problem. However, a recent work of Davison and Maki[4], with improvements by Taylor[5], can be adapted to provide a rapid stable solution of equation (6). It utilizes an approach suggested by Sage[3] (but also discouraged by that author) which interprets equation (6) as a $2n \times 2n$ first linear differential system.

Consider, for $N = 2n$, a N dimensional vector $\xi(t)$ satisfying

$$\dot{\xi}(t) = \left[\begin{array}{c|c} -F^T & H^T V_v^{-1} H \\ \hline GV_w G^T & F \end{array}\right]_{N \times N} \times \quad \xi(t) \triangleq \mathscr{R}(t)\,\xi(t). \tag{7}$$

This solution of equation (7) is characterized by the matrix exponential

$$\exp(\mathscr{R}t) = \left[\begin{array}{c|c} \phi_{11}(t)_{n\times n} & \phi_{12}(t)_{n\times n} \\ \hline \phi_{21}(t)_{n\times n} & \phi_{22}(t)_{n\times n} \end{array}\right]_{N \times N} \tag{8}$$

It can also be shown that

$$V_{\hat{x}}(t) = [\phi_{21}(t, t_0) + \phi_{22}(t, t_0)V_{\hat{x}}(0)][\phi_{11}(t, t_0) + \phi_{12}(t, t_0)V_{\hat{x}}(0)]^{-1}. \tag{9}$$

Therefore, computing $V_{\hat{x}}$ has been converted into a problem of computing the $n \times n$ matrices $\phi_{i,j}(t)$, $i = 1,2, j = 1,2$. The following algorithm can be used to efficiently produce those desired matrices.

Numerical solution

The modified Crank–Nicholson matrix approximation is given by [4, 5]

$$\exp(\mathscr{R}h) = C + O(h^5)$$

where

$$C = [I - h\mathscr{R}/2 + h^2\mathscr{R}^2/12]^{-1}[I + h\mathscr{R}/2 + h^2\mathscr{R}^2/12] \tag{10}$$

where h is a scalar (step size) and $O(h^5)$ denotes a fifth order of error accuracy, (Runge–Kutta RK4 methods provide only fourth order accuracy). The commutative property of the fundamental state transition matrix $\exp(\mathscr{R}t)$ admits the generation of $\phi_{i,j}(t)$, $i = 1,2$, $j = 1,2$, sequentially since

$$\begin{aligned} \exp(\mathscr{R}h) &= C \\ \exp(\mathscr{R}2h) &= C^2 \\ &\cdot \\ &\cdot \\ &\cdot \\ \exp(\mathscr{R}T) &= C^N. \end{aligned} \tag{11}$$

For some prespecified step size h, suppose $T = 2^k h$. A binary coded scheme to produce the required powers of C, namely

$$\begin{aligned} C^2 &= C^1 \cdot C^1 \\ C^3 &= C^2 \cdot C^1 \\ C^4 &= C^2 \cdot C^2 \\ &\text{etc.} \end{aligned}$$

would require $2^k N^3$ multiplicative operations. If $2^k \gg N^2$, it is shown in (5) that from the application of Bocher's formula, this can be reduced to only $2^k N^2$ multiplicative operations.

Bocher's formula yields the following results:†

(i) Let $C^1, C^2, \ldots C^{N-1}$ be computed.

(ii) Define

$$\begin{aligned} \alpha(N-1,0) &= T_1 \\ \alpha(N-2,0) &= (\alpha(N-1,0)\,T_1 + T_2)/2 \\ &\cdot \\ &\cdot \\ &\cdot \\ \alpha(0,0) &= \Big(\sum_{i=1}^{N-1} \alpha(i,0)T_i + T_N\Big)/N \end{aligned}$$

where $T_i = \text{trace}\,(C^i)$, $i = 1, \ldots, N-1$.

† Bocher's formula is erroneously transcribed in Ref. [6]. It is derived in Appendix A.

(iii) Define for $m > 0$

$$\alpha(N-1,m) = \alpha(N-2, m-1) - \alpha(N-1,0)\alpha(N-1, m-1)$$

$$\alpha(N-2,m) = \alpha(N-3, m-1) - \alpha(N-2,0)\alpha(N-1, m-1)$$

$$\alpha(0, m) = 0 - \alpha(0, 0)\alpha(N-1, m-1)$$

(iv) Then

$$C^{N+m} = \sum_{i=0}^{N-1} \alpha(i, m)C^i, \; 0 \le m \le 2^k - N.$$

Once (i) and (ii) have been precomputed, all successive powers of C can be generated recursively. Besides realizing a speed improvement, the entire program could be effectively embedded into the now available microprogrammable minicomputers. The proposed algorithm would require $N^3 + N^2$ memory word locations to support the recursive operation. It is interesting to note that the developed algorithm does not require production of the eigenvalues of C. Therefore, one of the main objections to fundamental solution based techniques, namely solving the characteristic polynomial $\det(\lambda I - C) = 0$, is not an issue. The solution technique proposed is outlined in Fig. 1.

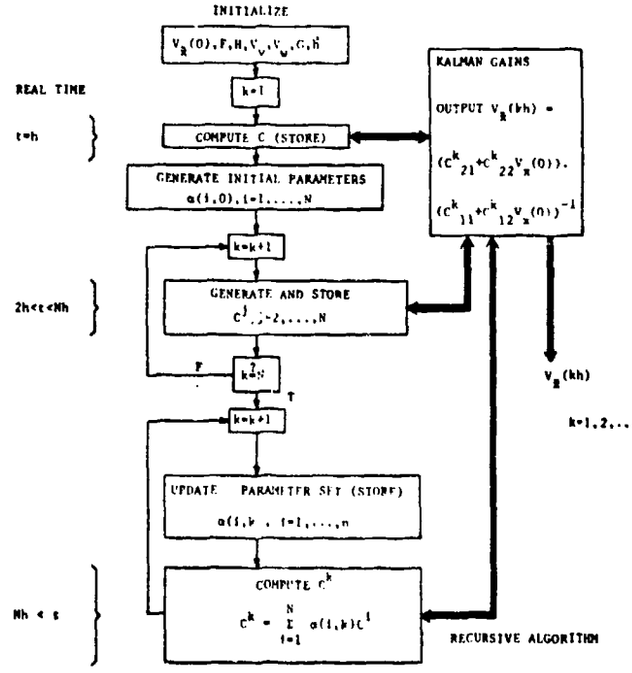


Fig. 1.

Heuristic extension

Kalman gains are usually most active near $t = 0$. As $t \to \infty$, $K(t)$ tends exponentially to some steady-state value provided the plant-observer pair are stable. Therefore, it would seem reasonable that a variable step-size algorithm could be developed which would accelerate the solution process. The variable step-size algorithm should give a dense time domain cover where $K(t)$ is most active (i.e. $t \simeq 0$) and be sparse where $K(t)$ is inactive ($K(t) \to \infty$). Consider then the construction of an algorithm which will produce a piecewise constant approximation to $K(t)$. Let this suboptimal gain matrix satisfy the following criterion:

Criteria (piecewise continuous approximation)

Let $K^a(t_i)$ be a piecewise continuous approximation of $K(t)$ over $t_i \leq t \leq t_{i+1}$, where

$$\|K^a(t_i) - K(t)\| \leq \varepsilon \tag{12}$$

$\varepsilon > 0$. Here ε serves as a prespecified admissible error bound on the approximation process. A small (large) ε would result in a finely (coarsely) refined approximation of $K(t)$. This thesis can be effectively accomplished by means of the binary coding scheme previously mentioned. Consider the following "variable interval—ε meteoric" Kalman gain algorithm.

Define the matrix norm of a $N \times N$ square matrix Λ to be

$$\begin{aligned} \|\Lambda\| &= \max \{|\Lambda_{ij}|\} \\ & i = 1, 2, \ldots, N \\ & j = 1, 2, \ldots, N. \end{aligned} \tag{13}$$

Procedure

No. 1 Choose h sufficiently small so that $0(h^5)$ error is tolerable

No. 2 Compute $C^1 = \exp(\mathscr{A}h)$

No. 3 Compute $C^{2(l+1)} = C^{2l} \cdot C^{2l}$; $l = 1, 2, 3, \ldots$

No. 4 Test: If $\|C^{2(l+1)} - C^{2l}\| < \varepsilon$

(i) if true

let $C^{2l} \cong \exp(\mathscr{A}t)$

for $2^l h \leq t < 2^{(l+1)}h$

(ii) if not true

reduce search interval to some $\bar{t}$ such that $2^l h \leq \bar{t} < 2^{(l+1)}h - \delta$ where

$$0 < \delta < 2^l h.$$

Return to No. 4.

Some obvious interval reducing methods would be an equal interval, dichotomous, or Fibonacci search methods. However, from a computational efficiency viewpoint, the following approach has proven most effective.

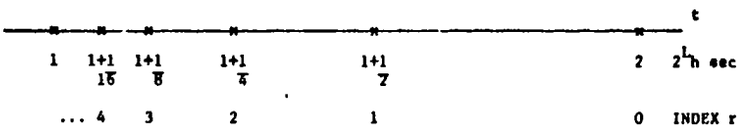


Fig. 2.

Suppose C^{2l} has been computed and accepted at time $t = 2^l h$. Suppose further that the last S computed matrices, namely $C^{2l}, C^{2(l-1)}, \ldots, C^{2(l-s)}$, are stored in memory. Compute $C^{2(l+1)}$ as in No. 3. If it fails test No. 4. Reduce the search interval by testing C^{2l} sequentially against

$$C^{2l+2^{(l-r)}} = C^{2l} \cdot C^{2^{(l-r)}}; r = \{1, 2, \ldots, S\}. \tag{14}$$

Here $C^{2l+2^{(l-r)}}$ can easily be generated by direct binary operation from matrices found in memory. The reduced interval is a monotonically decreasing sequence over the index set r. It is characterized in Fig. 2. In practice, the number of previously computed and stored matrices, indexed by S, is to be determined experimentally. Since memory requirements are generally considered to be a secondary goal when compared to computational speed,

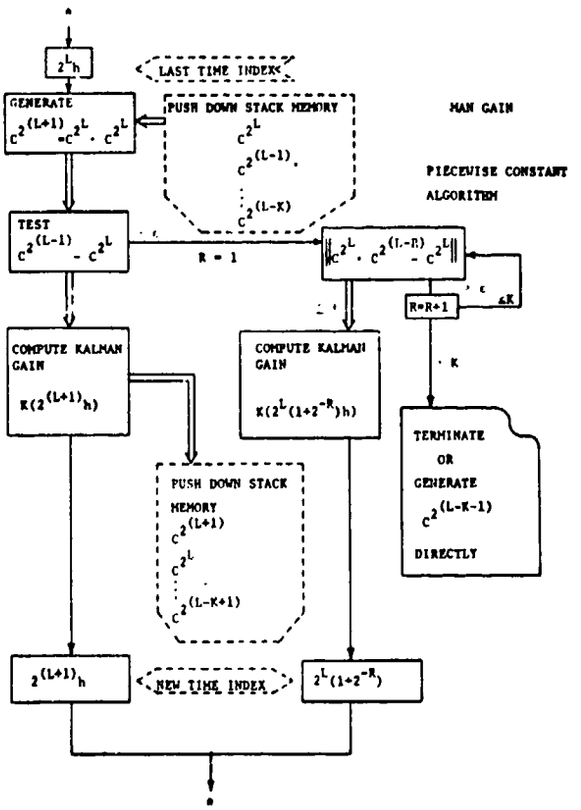


Fig. 3.

one can usually be optimistic in one's choice of S. If it should turn out that the choice of S was insufficient to support the piecewise constant approximation process, the required powers of C can be synthesized directly from binary operations on C. Therefore, a solution is always recoverable. The suggested approximation process is depicted in Fig. 3.

Steady-state Kalman gains

Steady-state Kalman gains can now easily be computed. Using a σ test, the Kalman gain at some time $t_l = 2^l h$ can be rapidly computed using base-2 algorithm. The Kalman gains are assumed to have non-oscillatory behavior, for t_l sufficiently large; then the steady value of $K(t)$ may be assigned the value $K(t_{l+1})$ if,

$$\|K(t_{l+1}) - K(t_l)\| < \sigma,\ t_l = 2^l h,\ \sigma > 0. \tag{15}$$

If the heuristic algorithm is used, which maintains a history of the last S "power of C" operations, namely

$$\{C^{2l}, C^{2(l-1)}, \ldots, C^{2(l-S)}\}$$

then it would be reasonable to assume that the last S Kalman gains are also stored in memory. Let the following $n \times n$ gain matrices be found in memory

$$\{K(t_l), K(t_{l-1}), \ldots, K(t_{l-S})\}. \tag{16}$$

Consider the Kalman gain matrix to have obtained a steady-state value if

$$\|\sum_{i=0}^{S} K(t_{l-i})\| < \sigma/(S+1) \tag{17}$$

and let the steady-state value of $K(t)$ be assigned the value $K(t_l)$.

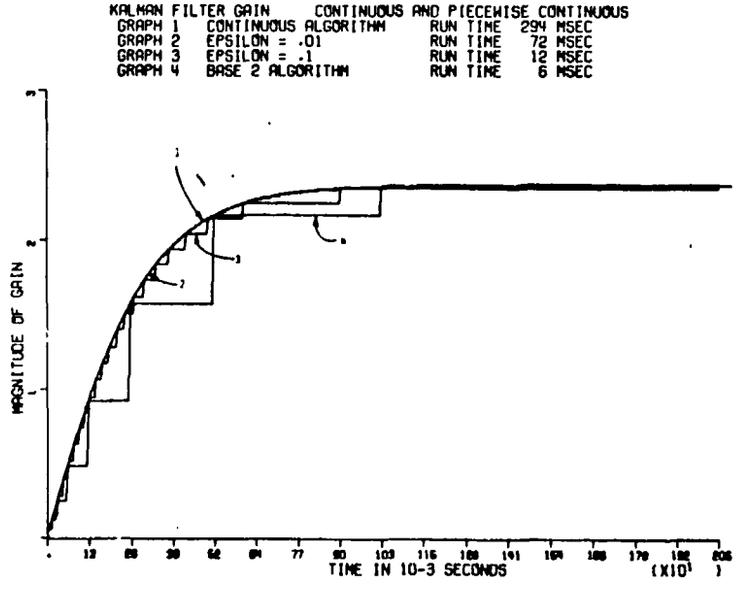


Fig. 4.

Example 1 (*scalar example*)

Find the Kalman gain associated with the system

$$x^0 = -0{\cdot}5x + w \qquad u \le t \le 2{\cdot}048$$

$$z = x + v$$

$$\operatorname{cov}(w(t), w(\tau)) = 2\delta(t - \tau)$$

$$\operatorname{cov}(v(t), v(\tau)) = 1/4\delta(t - \tau)$$

$$\operatorname{cov}(x0), x(0)) = 0.$$

The solution to the above problem can be found in Sage (p. 244), and is

$$K(t) = V_{\tilde{x}}(t)H^t V_v^{-1} = -1/2 + 1/2\sqrt{33}t \tanh\left(\sqrt{\frac{33}{2}}t + \tanh^{-1}\frac{1}{\sqrt{33}}\right).$$

The solution obtained using the new proposed algorithm, over $u \le t \le 2{\cdot}048$, in steps of 0·001 sec, is denoted in graph 1 of Fig. 4. It was always accurate to within 8 decimal places. It required 294 msec to complete the solution. Using the heuristic technique cited in the paper, an epsilon of $\varepsilon = 0{\cdot}01$ and $\varepsilon = 0{\cdot}1$ were tested (see equation 12). They are graphs 2 and 3 of Fig. 4 respectively. The test associated with $\varepsilon = 0{\cdot}01$ produced an excellent piecewise constant approximation to $K(t)$ in only 72 msec. Using the fastest algorithm possible, namely the base 2 algorithm, an approximation of $K(t)$ as $t \to \infty$ and produced in only 6 msec. This result appears as graph 4 of Fig. 4.

Example 2

Given

$$\dot{x}(t) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -3 & -3 \end{pmatrix} x(t) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} w(t)$$

$$z(t) = (100)x(t) + v(t)$$

where

$$V_w = 1$$

$$V_v = 1$$

$$V_{\tilde{x}(0)} = I.$$

The Kalman gain matrix $K(t)$, over $0 \le t \le 1$, was computed using conventional RK-4 as well as the proposed algorithm, using a step size h of 0·01 sec (the solution obtained using the new algorithm can be found in Appendix B).* Using a RK-4 approach, a solution was obtained in 38,320 msec. Using the new algorithm, a solution was obtained in only 16,312 msec.† Furthermore, any reduction in step size h caused the RK-4 method to diverge. However, the proposed algorithm was found to be stable and independent of the choice of h.

* Appendix B available from author on request.

† 15,455 msec where needed to compute C^L, $L = 3, 4, \ldots, 100$; 857 msec where needed to initialize the problem.

If the order of a system is increased from n_1 to n_2, there would be a resulting increase in time expended on the computation of $K(t)$. Suppose it took K_1 sec to compute $V_x(t)$ for a $n_1 \times n_1$ system. Then computing $V_x(t)$ through $R-K$ methods would require approximately

$$8\left(\frac{n_2}{n_1}\right)^4$$

more multiplications to generate the right-hand side of

$$\dot{V}_x(t) = FV_x(t) + V_x(t)F^T - V_x(t)H^TV_v^{-1}HV_x(t) + GV_wG^T.$$

Since there are

$$\left(\frac{n_2}{n_1}\right)^2$$

more integrals to solve, it would take approximately (modulo housekeeping programming)

$$K_1\left(8\left(\frac{n_2}{n_1}\right)^4 + \left(\frac{n_2}{n_1}\right)^2\right)$$

seconds to compute $V_x(t)$ for the $n_2 \times n_2$ system. The proposed algorithm would require

$$\left(\frac{2n_2}{2n_1}\right)^3 = \left(\frac{n_2}{n_1}\right)^3$$

more multiplications to construct

$$C^l = \sum_{i=0}^{N-1} \alpha_i C^i$$

and approximately

$$4\left(\frac{n_2}{n_1}\right)^4$$

more to compute

$$V_x(t) = [\phi_{21} + \phi_{22}V_x(0)][\phi_{11} + \phi_{12}V_x(0)]^{-1}.$$

If it took K_2 sec to compute the solution of a $n_1 \times n_1$ system, it would take approximately

$$K_2\left[4\left(\frac{n_2}{n_1}\right)^4 + \left(\frac{n_2}{n_1}\right)^3\right]$$

seconds to solve the $n_2 \times n_2$ problem. Therefore, there would be a general speed improvement of (for $n_2/n_1 > 1$)

$$\frac{K_1\left[8\left(\frac{n_2}{n_1}\right)^4 + \left(\frac{n_2}{n_1}\right)^2\right]}{K_2\left[4\left(\frac{n_2}{n_1}\right)^4 + \left(\frac{n_2}{n_1}\right)^3\right]} \cong \frac{2K_1}{K_2}$$

in favor of the proposed method. In terms of 3rd order benchmark problem, this speed

4 of 7

improvement factor equates to approx. 4. This can be a meaningful savings when the computation times become long.

However, the heuristic algorithm, previously discussed and tested in example 1, has been shown to be a great saver of computer time. Several orders of magnitude may be saved if a piecewise continuous approximation of $K(t)$ is acceptable.

CONCLUSIONS

The theory, methodology, and supporting examples of a new Kalman gain matrix algorithm have been presented. The results to date are most satisfactory in terms of accuracy and speed. The heuristic algorithms discussed have proven to be a very worthwhile trade off between accuracy and speed. Finally, an effective approach to the problem of estimating steady-state gains was discussed and supported with an example.

The developed algorithm and sample output is available from the author. It is coded in Fortran IV and appears in three parts. The first part is a program dedicated to the generation of the required powers of C. The second part interprets the powers of C as a Kalman gain matrix. The third part is a general matrix package. The program currently will handle a 10th order system but can easily be expanded upward.

REFERENCES

1. A. P. Sage, *Optimum Systems Control*. Prentice Hall, New Jersey. (1968).
2. T. O. Lewis and P. L. Odell, *Estimation in Linear Models*. Prentice Hall, New Jersey (1971).
3. A. P. Sage and J. Melsa, *Estimation Theory With Applications to Communication and Control*. McGraw-Hill, New York (1970).
4. E. J. Davison and M. C. Maki, The numerical solution of the matrix Ricatti equation, *IEEE Trans. Automatic Control Ac.* (1973).
5. F. J. Taylor, Components on "The numerical solution of the matrix Ricatti differential equation", *IEEE Trans. Automatic Control*. 1974.
6. Director and R. Rohrer, *Introduction to System Theory*. McGraw-Hill, New York (1972).

APPENDIX A

Bocher's formula

From the Cayley–Hamilton Theorem, a $n \times$ matrix A satisfies

$$A^n = \alpha_{0,0}I + \alpha_{1,0}A + \ldots + \alpha_{n-2,0}A^{n-2} + \alpha_{n-1,0}A^{n-1}$$

then

$$\begin{aligned} A^{n+1} = A^n \cdot A &= (\alpha_{0,0}I + \alpha_{1,0}A + \ldots + \alpha_{n-1,0}A^{n-1})A \\ &= (\alpha_{0,0}A + \alpha_{1,0}A^2 + \ldots) + \alpha_{n-1,0}A^n \\ &= (\alpha_{0,0}A + \alpha_{1,0}A^2 + \ldots) + \alpha_{n-1,0}(\alpha_{0,0}I + \alpha_{1,0}A + \ldots + \alpha_{n-1,0}A^{n-1}). \end{aligned}$$

Collecting terms, and defining A^{n+1} to be

$$A^{n+1} = \alpha_{0,1} I + \alpha_{1,1} A + \ldots + \alpha_{n-1,1} A^{n-1}$$

one obtains

$$\alpha_{0,1} = \alpha_{n-1,0} \cdot \alpha_{0,0}$$

$$\alpha_{1,1} = \alpha_{0,0} + \alpha_{n-1,0} \cdot \alpha_{1,0}$$

$$\vdots$$

$$\alpha_{n-1,1} = \alpha_{n-2,0} + \alpha_{n-1,0} \cdot \alpha_{n-1,0}.$$

Continuing in this manner, the following inductively follows

$$A^{n+m} = \sum_{i=0}^{n-1} \alpha_{i,m} A^i$$

where

$$\alpha_{0,m} = \alpha_{n-1,m-1} \alpha_{0,0}$$

$$\alpha_{1,m} = \alpha_{0,m-1} + \alpha_{n-1,m-1} \alpha_{1,0}$$

$$\vdots$$

$$\alpha_{n-1,m} = \alpha_{n-2,m-1} + \alpha_{n-1,m-1} \alpha_{n-1,0}.$$

APPENDIX G

TRANSIENT RESPONSE ANALYSIS ON MINI COMPUTERS

INTRODUCTION

Controls engineers have through the years, found the computer to be a valuable, if not indispensable tool. He has found ways to harness the power of both analog and digital devices. Hundreds of thousands of hours have been devoted to the study of control systems by computer methods. Simulation has become the rule rather than the exception. Many numerical integration techniques have been developed to accomplish the required simulation. In a recent work of J. Reitman, he noted [1].

> "Historically, the control system engineers developed the simulation methodology as an adjunct to the development of analog computers. Now extensive use is made of digital and hybrid digital-analog computer systems. To make the transition from analog to digital computers easier, a number of digital computer simulation languages have evolved: Mimic [2], [3], Midas [4], Pactolus [5], CSMP [6], [7], and CSSL [8]-[10]"

These simulations are performed on large to medium size computers. The recent availability of mini computers however has had very little impact in simulating the response of large dynamical systems. This is unfortunate when one considers the wealth of interactive I-O devices which would make simulation a highly animated experience. There are several reasons why the mini has fallen short as a simulation tool. Many of them are based on economics. However, from a technical point of view it can be noted from those who have attempted to do system simulation on a mini, become aware that a 16 bit word is often too small

to avoid the "curse" of numerical instability. Here a double precision 32 bit word (floating point) often numerically truncates the answer at each iteration of a numerical integration scheme so as to cause the answer to diverge or become a poor representation of the true solution. Of course, high level languages can be written to work with a 4 word (64 bit) floating point format. However, for the purpose of simulation the resultant routine is prohibitively slow. To overcome this problem a numerically stable and/or speed, an algorithm is presented which will overcome this cited numerical problem of conventional integration methods and thereby make it suitable for mini computer use. It is especially designed to efficiently compute the response of a linear constant coefficient control system to the common test inputs of (1) as step (2) a ramp, and (3) a constant acceleration input.

STATE VARIABLE MODEL

Let it be assumed that the control system under consideration satisfies the following state variable equations:

plant

$$\overset{o}{x}(t) = A\,x(t) + bu(t) : x(0) = x_o \qquad (1)$$

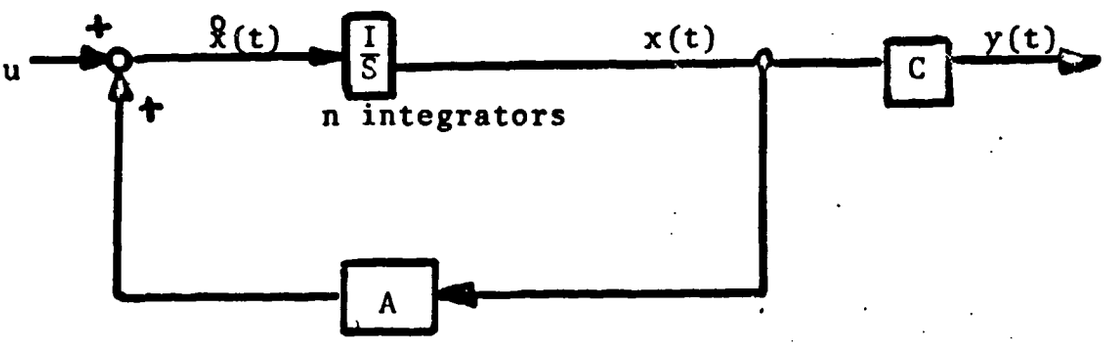
with observations

$$y(t) = Cx(t) \qquad (2)$$

where "o" means d/dt and x(t) and y(t) are n and m vectors respectively. The control u(t) is to be considered to be one of the following conventional test inputs.

$$u(t) = \left\{ \begin{array}{l} 0 = y(t) = \text{unforced response (impulse response)} \\ C = y(t) = \text{step response} \\ Ct = y(t) = \text{ramp response} \\ \frac{Ct^2}{2} = y(t) = \text{acceleration response} \end{array} \right\} \quad (3)$$

The proposed system is diagrammed in Figure 1.



Consider for the moment the following special cases

a) $u(t) = 0$

then the system equations become

$$\dot{x}(t) = A\,x(t),\ y(t) = C\,x(t) \quad (4)$$

b) $u(t) = c$

defining $x_{n+1}(t) = u(t) = c$, and noting

$$\dot{x}_{n+1}(t) = 0,\ x_{n+1}(0) = c$$

the state equations can be written as

$$\begin{bmatrix} \dot{x}(t) \\ \dots \\ \dot{x}_{n+1}(t) \end{bmatrix} = \left[\begin{array}{c:c} A & b \\ \hdashline 0 & 0 \end{array}\right] \begin{bmatrix} x(t) \\ \dots \\ x_{n+1}(t) \end{bmatrix}, \quad \begin{bmatrix} x(0) \\ \dots \\ x_{n+1}(0) \end{bmatrix} = \begin{bmatrix} x_o \\ \dots \\ c \end{bmatrix} \quad (5)$$

and $y(t) = [C \quad 0] \begin{bmatrix} x(t) \\ x_{n+1}(t) \end{bmatrix}$ (6)

c) $u(t) = ct$

defining $x_{n+1}(t) = u(t) = ct$

$$x_{n+2}(t) = \dot{x}_{n+1}(t) = c, \; x_{n+1}(0) = 0$$

and noting

$$\dot{x}_{n+2}(t) = 0, \; x_{n+2}(0) = c$$

the state equations becomes

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_{n+1}(t) \\ \dot{x}_{n+2}(t) \end{bmatrix} = \left[\begin{array}{c|c|c} A & b & 0 \\ \hline 0 & 0 & 1 \\ \hline 0 & 0 & 0 \end{array}\right] \begin{bmatrix} x(t) \\ x_{n+1}(t) \\ x_{n+2}(t) \end{bmatrix}, \quad \begin{bmatrix} x(0) \\ x_{n+1}(0) \\ x_{n+2}(0) \end{bmatrix} = \left[\begin{array}{c} x_o \\ \hline 0 \\ \hline c \end{array}\right] \quad (7)$$

and $y(t) = [C \quad 0 \quad 0] \begin{bmatrix} x(t) \\ x_1(t) \\ x_2(t) \end{bmatrix}$ (8)

d) $u(t) = ct^2/2$

defining $x_{n+1}(t) = u(t) = ct^2/2$

$$x_{n+2}(t) = \dot{x}_{n+1}(t) = ct, \; x_{n+1}(0) = 0$$

$$x_{n+3}(t) = \dot{x}_{n+2}(t) = c \; , \; x_{n+2}(0) = 0$$

and noting

$$\dot{x}_{n+3}(t) = 0, \; x_{n+3}(0) = c$$

the state equation becomes

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_{n+1}(t) \\ \dot{x}_{n+2}(t) \\ \dot{x}_{n+3}(t) \end{bmatrix} = \left[\begin{array}{c|c|c|c} A & b & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 \end{array}\right] \begin{bmatrix} x(t) \\ x_{n+1}(t) \\ x_{n+2}(t) \\ x_{n+3}(t) \end{bmatrix} + \begin{bmatrix} x(0) \\ x_{n+1}(0) \\ x_{n+2}(0) \\ x_{n+3}(0) \end{bmatrix} + \begin{bmatrix} X_o \\ 0 \\ 0 \\ c \end{bmatrix} \tag{9}$$

and

$$y(t) = [C \quad 0 \quad 0 \quad 0] \begin{bmatrix} x(t) \\ x_{n+1}(t) \\ x_{n+2}(t) \\ x_{n+3}(t) \end{bmatrix} \tag{10}$$

The equations found in (a) thru (d) are called the "augmented state equations" and can be generalized in terms of an

(i) augmented state vector, say X

(ii) augmented plant matrix, say A

(ii) augmented observation matrix, say C

such that

$$\dot{X}(t) = A X(t) \tag{11}$$

$$y(t) = C X(t) \tag{12}$$

Equations of the form of this can be solved through direct application of the "state transition matrix" or, as it is often called, the "matrix exponential", [11], [12]. This matrix is denoted

$$\exp(At) \tag{13}$$

defines the solution of (11) to be

$$X(t) = \exp(At)\, X(0) \qquad (14)$$

Several methods have been proposed which allow the user to compute the required matrix exponential. They are:

1. Liou method [13]
2. $(sI-A)^{-1}$ method [14]
3. matrix inversion Lemma method [15]
4. Davison method [16]

It is the Davison method, due to its numerical stability, which is particularly attractive. It generates the following closed form representation of exp (At) over some given step size h, namely.

$$\exp(Ah) \tilde{=} \left(I - \frac{hA}{2} + \frac{h^2A^2}{12}\right)^{-1} \left(I + \frac{hA}{2} + \frac{h^2A^2}{12}\right) \qquad (15)$$

The ability to define the matrix exponential over some small interval h is not a handicap in that it is well-known that

$$\begin{aligned} \exp(A2h) &= \exp(Ah) * \exp(Ah) \\ \exp(A3h) &= \exp(A2h) * \exp(Ah) \end{aligned} \qquad (16)$$

and so on. In general, $X(\ell h)$ can be computed recursively as follows

$$\begin{aligned} X(\ell h) &= \exp(A\ell h)X(0) \\ &= \exp(A(\ell - 1)h) \exp(Ah) X(0) \end{aligned} \qquad (17)$$

recursive

An efficient technique using Bochers formula has been published by Taylor [17].

However, those performing simulation on a dedicated mini are usually interested in resolving the trajection compactly in time during the transient period and allowing the resolution along the time axis "slip" as the solution approaches steady state. Of course, it is

assumed that the amplitude accuracy will be preserved independent of the time axis resolution. The algorithm proposed is extremely useful for such applications in that it can accelerate along the time axis at a very rapid rate. That is, suppose the current solution is

$x(\ell h)$, where

$$x(\ell h) = \exp(A\ell h)\, x(0)$$

and $y(\ell h) = C\, X(\ell h)$

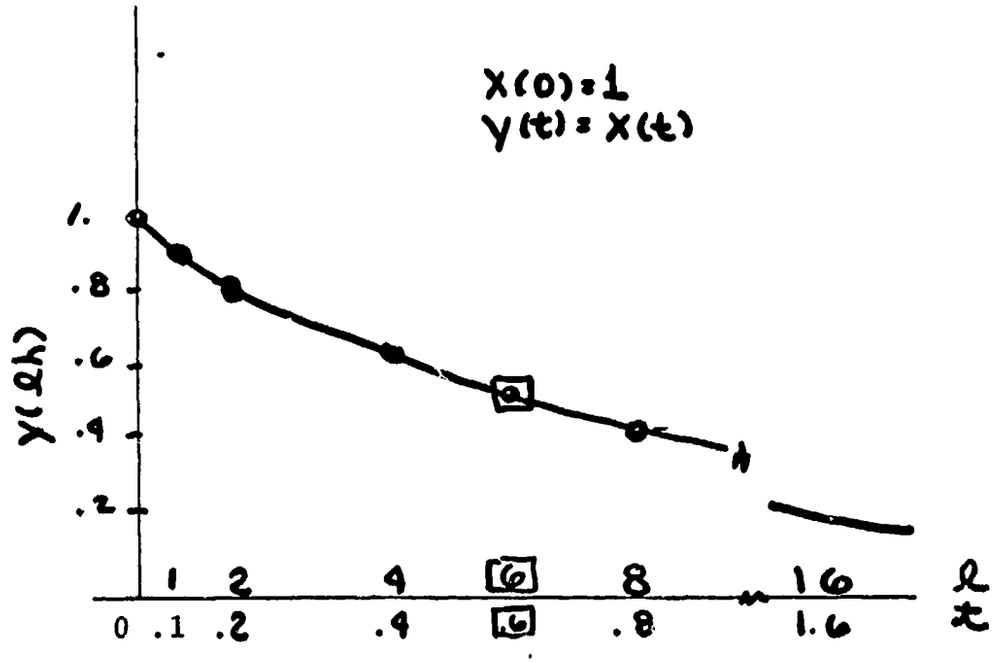
If $\exp(A\ell h)$ is stored, then the solution as $2\ell h$ can be obtained by simply forming

$$\exp(A2\ell h) = \exp(A\ell h) * \exp(A\ell h) \qquad (18)$$

and generating

$$X(2\ell h) = \exp(A2\ell h)\, X(0)$$

In general, after M operations, starting at $x(0)$, $X(2^{M-1}h)$ can be obtained. Thus the time axis can be scanned in base 2. As a byproduct, it can be noted that for M small, the solution is highly refined near $t = 0$. As M increases, (suppose the trajections are asymptomatically stable) the time axis in the steady-state region is coarsely refined. For example, consider the simple process $\dot{x}(t) = -x(t)$, which for h chooses to be .1 seconds, has the response $x(\ell h) = \exp(-\ell h)xo$ (see Fig. 2)



Base 2 algorithm

fig 2

Suppose that further amplitude resolution is desired. This could be achieved as follows (see Fig. 2). Suppose the solution currently resides at $\ell = 8$ ($t = .8$) which was the result of operating on the previously computed exp (4ℓ) with itself. Suppose exp (2ℓ) remained in memory, this would allow the user to compute exp $(6\ell)^2$ exp (4ℓ) = exp (2ℓ) and so on. The algorithm present in this work allocates additional storage (called the "push - down" stack) to allow the user to create a more dense solution space than available with the direct base 2 algorithm.

One last feature of the proposed algorithm shall be

developed. From the theory of infinite matricis, it can be shown that the steady state step response of the original system (equation 1 and 2) is given by

$$y(\text{Steady-State}) = C[I - \exp(Ah)]^{-1} bh\ [\ ]. \qquad (19)$$

Therefore, if one is simulating a step response, he may also choose to resolve his answer in terms of a percent of the steady state response. That is, suppose

$$\dot{x}(t) = -x(t) + 1(t)\ ,\ x(0) = 0$$

$$y(t) = x(t)$$

Therefore $y(t) = 1 - \exp(-t)$ or $y(\text{steady state}) = 1$. Also, if $h = .01$ then the steady state approximating equation yields an answer of $1.005008333 \cong 1$. If the user desires the amplitude resolved to at least 10% of the steady state value then he would require that no two adjacent amplitudes, say $y(\ell h)$ and $y(mh)$, where m is the successor of ℓ, differ by no more than 10% of 1, or .1. Suppose $\exp(A\ell)$, $\exp(A(\ell-1))$, through $\exp(A(\ell-r))$ are stored in core. One would test a base 2 "jump" to $y(2\ell)$. If $|y(2\ell) - y(\ell)| > .1$ then generate $y(\ell+\ell-1) = \exp(A\ell) * \exp(A(\ell-1))\ x(0)$. If $|y(\ell+\ell-1) = y(\ell)| > .1$, test $y(\ell)$ against $y(\ell+\ell+2)$ and so on and see figure 3.

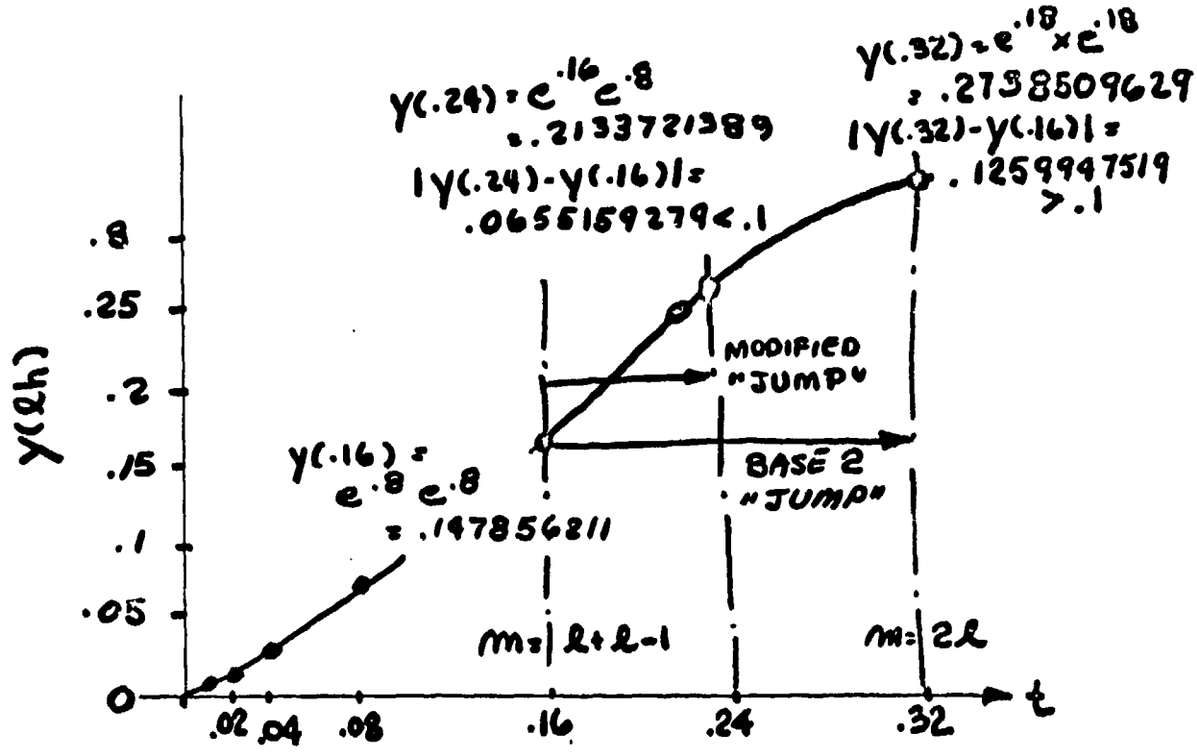

y(.24) = e^.16 e^.8
= .2133721389
|y(.24)-y(.16)| =
.0655159279 < .1
y(.32) = e^.18 × e^.18
= .2738509629
|y(.32)-y(.16)| =
.1259947519
> .1
MODIFIED
"JUMP"
BASE 2
"JUMP"
y(.16) =
e^.8 e^.8
= .1478562811
m = l+l-1
m = 2l
y(2h)
.8
.25
.2
.15
.1
.05
0
.02
.04
.08
.16
.24
.32
t

fig 3

Example

The following simple example shall be used to demonstrate the salient features of the developed algorithm. A second order underdamped system is considered and is given by

i) $$\frac{d^2z(t)}{dt^2} + 1.5\frac{dz(t)}{dt} + z(t) = 1(t)$$

$$z(0) = 0 \quad , \quad dz(0)/dt = 0$$

ii) $$y(t) = x(t)$$

The solution is known to be given by

$$y(t) = 1 - \frac{2}{\sqrt{3}} \sin\left(\frac{\sqrt{3}t}{2} + 60^{\circ}\right)$$

The state representation of the above system is

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -1 & -1.5 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} 1(t) = Ax + bu(t)$$

$$x(t) = \begin{bmatrix} z(t) \\ dz(t)/dt \end{bmatrix}, \quad x(0) = [0]$$

$$y(t) = [1 \quad 0]\, x(t) = Cx(t)$$

The augmented state variable representation which simulates a step response is (see equation (5))

$$\overset{\circ}{X}(t) = AX(t) = \begin{bmatrix} 0 & 1 & 0 \\ -1 & -1.5 & 1 \\ 0 & 0 & 0 \end{bmatrix} X(t) \;,\; X(0) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$y(t) = CX(t) = [100]\, X(t)$$

The developed algorithm requires the user specify matrices A, C, and the vector $X(0)$. If a modulo 2 simulation is desired, the user must supply

a) step size Dt

b) terminal time

c) set the size of the push-down stack to one "1".

If a modified modulo 2 simulation is desired, the user must supply

a) step size Δt

b) terminal time

c) choose desired size of push-down stack

d) choose desired percent resolution

e) members of x(t) to be resolved.

It should be noted that the resolution test on state $x_i(t)$ is based on a given percent of the steady state value of that state. Therefore, only those states which result in a finite steady state value should be considered as candidates to be resolved beyond the modulo 2 resolution.

The problem under study shall be approached two ways. A modulo-2 and modified modulo-2 solution shall be presented. For the example under consideration, the steady state value of $x_1(t)$ and $x_2(t)$ (using equation (19)) was found to be .999928 $\cong$ 1 and 8.40455 x 10 -3 $\cong$ 0 respectively. The state $x_1(t)$ shall be chosen to be resolved with 15% of its steady state value. That is, no two successive values of the modified modulo-2 simulation will differ by at most .15. The results of these two simulations are summarized in figure 4. It should be noted that modulo-2 algorithm is an accurate, but coarse, representation of y(t); whereas the modified modulo-2 algorithm is a vastly superior representation of the actual y(t). A more highly refined algorithm is required by

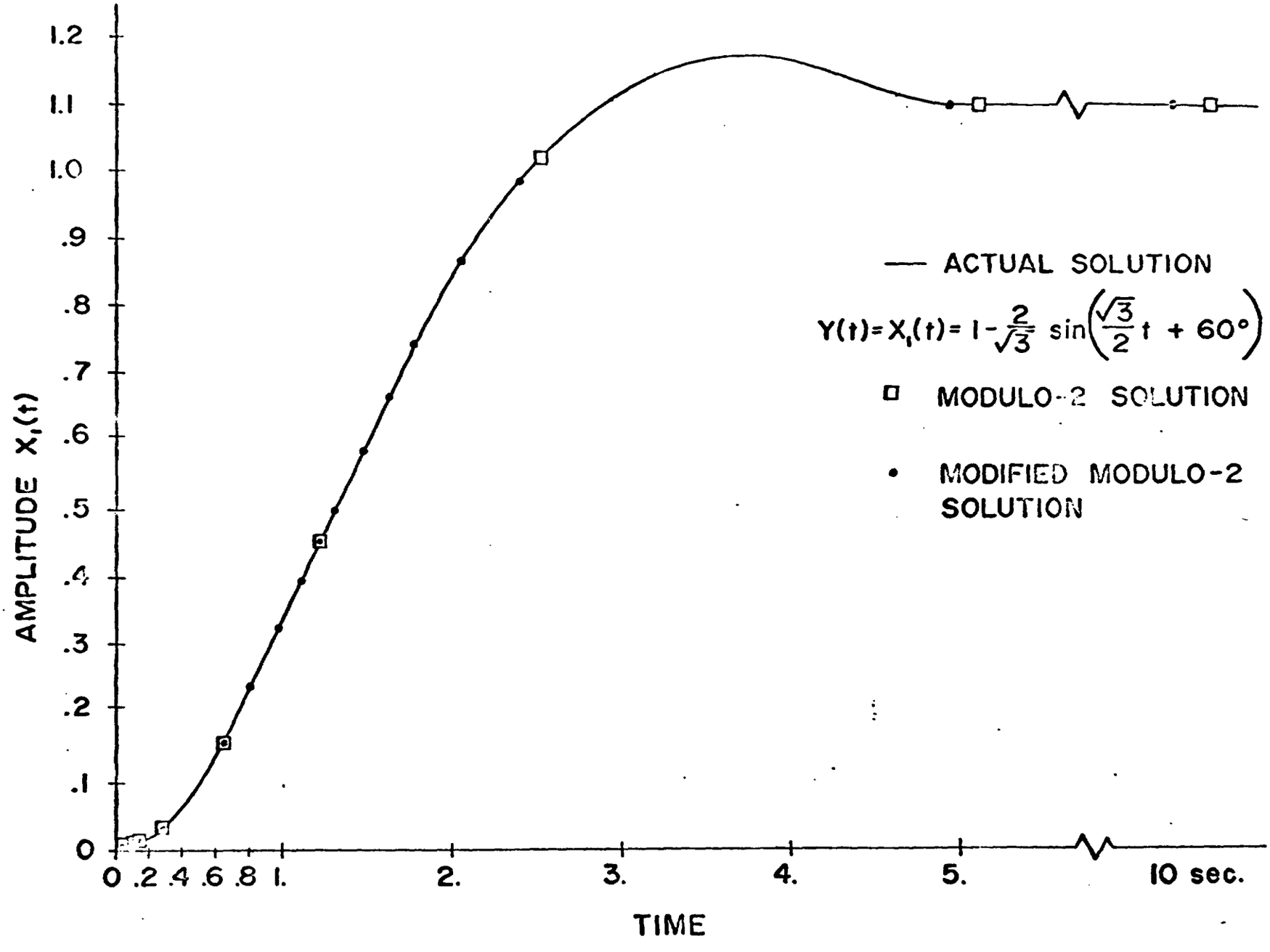
increasing the size of the push-down stack and decreasing the percent resolution value. If the "knee" about the point of maximum overshoot is to be further resolved, a special routine could be written. This routine would give $x_1(t)$ additional refinement whenever the slope of $x_1(t)$ changes signs. That is, a local minima or maxima is known to occur of points where the slope of that once passes through zero. With respect to the example problem, the region between sign changes of $x_2(t)$ (note $x_2(t) \equiv dx_1(t)/dt$), namely $2.4 \leq t \leq 4.8$ would be further resolved.

Besides being numerically stable, the algorithm was found to be considerably faster than conventional numerical integration formulas. The speed increase is due to the algorithm's ability to "leap frog" along the time axis. In the case of the modified base-2 algorithm, the "leap frogging" moves in ever-increasing step sizes provided the percent resolution test is not violated. In the considered example for a basic step size of .01 seconds, to complete a solution over $0 \leq t \leq 10$, it takes

i) 1000 solution steps by Runge-Kutta methods
ii) 18 solution steps by modified base-2 methods
iii) 9 solution steps by base-2 methods.

It should be now self-evident why the algorithm is fast, therefore, an economic design tool.

Fig 4


1.2
1.1
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
AMPLITUDE $X_1(t)$
0 .2 .4 .6 .8 1. 2. 3. 4. 5. 10 sec.
TIME
— ACTUAL SOLUTION
$Y(t)=X_1(t)=1-\frac{2}{\sqrt{3}}\sin\left(\frac{\sqrt{3}}{2}t+60°\right)$
□ MODULO-2 SOLUTION
• MODIFIED MODULO-2 SOLUTION

DESCRIPTION OF SOFTWARE

The following program is written in Extended Basic and was implemented on a Hewlett-Packard 2100-S minicomputer. Its use was described in the previous section. Briefly, the generation of $\exp(Ah)$ (matrix exponential approximation) is generated between statement 400 and 530. If needed, the steady state value of x(t) is generated between statement 645 and 685. From statement 890 to 930, the state $y(\ell) = C \exp(A\ell h) X(0)$ is produced. If an error tolerance is requested, it will be tested from statement 940 to 990. If the error tolerance is violated, it is sent to statement 2000 for resolvement. Statements 1110 to 1610 are dedicated to restacking the push down stack. The program will loop back to 900 until the terminal time is achieved. Once achieved, the program will process statement 1665 to 1780 to complete the solution. The initial parameters set consisting of A, C, and $X(0)$ are centered as data statements located at and beyond statement 300.

```
LIST
10  REM THE FOLLOWING DIMENSION MUST BE CONSISTENT WITH THE PROBLEM
20  DIM X[3],Y[3],Z[2],V[2],O[2],I[2],E[2,2],F[2,2],G[2,2],B[2,2]
30  DIM A[3,3],C[3,3],H[2,3],R[3,3],S[3,3],W[3,3]
40  REM DUMMY ARRARYS
50  DIM T[100,10],Q[250],U[10,250],L[10],D[10]
55  PRINT "DIMENSION OF STATE SPACE"
60  INPUT N6
65  PRINT "DIMENSION OF AUGMENTED STATE SPACE"
70  INPUT N
80  PRINT "DIMENSION OF MESSAGE SPACE?"
90  INPUT N5
100  PRINT "STEP SIZE IN SECONDS?"
110  INPUT D
120  PRINT "TERMINAL TIME IN SECONDS?"
130  INPUT F
140  PRINT "IF PLANT IS STABLE****RESPOND 1 (ONE)"
150  INPUT T9
160  IF T9#1 THEN 190
170  PRINT "PERCENT RESOLUTION DESIRED?"
180  INPUT E
181  FOR I=1 TO N5
182  PRINT "RESOLVE Y(";I;")? IF SO RESPOND";I;"0 OTHERWISE"
183  INPUT D[I]
184  NEXT I
190  PRINT "DESIRED SIZE OF PUSH DOWN STACK?"
195  INPUT N1
200  MAT  READ X
205  MAT  READ A
210  MAT  READ H
230  PRINT "INITIAL AUGMENTED STATE VECTOR"
240  MAT  PRINT X
250  PRINT "AUGMENTED PLANT MATRIX"
260  MAT  PRINT A
270  PRINT "OBSERVATION MATRIX"
280  MAT  PRINT H
290  REM LABLES 300 TO 390 ARE RESERVED FOR DATA STATEMENTS
300  DATA 0,0,1
310  DATA 0,1,0
320  DATA -1,-1,1
330  DATA 0,0,0
340  DATA 1,0,0
350  DATA 0,1,0
400  REM COMPUTE C
410  MAT R=A*A
420  MAT R=(D*D/12)*R
430  MAT S=(D/2)*A
440  MAT C=R+S
450  MAT S=R-S
460  MAT R=IDN[N,N]
470  MAT S=S+R
480  MAT C=C+R
490  MAT W=INV(R)
500  MAT R=W*C
```

```
500  MAT R=W*C
510  MAT C=R
520  PRINT "MATRIX EXPONENTIAL APPROXIMATION"
521  GOTO 540
530  MAT  PRINT C
540  REM PERCENT RESOLUTION TEST
550  IF T9#1 THEN 700
600  FOR K=1 TO N6
605  FOR J=1 TO N6
610  LET E[K,J]=C[K,J]
615  NEXT J
620  NEXT K
625  MAT F=IDN[N6,N6]
630  MAT G=F-E
635  PRINT "IF MATRIX IS SINGULAR ERROR CODE WILL FOLLOW"
640  PRINT " REPEAT PROBLEM WITHOUT ERROR TOLLERANCE"
645  MAT E=INV(G)
650  FOR K=1 TO N5
655  LET S=0
660  FOR J=1 TO N6
665  LET S=H[K,J]*E[J,N6]+S
670  NEXT J
675  LET V[K]=S*D*.5
680  PRINT "APPROX STEADY STATE VALUE OF Y(";K;")=";V[K]
685  NEXT K
700  MAT T=ZER
710  MAT Z=ZER
720  MAT L=ZER
750  MAT R=C
755  MAT I=H*X
760  FOR K=1 TO N5
770  LET U[K,1]=I[K]
780  NEXT K
790  LET Q[1]=D
800  LET L[1]=D
810  REM M1 IS THE COMPLETED SAMPLE COUNTER
820  REM N2 IS AN EXHAUSTIVE STACK COUNTER
830  REM N3 IS A MODULO 10 COUNTER
840  FOR K=1 TO N
850  FOR L=1 TO N
860  LET T[K,L]=C[K,L]
870  NEXT L
880  NEXT K
890  LET M1=N2=1
900  LET N3=0
910  MAT S=R*R
920  MAT Y=S*X
930  MAT I=H*Y
940  IF T9#1 THEN 1100
950  MAT O=Z-I
960  FOR K=1 TO N5
965  IF K#D[K] THEN 990
970  LET S1=ABS(O[K]*100/V[K])
980  IF S1 >= E THEN 2000
990  NEXT K
1000  MAT Z=I
1100  LET T1=L[1]+L[N3+1]
1110  REM RESTACK DATA
```

```
1120  FOR K=1 TO N1-1
1130  LET L(N1+1-K)=L(N1-K)
1140  NEXT K
1150  FOR K=1 TO N1-1
1160  LET K1=(N1-K)*10+1.00000E-04
1170  LET K1=INT(K1)
1180  FOR K3=1 TO N
1190  FOR K4=1 TO N
1200  LET T(K1+K3,K4)=T(K1-10+K3,K4)
1210  NEXT K4
1220  NEXT K3
1230  NEXT K
1500  FOR K=1 TO N
1510  FOR L=1 TO N
1520  LET T(K,L)=S(K,L)
1530  NEXT L
1540  NEXT K
1550  LET M1=M1+1
1560  MAT I=H*Y
1570  FOR K=1 TO N5
1580  LET U(K,M1)=I(K)
1590  NEXT K
1600  LET Q(M1)=T1
1610  LET L(1)=T1
1650  MAT R=S
1660  IF T1<F*2 THEN 900
1665  LET Q(1)=0
1670  FOR I=2 TO M1
1675  LET Q(I)=Q(I)/2
1680  NEXT I
1700  PRINT "IN TUBULAR FORM THE OUTPUT STATES (MESSAGE) ARE"
1710  FOR L=1 TO M1
1720  PRINT "TIME";Q(L)
1730  FOR K=1 TO N5
1740  PRINT U(K,L)
1750  NEXT K
1760  NEXT L
1770  PRINT "FINIS"
1780  STOP
2000  IF M1 >= N1 THEN 2040
2010  REM SET ALLOWABLE STACK PARAMETERS
2020  IF M1-N3 <= 0 THEN 3000
2040  IF N3 >= N1-1 THEN 4000
2050  FOR K=1 TO N
2060  FOR L=1 TO N
2070  LET W=INT(10*N3+10.001)
2075  LET S=0
2080  FOR J=1 TO N
2085  LET S=T(W+K,J)*T(J,L)+S
2090  NEXT J
2095  LET R(K,L)=S
2100  NEXT L
2105  NEXT K
2200  MAT S=R
2210  LET N3=N3+1
2220  GOTO 920
```

```
3000  PRINT "STACK DEPLETED AT STEP",M1
3010  PRINT "LAST ERROR MAGNITUTE WAS",S1
3020  PRINT "DO YOU WISH TO CONTINUE WITH NEW ERROR TOLLERANCE"
3030  PRINT " IF SO RESPOND 1(ONE)"
3040  INPUT F8
3050  IF F8=1 THEN 3100
3060  STOP
3100  PRINT " ENTER NEW PRECENT ERROR "
3110  INPUT E
3120  GOTO 900
4000  PRINT "STACKDEPLED AT STEP",M1
4010  PRINT "THE LAST ERROR MAGNITUDE WAS",S1
4020  PRINT "NEW STACK SIZE*****PREVIOUS SIZE";N1
4030  PRINT "OR MAXIMAL STACK SIZE****IF SO RESPON 1(ONE)"
4040  INPUT F8
4050  IF F8=1 THEN 4100
4060  STOP
4100  PRINT "NEW ERROR TOLERANCE"
4110  INPUT E
4120  PRINT "NEW STACK SIZE*****PREVIOUS SIZE",N1
4130  INPUT N1
4140  GOTO 900
5000  END

STOP
```

References

1. J. Reitman, "The Engineering Role of Simulation," IEEE Transactions on Systems, Man, and Cybernetics, March 1974.

2. H. E. Peterson and F. J. Sansom, "MIMIC-- a digital simulator program," Wright-Patterson AFB, Ohio, SECA Internal Memo 65-12, May 1965.

3. F. S. Sansom and H. E. Peterson, "MIMIC programming manual," Wright-Patterson AFB, Ohio, Tech. Rep. SEG-TR-67-31, July 1967.

4. R. T. Harnett, F. J. Sansom, and L. M. Warshowsky, "MIDAS, an analog approach to digital computation," Simulation, Vol. 3, May 1964.

5. R. D. Brennan, "PACTOLUS, a simulator language which makes a digital computer feel like an analog computer," Simulation, Vol. 6, August 1964.

6. R. D. Brennan and M. Y. Silberberg, "Two continuous system modeling programs," IBM Syst.J., Vol. 6, No. 4, 1967.

7. System/360 Continuous System Modeling Program User's Manual, IBM Form #H20-0367, 1969.

8. Simulation Councils Inc. Software Committee, "The SCI continuous system simulation language (CSSL)," Simulation, Vol. 9, No. 6, 1967.

9. A. B. Trevor and J. V. Wait, "DARE IIIb--a CSSL-type batch-mode simulation language for CDC 6000 series computer," Simulation, Vol. 18, No. 6, 1972.

10. Continuous System Language III (CSSL 3) User's Guide, Control Data Corp., Publ. 17304400, 1971.

11. DeRusso, Roy and Close, State Variables for Engineers, Wiley.

12. Eveleigh, Control System Design, McGraw-Hill.

13. M. L. Liou, "A novel method of evaluating transient response", Proc. of the IEEE, January 1966.

14. Gupta and Hasdorff, Fundamentals of Automatic Control, Wiley.

15. F. Taylor, "A novel inversion of (sI - A)", Int. J. of Systems Sci., Vol. 5, No. 2, 1974.

16. E. J. Davison, "A high-order Crank-Nicholson technique for solving differential equations", Compt. J., August 1967.

17. F. Taylor, "Comments on 'The Solution of the Matrix Ricatti Differential Equation'", IEEE Trans. on Automatic Control, April 1974.