SEQUENTIAL ESTIMATION

and

PATTERN RECOGNITION

by

CHAO - HUAN HUANG

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF ENGINEERING

THE UNIVERSITY OF TEXAS AT EL PASO

May 1974

UMI Number: EP01226

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

SEQUENTIAL ESTIMATION

and

PATTERN RECOGNITION

APPROVED BY SUPERVISORY COMMITTEE

Fred J. Taylor

CHAIRMAN

Jack Smith

E. A. Dean

Rudolph Gomez

DEAN OF GRADUATE SCHOOL

This work was supported in part by contract #DAAD07-74-C-0103 from the Atmospheric Sciences Laboratory, White Sands Missile Range, New Mexico.

ACKNOWLEDGEMENTS

The Author is very much indebted to Dr. Fred Taylor for his invaluable guidance and assistance during the preparation of this work. Thanks to Dr. Jack Smith and Dr. Alan Dean for criticisms and contributions. And also to Mr. Wang Tang and Miss Mei-Yun Wu for help.

And finally thanks to Miss Lallie Scott for correcting and typing the thesis and Mrs. B. Hallander for office help.

Abstract

Kalman filtering methods are known to be the optimal method of processing noise corrupted data belonging to a linear system. In this work, sequential and finite fading memory Kalman filter theory is developed and extended to the problem of estimating the operation curves in power systems for economic dispatch. And also to process the optical scintillation data for the purpose of remote wind speed sensing numerical experimentation which is performed on field recorded power and scintillation data. The result of this experiment is quantitatively and qualitatively analyzed.

TABLE OF CONTENTS

1.0 INTRODUCTION

Linear minimal variance filtering has been of interest to system engineers since the early 1960's. Kalman gave a set of recursive equations for estimating the state of a linear dynamic system, provided that the noise statistics corrupting the system state's can be specified [1].

However, the problem of divergence can occur when these filters are implemented [2]. To solve this problem, a filter called "Finite Fading Memory Filter," Taylor [3], is developed, which combines the policy of "Fading Memory Filter" of Sorenson [4] and "Finite Memory Filter" of Jazwinski [5]. This filter provides two degrees of design freedom to choose, namely δ, a discount factor and M, the length of the memory window. The filter developed is also sequential and can therefore be used for real time data processing.

In part one, the Kalman filter is used to estimate the coefficients of heat rate curves of power systems for economic dispatch. Numerical results are compared with the results obtained from the formulas used by the El Paso Electric Company and that obtained by using the 4th order polynomial curve fitting technique of Dallas Power and Light Company.

In part two, a Kalman filter is configured as a supervised learning machine. This work leads to a recursive remote wind speed measurement system based upon estimates of the spectral signature of a coherent beam propagating through a turbulent media. Numerical estimates of wind speed are produced from field derived data and the results are compared.

2.0 THEORY

2.1 The Kalman Sequential Filter

Let us consider a system whose state as function of time is an n-dimensional discrete-time process x(k)

$$x(k + 1) = \Phi(k + 1, k)\ x(k) \qquad (1) \quad x(k) \epsilon E^n$$

Suppose that we are interested in knowing the value of x(k) for some fixed k, but that x(k) is not directly accessible; suppose further we have a noise corrupted observation model

$$y(k) = H(k)\ x(k) + \nu(k) \qquad (2) \quad \nu(k) \epsilon E^m$$

and now from these observations, we wish to infer the value of x(k).

Since only the measurement y(1) . . . y(j) is available from which to estimate x(k), intuitively, we will define the estimate of x(k) base on j observations as

$$\tilde{x}(k/j) = \Phi_k(y(i),\ i=1,\ .\ .\ .\ j)$$

and also define the error measurement $\tilde{x}(k/j)$

$$\tilde{x}(k/j) = x(k) - \tilde{x}(k/j)$$

where x(k) is the true state value at stage k.

Since x(k) is a random process, the problem therefore becomes: "Given the observations y(1) . . . y(j), determine a estimate $\tilde{x}(k/j) = \Phi_k(y(1),\ .\ .\ .\ y(j))$ of x(k), such that it will minimize the objective function $E[\tilde{x}(k/j)\ \tilde{x}^T(k/j)]$." The reason for choosing $E[\tilde{x}(k/j)\ \tilde{x}^T(k/j)]$ is because x(k) is a random process, and we want to minimize the variance of the error process $\tilde{x}(k/j)$.

A set of equations was derived by Kalman [6] in 1960. These algorithms are listed below and it is often called Kalman filter or minimum variance filter. First assume v(k) is a white noise process with

$$E[v(k)] = 0$$

$$\mathrm{cov}[v(j)v^T(k)] = R_v(k) \quad (k-j)$$

$$\mathrm{cov}[x(o)v^T(k)] = 0$$

$$E[x(o)] = x_o$$

$$\mathrm{Var}[x(o)] = E[x(o)x^T(o)] = P_o \qquad (n \times n) \tag{3}$$

The Kalman filter algorithm:

$$\hat{x}(k/k) = [A^T(k)\bar{R}_v^{-1}(k)A(k)]^{-1} A^T(k) \cdot \bar{R}_v^{-1}(k)y(k) \tag{4}$$

where

$$A(k) = \begin{bmatrix} H(1) & \Phi & (1,k) \\ & \vdots & \\ H(k-1) & \Phi & (k-1,\ k) \\ H(k) & \Phi & (k,k) \end{bmatrix} \qquad (m\cdot k) \times n$$

$$\bar{R}v(k) = \begin{bmatrix} Rv(1) & & \\ & \ddots & \\ & & Rv(k) \end{bmatrix}_{(m\cdot k) \times (m\cdot k)} \qquad \bar{Y}(k) = \begin{bmatrix} Y(1) \\ \vdots \\ Y(k) \end{bmatrix}_{(m\ k) \times 1}$$

According to the assumption that V(k) is a white noise then $\hat{x}(k/k)$ can be accomplished sequentially as follows

$$\hat{x}(k/k) = \phi(k,k-1)\hat{x}(k-1/k-1) + k(k)[Y(k) - H(k)\phi(k,k-1)\hat{x}(k-1/k-1)] \tag{5}$$

The Kalman gain

$$k(k) = P[k/k-1]\ H^T(k)\ [R_v(k) + H(k)P(k/k-1)\ H^T(k)]^{-1} \tag{6}$$

A priori error covariance

$$P[k/k-1] = \phi(k,k-1)\ P(k-1/k-1)\ \phi^T(k,k-1) \tag{7}$$

A posteriori error covariance

$$P[k/k] = P[k/k-1] - k(k)H(k)P(k/k-1) \tag{8}$$

with initial condition x_o and $P(o/o) = P_o$.

In using this Kalman filter algorithm knowledge of the a priori statistics, x_o and P_o and $R_v(k)$ for $k = 0, 1 \ldots k$ are assumed to be known.

Usually if the system is stationary and $R_v(k)$ really represent the noise variance at stage k, for $k = 0, 1 \ldots k$, the estimate of x will converge and the variance of $x - \hat{x}$ (which is the error variance) will approach the steady state value as k becomes large. If the assumptions associated with these "a priori" statistics do not represent the actual observation noise variance, then divergence may result. Divergence means that the difference between the theoretical estimated and actual system values diverge in time.

2.2 The Finite Fading Memory Filtering (F.F.F.)

In order to suppress the divergence, a Finite Fading Memory Filter is developed [3]. This filter is defined through the use of two parameters, they are

i) δ, (a discount factor); δ establishes a discount policy that will weigh previous observation to be "less" important than the current observations

ii) M, (the length of a memory window); in the interval M, data is assumed to be related to current assumptions and all of the data outside of this data window is assumed to be inconsistent with the current statistical assumptions.

For convenience, eq (5) and eq (6) are repeated below

$$\hat{x}(k/k) = \phi(k,k-1)\hat{x}(k-1/k-1) + k(k)[Y(k) - H(k)\phi(k,k-1)\hat{x}(k-1/k-1)] \tag{5}$$

$$k(k) = P(k/k-1)\ H^T(k)\ [R_v(k) + H(k)P(k/k-1)\ H^T(k)]^{-1} \tag{6}$$

As we can see in (6), the Kalman gain is proportional to the inverse of the noise covariance, $R_v(k)$. So, when $R_v(k)$ is large, the estimate of $\hat{x}(k/k)$ may be bad. Therefore, we have less confidence in the innovation process $[Y(k) - H(k)\phi(k,k-1)\hat{x}(k-1/k-1)]$. When $R_v(k)$ is large, $k(k)$ will be small; then the correcting term $k(k)[Y(k) - H(k)\phi(k,k-1)\hat{x}(k-1/k-1)]$ will contribute weakly to the one stage prediction $\phi(k,k-1)\hat{x}(k-1/k-1)$. In other words, the larger the noise covariance (implying high noise corruption), the less desirable it will be to alter the previous estimation.

According to this, a discounting policy is set up as follows. For the past data that we do not think is still consistent with the system, we will artificially increase their error covariance. Let us assume $R_v(\ell)$ is the noise covariance at sample stage ℓ, and $R_v(\ell/k)$ is the discounted noise covariance when system is at stage k and looking backward to stage ℓ.

Define

$$R_v(\ell/k) = \delta^{k-\ell}\ R_v(\ell) \qquad \delta \geq 1 \quad k \geq \ell$$

That is the assumed covariance of the past noise covariance shall be artificially and monotonically increased as shown in Fig. 1-1A.

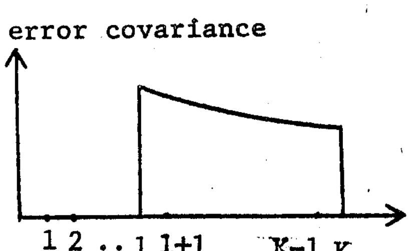

Fig. 1-1A. The new noise covariance when system is at sample k and looking backward to sample ℓ.

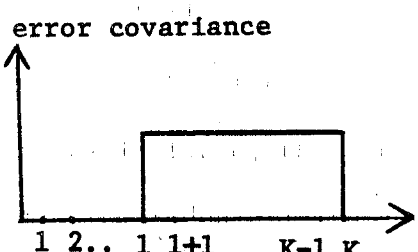

Fig. 1-1B. The original noise covariance.

Fig. 1-1B is the original unchanged noise covariance. Now, with the same noise free linear discrete plant $x(k+1) = \Phi(k+1, k)\, x(k)$

and observation $y(k) = H(k)\, x(k) + v(k)$

The F.F.F. Algorithm will be defined as follows:

One stage estimation

$$\hat{x}(k+1) = \Phi(k+1,k)\hat{x}(k) + k(k+1)\,[z(k+1) - B(k+1)\Phi(k+1,k)\hat{x}(k)] \tag{9}$$

F.F.F. gain

$$k(k+1) = P(k+1/k)\, B^T(k+1)\, [W(k+1) + B(k+1)P(k+1/k)B^T(k+1)]^{-1} \tag{10}$$

F.F.F. A-priori covariance

$$P(k+1/k) = \Phi(k+1,k)\, P(k/k)\Phi^T(k+1,k) \tag{11}$$

F.F.F. A-posteriori covariance

$$P(k+1/k+1) = \delta[P(k+1/k) - k(k+1)\, B(k+1)\, P(k+1/k)] \tag{12}$$

where

$$B(k+1) = \begin{bmatrix} H(k+1) \\ H(k-m)\Phi(k-m,\ k+1) \end{bmatrix} \quad (2m \times n)$$

$$W(k+1) = \begin{bmatrix} \frac{1}{\delta} Rv(k+1/k+1) & 0 \\ 0 & -Rv(k-m/k) \end{bmatrix} \quad (2m \times 2m)$$

$$Z(k+1) = \begin{bmatrix} Y(k+1) \\ Y(k-m) \end{bmatrix} \quad (2m \times 1)$$

PROOF

Define for the nth sample

$$A(\ell/m,n) = \begin{bmatrix} H(\ell-m)\Phi(\ell-m,n) \\ \vdots \\ H(\ell)\Phi(\ell,n) \end{bmatrix} \tag{13}$$

$$V(\ell/m,n) = \begin{bmatrix} R_v(\ell-m/n), & \cdots & 0 \\ 0 & \ddots & \vdots \\ 0 & & 0 \\ 0, & \cdots & R_v(\ell/n) \end{bmatrix} \tag{14}$$

$$Y(\ell/m) = [Y^T(\ell-m), \; . \; . \; . \; . \; , \; Y^T(\ell)]^T \tag{15}$$

Assumed a priori noise covariance be discounted as follows:

$$Rv(j/k) = \delta^{(k-j)} Rv(j/j)$$

$$\text{then } Rv(k/k+1) = \delta \; Rv(k/k) \qquad k>j, \; \delta>1$$

$$\text{so } V^{-1}(k/m-1,k) = \begin{bmatrix} Rv(k-m+1/k), & 0, & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0, & \cdots & 0, & Rv(k/k) \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} \frac{1}{\delta}Rv(k-m+1/k+1), & 0, & \cdots & 0 \\ 0 & & \ddots & \vdots \\ 0 & \cdots & \cdots & \frac{1}{\delta}R(k/k+1) \end{bmatrix}^{-1}$$

$$= \delta V^{-1}(k/m-1, \; k+1) \tag{16}$$

From the projection Lemma [7], define

$$P(j/j) = [A^T(j/m,j) \; V^{-1}(j/m,j) \; A(j/m,j)]^{-1} \tag{17}$$

and

$$A(k/m-1, \; k+1) = A(k/m-1,k) \; \Phi(k,k+1) \tag{18}$$

Now, $P(k+1/k+1) = [A^T(k+1/m, \; k+1) \; V^{-1}(k+1/m, \; k+1) \; A(k+1/m, \; k+1)]^{-1}$

$$= \left[\begin{bmatrix} H(k+1-m)\Phi(k+1-m,k+1) \\ \vdots \\ H(k)\Phi(k,k+1) \\ H(k+1)\Phi(k+1, \; k+1) \end{bmatrix}^T \begin{bmatrix} Rv(k+1-m/k+1) & \cdots & 0 \\ 0 & \ddots & \vdots \\ 0 & & \vdots \\ \vdots & Rv(k/k+1), & 0 \\ 0 & \cdots & Rv(k+1/k+1) \end{bmatrix}^{-1} \begin{bmatrix} H(k+1-m)\Phi(k+1-m,k+1) \\ \vdots \\ H(k)\Phi(k,k+1) \\ H(k+1)\Phi(k+1,k+1) \end{bmatrix} \right]^{-1}$$

$$= [H^T(k+1)Rv^{-1}(k+1/k+1) \; H(k+1) \; +$$

$$\begin{bmatrix} H(k+1-m)\phi(k+1-m,k+1) \\ \vdots \\ H(k)\phi(k,k+1) \end{bmatrix} \begin{bmatrix} Rv^{-1}(k+1-m/k+1). & 0 & 0 & 0 \\ 0 & \ddots & & \vdots \\ 0 & & \ddots & \vdots \\ 0 & & Rv^{-1}(k/k+1) & \end{bmatrix} \begin{bmatrix} H(k+1-m)\phi(k+1-m,k+1) \\ \vdots \\ H(k)\phi(k,k+1) \end{bmatrix} \Bigg]^{-1}$$

$$= [H^T(k+1)Rv^{-1}(k+1/k+1)H(k+1) + A^T(k/m-1,k+1)V^{-1}(k/m-1,k+1)A(k/m-1,k+1)]^{-1}$$

and use (16) and (18)

$$= \left[H^T(k+1)Rv^{-1}(k+1/k+1)H(k+1) + \Phi^T(k,k+1)A^T(k/m-1,k)\frac{1}{\delta}V^{-1}(k/m-1,k)A(k/m-1,k)\Phi(k,k+1)\right]^{-1}$$

$$= \left[H^T(k+1)Rv^{-1}(k+1/k+1)H(k+1) + \frac{1}{\delta}\Phi^T(k,k+1)\begin{pmatrix} H(k+1-m)\Phi(k+1-m,k) \\ \vdots \\ H(k)\Phi(k,k)\end{pmatrix}^T \begin{pmatrix} R_v^{-1}(k+1-m/k), & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0, & \cdots & Rv^{-1}(k/k)\end{pmatrix}\begin{pmatrix} H(k+1-m)\Phi(k+1-m,k) \\ \vdots \\ H(k)\Phi(k,k)\end{pmatrix}\Phi(k,k+1)\right]^{-1}$$

$$= \left[H^T(k+1)R_v^{-1}(k+1/k+1)H(k+1) + \frac{1}{\delta}{}^T(k,k+1)\begin{pmatrix} H(k-m)\Phi(k-m,k) \\ H(k+1-m)\Phi(k+1-m,k) \\ \vdots \\ H(k)\Phi(k,k)\end{pmatrix}^T \begin{pmatrix} R_v^{-1}(k-m/k), & \cdots & & 0 \\ \vdots & R_v^{-1}(k-m+1/k), & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & R_v^{-1}(k/k)\end{pmatrix} \begin{pmatrix} H(k-m)\Phi(k-m,k) \\ H(k+1-m)\Phi(k+1-m,k) \\ \vdots \\ H(k)\Phi(k,k)\end{pmatrix}\Phi(k,k+1) - \frac{1}{\delta}\Phi^T(k,k+1)\Phi^T(k-m,k)H^T(k-m) R_v^{-1}(k-m/k)H(k-m)\Phi(k-m,k)\Phi(k,k+1)\right]^{-1}$$

$$= [H^T(k+1)R_v^{-1}(k+1/k+1)H(k+1) - \frac{1}{\delta}\Phi^T(k-m,k+1)H^T(k-m)R_v^{-1}(k-m/k)H(k-m)\Phi(k-m,k+1) + \frac{1}{\delta}\Phi^T(k,k+1)A^T(k/m,k)V^{-1}(k/m,k)A(k/m,k)\Phi(k,k+1)]^{-1}$$

$$= [H^T(k+1)R_v^{-1}(k+1/k+1)H(k+1) - \frac{1}{\delta}\Phi^T(k-m,k+1)\ H^T(k-m)R_v^{-1}(k-m/k)H(k-m)\Phi(k-m,k+1) + \frac{1}{\delta}\Phi^T(k,k+1)P^{-1}(k/k)\Phi(k,k+1)]^{-1}$$

Now $P(k+1/k) = \Phi(k+1,k)P(k/k)\Phi^T(k+1,k)$

$$P^{-1}(k+1/k) = [\Phi^T(k+1,k)]^{-1}P^{-1}(k/k)\Phi^{-1}(k+1,k)$$

$$= \Phi^T(k,k+1)P^{-1}(k/k)\Phi(k,k+1)$$

$$= [H^T(k+1)R_v^{-1}(k+1/k+1)H(k+1) - \frac{1}{\delta}\Phi^T(k-m,k+1)H^T(k-m)R_v^{-1}(k-m/k)$$
$$H(k-m)\Phi(k-m,k+1) + \frac{1}{\delta}P^{-1}(k+1/k)]^{-1}$$

$$= \delta[(H^T(k+1) \mid \Phi^T(k-m,k+1)H^T(k-m))\left(\begin{array}{c|c}\delta R_v^{-1}(k+1/k+1) & 0 \\ \hline 0 & -R_v^{-1}(k-m/k)\end{array}\right)\left(\begin{array}{c}H(k+1) \\ \hline H(k-m)\Phi(k-m,k+1)\end{array}\right)$$
$$+ P^{-1}(k+1/k)]^{-1}$$

$$= \delta[(H^T(k+1) \mid \Phi^T(k-m,k+1)H^T(k-m))\left(\begin{array}{c|c}\frac{1}{\delta}Rv(k+1/k+1) & 0 \\ \hline 0 & -Rv(k-m/k)\end{array}\right)^{-1}\cdot\left(\begin{array}{c}H(k+1) \\ \hline H(k-m)\Phi(k-m,k+1)\end{array}\right)$$
$$+ P^{-1}(k+1/k)]^{-1}$$

$$= \delta[B^T(k+1)W^{-1}(k+1)B(k+1) + P^{-1}(k+1/k)]^{-1}$$

Now use the matrix Inversion Lemma [8]

$$[B^TW^{-1}B + P^{-1}]^{-1} = P - PB^T[BPB^T + W]^{-1}BP$$

---

define $k = PB^T[BPB^T + W]^{-1}$

$$= P - KBP$$

finally $P(k+1/k+1) = \delta[P(k+1/k) - k(k+1)B(k+1)P(k+1/k)]$

Secondly, from the projection Lemma, the "best" estimate

$$\hat{x}(k+1/m) = [A^T(k+1/m,k+1)V^{-1}(k+1/m,k+1)A(k+1/m,k+1)]^{-1}\cdot$$
$$A^T(k+1/m,k+1)V^{-1}(k+1/m,k+1)Y(k+1/m)$$

$$= P(k+1/k+1)A^T(k+1/m,k+1)V^{-1}(k+1/m,k+1)Y(k+1/m)$$

$$= \delta[P(k+1/k) - k(k+1)B(k+1)P(k+1/k)]$$
$$\cdot A^T(k+1/m,k+1)V^{-1}(k+1/m,k+1)Y(k+1/m)$$

If we denote the second term as D, then

$$D = A^T(k+1/m,k+1)V^{-1}(k+1/m,k+1)Y(k+1/m)$$

$$= \begin{bmatrix} H(k+1-m)\Phi(k+1-m,k+1) \\ \vdots \\ H(k)\Phi(k,k+1) \\ H(k+1)\Phi(k+1,k+1) \end{bmatrix}^T \begin{bmatrix} R_v^{-1}(k+1-m/k+1) & \cdots & & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & R_v^{-1}(k/k+1) & \vdots \\ 0 & \cdots & \cdots & R_v^{-1}(k+1/k+1) \end{bmatrix} \begin{bmatrix} Y(k+1-m) \\ Y(k-m) \\ \vdots \\ Y(k) \\ Y(k+1) \end{bmatrix}$$

$$= H(k+1)R_v^{-1}(k+1/k+1)Y(k+1) + A^T(k/m-1,k+1)V^{-1}(k/m-1,k+1)Y(k/m-1)$$

$$= H(k+1)R_v^{-1}(k+1/k+1)Y(k+1) + \Phi^T(k,k+1)A^T(k/m-1,k)\frac{1}{\delta}\ V^{-1}(k/m-1,k)Y(k/m-1)$$

$$= H(k+1)R_v^{-1}(k+1/k+1)Y(k+1) + \Phi^T(k,k+1)\begin{bmatrix} H(k+1-m,k)\Phi(k+1-m,k) \\ \vdots \\ H(k)\Phi(k,k) \end{bmatrix}^T$$

$$\frac{1}{\delta}\begin{bmatrix} R_v^{-1}(k+1-m/k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R_v^{-1}(k/k) \end{bmatrix}\begin{bmatrix} Y(k+1-m) \\ \vdots \\ Y(k) \end{bmatrix}$$

$$= H(k+1)R_v^{-1}(k+1/k+1)Y(k+1) - \frac{1}{\delta}\ \Phi^T(k,k+1)\Phi^T(k-m,k)H^T(k-m)R_v^{-1}(k-m/k)Y(k-m)$$

$$+ \frac{1}{\delta}\ \Phi^T(k,k+1)\begin{bmatrix} H(k-m)\Phi(k-m,k) \\ H(k+1-m)\Phi(k+1-m,k) \\ \vdots \\ H(k)\Phi(k,k) \end{bmatrix}\begin{bmatrix} R_v^{-1}(k-m/k) & \cdots & \cdots & 0 \\ \vdots & R_v^{-1}(k+1-m/k) & & \vdots \\ & & \ddots & \vdots \\ 0 & \cdots & \cdots & R_v^{-1}(k/k) \end{bmatrix}\begin{bmatrix} Y(k-m) \\ Y(k+1-m) \\ \vdots \\ Y(k) \end{bmatrix}$$

$$= H(k+1)R_v^{-1}(k+1/k+1)Y(k+1) - \frac{1}{\delta}\ \Phi^T(k-m,k+1)H^T(k-m)R_v^{-1}(k-m/k)Y(k-m)$$

$$+ \frac{1}{\delta}\ \Phi^T(k,k+1)A^T(k/m,k)V^{-1}(k/m,k)Y(k/m)$$

$$= \frac{1}{\delta}[\,(H^T(k+1) \mid \Phi^T(k-m,k+1)H^T(k-m))\left(\begin{array}{c|c} \frac{1}{\delta}R_v(k+1/k+1) & 0 \\ \hline 0 & -R_v(k-m/k) \end{array}\right)^{-1}\left(\begin{array}{c} Y(k+1) \\ \hline Y(k-m) \end{array}\right)$$

$$+ \Phi^T(k,k+1)A^T(k/m,k)V^{-1}(k/m,k)Y(k/m)\,]$$

$$= \frac{1}{\delta}[B^T(k+1)W^{-1}(k+1)Z(k+1) + \Phi^T(k,k+1)A^T(k/m,k)V^{-1}(k/m,k)Y(k/m)]$$

Now

$$\hat{x}(k+1/m) = \delta[P(k+1/k) - K(k+1)B(k+1)P(k+1/k)] \cdot D$$

$$= \delta[P(k+1/k) - k(k+1)B(k+1)P(k+1/k)] \cdot \frac{1}{\delta}[B^T(k+1)W^{-1}(k+1)Z(k+1) + \Phi^T(k,k+1)$$

$$A^T(k/m,k)V^{-1}(k/m,k)Y(k/m)]$$

$$= [\Phi(k+1,k)P(k/k)\Phi^T(k+1,k) - k(k+1)B(k+1)\Phi(k+1,k)P(k/k)\Phi^T(k+1,k)] \cdot$$
$$[B^T(k+1)W^{-1}(k+1)Z(k+1) + \Phi^T(k,k+1)A^T(k/m,k)V^{-1}(k/m,k)Y(k/m)]$$

$$= [\Phi(k+1,k)P(k/k)\Phi^T(k+1,k)\Phi^T(k,k+1)A^T(k/m,k)V^{-1}(k/m,k)Y(k/m)] - [k(k+1)$$
$$B(k+1)\Phi(k+1,k)P(k/k)\Phi^T(k+1,k)\Phi^T(k,k+1)A^T(k/m,k)V^{-1}(k/m,k)Y(k/m)]$$
$$+ [\Phi(k+1,k)P(k/k)\Phi^T(k+1,k)B^T(k+1)W^{-1}(k+1)Z(k+1)] - [k(k+1)B(k+1)$$
$$\Phi(k+1,k)P(k/k)\Phi^T(k+1,k)B^T(k+1)W^{-1}(k+1)Z(k+1)]$$

$$= [\Phi(k+1,k)P(k/k)A^T(k/m,k)V^{-1}(k/m,k)Y(k/m,k)] - [KB\Phi(k+1,k)P(k/k)A^T(k/m,k)$$
$$V^{-1}(k/m,k)Y(k/m,k)] + [P(k+1/k)B^TW^{-1}Z] - [KBP(k+1/k)B^TW^{-1}Z]$$

$$= [\Phi(k+1,k)\hat{x}(k/m)] - KB\Phi(k+1,k)\hat{x}(k/m) + [PB^T - KBPB^T]W^{-1}Z$$

Now

$$[PB^T - KBPB^T]W^{-1}Z = [PB^T - PB^T(W + BPB^T)^{-1}BPB^T - PB^T(W + BPB^T)^{-1}W +$$
$$PB^T(W + BPB^T)^{-1}W]W^{-1}Z$$

$$= [PB^T - PB^T(W + BPB^T)^{-1}(BPB^T + W) + PB^T(W + BPB^T)^{-1}W]W^{-1}Z$$

$$= [PB^T - PB^T + PB^T(W + BPB^T)^{-1}W]W^{-1}Z$$

$$= [PB^T(W + BPB^T)^{-1}]Z$$

$$= k(k+1)Z(k+1)$$

Finally $\hat{x}(k+1/m) = \Phi(k+1,k)\hat{x}(k/m) + k(k+1)[z(k+1) - B(k+1)\Phi(k+1,k)\hat{x}(k/m)]$

Q.E.D.

It can be noted that the optimal FFF estimation is linear and computed recursively (see eq (9)). The recursive nature of the algorithm will allow the estimate to be computed "on-line" and therefore support "real-time" estimation. In addition, the implementation of the FFF filter will require more memory than that associated with a Kalman filter. The memory will be configurated as a "first in - last out" stack. Since contemporary

memory costs are low and decreasing, the additional memory requirement of the FFF filter does not detract from its applicability.

2.3 Example 1

To show the adaptive qualities, a very simple example is as follows. Suppose we have a simple stationary plant

$$x(k+1) = \Phi(k+1,k)x(k)$$

$$y(k) = x(k) + V(k)$$

Let $\Phi(k,k+1) = \begin{cases} 1 & k \neq 50,\ k \neq 75 \\ 6/5 & k = 50 \\ 7/6 & k = 75 \end{cases}$

then $x(k) = \begin{cases} x(0) & k \leq 50 \\ 1.2x(0) & 50 < k \leq 75 \\ 1.4x(0) & 75 < k \leq 100 \end{cases}$

Let $x(0) = 30$ and $R_v(k) = 1$

The algorithm;

$$B = \left[\frac{1}{1}\right],\ W = \left[\begin{array}{c|c} 1/\delta & 0 \\ \hline 0 & -\delta^M \end{array}\right],\ Z = \left[\frac{Y}{Y(1)}\right]$$

$$K = PB^T[W + BPB^T]^{-1}$$

$$P = (\delta)*[P - KBP]$$

$$\hat{x} = \hat{x} + K[Z - B\hat{x}]$$

The results for different δ, and M are shown in Fig. 1-2. Program 1A is for sequential estimation and Program 1B is for FFF estimation. Both can be found in Appendix (2). As we can see from Fig. 1-2, the FFF estimate can follow the structural changes of the system, but the result will depend on the choices of δ and M.

In this example M = 20 seems to be a reasonable choice, since the value of x is defined over 30 second blocks of time. When we use $\delta = 1$, M = 30, the result is inferior to that of a regular sequential estimation. The

reason is P, the error covariance will decrease with the increasing of the number of observations, and $M = 30$ did not provide a sufficiently rich sample space to specify the system.

In order to solve this problem, we can increase the length of the data window and/or use a heavy discount on these data which we think is no longer important, when using $\delta = 1.1$, $M = 50$ the result was found to be better than $\delta = 1$, $M = 30$. However, it is still not a good choice. Using $\delta = 1.2$, $M = 50$ the system estimation is superior to the others. Therefore, we can see that the performance of a FFF filter will directly depend on the choices of δ and M. However with a reasonable choice of δ and M, the estimation algorithm has been shown to be able to follow structural changes in a linear system.

In using the FFF filter, the first M observation should be processed through a regular sequential filter. And after the observation index becomes greater than M the data will be processed by FFF filter. This is accomplished by shifting the data window and discounting the past history at the same time.

In summary, the FFF filter is easily implemented and can also process data on line. The only disadvantage is that instead of inverting a $(m \times m)$ matrix [eq (6)], now a $(2m \times 2m)$ matrix has to be inverted [eq (10)] which will increase the computational time.

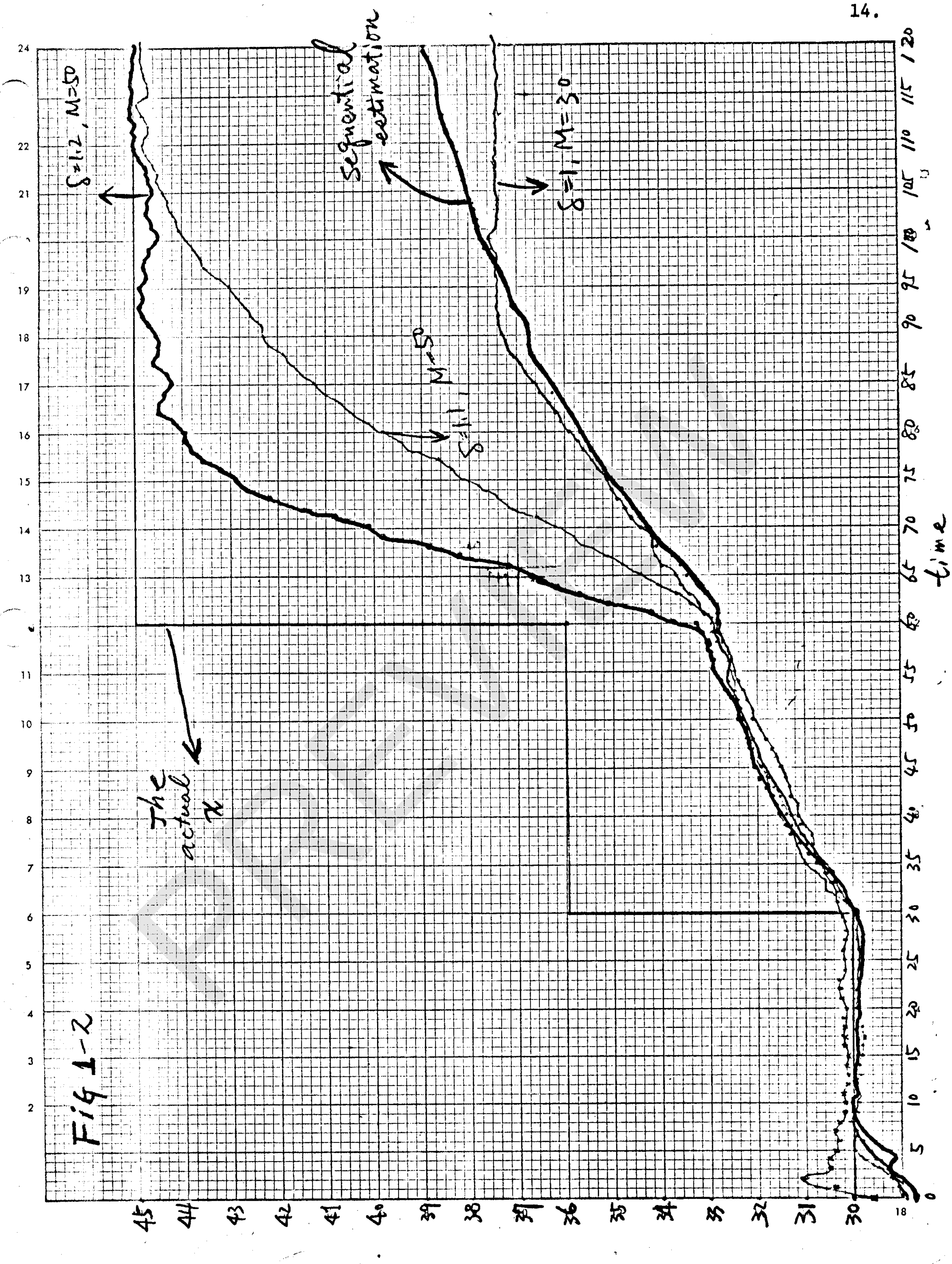
δ=1.2, M=50
Sequential estimation
δ=1, M=30
δ=1.1, M=50
The actual x
time

Fig 1-2

3.0 APPLICATIONS

3.1.0 Application 1: Sequential estimation for heat rate curves in power system

3.1.1 Introduction

Numerous papers have been published on the subject of economic operation of power systems. Fuel cost studies are performed with the aid of digital computers and require generator performance data in the form of heat rate curve, both incremental or net.

Least square curve fitting techniques are generally used to define an optimal polynomial fit to experimentally derived heat-rate data. Due to the changes in operating conditions and the aging of the system, this curve must be updated and reprocessed regularly. This is undesirable because the numerical and data collection requirements associated with each unique least-square operation is enormous.

In this part, a Kalman filter is used to accomplish the polynomial modeling task. One of the most significant features of Kalman filter is its recursive form. This makes it possible to process the data for heat rate curve in real time. The output coefficients will always be consistent with the system. This approach requires only current data be held in a temporary memory location whereas the conventional least-square method requires the entire data history be analyzed in-situ. Knowledge of the noise statistics is assumed to be known.

The objective of this work is to implement a sequential Kalman filter which can estimate the heat rate curve in an on line fashion by choosing right basis functions. Finally the result of this approach will be compared to that obtained by using currently accepted heat-rate estimation method. The possibility of adding finite fading memory is also discussed and tested through simulated data.

3.1.2 Problem Statement

Often one is interested in representing a function y(x) as follows:

$$y(x) = \sum_{k=1}^{n} a_k H_k(x)$$

where $H_k(x)$'s are some specified function of x and the a_k are chosen so as to insure an optimal fit to y(x) in some sense.

We can view the curve least square data fitting problem as an estimation problem, that is

$$y(x) = \sum_{k=1}^{n} a_k H_k(x) + v$$

The problem becomes "from observations y(x) and $H_k(x)$, find the best estimate of a_k which will minimize $\int_x [y(x) - \sum_{k=1}^{n} a_k H_k(x)]^2 dx$"

or in vector form

$$y = HA + v$$

$$H = [H_1(x), H_2(x), \ldots H_n(x)]$$

$$A = [a_1, a_2, \ldots a_n]^T$$

$$y \in E^1 \quad H, (1 \times n), A, (n \times 1)$$

Define the plant and observation

$$A(k+1) = A(k)$$

$$y(k) = H(k)A(k) + v(k)$$

y(k), H(k) is the sample value at sample k, the Kalman filter algorithm becomes

$$P(k/k-1) = \Phi(k,k-1)P(k-1/k-1)\Phi^T(k,k-1) = P(k-1/k-1) \tag{19}$$

$$k(k) = P(k-1/k-1)H^T(k)[Rv(k) + H(k)P(k-1/k-1)H^T(k)]^{-1} \tag{20}$$

$$P(k/k) = P(k-1/k-1) - k(k)H(k)P(k-1/k-1) \tag{21}$$

$$\hat{A}(k/k) = \hat{A}(k-1/k-1) + k(k)[y(k) - H(k)\hat{A}(k/k)] \tag{22}$$

A sequential Kalman filter can be built to estimate the heat rate curve in a on-line fashion. Here y would be the measured BTU/H input. The H's would be the set of basis function's that the user considers to be most related to the input.

The use of the algorithm does not require any permanent storage of y and H. In addition, no matter how many basis functions are chosen, only a scalar inversion [see eq (20)] is required to implement the filter.

3.1.3 Example 2

Before processing field derived data, an experiment over simulated data will be presented. The simulated data set consists of 64 points representing input BTU/H and output KWH. The data set was contrived to look like a theoretical heat rate curve in power system.

It is well known that the performance of an electrical generator varies with the changes of cooling water temperature. Therefore, the data set are assumed to be varied with the temperature by decreasing the data value of the input BTU/H by 2 percent for each one of change over a temperature range from 85°F to 75°F. The data is plotted in Fig. 1-3 and listed in Appendix (1) as D-1-1. The program can be found in Appendix (2), program 1-2.

In this example, basis functions are chosen to be the constant, KWH, Temp, Temp x KWH, Sin (ω x KWH), cos (ω x KWH), Sin (2ω x KWH), cos (2ω x KWH), where ω is a frequency which can be obtained approximately by processing the data through a Fast Fourier Transform. In this example ω is 0.06.