NASA TECHNICAL
MEMORANDUM

NASA TM 73,211

NASA TM 73,211

VARIATIONAL ALGORITHMS FOR NONLINEAR SMOOTHING APPLICATIONS

Ralph E. Bach, Jr.

Ames Research Center
Moffett Field, Calif. 94035

June 1977

1. Report No. NASA TM 73,211	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle VARIATIONAL ALGORITHMS FOR NONLINEAR SMOOTHING APPLICATIONS		5. Report Date 6. Performing Organization Code
7. Author(s) Ralph E. Bach, Jr.*		8. Performing Organization Report No. A-6939
9. Performing Organization Name and Address Ames Research Center Moffett Field, Calif. 94035		10. Work Unit No. 505-08-23 11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D. C. 20546		13. Type of Report and Period Covered Technical Memorandum 14. Sponsoring Agency Code

15. Supplementary Notes *R. E. Bach, Jr., is with the Department of Electrical Engineering, Northeastern University, Boston, Mass. 02115. He is a visiting Research Scientist at Ames Research Center under an Intergovernmental Personnel Agreement.

16. Abstract

This report presents a variational approach for solving a nonlinear, fixed-interval smoothing problem with application to offline processing of noisy data for trajectory reconstruction and parameter estimation. The nonlinear problem is solved as a sequence of linear two-point boundary-value problems (TPBVP). Second-order convergence properties are demonstrated. Algorithms for both continuous and discrete versions of the problem are given, and example solutions are provided.

17. Key Words (Suggested by Author(s)) Nonlinear smoothing Parameter estimation		18. Distribution Statement Unlimited STAR Category - 65	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 37	22. Price* $3.75

*For sale by the National Technical Information Service, Springfield, Virginia 22161

VARIATIONAL ALGORI HMS FOR NONLINEAR SMOOTHING APPLICATIONS

Ralph E. Bach, Jr.*

Ames Research Center

SUMMARY

This report presents a variational approach for solving a nonlinear, fixed-interval smoothing problem with application to offline processing of noisy data for trajectory reconstruction and parameter estimation. The non-linear problem is solved as a sequence of linear two-point boundary-value problems (TPBVP). Second-order convergence properties are demonstrated. Algorithms for both continuous and discrete versions of the problem are given, and example solutions are provided.

I. INTRODUCTION

Smoothing applications generally involve offline processing of noisy data records for trajectory reconstruction and parameter estimation. The fixed-interval smoothing problem is conveniently formulated as one of minimizing a suitable performance measure subject to dynamic constraints. This formulation is equivalent to a Bolza problem in the calculus of variations (ref. 1). Bryson and Frazier (ref. 2) first gave a solution for the linear, continuous case, in which a "sweep" method was used to solve the resulting two-point boundary-value problem (TPBVP). Cox (ref. 3) later formulated and solved a linear, discrete problem in a similar way.

Solution of the nonlinear smoothing problem requires an iterative procedure that converges and is computationally feasible. Many techniques have been proposed (ref. 4), but only a few have been applied in practice. In one approach, an approximate sweep method is used to solve the nonlinear TPBVP of

*R. E. Bach, Jr., is with the Department of Electrical Engineering, Northeastern University, Boston, Mass. 02115. He is a visiting Research Scientist at Ames Research Center under an Intergovernmental Personnel Agreement.

the Bolza formulation. The resulting algorithm consists of an extended Kálmán filter-smoother. A typical implementation is described in reference 5. While this algorithm is commonly used, its convergence properties are difficult to predict because linearization is about a filtered trajectory.

A second-order variational procedure is applied here to the nonlinear smoothing problem. This approach leads to an algorithm that combines a Newton-Raphson computation of parameter changes with a stable information filter-smoother to improve the state estimates at each iteration. Linearization is about a smoothed trajectory, and quadratic convergence to a minimum of the performance measure is demonstrated. In the absence of process noise, the algorithm is shown to be mathematically equivalent to other Newton-Raphson methods used for parameter estimation (refs. 6 and 7).

The basic analytical approach used to derive the algorithm is not new: it is an example of the "successive sweep" method of McReynolds and Bryson (ref. 8), originally devised to solve a continuous optimal control problem. Furthermore, Sage and Melsa (ref. 9) outlined the application of this method in solving the discrete smoothing problem. The development presented here extends and unifies previous work, and also directs attention to an apparently neglected but very useful tool for state and parameter estimation.

II. STATEMENT OF PROBLEM

The continuous version of the smoothing problem can be stated as follows: Given a system of the form

$$\dot{x} = f(x,w) \ , \qquad x(t_o) = x_o \tag{2.1}$$

with a continuous measurement

$$z = h(x) + v \tag{2.2}$$

available over the interval (t_0,t_f), determine x_0, $x = x(t)$, and $w = w(t)$ so that a performance measure

$$J = (1/2)(x_0 - \bar{x}_0)^T P_0^{-1}(x_0 - \bar{x}_0) + (1/2)\int_{t_0}^{t_f}(w^T Q^{-1} w + v^T R^{-1} v)dt \tag{2.3}$$

is minimized. In this formulation, $\bar{x}_0$ is an a priori estimate of the state at $t = t_0$, and P_0, Q, and R are weighting matrices. The vector $w = w(t)$ can be considered an "unknown" input, and is often modeled as process noise, while $v = v(t)$ is the "error" in the measurement.

The discrete version of the smoothing problem is formulated in a similar manner: Given a system of the form

$$x(i + 1) = f[x(i), w(i)] , \qquad x(0) = x_0 \tag{2.4}$$

with a measurement sequence

$$z(i) = h[x(i)] + v(i) , \qquad i = 1, 2, \ldots, N \tag{2.5}$$

determine x_0 and sequences $[x(i)]$ and $[w(i)]$ so that a performance measure

$$J = (1/2)(x_0 - \bar{x}_0)^T P_0^{-1}(x_0 - \bar{x}_0) + (1/2) \sum_{i=0}^{N-1} [w^T(i)Q^{-1}w(i) + v^T(i+1)R^{-1}v(i+1)] \tag{2.6}$$

is minimized.

The reader can consult reference 9 for a Bayesian maximum-likelihood interpretation of criteria (2.3) and (2.6). For this discussion, a weighted least-squares interpretation would also be appropriate. The continuous and discrete versions of the smoothing problem are each examples of a Bolza problem, which are solved here by use of a second-order variational procedure. Note that the smoothing formulation permits the inclusion of constant parameters as state variables. Thus, parameter identification is a special case of state-variable estimation.

III. CONTINUOUS ALGORITHM

To develop an algorithm for solving the continuous smoothing problem, first adjoin the dynamic constraint (2.1) to the performance measure (2.3) with Lagrange multiplier $\lambda = \lambda(t)$ to obtain

$$\bar{J} = \phi + \int_{t_0}^{t_f} (\mathcal{H} - \lambda^T \dot{x})dt \tag{3.1}$$

where *a priori* information is included in

$$\phi = (1/2)(x_o - \bar{x}_o)^T P_o^{-1}(x_o - \bar{x}_o) \qquad (3.2)$$

and the Hamiltonian is defined as

$$\mathcal{H} = (1/2)(w^T Q^{-1} w + v^T R^{-1} v) + \lambda^T f(x,w) \ . \qquad (3.3)$$

The necessary conditions for a minimum $\bar{J}$ are determined in the usual way by requiring that the first variation vanish (ref. 1). These conditions are given by (2.1) and

$$\dot{\lambda} = -\mathcal{H}_x^T \ , \qquad \lambda(t_f) = 0 \qquad (3.4)$$

$$0 = \mathcal{H}_w^T \ ; \qquad \lambda(t_o) = -(\phi_x^T)_o \qquad (3.5)$$

where the letter subscript indicates a partial derivative operation. Equations (2.1), (3.4), and (3.5) define a nonlinear TPBVP. It is simple enough to choose an x_o, $w(t)$ and to generate nominal trajectories $x(t)$, $\lambda(t)$ by solving (2.1) and (3.4). Such trajectories, however, are not likely to yield a minimum performance measure (neither condition of (3.5) is satisfied).

The algorithm is developed by considering the effect on the augmented performance measure caused by changes $\delta x(t)$, $\delta w(t)$, and $\delta\lambda(t)$ from nominal trajectories. Equation (3.1) is expanded to second order to obtain:

$$\delta\bar{J} = (\phi_x + \lambda^T)_o \delta x_o + \int_{t_o}^{t_f} \mathcal{H}_w \delta w \, dt + (1/2)\delta x_o^T (\phi_{xx})_o \delta x_o$$

$$+ (1/2)\int_{t_o}^{t_f} (\delta x^T \delta w^T) \begin{bmatrix} \mathcal{H}_{xx} & \mathcal{H}_{xw} \\ \mathcal{H}_{wx} & \mathcal{H}_{ww} \end{bmatrix} \begin{bmatrix} \delta x \\ \delta w \end{bmatrix} dt + \int_{t_o}^{t_f} \delta\lambda^T (f_x \delta x + f_w \delta w - \delta\dot{x}) dt \ . \qquad (3.6)$$

The objective now is to determine δx_o, $\delta x(t)$, and $\delta w(t)$ so that $\delta\bar{J}$ is as large a negative number as possible at each iteration. Convergence is attained when $\bar{J}$ reaches a minimum. In this "accessory minimization" problem, $\delta\lambda = \delta\lambda(t)$ acts as a Lagrange multiplier for a dynamic constraint, equivalent to a first-order expansion of (2.1). The necessary conditions for minimizing $\delta\bar{J}$ in (3.6) are given by

$$\delta\dot{x} = f_x\delta x + f_w\delta w \,, \qquad \delta x(t_0) = \delta x_0 \tag{3.7}$$

$$\delta\dot{\lambda} = -\mathcal{H}_{xx}\delta x - \mathcal{H}_{xw}\delta w - f_x^{T}\delta\lambda \,, \qquad \delta\lambda(t_f) = 0 \tag{3.8}$$

$$\delta\lambda(t_0) = -(\phi_x^{T} + \lambda + \phi_{xx}\delta x)_0 \tag{3.9}$$

$$\delta w = -\mathcal{H}_{ww}^{-1}(f_w^{T}\delta\lambda + \mathcal{H}_w^{T} + \mathcal{H}_{wx}\delta x) \,. \tag{3.10}$$

Substituting (3.10) into (3.7) and (3.8) yields the linear, nonhomogeneous TPBVP:

$$\delta\dot{x} = (f_x - f_w\mathcal{H}_{ww}^{-1}\mathcal{H}_{wx})\delta x - f_w\mathcal{H}_{ww}^{-1}f_w^{T}\delta\lambda - f_w\mathcal{H}_{ww}^{-1}\mathcal{H}_w^{T} \,, \qquad \delta x_0(\text{unknown}) \tag{3.11}$$

$$\delta\dot{\lambda} = -(\mathcal{H}_{xx} - \mathcal{H}_{xw}\mathcal{H}_{ww}^{-1}\mathcal{H}_{wx})\delta x - (f_x - f_w\mathcal{H}_{ww}^{-1}\mathcal{H}_{wx})^{T}\delta\lambda + \mathcal{H}_{xw}\mathcal{H}_{ww}^{-1}\mathcal{H}_w^{T} \,,$$
$$\delta\lambda(t_f) = 0 \,. \tag{3.12}$$

Any one of a number of "sweep" solutions may be used for a linear TPBVP. Here it is convenient to introduce $M = M(t)$ and $m = m(t)$ so that

$$\delta\lambda = M\delta x + m \;; \qquad M(t_f) = 0 \;; \qquad m(t_f) = 0 \,. \tag{3.13}$$

Now differentiate (3.13) and use (3.11) and (3.12) to obtain the differential equations:

$$\dot{M} = -f_x^{T}M - Mf_x - \mathcal{H}_{xx} + L\mathcal{H}_{ww}^{-1}L^{T} \tag{3.14}$$

$$\dot{m} = -f_x^{T}m + L\mathcal{H}_{ww}^{-1}(f_w^{T}m + \mathcal{H}_w^{T}) \tag{3.15}$$

where

$$L = \mathcal{H}_{xw} + Mf_w \,. \tag{3.16}$$

Equations (3.14) and (3.15) constitute a "backward information filter," i.e., their solutions are propagated backward from $t = t_f$ until $t = t_0$. The change in x_0 needed to initialize the forward smoothing pass is determined by substituting (3.13) into (3.9) and solving for δx_0:

$$\delta x_0 = -(P_0^{-1} + M_0)^{-1}[P_0^{-1}(x_0 - \bar{x}_0) + \alpha_0] \tag{3.17}$$

where

$$(\phi_x^{T})_0 = P_0^{-1}(x_0 - \bar{x}_0) \;; \qquad (\phi_{xx})_0 = P_0^{-1} \tag{3.18}$$

$$\alpha_o = \lambda(t_o) + m(t_o) \; ; \quad M_o = M(t_o)$$

are used. Hence a Newton-Raphson calculation yields the minimizing change of initial condition. To complete the TPBVP solution, integrate the "forward information smoother," (3.7), from $t = t_o$ until $t = t_f$, where the change in the forcing function, obtained from (3.10) and (3.13), is given by

$$\delta w = -\mathcal{H}_{ww}^{-1}(f_w{}^T m + \mathcal{H}_w{}^T + L^T \delta x) \; . \tag{3.20}$$

The foregoing procedure can be applied iteratively until the performance measure (2.3) reaches a minimum. Convergence properties of the algorithm can be demonstrated by observing that the change in performance measure $\delta\bar{J}$ reduces to a sum of quadratic forms. The derivation is facilitated by adding the zero value

$$-(1/2) \int_{t_o}^{t_f} \delta x^T M(f_x \delta x + f_w \delta w - \delta\dot{x}) dt$$

to (3.6) and substituting for $\delta\lambda$ from (3.13). After some manipulation, the expression for $\delta\bar{J}$ becomes

$$\delta\bar{J} = [(x_o - \bar{x}_o)^T P_o^{-1} + \alpha_o{}^T]\delta x_o + (1/2)\delta x_o{}^T(P_o^{-1} + M_o)\delta x_o$$
$$- (1/2) \int_{t_o}^{t_f} (m^T f_w + \mathcal{H}_w)\mathcal{H}_{ww}^{-1}(f_w{}^T m + \mathcal{H}_w{}^T) dt \; . \tag{3.21}$$

Now, substituting from (3.17) yields

$$\delta\bar{J} = -(1/2)\delta x_o{}^T(P_o^{-1} + M_o)\delta x_o - (1/2) \int_{t_o}^{t_f} (m^T f_w + \mathcal{H}_w)\mathcal{H}_{ww}^{-1}(f_w{}^T m + \mathcal{H}_w{}^T) dt \; . \tag{3.22}$$

Thus, convergence of the algorithm is assured when

$$(P_o^{-1} + M_o) > 0 \; ; \quad \mathcal{H}_{ww} > 0 \; . \tag{3.23}$$

Practical Implementation

Some attention has been given to the problem of increasing the radius of convergence of the algorithm. An effective procedure, which has been verified experimentally, is to eliminate from $\mathcal{H}_{xx}$, $\mathcal{H}_{xw}$, and $\mathcal{H}_{ww}$ all second-partial derivatives involving $f(x,w)$ and $h(x)$. The result is

$$\mathcal{H}_{xx} = h_x^T R^{-1} h_x \; ; \qquad \mathcal{H}_{xw} = 0 \; ; \qquad \mathcal{H}_{ww} = Q^{-1} \; . \tag{3.24}$$

In some cases, of course, one or more of the relations in (3.24) may be exact. Note that the costate λ is no longer needed to calculate $\mathcal{H}_{xx}$, $\mathcal{H}_{xw}$, or $\mathcal{H}_{ww}$. The accessory minimization problem can now be simplified considerably. Introduce a new variable $\rho(t)$ so that

$$\rho = \nu + \lambda \tag{3.25}$$

and use (3.24), (3.25), and

$$\mathcal{H}_w^T = f_w^T \lambda + Q^{-1} w \tag{3.26}$$

$$\mathcal{H}_x^T = f_x^T \lambda - h_x^T R^{-1} v \tag{3.27}$$

in (3.7) and (3.8) to obtain an equivalent linear TPBVP:

$$\delta\dot{x} = f_x \delta x - f_w Q f_w^T \rho - f_w w \; , \qquad \delta x(t_0) = \delta x_0 \tag{3.28}$$

$$\dot{\rho} = -h_x^T R^{-1} h_x \delta x - f_x^T \rho + h_x^T R^{-1} v \; , \qquad \rho(t_f) = 0 \; . \tag{3.29}$$

To solve (3.28) and (3.29), use the sweep

$$\rho = M\delta x + \alpha \; ; \qquad M(t_f) = 0 \; ; \qquad \alpha(t_f) = 0 \; . \tag{3.30}$$

Now, differentiate (3.30) and use (3.28) and (3.29) to obtain the differential equations:

$$\dot{M} = -f_x^T M - M f_x - h_x^T R^{-1} h_x + M f_w Q f_w^T M \tag{3.31}$$

$$\dot{\alpha} = -(f_x - f_w Q f_w^T M)^T \alpha + M f_w w + h_x^T R^{-1} v \tag{3.32}$$

which constitute the backward information filter. The forward information smoother, determined by substituting (3.30) into (3.28), is given by

$$\delta\dot{x} = (f_x - f_w Q f_w^T M)\delta x - f_w(w + Q f_w^T \alpha) \tag{3.33}$$

where δx_0, as determined using (3.25) and (3.30) in (3.9), is the same as (3.17). The change in the unknown forcing function is found from (3.10) to be

$$\delta w = -w - Q f_w^T(\alpha + M\delta x) \; . \tag{3.34}$$

The steps in the algorithm are summarized below.

(1) Use x_0, $w(t)$ obtained from the preceding iteration (or an initial guess) to compute a smoothed trajectory $x(t)$ from (2.1) and the performance measure from (2.3).

(2) Solve the "backward information filter" (3.31) and (3.32) to obtain $M(t)$ and $\alpha(t)$; store the elements necessary for the next step.

(3) Perform the Newton-Raphson computation (3.17) for δx_0 and solve the "forward smoother" (3.33) and (3.34) to determine $\delta w(t)$.

(4) Update x_0 and $w(t)$ and iterate until δx_0 and $\delta w(t)$ are "sufficiently" small and J is minimized.

Some comments concerning the algorithm presented here are in order. First, (3.31) to (3.34) are recognized as the usual backward information filter, forward information smoother solution of a linear smoothing problem. In fact, any one of the classical solutions (refs. 4 and 10) may be implemented for the linear TPBVP of (3.28) and (3.29).[1] The important point here is that the nonlinear smoothing problem can be solved as a converging sequence of linear TPBVP solutions. Also, note that the approximations of (3.24) affect only the radius of convergence: when convergence is attained, $\delta x(t)$, $\delta w(t)$, and $\delta\lambda(t)$ vanish and $\alpha = \beta = 0$, so that (3.32), (3.34), and (3.9) reduce to

$$\dot{\lambda} = -f_x{}^T\lambda + h_x{}^T R^{-1} v\ , \qquad \lambda(t_f) = 0 \tag{3.35}$$

$$w = -Qf_w{}^T\lambda\ ; \qquad \lambda(t_0) = -P_0^{-1}(x_0 - \bar{x}_0)\ . \tag{3.36}$$

These equations are equivalent to (3.4) and (3.5), the conditions necessary for a minimum performance measure.

Finally, note that, if there is no process noise associated with the model, the equations for $M(t)$ and $\alpha(t)$ are much simpler. Storage requirements are reduced since the forward smoother equations need not be solved. The algorithm becomes a second-order procedure for parameter identification, which is equivalent to but easier to implement than the modified Newton-

[1] Refer to appendix C for another (equivalent) algorithm for solution of the continuous smoothing problem.

Raphson method (refs. 6 and 7). Details of this equivalence are presented in appendix A.

Linear Case

For a linear system given by

$$\dot{x} = Fx + Gu + \Gamma w , \quad x(t_0) = x_0 \tag{3.37}$$

$$z = Hx + Du + v \tag{3.38}$$

where u is a known forcing function, it is easy to demonstrate that convergence to a minimum performance measure is attained in one iteration of the algorithm. In (3.31) - (3.34), use

$$f_x = F ; \quad f_w = \Gamma ; \quad h_x = H \tag{3.39}$$

and, to eliminate dependence on the previous (or nominal) solution (x_n, w_n), make a change of variable

$$\delta x = x - x_n ; \quad \alpha = \alpha_L + Mx_n . \tag{3.40}$$

The resulting information filter-smoother equations are given by

$$\left.\begin{aligned} \dot{M} &= -F^T M - MF - H^T R^{-1} H + M\Gamma Q\Gamma^T M ; \\ \dot{\alpha}_L &= -(F - \Gamma Q\Gamma^T M)^T \alpha_L - MGu + H^T R^{-1}(z - Du) \end{aligned}\right\} \tag{3.41}$$

with $\alpha_L(t_f) = 0$, $M(t_f) = 0$, and

$$\left.\begin{aligned} \dot{x} &= (F - \Gamma Q\Gamma^T M)x + Gu - \Gamma Q\Gamma^T \alpha_L ; \\ x_0 &= -(P_0^{-1} + M_0)^{-1} (\alpha_{L0} - P_0^{-1}\bar{x}_0) ; \\ w &= -Q\Gamma^T(\alpha_L + Mx) . \end{aligned}\right\} \tag{3.42}$$

IV. DISCRETE ALGORITHM

Derivation of the algorithm for solving the discrete smoothing problem very closely parallels the development in the preceding section. Adjoin the constraint (2.4) to the performance measure (2.6) with a Lagrange multiplier to obtain

$$\bar{J} = \phi + \sum_{i=0}^{N-1} [\mathcal{H} - \lambda^T(i+1)x(i+1)] \tag{4.1}$$

where ϕ is as given in (3.2) and

$$\mathcal{H} = (1/2)[w^T(i)Q^{-1}w(i) + v^T(i+1)R^{-1}v(i+1)] + \lambda^T(i+1)f[x(i), w(i)] . \tag{4.2}$$

In what follows, it should be understood that

$$\mathcal{H}_x = \partial\mathcal{H}/\partial x(i) \ ; \qquad \mathcal{H}_w = \partial\mathcal{H}/\partial w(i) \ . \tag{4.3}$$

The usual variational procedure requires that

$$\lambda(i) = \mathcal{H}_x^T \ , \qquad \lambda(N) = 0 \tag{4.4}$$

$$0 = \mathcal{H}_w^T \ ; \qquad \lambda(0) = -(\phi_x^T)_o \tag{4.5}$$

be satisfied at a minimum $\bar{J}$. The discrete algorithm follows from a second-order expansion of the augmented performance measure in terms of changes $[\delta x(i)]$, $[\delta w(i)]$, and $[\delta\lambda(i)]$ from nominal trajectories generated by solving (2.4) and (4.4) for some x_o and sequence $[w(i)]$. The expansion is given by

$$\begin{aligned}\delta\bar{J} = {} & [(\phi_x)_o + \lambda^T(0)]\delta x_o + \sum_{i=0}^{N-1} \mathcal{H}_w \delta w(i) + (1/2)\delta x_o^T(\phi_{xx})_o\delta x_o \\ & + (1/2) \sum_{i=0}^{N-1} [\delta x^T(i) \ \delta w^T(i)] \begin{bmatrix} \mathcal{H}_{xx} & \mathcal{H}_{xw} \\ \mathcal{H}_{wx} & \mathcal{H}_{ww} \end{bmatrix} \begin{bmatrix} \delta x(i) \\ \delta w(i) \end{bmatrix} \\ & + \sum_{i=0}^{N-1} \delta\lambda^T(i+1)[f_x\delta x(i) + f_w\delta w(i) - \delta x(i+1)] \ . \end{aligned} \tag{4.6}$$

The objective now is to choose $\delta x(i)$ and $\delta w(i)$ for $i = 0, 1, \ldots, N-1$ to make $\delta^2 \bar{J}$ as large a negative number as possible. Note that $\delta\lambda(i+1)$ acts as a Lagrange multiplier for a dynamic constraint equivalent to a first-order expansion of (2.5). The necessary conditions for minimizing $\bar{J}$ in (4.6) are

$$\delta x(i+1) = f_x \delta x(i) + f_w \delta w(i) , \qquad \delta x(0) = \delta x_0 \tag{4.7}$$

$$\delta\lambda(i) = \mathcal{H}_{xx} \delta x(i) + \mathcal{H}_{xw} \delta w(i) + f_x^T \delta\lambda(i+1) ,$$
$$\delta\lambda(N) = 0 \tag{4.8}$$

$$\delta\lambda(0) = -[(\phi_x^T)_0 + \lambda(0) + (\phi_{xx})_0 \delta x_0] \tag{4.9}$$

$$\delta w(i) = -\mathcal{H}_{ww}^{-1}[f_w^T \delta\lambda(i+1) + \mathcal{H}_w^T + \mathcal{H}_{wx} \delta x(i)] . \tag{4.10}$$

Substituting (4.10) into (4.7) and (4.8) yields the linear, nonhomogeneous TPBVP:

$$\delta x(i+1) = (f_x - f_w \mathcal{H}_{ww}^{-1} \mathcal{H}_{wx}) \delta x(i) - f_w \mathcal{H}_{ww}^{-1} f_w^T \delta\lambda(i+1)$$
$$- f_w \mathcal{H}_{ww}^{-1} \mathcal{H}_w^T , \qquad \delta x_0 \text{(unknown)} \tag{4.11}$$

$$\delta\lambda(i) = (\mathcal{H}_{xx} - \mathcal{H}_{xw} \mathcal{H}_{ww}^{-1} \mathcal{H}_{wx}) \delta x(i) + (f_x - f_w \mathcal{H}_{ww}^{-1} \mathcal{H}_{wx})^T \delta\lambda(i+1)$$
$$- \mathcal{H}_{xw} \mathcal{H}_{ww}^{-1} \mathcal{H}_w^T , \qquad \delta\lambda(N) = 0 . \tag{4.12}$$

A backward sweep solution of (4.11) and (4.12) may be obtained. Let

$$\delta\lambda(i) = M(i) \delta x(i) + m(i) ; \qquad M(N) = 0 ; \qquad m(N) = 0 . \tag{4.13}$$

The result can be written as

$$M(i) = f_x^T M(i+1) f_x + \mathcal{H}_{xx} - L\bar{Q}L^T \tag{4.14}$$

$$m(i) = f_x^T m(i+1) - L\bar{Q}[f_w^T m(i+1) + \mathcal{H}_w^T] \tag{4.15}$$

where

$$L = \mathcal{H}_{xw} + f_x^T M(i+1) f_w ; \qquad \bar{Q} = [\mathcal{H}_{ww} + f_w^T M(i+1) f_w]^{-1} . \tag{4.16}$$

Now, solve (4.14) and (4.15) backward, from $i = N - 1$ to $i = 0$. To initialize the forward smoothing pass, substitute (4.13) into (4.9) and solve for δx_0 to obtain

$$\delta x_0 = -(P_0^{-1} + M_0)^{-1}[P_0^{-1}(x_0 - \bar{x}_0) + \alpha_0] \tag{4.17}$$

where

$$(\phi_x^T)_0 = P_0^{-1}(x_0 - \bar{x}_0) \; ; \qquad (\phi_{xx})_0 = P_0^{-1} \tag{4.18}$$

$$\alpha_0 = \lambda(0) + m(0) \; ; \qquad M_0 = M(0) \tag{4.19}$$

are used. The expression for the change of initial condition is seen to be a Newton-Raphson calculation. To obtain the change in the forcing function, solve (4.7) forward, with

$$\delta w(i) = -\bar{Q}[f_w^T m(i + 1) + \mathcal{H}_w^T + L^T \delta x(i)] \; . \tag{4.20}$$

The algorithm can be applied iteratively until the performance measure (2.6) reaches a minimum. Convergence properties can be demonstrated by observing that $\delta\bar{J}$ in (4.6) can be expressed as a sum of quadratic forms. The derivation is facilitated by adding the zero value

$$-(1/2) \sum_{i=0}^{N-1} \delta x^T(i + 1)M(i + 1)[f_x \delta x(i) + f_w \delta w(i) - \delta x(i + 1)]$$

to (4.6) and substituting for $\delta\lambda(i + 1)$ from (4.13). The expression for $\delta\bar{J}$ eventually reduces to

$$\begin{aligned}\delta\bar{J} = {} & [(x_0 - \bar{x}_0)^T P_0^{-1} + \alpha_0^T]\delta x_0 + (1/2)\delta x_0^T(P_0^{-1} + M_0)\delta x_0 \\ & -(1/2) \sum_{i=0}^{N-1} [m^T(i + 1)f_w + \mathcal{H}_w]\bar{Q}[f_w^T m(i + 1) + \mathcal{H}_w^T] \; .\end{aligned} \tag{4.21}$$

Note that the gradient of $\bar{J}$ with respect to x_0 appears in the first term of (4.21); the hessian $(\partial^2\bar{J}/\partial x_0^2)$ is found in the second term. Now, substitute the minimizing δx_0 from (4.17) and obtain

$$\delta\bar{J} = -(1/2)\delta x_0^T(P_0^{-1} + M_0)\delta x_0 - (1/2)\sum_{i=0}^{N-1}[m^T(i + 1)f_w + \mathcal{H}_w]\bar{Q}[f_w^T m(i + 1) + \mathcal{H}_w^T] \; . \tag{4.22}$$

Convergence of the discrete algorithm is assured for

$$(P_0^{-1} + M_0) > 0 \ ; \qquad \bar{Q} > 0 \ . \tag{4.23}$$

Practical Implementation

The radius of convergence can be effectively increased and the computations simplified by again eliminating all second-partial derivatives from $\mathcal{H}_{xx}$, $\mathcal{H}_{xw}$, and $\mathcal{H}_{ww}$. In the discrete case,

$$\mathcal{H}_{xx} \cong f_x^T h_x^T R^{-1} h_x f_x \ ; \qquad \mathcal{H}_{xw} \cong f_x^T h_x^T R^{-1} h_x f_w \ ;$$
$$\mathcal{H}_{ww} \cong f_w^T h_x^T R^{-1} h_x f_w + Q^{-1} \ . \tag{4.24}$$

To obtain a simplified linear TPBVP, introduce the variable

$$\rho(i) = \lambda(i) + \delta\lambda(i) \tag{4.25}$$

and use (4.24), (4.25) and

$$\mathcal{H}_x^T = f_x^T[\lambda(i + 1) - h_x^T R^{-1} v(i + 1)] \tag{4.26}$$

$$\mathcal{H}_w^T = f_w^T[\lambda(i + 1) - h_x^T R^{-1} v(i + 1)] + Q^{-1} w(i) \tag{4.27}$$

in (4.7) and (4.8). The result is

$$\delta x(i + 1) = f_x \delta x(i) - f_w Q f_w^T f_x^{-T} \rho(i) - f_w w(i) \ , \qquad \delta x(0) = \delta x_o \tag{4.28}$$

$$\rho(i) = f_x^T \left\{ \rho(i + 1) - h_x^T R^{-1} [v(i + 1) - h_x \delta x(i + 1)] \right\} , \quad \rho(N) = 0 \ . \tag{4.29}$$

A sweep solution of (4.28) and (4.29) with

$$\rho(i) = M(i)\delta x(i) + \alpha(i) \ ; \qquad M(N) = 0 \ ; \qquad \alpha(N) = 0 \tag{4.30}$$

yields the backward information filter,[*] represented by a measurement update:

$$\left. \begin{aligned} \beta(i + 1) &= \alpha(i + 1) - h_x^T R^{-1} v(i + 1) \ ; \\ P(i + 1) &= M(i + 1) + h_x^T R^{-1} h_x \ , \end{aligned} \right\} \tag{4.31}$$

[*] Refer to appendix D for another (equivalent) algorithm for solution of the discrete smoothing problem.

and a time update:

$$\bar{Q} = [Q^{-1} + f_w^T P(i + 1) f_w]^{-1} \; ; \qquad C(i) = \bar{Q} f_w^T P(i + 1) \qquad (4.32)$$

$$\left.\begin{aligned} \alpha(i) &= f_x^T [I - f_w C(i)]^T [\beta(i + 1) - P(i + 1) f_w w(i)] \; ; \\ M(i) &= f_x^T [I - f_w C(i)]^T P(i + 1) f_x \; . \end{aligned}\right\} \qquad (4.33)$$

The change δx_0 required to initialize the forward smoother (4.7) is again given by (4.17), while the change in forcing function can be written:

$$\delta w(i) = -[I - C(i) f_w][w(i) + Q f_w^T \beta(i + 1)] - C(i) f_x \delta x(i) \; . \qquad (4.34)$$

The steps in the discrete algorithm are summarized below.

(1) Use x_0, $[w(i)]$ obtained from the preceding iteration (or an initial guess) to compute a smoothed trajectory $[x(i)]$ from (2.4) and the performance measure from (2.6).

(2) Solve the "backward information filter" (4.31) and (4.33) to obtain $[M(i)]$ and $[\alpha(i)]$; store elements necessary for the next step.

(3) Perform the Newton-Raphson computation (4.17) for δx_0 and solve the "forward smoother" (4.7) and (4.34) to determine $[\delta w(i)]$.

(4) Update x_0 and $[w(i)]$ and iterate until δx_0, $[\delta w(i)]$ are "sufficiently" small and J is minimized.

When the algorithm converges, $\delta x(i)$, $\delta w(i)$, and $\delta\lambda(i)$ vanish for all i and $\alpha(i) = \rho(i) = \lambda(i)$. It can be shown that (4.31), (4.33), and (4.9) reduce to

$$\lambda(i) = f_x^T [\lambda(i + 1) - h_x^T R^{-1} v(i + 1)] \; , \qquad \lambda(N) = 0 \qquad (4.35)$$

$$w(i) = -Q f_w^T f_x^{-T} \lambda(i) \; ; \qquad \lambda(0) = -P_0^{-1}(x_0 - \bar{x}_0) \qquad (4.36)$$

which are equivalent to (4.4) and (4.5), the necessary conditions for a minimum performance measure. Note also that, for no process noise, the algorithm simplifies to a second-order parameter identification method (mathematically equivalent to the modified Newton-Raphson method — see appendix B for details).

V. EXAMPLES

Example 1

A typical smoothing application involves estimation of first and second derivatives, given a noisy data record. A continuous model for this problem is shown in Fig. 1. Note that a weighted least-squares interpretation of the performance measure should be made in this case since the unknown input will be deterministic in nature. Although this application is a linear problem, it provides a useful check on the algorithm.

For computer simulation, it is helpful to replace the continuous model with a discrete formulation given by

$$x(i + 1) = Fx(i) + Gw(i) \ , \qquad x(0) = x_0 \tag{5.1}$$

$$z(i) = Hx(i) + v(i) \tag{5.2}$$

where

$$F = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} , \qquad G = \begin{bmatrix} 0 \\ h \end{bmatrix} , \qquad H = [1 \quad 0] \tag{5.3}$$

for a time step h and

$$x(i) = \begin{bmatrix} y(i) \\ \dot{y}(i) \end{bmatrix} , \qquad w(i) = \ddot{y}(i) \ . \tag{5.4}$$

The data record $[z(i)]$ is assumed to be available for $i = 1, 2, \ldots, N$. The problem is to choose $y(0)$, $\dot{y}(0)$, and $w(i) = \ddot{y}(i)$, $i = 0, 1, \ldots, N-1$, to minimize

$$J = (1/2) \sum_{i=0}^{N-1} \{w^2(i)/Q + [z(i + 1) - y(i + 1)]^2/R\} \tag{5.5}$$

where no a priori knowledge of the initial conditions is given.

Implementation of the discrete algorithm (section IV) yields the information filter

$$M_{11}(i - 1) = M_{11}(i) + 1/R - h^2\bar{Q}M_{12}^2(i) \tag{5.6}$$

$$M_{12}(i - 1) = (\bar{Q}/Q)M_{12}(i) + hM_{11}(i - 1) \tag{5.7}$$

$$M_{22}(i - 1) = (\bar{Q}/Q)[M_{22}(i) + hM_{12}(i)] + hM_{12}(i - 1) \tag{5.8}$$

$$\alpha_1(i - 1) = \alpha_1(i) - v(i)/R - h\bar{Q}M_{12}(i)[h\alpha_2(i) + w(i - 1)/Q] \tag{5.9}$$

$$\alpha_2(i - 1) = (\bar{Q}/Q)[\alpha_2(i) - hM_{22}(i)w(i - 1)] + h\alpha_1(i - 1) \tag{5.10}$$

where

$$\bar{Q} = Q/[1 + h^2QM_{22}(i)] \ . \tag{5.11}$$

Equations (5.6) to (5.10) are solved "backward," that is, for $i = N, N - 1, \ldots, 1$, with

$$M_{11}(N) = M_{12}(N) = M_{22}(N) = \alpha_1(N) = \alpha_2(N) = 0 \ . \tag{5.12}$$

Changes in initial conditions are then determined by solving

$$\delta x_0 = -\begin{bmatrix} M_{11}(0) & M_{12}(0) \\ M_{12}(0) & M_{22}(0) \end{bmatrix}^{-1} \begin{bmatrix} \alpha_1(0) \\ \alpha_2(0) \end{bmatrix} \tag{5.13}$$

and the result is used to initialize the "forward" smoother, which is given by

$$\delta w(i) = -\bar{Q}\{w(i)/Q + h\alpha_2(i + 1) + hM_{12}(i + 1)[\delta x_1(i) + h\delta x_2(i)]$$
$$+ hM_{22}(i + 1)\delta x_2(i)\} \tag{5.14}$$

$$\delta x_1(i + 1) = \delta x_1(i) + h\delta x_2(i) \tag{5.15}$$

$$\delta x_2(i + 1) = \delta x_2(i) + h\delta w(i) \ . \tag{5.16}$$

A test case was run to simulate an aircraft descending from an altitude of 1100 m to touchdown in 80 sec. The initial vertical velocity was -2.5 m/sec. Note the "true" vertical acceleration waveform as shown in Fig. 3(b). The data record (h = 1 sec) was obtained by integrating the acceleration twice and adding random noise of 10 m (rms) to simulate barometric altimeter measurements. An analysis was performed with the starting sequence

$\hat{w}(i) = 0$, $i = 0, 1, \ldots, N - 1$, and several pairs of starting values for the initial conditions. In each case, convergence to

$$\hat{y}(0) = 1103 \text{ m} , \qquad \hat{\dot{y}}(0) = -2.79 \text{ m/sec} \tag{5.17}$$

was essentially complete after two iterations. The trajectory estimates are shown in Figs. 2 and 3.

It should be noted that the sequence $[\hat{w}(i)]$ was sensitive to the choice of Q and R. The results presented here were obtained with

$$Q = 1.0 \; ; \qquad R = 100 \tag{5.18}$$

which were the mean-squared values of the $[w(i)]$, $[v(i)]$ sequences, respectively. An analogous problem exists in applying a "moving-arc" polynomial smoother (ref. 11) for estimating position, velocity, and acceleration from measurements of position. With that technique, a least-squares fit of a second-degree polynomial to n consecutive data points is "moved" through the data; the values of the polynomial and its derivatives at the midpoint of each group provide the estimation sequence. Naturally, the choice of n will influence the smoothed waveforms, and some experimentation may be required to determine a suitable value.

Example 2

Consider the problem of estimating the parameters y_o and p for the system shown in figure 4, where both input and state measurements are corrupted by additive noise. In the estimation model, $a = a(t)$ is a known input, $w = w(t)$ is process noise, and $v = v(t)$ is measurement noise. Here, it is appropriate to use a Bayesian maximum-likelihood interpretation of the performance measure (ref. 9).

A discrete version of the model, suitable for digital-computer simulation, is given by

$$y(i + 1) = \phi y(i) + ha(i) + hw(i) , \qquad y(0) = y_o \tag{5.19}$$

$$p(i + 1) = p(i) , \qquad p(0) = p \tag{5.20}$$

and

$$z(i) = y(i) + v(i) \tag{5.21}$$

where

$$\phi = 1 + hp \ . \tag{5.22}$$

The sequences [w(i)], [v(i)] are assumed to be zero-mean and white, with covariances Q and R, respectively. Now the problem is to choose y_o, p, and w(i), i = 0, 1, ..., N - 1, to minimize

$$J = (1/2) \sum_{i=0}^{N-1} [w^2(i)/Q + v^2(i + 1)/R] \tag{5.23}$$

where, again, no a priori information about y_o and p is considered.

A data record was generated with p = -1.0, y_o = 0, h = 0.02, N = 300, and a unit-doublet input sequence, with each pulse lasting 1/4 of the record. The covariance of the state measurement noise was chosen to be R = 0.01, and values of Q/R = 0, 1, 10, 100, and 1000 were used in the simulation. Smoothing of the data was done by use of the discrete algorithm in section IV. Two analyses of each record were performed — in the first, only p and y_o were estimated, with $\hat{w}(i) = 0$ for all i. Results for 100 Monte Carlo runs are shown in Table I. The second analysis included estimation of [w(i)] (Table II).

A comparison of the experimental results shown in Tables I and II indicates that some parameter estimates are slightly less efficient when the sequence [w(i)] is estimated. This behavior was reported earlier (ref. 12) and can be explained heuristically. The N elements of [$\hat{w}$(i)] are additional degrees of freedom in the minimization procedure, and permit better "fits" to the data. In some cases, however, these N elements may reduce the influence of the other parameters on the performance measure. Note the apparent bias of the parameter estimates in Table I for large values of Q/R compared with the corresponding estimates in Table II.

By using the complete smoothing model, one has the distinct advantage of obtaining reliable predictions for parameter estimation errors as part of the solution. The approximation for the Cramer-Rao lower bound on the error variance for the ith parameter estimate is given by

$$(\sigma_i^2)^{cr} = (M_0^{-1})_{ii} \ . \tag{5.24}$$

Note the much improved correspondence between values of σ^{mc} and σ^{cr} (rms value for 100 runs) given in Table II compared to those given in Table I. These results, however, depend on good a priori knowledge of Q and R.

VI. CONCLUDING REMARKS

Continuous and discrete versions of a nonlinear smoothing algorithm were derived. Convergence characteristics were demonstrated to be of the second order. Examples were provided to illustrate application of the algorithm for state and parameter estimation in the presence of unknown inputs that are deterministic and stochastic in nature.

Mention should be made of the effort to increase the radius of convergence of the algorithm. Eliminating second-partial derivatives from $\mathcal{H}_{xx}$, $\mathcal{H}_{xw}$, and $\mathcal{H}_{ww}$ provides an effective approximation to the hessian matrix. For a system without process noise, convergence properties were shown to be equivalent to those obtained with the modified Newton-Raphson method. It is conjectured, however, that the approximation can be used in general, with good results.

It is too early to speculate whether the implementation of the smoothing algorithm described here will be more effective than existing extended Kálmán filter-smoother procedures. However, application of the algorithm for smoothing aircraft flight-test data is anticipated, and a comparison of results with those obtained by use of the filter-smoother will be included in the study.

ACKNOWLEDGMENT

The author gratefully acknowledges the many useful discussions with Mr. Rodney Wingrove of Ames Research Center. His constant encouragement and that of Mr. Brent Creer, Branch Chief, were very helpful in the preparation of this report.

APPENDIX A

THE MNR METHOD AND SMOOTHING
(CONTINUOUS CASE)

The modified Newton-Raphson (MNR) method (refs. 6 and 7) is widely used for offline parameter identification. The continuous version of the algorithm computes a change in initial condition

$$\delta x_o = -\left[P_o^{-1} + \int_{t_o}^{t_f} S^T R^{-1} S \, d\tau\right]^{-1} \left[P_o^{-1}(x_o - \bar{x}_o) - \int_{t_o}^{t_f} S^T R^{-1} v \, d\tau\right] \quad (A.1)$$

at each iteration for a system

$$\dot{x} = f(x) \ , \qquad x_o \quad (A.2)$$

$$z = h(x) + v \quad (A.3)$$

until the performance measure

$$J = (1/2)(x_o - \bar{x}_o)^T P_o^{-1}(x_o - \bar{x}_o) + (1/2) \int_{t_o}^{t_f} v^T R^{-1} v \, d\tau \quad (A.4)$$

is minimized. In this formulation,

$$S = h_x \phi(t,t_o) \ ; \qquad \dot{\phi} = f_x \phi \ , \qquad \phi(t_o,t_o) = I \ . \quad (A.5)$$

Here it is shown that the MNR method and the nonlinear smoothing algorithm derived in section III are equivalent for a system without process noise (w = 0), and with the approximation

$$\mathcal{H}_{xx} \simeq h_x{}^T R^{-1} h_x \ . \quad (A.6)$$

First, let

$$\phi^T \alpha = -\int_t^{t_f} S^T R^{-1} v \, d\tau \quad (A.7)$$

and differentiate both sides to obtain

$$\dot{\phi}^T \alpha + \phi^T \dot{\alpha} = S^T R^{-1} v \ . \quad (A.8)$$

Now, substitute from (A.5) into (A.8). It is easily seen that

$$\dot{\alpha} = -f_x{}^T\alpha + h_x{}^T R^{-1} v \ . \tag{A.9}$$

Note that backward integration of (A.9) from a boundary condition of $\alpha(t_f) = 0$ yields

$$\alpha(t_o) = -\int_{t_o}^{t_f} S^T R^{-1} v \, d\tau \ . \tag{A.10}$$

The development of the information matrix follows in similar fashion. Let

$$\phi^T M \phi = \int_{t}^{t_f} S^T R^{-1} S \, d\tau \tag{A.11}$$

and differentiate both sides to obtain

$$\dot{\phi}^T M\phi + \phi^T \dot{M}\phi + \phi^T M\dot{\phi} = -S^T R^{-1} S \ . \tag{A.12}$$

Now, substitute from (A.5) into (A.12). It can be seen that

$$\dot{M} = -Mf_x - f_x{}^T M - h_x{}^T R^{-1} h_x \tag{A.13}$$

so that backward integration from a boundary condition of $M(t_f) = 0$ yields

$$M(t_o) = \int_{t_o}^{t_f} S^T R^{-1} S \, d\tau \ . \tag{A.14}$$

Hence, the MNR method is equivalent to the nonlinear smoothing algorithm presented here for no process noise and with the aforementioned approximation of $\mathcal{H}_{xx}$.

APPENDIX B

THE MNR METHOD AND SMOOTHING (DISCRETE CASE)

The discrete version of the modified Newton-Raphson (MNR) method (refs. 6 and 7) determines the initial condition for a system modelled as

$$x(i + 1) = f[x(i)] \;, \qquad x_o \text{ unknown} \tag{B.1}$$

$$z(i + 1) = h[x(i + 1)] + v(i + 1) \tag{B.2}$$

such that a performance measure

$$J = (1/2)(x_o - \bar{x}_o)^T P_o^{-1}(x_o - \bar{x}_o) + (1/2)\sum_{i=0}^{N-1} v^T(i + 1)R^{-1}v(i + 1) \tag{B.3}$$

is minimized. In this formulation, $\bar{x}_o$ is an <u>a priori</u> estimate and P_o, R are weighting matrices. The MNR method computes a change in initial condition at each iteration of

$$\delta x_o = -\left[P_o^{-1} + \sum_{i=0}^{N-1} S^T(i + 1)R^{-1}S(i + 1)\right]^{-1}\left[P_o^{-1}(x_o - \bar{x}_o)\right.$$
$$\left. - \sum_{i=0}^{N-1} S^T(i + 1)R^{-1}v(i + 1)\right] \tag{B.4}$$

where

$$S(i + 1) = h_x\phi(i + 1) \;; \qquad \phi(i + 1) = f_x\phi(i) \;, \qquad \phi(0) = I \tag{B.5}$$

and $\phi(i)$ is the "sensitivity function"

$$\phi(i) = \partial x(i)/\partial x_o \;. \tag{B.6}$$

It is shown here that the MNR method and the nonlinear smoothing algorithm derived in section IV are mathematically equivalent for a system without process noise and with the approximation

$$\mathcal{H}_{xx} \simeq f_x^T h_x^T R^{-1} h_x f_x \;. \tag{B.7}$$

First, let

$$\phi^T(i)\alpha(i) = -\sum_{j=i}^{N-1} S^T(j+1)R^{-1}v(j+1) \tag{B.8}$$

and form the difference

$$\phi^T(i)\alpha(i) - \phi^T(i+1)\alpha(i+1) = -S^T(i+1)R^{-1}v(i+1) \ . \tag{B.9}$$

Rearrange terms and substitute from (B.5) to obtain

$$\alpha(i) = f_x{}^T[\alpha(i+1) - h_x{}^T R^{-1}v(i+1)] \tag{B.10}$$

which is equivalent to the relation for the costate variable given in (4.33). Notice that (B.8) with $i = 0$ becomes

$$\alpha(0) = -\sum_{j=0}^{N-1} S^T(j+1)R^{-1}v(j+1) \ . \tag{B.11}$$

The development for the information matrix follows in similar fashion. Let

$$\phi^T(i)M(i)\phi(i) = \sum_{j=i}^{N-1} S^T(j+1)R^{-1}S(j+1) \tag{B.12}$$

and form the difference

$$\phi^T(i)M(i)\phi(i) - \phi^T(i+1)M(i+1)\phi(i+1) = S^T(i+1)R^{-1}S(i+1) . \tag{B.13}$$

Rearrange terms and substitute from (B.5) to obtain

$$M(i) = f_x{}^T[M(i+1) + h_x{}^T R^{-1}h_x]f_x \tag{B.14}$$

which is equivalent to the expression for the information matrix given in (4.33). Notice that (B.13) with $i = 0$ becomes

$$M(0) = \sum_{j=0}^{N-1} S^T(j+1)R^{-1}S(j+1) \tag{B.15}$$

With the substitution of (B.11) and (B.15) into (B.4), the expression for δx_0 is the same as that given in (4.17). Hence, the MNR method is equivalent to the discrete smoothing algorithm presented in section IV, for the case of no process noise, and with the aforementioned approximation of $\mathcal{H}_{xx}$.

APPENDIX C

A COVARIANCE ALGORITHM FOR CONTINUOUS SMOOTHING

In this appendix, another algorithm for solution of the continuous nonlinear smoothing problem is presented. The algorithm provides a "forward covariance filter," "backward covariance smoother" solution of the linear TPBVP that results from the accessory minimization problem defined in section III. There it was shown that the approximations of (3.24) simplified the algorithm and extended its radius of convergence. By introducing the variable

$$\rho = \lambda + \delta\lambda \tag{C.1}$$

an equivalent linear TPBVP was determined as

$$\delta\dot{x} = f_x\delta x - f_w Q f_w^T \rho - f_w w \ , \qquad \delta x(t_o) = \delta x_o \tag{C.2}$$

$$\dot{\rho} = -h_x^T R^{-1} h_x \delta x - f_x^T \rho + h_x^T R^{-1} v \ , \qquad \rho(t_f) = 0 \ . \tag{C.3}$$

Another solution to (C.2) and (C.3) can be obtained using the sweep

$$\delta x = \delta\hat{x} - P\rho \tag{C.4}$$

where $\delta\hat{x} = \delta\hat{x}(t)$ and $P = P(t)$. In order to satisfy the boundary condition of (3.9), set

$$\delta\hat{x}(t_o) = \bar{x}_o - x_o \ ; \qquad P(t_o) = P_o \ . \tag{C.5}$$

Now, differentiate (C.4) and use (C.2) and (C.3) to obtain the differential equations

$$\dot{P} = f_x P + P f_x^T + f_w Q f_w^T - KRK^T \tag{C.6}$$

$$\delta\dot{\hat{x}} = f_x \delta\hat{x} - f_w w + K(v - h_x \delta\hat{x}) \tag{C.7}$$

where

$$K = Ph^T R^{-1} \ . \tag{C.8}$$

Equations (C.6) and (C.7) constitute a forward covariance filter, that is, their solutions are propagated forward from $t = t_o$ until $t = t_f$. To complete the solution of the TPBVP, substitute (C.4) into (C.3) and obtain the backward smoother

$$\dot{\rho} = -(f_x - Kh_x)^T\rho + h_x{}^T R^{-1}(v - h_x\delta\hat{x}) \ , \qquad \rho(t_f) = 0 \ . \tag{C.9}$$

While (C.9) is being propagated backward from $t = t_f$ until $t = t_o$, the updated value of the unknown forcing function can be determined from

$$w = -Qf_w{}^T\rho \ . \tag{C.10}$$

Equation (C.10) follows directly from (3.10), (3.28), and (C.1). At the conclusion of the backward pass, the updated initial condition can be determined from

$$x_o = \bar{x}_o - P_o\rho_o \tag{C.11}$$

where $\rho_o = \rho(t_o)$. This relation is obtained from (C.4) and (C.5).

The steps in the algorithm can now be summarized as follows:

(1) Use x_o,w(t) obtained from the preceding iteration (or an initial guess) to compute a smoothed trajectory x(t) from (2.1) and the performance measure from (2.3).

(2) Solve the "forward covariance filter" (C.6) and (C.7) to obtain K(t) and $\delta\hat{x}(t)$. Store the elements necessary for the next step.

(3) Solve the "backward covariance smoother" (C.9) and evaluate w(t) from (C.10). Determine the updated initial condition x_o from (C.11).

(4) Iterate until the performance measure has reached a minimum.

The algorithm presented here is, in effect, a linear Kálmán filter-smoother. It has the advantage of not requiring a matrix inversion to determine δx_o. However, it may be difficult to "start" the filter in the absence of a priori information. No such difficulty will be experienced using the algorithm of section III.

Notice that when the algorithm converges, $\delta x(t)$, $\delta w(t)$, and $\delta\lambda(t)$ vanish, so that $\rho(t) = \lambda(t)$. It is easy to show that (C.3), (C.10), and (C.11) are equivalent to (3.4) and (3.5), the conditions necessary for a minimum performance measure.

For the linear system described by (3.37) to (3.39), the covariance algorithm also converges in one iteration. To demonstrate this fact, make a change of variable

$$\delta x = x - x_n \ ; \qquad \delta\hat{x} = \hat{x} - x_n \ . \tag{C.12}$$

The resulting covariance filter-smoother equations are given by

$$\left.\begin{aligned} \dot{P} &= FP + PF^T + \Gamma Q\Gamma^T - KRK \ ; \\ \dot{\hat{x}} &= F\hat{x} + Gu + K[z - (H\hat{x} + Du)]; \end{aligned}\right\} \tag{C.13}$$

where $K = PH^T R^{-1}$, $\quad P(t_o) = P_o$, $\quad \hat{x}(t_o) = \bar{x}_o$, $\quad$ and

$$\left.\begin{aligned} \dot{\rho} &= -(F - KH)^T\rho + H^T R^{-1}[z - (H\hat{x} + Du)]; \\ w &= -Q\Gamma^T\rho \end{aligned}\right\} \tag{C.14}$$

with $\rho(t_f) = 0$. At the conclusion of the backward pass, the initial condition is obtained from

$$x_o = \bar{x}_o - P_o\rho_o \tag{C.15}$$

APPENDIX D

A COVARIANCE ALGORITHM FOR DISCRETE SMOOTHING

In this appendix, a discrete "forward covariance filter," "backward covariance smoother" algorithm for nonlinear smoothing is derived. The derivation will be done using the approximations of (4.24) to simplify the accessory minimization problem and extend the convergence interval. The development of the discrete algorithm is similar to that presented in appendix C for the continuous case. Recall that in section IV, introduction of the variable

$$\rho(i) = \lambda(i) + \delta\lambda(i) \tag{D.1}$$

led to the equivalent linear TPBVP

$$\delta x(i + 1) = f_x \delta x(i) - f_w Q f_w^T f_x^{-T} \rho(i) - f_w w(i) \ , \qquad \delta x(0) = \delta x_o \tag{D.2}$$

$$\rho(i) = f_x^T \{\rho(i + 1) - h_x^T R^{-1} [v(i + 1) - h_x \delta x(i + 1)]\} \ , \quad \rho(N) = 0 \ . \tag{D.3}$$

To solve (D.2) and (D.3), the sweep

$$\delta x(i) = \delta\hat{x}(i) - P(i)\rho(i) \tag{D.4}$$

will be used. The boundary condition of (4.9) will be satisfied if

$$\delta\hat{x}(0) = \bar{x}_o - x_o \ ; \qquad P(0) = P_o \ . \tag{D.5}$$

Now, use (D.4) in (D.2) and (D.3) to effect a separation of solutions. After some algebraic manipulation, the equations for the forward covariance filter are obtained in the form of a time update:

$$\left.\begin{aligned} \delta\bar{x}(i + 1) &= f_x \delta\hat{x}(i) - f_w w(i) \ ; \\ M(i + 1) &= f_x P(i) f_x^T + f_w Q f_w^T \ , \end{aligned}\right\} \tag{D.6}$$

and a measurement update:

$$\left.\begin{aligned} \delta\hat{x}(i + 1) &= \delta\bar{x}(i + 1) + K(i + 1)[v(i + 1) - h_x \delta\bar{x}(i + 1)] \ ; \\ K(i + 1) &= M(i + 1) h_x^T [R + h_x M(i + 1) h_x^T]^{-1} \ ; \\ P(i + 1) &= [I - K(i + 1) h_x] M(i + 1) \ . \end{aligned}\right\} \tag{D.7}$$

In order to complete the solution of the TPBVP, it is computationally advantageous to make a change of variable

$$\rho(i) = f_x^T \beta(i + 1) . \tag{D.8}$$

Substitute (D.4) and (D.8) into (D.3) and make use of the fact that

$$K(i) = P(i)\ h_x^T R^{-1} \tag{D.9}$$

to express the backward covariance smoother as

$$\left.\begin{aligned} \beta(i) &= [I - K(i)h_x]^T \rho(i) - h_x^T R^{-1}[v(i) - h_x^T \delta\hat{x}(i)] \ , \qquad \rho(N) = 0 \ ; \\ \rho(i - 1) &= f_x^T \beta(i) \ . \end{aligned}\right\} \tag{D.10}$$

During the backward pass, updated values of the forcing function estimate can be calculated from

$$w(i - 1) = -Q f_w^T \beta(i) . \tag{D.11}$$

Finally, the initial condition for the next iteration is given by

$$x_o = \bar{x}_o - P_o \rho(0) . \tag{D.12}$$

Equation (D.11) is obtained using (4.24), (4.27), (4.7), and (D.10) with (4.10), while (D.12) follows from (D.4) and (D.5).

The steps in the algorithm are summarized as follows:

(1) Use x_o and $[w(i)]$ obtained from the preceding iteration (or an initial guess) to compute a smoothed trajectory $[x(i)]$ from (2.4), and the value of J from (2.6).

(2) Solve the "forward covariance filter" (D.6) and (D.7) to obtain $[K(i)]$ and $[\delta\hat{x}(i)]$. Store elements necessary for the next step.

(3) Solve the "backward covariance smoother" (D.10) and evaluate $[w(i)]$ from (D.11). Determine the updated initial condition x_o from (D.12).

(4) Iterate until the performance measure J is minimized.

REFERENCES

1. Bryson, A. E.; and Ho, Y. C.: Applied Optimal Control. Waltham MA: Blaisdell, 1969, Ch. 2 and 6.

2. Bryson, A. E.; and Frazier, M.: Smoothing for linear and nonlinear dynamic systems, Wright-Patterson Air Force Base, Ohio, TDR 63-119, pp. 354-364, 1963.

3. Cox, H.: On the estimation of state variables and parameters for noisy systems, IEEE Trans. Automat. Contr., vol. AC-9, pp. 5-12, Jan. 1964.

4. Meditch, J. S.: A survey of data smoothing for linear and nonlinear dynamic systems, Automatica, vol. 9, pp. 151-162, 1973.

5. Gelb, A., ed.: Applied Optimal Estimation. Cambridge, MA: M.I.T. Press, 1974, Ch. 6.

6. Grove, R. D.; Bowles, R. L.; and Mayhew, S. C.: A procedure for estimating stability and control parameters from flight-test data by using maximum-likelihood methods. NASA TN D-6735, May 1972.

7. Wingrove, R. C.: Quasilinearization technique for estimating states from flight data, J. Aircraft, vol. 10, no. 5, pp. 303-307, May 1973.

8. McReynolds, S. R.; and Bryson, A. E.: A successive sweep method for solving optimal programming problems, Proc. JACC, Aug. 1965, pp. 551-555.

9. Sage, A. P.; and Melsa, J. L.: System Identification. New York: Academic Press, 1971, Ch. 3 and 4.

10. Meditch, J. S.: Stochastic Optimal Linear Estimation and Control. New York, N.Y.: McGraw-Hill, 1969.

11. Comstock, D. W., ed.: Handbook of Data Reduction Methods. White Sands Missile Range, New Mexico, Tech. Rept., March 1969.

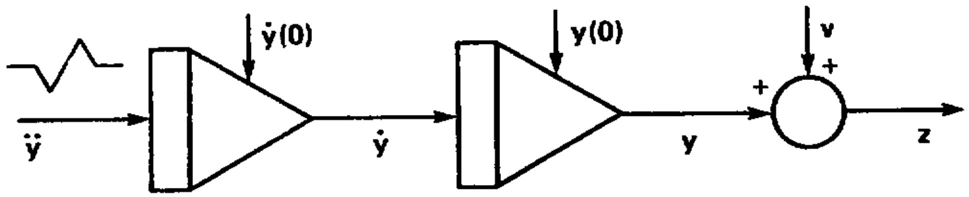
12. Bach, R. E., Jr.; and Chen, F. Y.: Filtering and smoothing techniques for parameter estimation, presented at the 9th Asilomar Conference on Circuits, Systems, and Computers, Pacific Grove, CA, Nov. 1975.

TABLE I.- MONTE-CARLO RESULTS: ESTIMATION OF p, y_o ONLY

Q/R	$\hat{p}$	σ_p^{mc}	σ_p^{cr}	$\hat{y}_o$	$\sigma_{y_o}^{mc}$	$\sigma_{y_o}^{cr}$
0	-0.997	0.024	0.025	0.002	0.023	0.022
1	-1.000	0.029	0.025	0.004	0.023	0.022
10	-1.001	0.057	0.025	0.003	0.035	0.022
100	-1.036	0.164	0.026	0.007	0.092	0.022
1000	-1.524	0.648	0.041	0.076	0.245	0.026

TABLE II.- MONTE-CARLO RESULTS: ESTIMATION OF p, y_o, $[w(i)]$

Q/R	$\hat{p}$	σ_p^{mc}	σ_p^{cr}	$\hat{y}_o$	$\sigma_{y_o}^{mc}$	$\sigma_{y_o}^{cr}$
0	-0.997	0.024	0.025	0.002	0.023	0.022
1	-1.000	0.029	0.030	0.005	0.024	0.023
10	-0.999	0.057	0.056	0.003	0.031	0.030
100	-0.993	0.116	0.157	0.001	0.054	0.050
1000	-0.978	0.516	0.486	0.001	0.093	0.096



$x_1 = y; x_2 = \dot{y}; w = \ddot{y}:$

$\dot{x}_1 = x_2$
$\dot{x}_2 = w$ $\quad z = x_1 + v$

Figure 1.- Continuous model for derivative estimation.

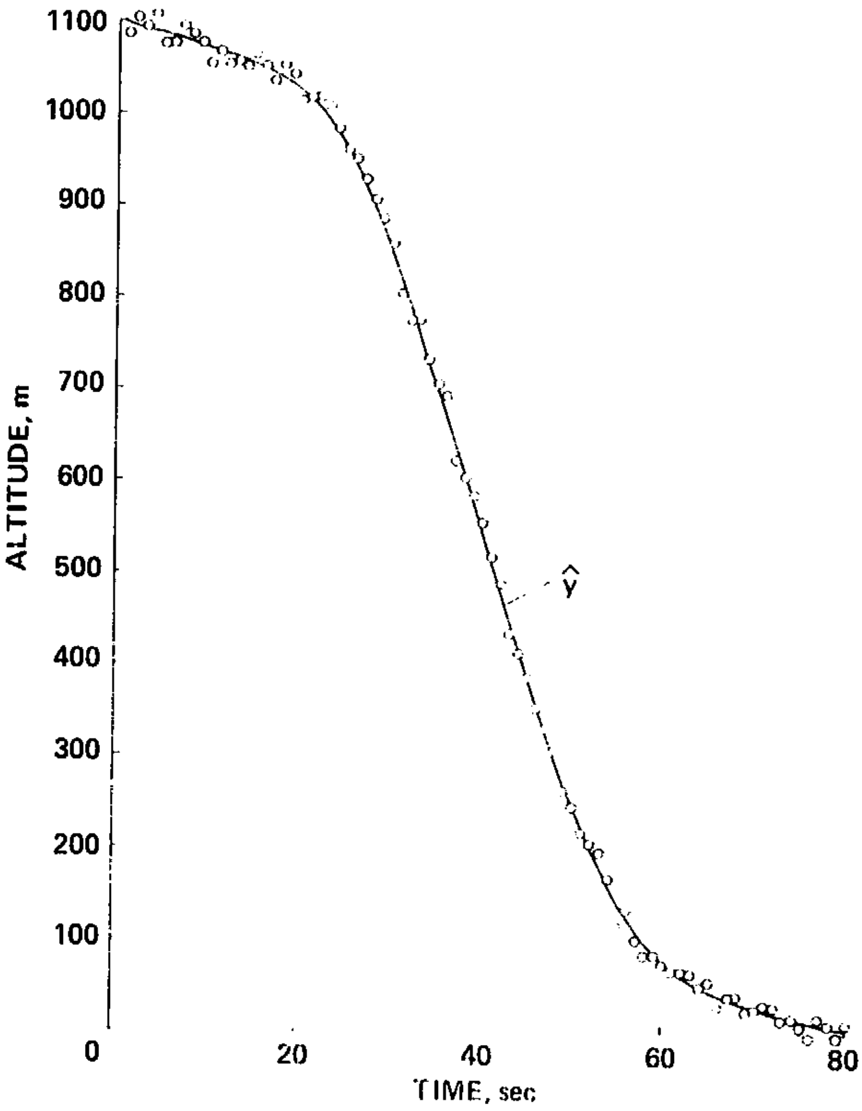
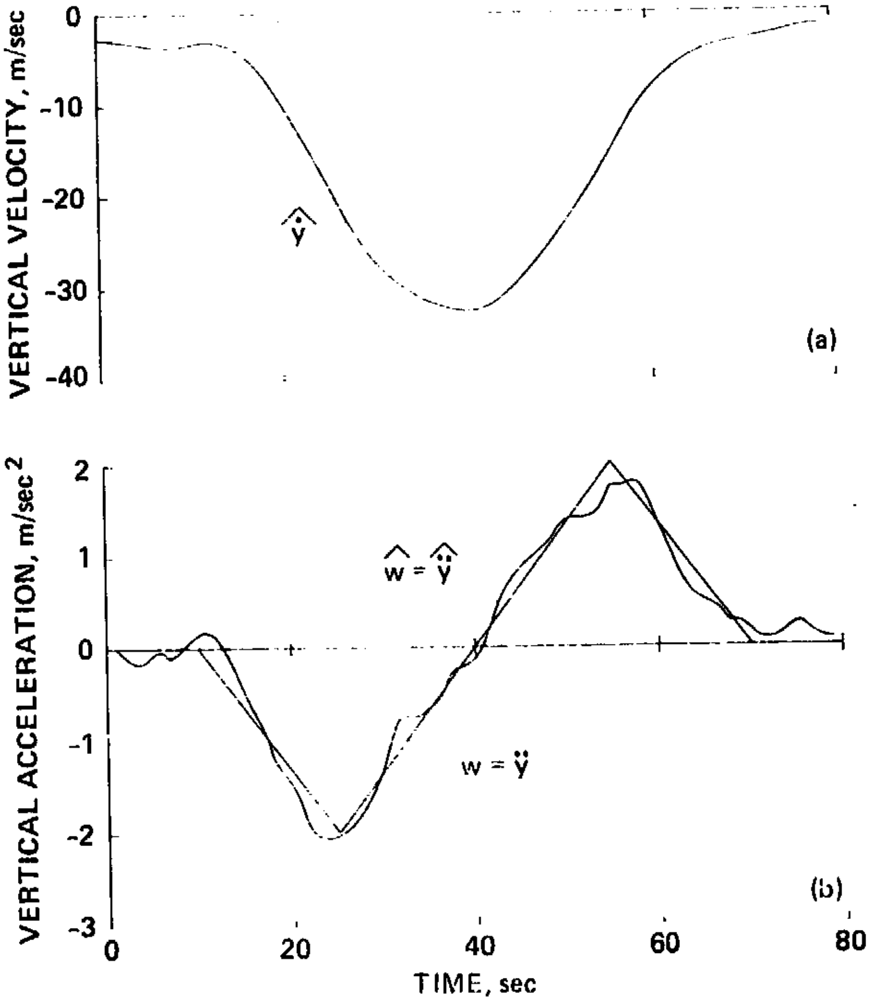


Figure 2.- Altitude data record and estimate $\hat{y}$.



(a) Vertical velocity estimate $\hat{\dot{y}}$.

(b) True acceleration $\ddot{y}$, and estimate $\hat{\ddot{y}}$.

Figure 3.- Trajectories for derivative estimation problem.

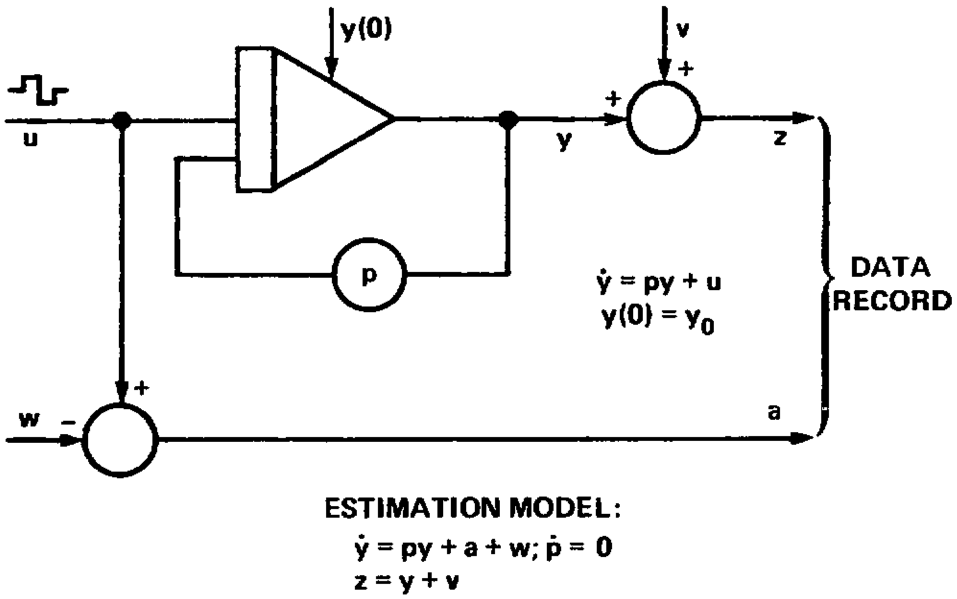


Figure 4.- Continuous model for parameter estimation.