Synthetic Approach to Optimal Filtering

James Ting-Ho Lo, *Member, IEEE*

***Abstract*—As opposed to the analytic approach used in the modern theory of optimal filtering, a synthetic approach is presented. The signal/sensor data, which are generated by either computer simulation or actual experiments, are synthesized into a filter by training a recurrent multilayer perceptron (RMLP) with at least one hidden layer of fully or partially interconnected neurons and with or without output feedbacks.**

The RMLP, after adequate training, is a recursive filter optimal for the given structure, with the lagged feedbacks carrying the optimal conditional statistics at each time point. Above all, it converges to the minimum variance filter as the number of hidden neurons increases. We call such an RMLP a neural filter.

Simulation results show that the neural filters with only a few hidden neurons consistently outperform the extended Kalman filter and even the iterated extended Kalman filter for the simple nonlinear signal/sensor systems considered.

I. Introduction

THE MODERN THEORY of optimal filtering is concerned mainly with the problem of estimating a known function $\psi(x(t))$ of a signal process $x(t)$ given measurements $y^t := \{y(\tau), 1 \leq \tau \leq t\}$ that are described by

$$x(t+1) = f(x(t)) + G(x(t))w(t) \tag{1.1}$$

$$y(t) = h(x(t)) + v(t) \tag{1.2}$$

where $w(t)$ and $v(t)$ are Gaussian noise processes with given statistics and f, G, and h are known measurable functions with such appropriate dimensions and properties that x and y exist and are unique [1], [9].

The methodology used in the modern theory is analytic in nature. A solution is supposed to consist of analytic formulas and/or equations that describe the structures and determine the parameters of the filter. To reach such a solution, deductive reasoning is used and more often than not, many assumptions are made to make some special cases analytically tractable. For instance, the celebrated Kalman filter was derived under the assumption that the functions f and h are linear in $x(t)$, the function G does not depend on $x(t)$, and $w(t)$ and $v(t)$ are Gaussian sequences [7], [8]. In fact, the general model, (1.1) and (1.2), contains such assumptions as the Markov property, the Gaussian distributions, and the additive measurement noises.

Manuscript received August 6, 1992; revised February 23, 1993. This work was supported in part by Rome Laboratory under Contract No. F30602-91-C-0033. A shorter version of this paper was presented and won the Best Paper Award at the 1992 Workshop on Neural Networks (WNN-92/Houston), which was sponsored by the Society for Computer Simulation, cosponsored by NASA in cooperation with SPIE and INNS, with the IEEE Neural Network Council as a participating Society.

The author is with the Department of Mathematics and Statistics, University of Maryland Baltimore County, Baltimore, MD 21228 USA.

IEEE Log Number 9208868.

The purpose of this paper is to present an alternative approach based on synthesis. Instead of deriving formulas and/or equations, we synthesize realizations of $x(t)$ and $y(t)$ into a filter. If a mathematical model such as (1.1) and (1.2) is available, these realizations are easily generated by computer simulations. Otherwise, they can be collected by actual experiments. Virtually none of aforementioned assumptions is required for the synthesis.

This synthetic approach was inspired by the recent resurgence of the study on artificial neural networks, especially the multilayer perceptrons (MLP's) and MLP-based recurrent neural networks. The synaptic weights of such a network are not evaluated by formulas or equations, but are determined by training the network using the desired input/output pairs [3], [6], [10], [13], [14]. This training is the main ingredient of synthesizing. Of course, we do not have to use these neural networks for synthesizing filters. Nevertheless, there are three reasons for using them: First, any measurable function with a compact support can be approximated to any degree of accuracy by an MLP with even a single layer of hidden neurons [2], [4], [5]. Second, the feedbacks in a recurrent network provide the "state" in building a dynamical system. Third, the massively parallel nature of MLP's and MLP-based recurrent networks make them most suitable for real-time filtering.

To simplify our discussion, we will, in this paper, mainly consider an MLP with a single hidden layer of neurons, whose states are fully [3], [12] or partially fed back to one another. Our results can be easily generalized or extended to many other architectures.

In training a recurrent MLP into a filter, the measurement sequence $y(t)$ are used as the inputs consecutively in time and the signal sequence $\psi(x(t))$ are the corresponding desired outputs. We note that an estimate of $\psi(x(t))$is what the recurrent MLP is intended to approximate and is thus supposed to be the desired output for training. However, a good estimate such as the minimum-variance estimate is very difficult to obtain beforehand. The use of $\psi(x(t))$ as the desired output will be justified later on. A requirement here that is not needed in the modern theory of optimal filtering is that $x(t)$ and $y(t)$ stay in a compact set. The fullfilment of it is easy to justify in the real world. The model, (1.1) and (1.2), violates this requirement. However, it is only an idealization, whose $x(t)$ and $y(t)$ may stray out of a compact region with arbitrarily small probability provided that the region is sufficiently large.

The training process minimizes the mean square error between the network outputs $\hat{\psi}(x(t))$ and the desired outputs $\psi(x(t))$. Consequently, after adequate training, the estimates $\hat{\psi}(x(t))$ produced by the network are minimum-variance for

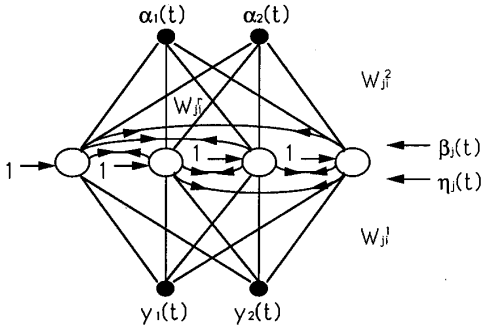


Fig. 1. A recurrent MLP with fully interconnected hidden neurons. The symbols are defined at the beginning of Section II. The dynamics are described by (2.1)–(2.3).

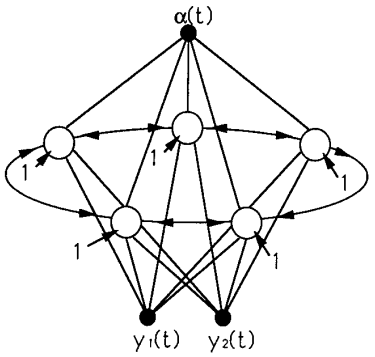


Fig. 2. A recurrent MLP with ring-connected hidden neurons. This is a special case of the RMLP shown in Fig. 1. The dynamics here are described by (2.1), (2.4) and (2.3).

the given network architecture. It is easy to see that the mean square error between $\hat{\psi}(x(t))$ and $\psi(x(t))$ after adequate training decreases, as the number of hidden neurons in the recurrent MLP increases. We call such a recurrent MLP a neural filter. The main theorem in this paper is that the output $\hat{\psi}(x(t))$ of a neural filter converges to the minimum variance estimate, which is also the conditional expectation, $E[\psi(x(t))|y^t]$, as the number of fully interconnected hidden neurons increases.

Four numerical examples are thoroughly worked out and reported. Two of them show that even the neural filters with a few hidden neurons consistently outperform the extended Kalman filter and the iterated extended Kalman filter for nonlinear systems of the type (1.1) and (1.2). The third shows that a neural filter with a sufficient number of neurons is indistinguishable from the Kalman filter for a linear system of the type (1.1) and (1.2). In the last example, neither the signal nor the measurement can be described by (1.1) or (1.2). Nevertheless, the neural filter works satisfactorily.

A close look at the proof of the main theorem indicates that fully interconnectedness of the hidden layer is not necessary for the convergence to the minimum variance filter. Another architecture is also tested in the numerical examples. The hidden neurons in this architecture are arranged in a ring and each is fed back to its two neighbors after a unit time delay. As it stands, the main theorem does not apply to this architecture. However, in all the four examples, the filtering performance of a recurrent MLP with this architecture is very close to that of a recurrent MLP with fully interconnected hidden neurons. Notice that the total number of connections in the architecture with fully interconnected hidden neurons is much greater than that in the architecture with the ring topology, especially when the number of neurons is large. This raises the question, "What architecture(s) need(s) the smallest numbers of neurons and connections for approximating the minimum variance filter to any degree of accuracy?" The question is under investigation and an answer will soon be reported.

II. MLP's with Interconnected Hidden Neurons

A typical recurrent MLP (RMLP) with a single hidden layer of fully interconnected neurons is illustrated in Fig. 1. It has 2 input variables $y_i(t)$, 4 hidden neurons, and 2 output variables $\alpha_i(t)$, where t indicates the dependence on time. Denote the weight from the ith input to neuron j by w^1_{ji}, the weight from neuron i to the jth output by w^2_{ji}, and the weight for the lagged feedback from neuron i to neuron j by w^r_{ji}. The activation level $\beta_j(t)$ of and the weighted sum $\eta_j(t)$ in neuron j satisfy

$$\beta_{j(t)} = a(\eta_j(t)), \tag{2.1}$$

$$\eta_j(t) = w^1_{j0} + \sum_{i=1}^{m} w^1_{ji} y_i(t) + \sum_{i=1}^{q} w^r_{ji} \beta_i(t-1), \tag{2.2}$$

where a is a monotone increasing neuron activation function, say tanh. The output $\alpha_i(t)$ is then determined by

$$\alpha_i(t) = w^2_{i0} + \sum_{j=1}^{q} w^2_{ij} \beta_j(t), \text{ for } i = 1, \cdots, k, \tag{2.3}$$

where m, q, and k are the numbers of inputs, neurons, and outputs, respectively. It is assumed in this paper that the activation levels are initialized at $\beta_j(0) = B$, $j = 1, \cdots, q$, where B is a number that the probability that any $y_{i(t)}$ equals B is 0. In many applications, it is better to optimize $\beta_j(0)$ together with w in training. Optimized $\beta_j(0)$ carry optimal statistics of $x(0)$.

A typical RMLP with a single hidden layer of ring-connected neurons is illustrated in Fig. 2. It has 2 input variables $y_i(t)$, 5 hidden neurons, and 2 output variables $\alpha_i(t)$. Using the same symbols as defined above, the RMLP is specified by equations (2.1), (2.3), and

$$\eta_j(t) = w^1_{j0} + \sum_{i=1}^{m} w^1_{ji} y_i(t) + w^r_{j\rho(j-1)} \beta_{\rho(j-1)}(t-1) + w^r_{j\rho(j+1)} \beta_{\rho(j+1)}(t-1), \tag{2.4}$$

where the function ρ is defined by $\rho(j) = j$, if $1 \le j \le q$; $= 1$, if $j = q+1$; $= q$, if $j = 0$. It is also assumed here that $\beta_j(0) = 0$, $j = 1, \cdots, q$.

As discussed in the preceding section, by either computer simulation or experiments, we may have available a finite set of signal/measurement sequences, $(x(t,\omega), y(t,\omega))$, $t = 1, \cdots, T$, $\omega \in S$. It is assumed that the finite set S is a sample from the sample space Ω and adequately reflects the

joint probability distributions of the signal and measurement prosesses $x(t)$ and $y(t)$.

Suppose that $\psi(x(t)) = [\psi_1(x(t)), \cdots, \psi_k(x(t))]^T$ are what we want to estimate and are the desired outputs for the actual outputs $\alpha_1(t), \cdots, \alpha_k(t)$. Using the m components of the measurement vector $y(t) = [y_1(t), \cdots, y_m(t)]^T$ as the inputs, the training data consists of the I/O pairs $(y(t,\omega), \psi(x(t,\omega)))$, $t = 1, \cdots, T$, $\omega \in S$.

The training is to minimize, by the variation of the RMLP weights w, the mean square error

$$C(w) = \frac{1}{T(\#S)} \sum_{\omega \in S} \sum_{t=1}^{T} \sum_{l=1}^{k} (\alpha_l(t,\omega) - \psi_l(x(t,\omega)))^2 \quad (2.5)$$

where $\#S$ is the number of elements in S and $\alpha_l(t,\omega)$ is the lth actual RMLP output at time t corresponding to the input sequence $y(\tau,\omega)$, $\tau = 1, \cdots, t$. We note here that $\alpha_l(t,\omega)$ is a function of $y(\tau,\omega)$, $\tau = 1, \cdots, t$.

A good training method, which is used in our simulation, is first to evaluate the gradient of the above error function with respect to w by the time-dependent recurrent backpropagation by Werbos [15] and Pearlmutter [11] and then to use it in a conjugate gradient algorithm for optimization.

In general, many RMLP's are trained and one is selected in comparison of estimation accuracy versus network complexity to maximize cost-effectiveness. This selected RMLP is called a neural filter.

III. Convergence to the Minimum Variance Filter

The collection of all k-dimensional random vectors $x = [x_1, \cdots, x_k]^T$ is a Hilbert space L_2 with the norm $E[\|x\|^2] = E[\sum_{l=1}^{k} x_l^2]$, where E denotes the expectation or the average over the sample space Ω with the σ-field $\mathcal{A}$ of events and the probability measure P.

By simple calculation, we have for any Borel-measurable function f of $y^t := \{y(1), \cdots, y(t)\}$,

$$\begin{aligned} C' &:= \frac{1}{T} \sum_{t=1}^{T} E[\|\psi(x(t)) - f(y^t)\|^2] \\ &= \frac{1}{T} \sum_{t=1}^{T} \{E[\|\psi(x(t)) - E[\psi(x(t))|y^t]\|^2] \\ &\quad + E[\|E[\psi(x(t))|y^t] - f(y^t)\|^2]\}, \end{aligned} \quad (3.1)$$

which shows that the conditional expectation $E[\psi(x(t))\ |y^t]$ is the minimum variance estimate of $\psi(x(t))$ given y^t.

Under the assumption that averaging over S is indistinguishable from taking expectation over Ω, we see, by comparing $C(\omega)$ in (2.5) and C' in (3.1), that

$$\begin{aligned} C(w) &= \frac{1}{T(\#S)} \sum_{\omega \in S} \sum_{t=1}^{T} \|\psi(x(t,\omega)) - E[\psi(x(t))|y^t(\omega)]\|^2 \\ &\quad + \frac{1}{T(\#S)} \sum_{\omega \in S} \sum_{t=1}^{T} \|\alpha(t,\omega) - E[\psi(x(t))|y^t(\omega)]\|^2 \end{aligned}$$

Hence, minimizing $C(w)$ is equivalent to minimizing the second term above. This justifies using $\psi(x(t))$ as the desired output of the RMLP under training.

Recall the neural filter architecture described in Section II. We observe that adding a neuron to a hidden layer decreases $\min_w C(w)$. We are now ready to state and prove the main theorem of this paper.

Theorem: Consider the n-dimensional random process $x(t)$ and the m-dimensional random process $y(t)$, $t = 1, \cdots, T$ defined on a probability space $(\Omega, \mathcal{A}, P)$. Assume that the range $\{y(t,\omega) | t = 1, \cdots, T, \omega \in \Omega\} \subset R^m$ is compact and ψ is an arbitrary k-dimensional Borel function of $x(t)$ with finite second moments $E[\|\psi(x(t))\|^2]$, $t = 1, \cdots, T$. Let $\alpha(t)$ denote the k-dimensional output at time t of an RMLP that has taken the inputs, $y(1), \cdots, y(t)$, in the given order.

1) *Given $\epsilon > 0$, there exists an RMLP with one hidden layer of fully interconnected neurons such that*

$$\frac{1}{T} \sum_{t=1}^{T} E[\|\alpha(t) - E[\psi(x(t))|y^t]\|^2] < \epsilon.$$

2) *If the RMLP has one hidden layer of N neurons fully interconnected and the output $\alpha(t)$ is written as $\alpha(t; N)$ here to indicate its dependency on N, then*

$$r(N) := \min_w \frac{1}{T} \sum_{t=1}^{T} E[\|\alpha(t,N) - E[\psi(x(t))|y^t]\|^2] \quad (3.2)$$

is monotone decreasing and converges to 0 as N approaches infinity.

Proof: Since (a) is an immediate consequence of (b), we shall only prove (b).

It is obvious that $r(N)$, $N = 1, 2, \cdots$, are all bounded from below by 0. Hence the monotone decreasing sequence $r(N)$ converges and we denote the limit by $r(\infty)$. To prove that $r(\infty) = 0$, it suffices to show that for any $\epsilon > 0$, there is an integer M such that $|r(M)| < \epsilon$. This will be shown in the following for the case in which the measurement process $y(t)$ is scalar-valued, i.e., $m = 1$. The proof for $m > 1$ is similar and omitted.

As illustrated in Fig. 3, the first $T-1$ neurons can be so interconnected that for $t = 0, \cdots, T-1$, $\beta_1(t+1) = a(y(t+1))$ and for $\tau = 2, \cdots, T-1$,

$$\beta_\tau(t+1) = a(\beta_{\tau-1}(t)). \quad (3.3)$$

These neurons are initialized at $\beta_\tau(0) = B$, $\tau = 1, \cdots, T-1$, where B is a point in R^1 outside the compact set $\{y(t,\omega) | t = 1, \cdots, T, \omega \in \Omega\}$. At time $t+1$, the lagged feedbacks from them are

$$\begin{aligned} &(\beta_{T-1}(t), \cdots, \beta_{t+1}(t), \beta_t(t), \beta_{t-1}(t), \cdots, \beta_1(t)) \\ &\quad = (B, \cdots, B, a^{\circ t}(y(1)), a^{\circ(t-1)}(y(2)), \cdots, a(y(t))), \end{aligned}$$

where $a^{\circ t}$ denotes the t-fold composition $a \circ a \circ \cdots \circ a$ of a. Including both these feedbacks and the input node, we have the T-dimensional vector at time $t+1$,

$$z(t+1) = [\beta_{T-1}(t), \cdots, \beta_1(t), y(t+1)]^T,$$

to use as the inputs to an MLP with one hidden layer.

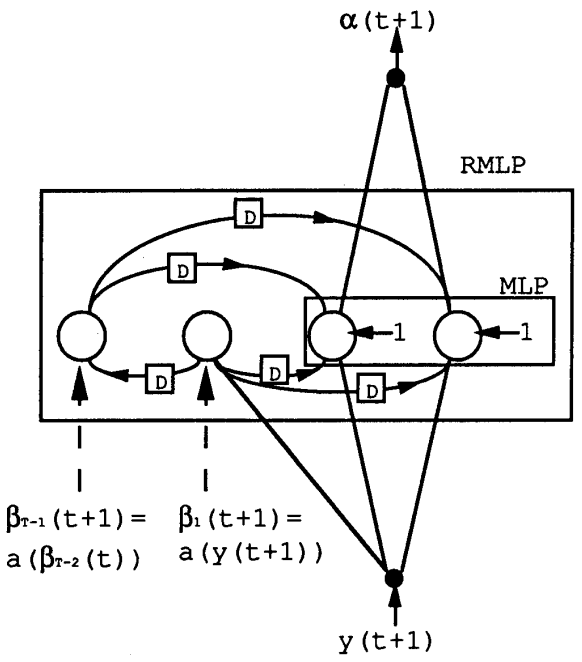


Fig. 3. A special recurrent MLP used in the proof of the main theorem.

Recall that $E[\psi(x(t+1))|a^{ot}(y(1)), a^{o(t-1)}(y(2)), \cdots, a(y(t)), y(t+1)]$ is a Borel measurable function of $a^{ot}(y(1))$, $\cdots$, $a(y(t))$ and $y(t+1)$. Let us denote it by $\lambda_{t+1}(a^{ot}(y(1)), \cdots, a(y(t)), y(t+1))$. Now let us consider the mapping $f : R^T \to R^k$ defined by

$$f(\xi_1, \cdots, \xi_T) = \begin{cases} \lambda_1(\xi_T), & \text{if } \xi_1 = \cdots = \xi_{T-1} = B, \xi_T \neq B \text{ and } \lambda_1(\xi_T) \text{ defined;} \\ \lambda_2(\xi_{T-1}, \xi_T), & \text{if } \xi_1 = \cdots = \xi_{T-2} = B, \xi_{T-1} \neq B, \xi_T \neq B \text{ and } \lambda_2(\xi_{T-1}, \xi_T) \text{ defined;} \\ \vdots & \\ \lambda_T(\xi_1, \cdots, \xi_T), & \text{if } \xi_1 \neq B, \cdots, \xi_T \neq B \text{ and } \lambda_T(\xi_1, \cdots, \xi_T) \text{ defined;} \\ 0, & \text{otherwise.} \end{cases}$$

Let us consider also the measures μ_t defined on the Borel sets in R^T by, denoting $d\xi_1 \times \cdots \times d\xi_T$ by $d\xi$,

$$\mu_1(d\xi) = \begin{cases} \frac{1}{T} P(y(1) \in d\xi_T), & \text{if } B \in d\xi_t, \text{ for } t < T, \text{ and } B \notin d\xi_t, \text{ for } t \not< T, \\ 0, & \text{otherwise;} \end{cases}$$

$$\mu_2(d\xi) = \begin{cases} \frac{1}{T} P(a(y(1)) \in d\xi_{T-1}, y(2) \in d\xi_T), & \text{if } B \in d\xi_t, \text{ for } t < T-1, \text{ and } B \notin d\xi_t, \text{ for } t \not< T-1, \\ 0, & \text{otherwise;} \end{cases}$$

$$\vdots$$

$$\mu_T(d\xi) = \frac{1}{T} P(a^{o(T-1)}(y(1)) \in d\xi_1, \cdots, a(y(T-1)) \in d\xi_{T-1}, y(T) \in d\xi_T).$$

Notice that $\mu = \mu_1 + \cdots + \mu_T$ is also a measure on the Borel sets in R^T and that

$$\begin{aligned} \int \|f(\xi_1, \cdots, \xi_T)\|^2 \mu(d\xi) &= \frac{1}{T}\{\int \|\lambda_1(\xi_T)\|^2 \mu_1(d\xi) \\ &\quad + \int \|\lambda_2(\xi_{T-1}, \xi_T)\|^2 \mu_2(d\xi) \\ &\quad + \cdots \\ &\quad + \int \|\lambda_T(\xi_1, \cdots, \xi_T)\|^2 \mu_T(d\xi)\} \\ &= \frac{1}{T}\{E[\|E[\psi(x(1))|y(1)]\|^2] + \cdots \\ &\quad + E[\|E[\psi(x(T))|a^{o(T-1)}(y(1)), \cdots, y(T)]\|^2]\} \\ &\leq \frac{1}{T}\{E[\|\psi(x(1))\|^2] + \cdots \\ &\quad + E[\|\psi(x(T))\|^2]\} \\ &< \infty. \end{aligned}$$

Since $y(t)$, $t = 1, \cdots, T$, are assumed to have a compact range, it is easy to see that $a^{ot}(y(\tau))$, for $t = 0, \cdots, T-1$ and $\tau = 1, \cdots, T$, all have compact ranges. Hence there is a compact set W in R^T such that $\mu(W) = 1$.

Consider now the normed space $L_2(R^T, \mu)$ with the norm defined by $[\int \|g(\xi_1, \cdots, \xi_T)\|^2 \mu(d\xi)]^{1/2}$ for each k-dimensional function $g \in L_2(R^T, \mu)$. By Corollary 2.2 of the paper by Hornik, Stinchcombe, and White [5], the k-dimensional functions represented by all the MLP's with k output nodes and a single layer of hidden neurons are dense in $L_2(R^T, \mu)$. It then follows that for any $\epsilon > 0$, there is such an MLP that the function $g(\xi_1, \cdots, \xi_T)$ that the MLP represents satisfies

$$\int \|f(\xi_1, \cdots, \xi_T) - g(\xi_1, \cdots, \xi_T)\|^2 \mu(d\xi) < \epsilon.$$

Let us translate the left hand side of this inequality from $L_2(R^T, \mu)$ back to the probability space $(\Omega, \mathcal{A}, P)$:

$$\begin{aligned} &\int \|f(\xi_1, \cdots, \xi_T) - g(\xi_1, \cdots, \xi_T)\|^2 \mu(d\xi) \\ &= \frac{1}{T}\{\int \|\lambda_1(\xi_T) - g(B, \cdots, B, \xi_T)\|^2 \mu_1(d\xi) \\ &\quad + \int \|\lambda_2(\xi_{T-1}, \xi_T) - g(B, \cdots, B, \xi_{T-1}, \xi_T)\|^2 \mu_2(d\xi) \\ &\quad + \cdots \\ &\quad + \int \|\lambda_T(\xi_1, \cdots, \xi_T) - g(\xi_1, \cdots, \xi_T)\|^2 \mu_T(d\xi)\} \\ &= \frac{1}{T} \sum_{t=1}^{T} E[\|E[\psi(x(t))|y^t] \\ &\quad - g(B, \cdots, B, a^{o(t-1)}(y(1)), a^{o(t-2)}(y(2)), \cdots, y(t))\|^2] \end{aligned}$$

where the last equality holds, because the σ-fields generated by y^t and $\{a^{o(t-1)}(y(1)), a^{o(t-2)}(y(2)), \cdots, y(t)\}$ are identical due to the monotonicity of a.

We notice that the foregoing approximation MLP, $g(\xi_1, \cdots, \xi_T)$, and the $T-1$ neurons (3.3) discussed earlier form an

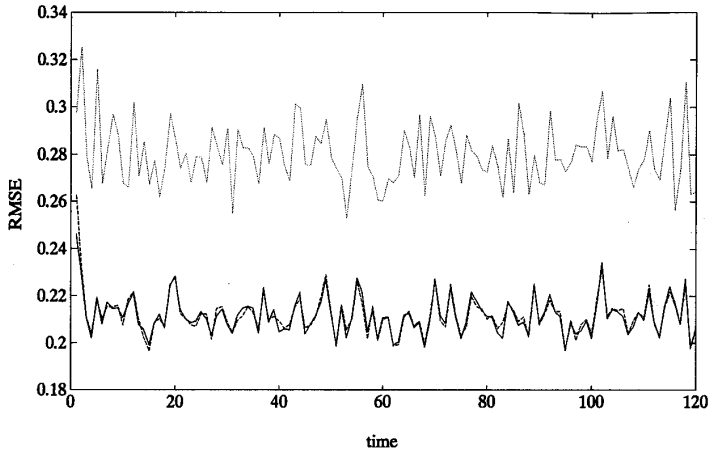


Fig. 4. RMSE's versus time of IEKF ($\cdots$), NFFN (—) and NFRN (---) for Example 1. The RMSE's over 120 time points are 0.2806, 0.2120, and 0.2122, respectively.

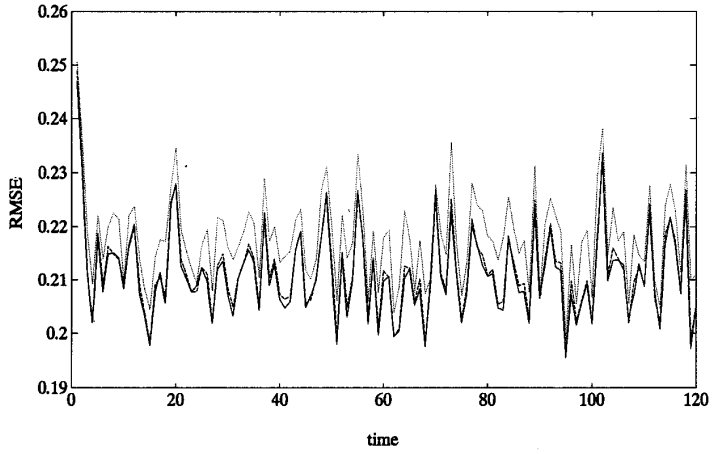


Fig. 5. RMSE's versus time of NFFN's with 2 neurons ($\cdots$), 4 neurons (---), and 8 neurons (—) for Example 1. The RMSE's over 120 time points are 0.2181, 0.2121, and 0.2114, respectively.

RMLP with a single hidden layer. The number N of hidden neurons of this RMLP is the sum of that of the MLP and $T-1$. Recalling (3.2), we have

$$r(N) \leq \frac{1}{T}\sum_{t=1}^{T} E[\|E[\psi(x(t))|y^t] - g(B,\cdots,B, a^{\circ(t-1)}(y(1)), a^{\circ(t-2)}(y(2)),\cdots,y(t))\|^2] < \epsilon,$$

since $r(N)$ minimizes the error function in (3.2) by the variation of the weights of an RMLP with exactly the same architecture as for the RMLP constructed earlier. This completes the proof.

In real-world application of a filter, the time interval in which the filter is applied is usually very long, i.e., $T \gg 1$. Having more than $T-1$ neurons is impractical. Nevertheless, the above theorem guarantees you may get arbitrarily close to the minimum variance filter, provided that a sufficient number of neurons are used.

IV. Differentiating an RMLP with Interconnected Hidden Neurons

In this section, we shall briefly review an efficient method of evaluating the gradient dC/dw, which was derived and called the time-dependent recurrent backpropagation (TDRBP) by Werbos [15] and Pearlmutter [11]. To simplify our discussion, let us consider an RMLP, for $T=3$, with a single input $y(t)$, a single input $x(t)$, and a hidden layer of q fully interconnected neurons, which are governed by (2.1), (2.2), and (2.3). The training process minimizes, by the variation of w, $C(w)$ in (2.5), which we write as

$$C(w) = \frac{1}{2}\sum_{t=1}^{T}\sum_{\omega\in S}(\alpha(t,\omega) - \xi(t,\omega))^2,$$

where $T(\#S)/2$ is suppressed for simplification.

Before deriving the formulas for $dC(w)/dw$, we observe that $\alpha(t,\omega)$ and $\beta_i(t,\omega)$, $i=1,\cdots,q$ are functions of w,

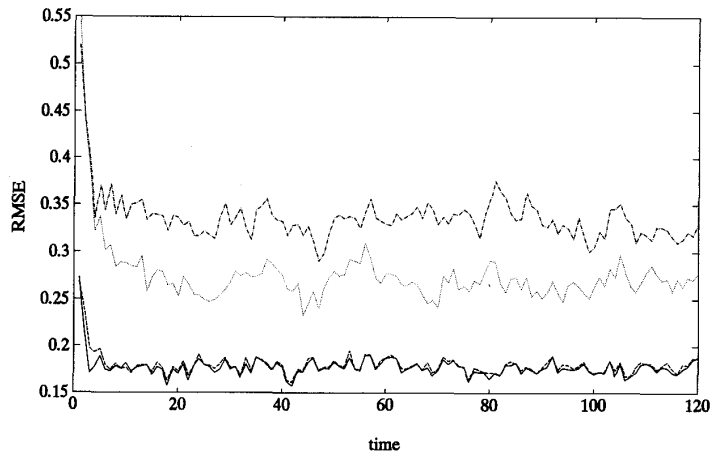


Fig. 6. RMSE's versus time of EKF (-·-·), IEKF (· · ·), NFFN (—), and NFRN (---) for Example 2. The RMSE's over 120 time points are 0.3373, 0.2778, 0.1779, and 0.1804, respectively.

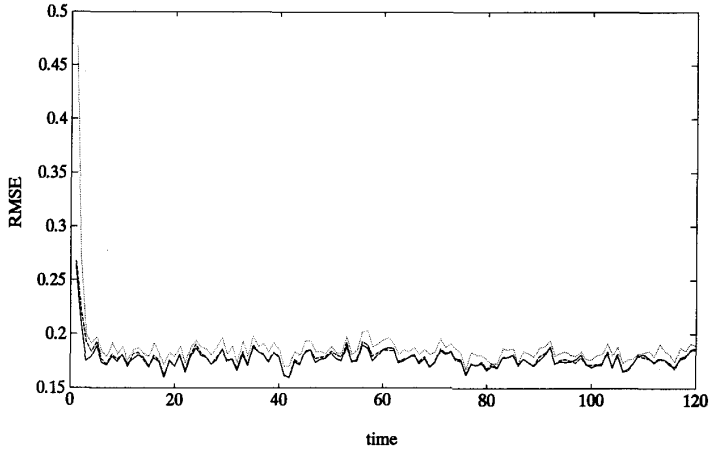


Fig. 7. RMSE's versus time of NFFN's with 2 neurons (· · ·), 4 neurons (---), and 8 neurons (—) for Example 2. The RMSE's over 120 time points are 0.1897, 0.1788, and 0.1777, respectively.

$y(t,\omega)$, and $\beta_i(t-1,\omega)$, $i=1,\cdots,q$. By the chain rule of differentiation, we have

$$\begin{aligned}\frac{dC}{dw} &= \sum_t \sum_\omega \frac{\partial C}{\partial \alpha(t,\omega)} \cdot \frac{d\alpha(t,\omega)}{dw} \\ &= \sum_\omega \{ \frac{\partial C}{\partial \alpha(1,\omega)} \cdot \frac{\partial \alpha(1,\omega)}{\partial w} \\ &\quad + \sum_{t=2}^{3} \frac{\partial C}{\partial \alpha(t,\omega)} [\sum_i \frac{\partial \alpha(t,\omega)}{\partial \beta_i(t-1,\omega)} \cdot \frac{d\beta_i(t-1,\omega)}{dw} \\ &\quad + \frac{\partial \alpha(t,\omega)}{\partial w}]\} \\ &= \sum_\omega \{ \frac{\partial C}{\partial \alpha(1,\omega)} \cdot \frac{\partial \alpha(1,\omega)}{\partial w} \\ &\quad + \frac{\partial C}{\partial \alpha(2,\omega)} [\sum_i \frac{\partial \alpha(2,\omega)}{\partial \beta_i(1,\omega)} \cdot \frac{\partial \beta_i(1,\omega)}{\partial w} \\ &\quad + \frac{\partial \alpha(2,\omega)}{\partial w}] + \frac{\partial C}{\partial \alpha(3,\omega)} (\sum_j \frac{\partial \alpha(3,\omega)}{\partial \beta_j(2,\omega)} \\ &\quad [\sum_i \frac{\partial \beta_j(2,\omega)}{\partial \beta_i(1,\omega)} \cdot \frac{\partial \beta_i(1,\omega)}{\partial w} + \frac{\partial \beta_j(2,\omega)}{\partial w}] + \frac{\partial \alpha(3,\omega)}{\partial w})\} \\ &= \sum_\omega \{ \sum_t \frac{\partial C}{\partial \alpha(t,\omega)} \cdot \frac{\partial \alpha(t,\omega)}{\partial w} \\ &\quad + \sum_i [\frac{\partial C}{\partial \alpha(2,\omega)} \cdot \frac{\partial \alpha(2,\omega)}{\partial \beta_i(1,\omega)} \\ &\quad + \frac{\partial C}{\partial \alpha(3,\omega)} \sum_j \frac{\partial \alpha(3,\omega)}{\partial \beta_j(2,w)} \cdot \frac{\partial \beta_j(2,\omega)}{\partial \beta_i(1,\omega)}] \cdot \frac{\beta_i(1,\omega)}{\partial w} \\ &\quad + \frac{\partial C}{\partial \alpha(3,\omega)} \sum_i \frac{\partial \alpha(3,\omega)}{\partial \beta_i(2,\omega)} \cdot \frac{\partial \beta_i(2,\omega)}{\partial w} \} \\ &= \sum_\omega \{ \sum_t \frac{\partial C}{\partial \alpha(t,\omega)} \cdot \frac{\partial \alpha(t,\omega)}{\partial w} \\ &\quad + \sum_{t=1}^{T} \sum_i \frac{\partial^+ C}{\partial \beta_i(t,\omega)} \cdot \frac{\partial \beta_i(t,\omega)}{\partial w} \}, \end{aligned} \tag{4.1}$$

where $\partial^+ C/\partial \beta_i(t,\omega)$ is called an ordered partial derivative by Werbos [15] and denotes the partial derivative of C w.r.t.

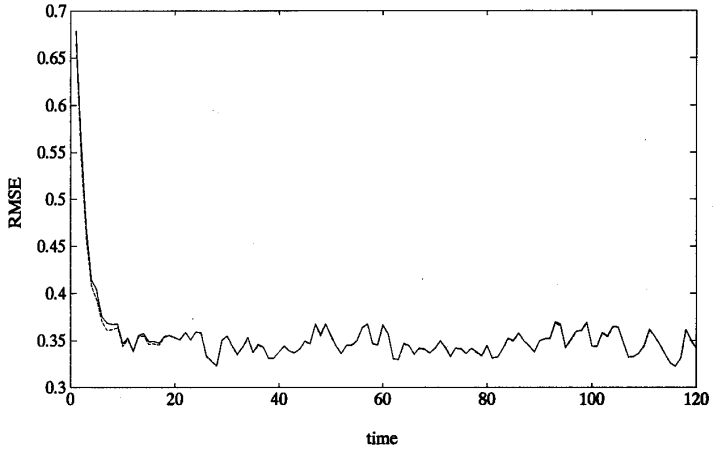


Fig. 8. RMSE's versus time of the Kalman filter (---) and NFRN (—) for Example 3. The RMSE's over 120 time points are 0.3549 and 0.3563, respectively.

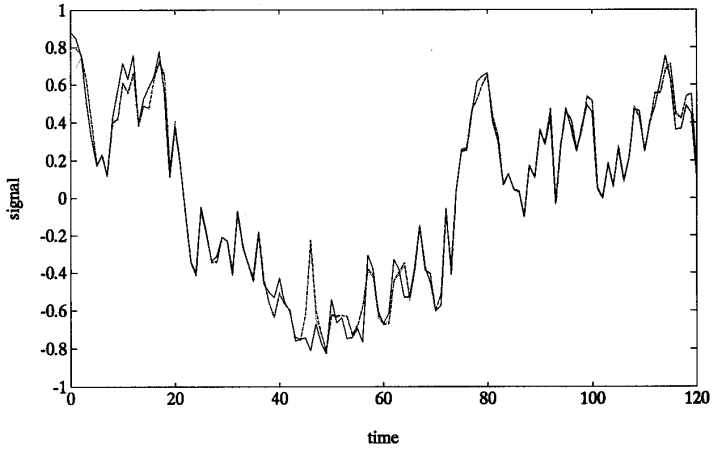


Fig. 9. The true signal (—) and its estimates produced by NFFN (---) and NFRN ($\cdots$) versus time for Example 4.

$\beta_i(t,\omega)$ taking into consideration the dependency of $\alpha(t+1,\omega)$ on $\beta_i(t,\omega)$. We notice that the ordered partial derivatives satisfy the recurrent relationship

$$\frac{\partial^+ C}{\partial \beta_i(t,\omega)} = \frac{\partial C}{\partial \alpha(t+1,\omega)} \cdot \frac{\partial \alpha(t+1,\omega)}{\partial \beta_i(t,\omega)} + \sum_j \frac{\partial^+ C}{\partial \beta_j(t+1,\omega)} \cdot \frac{\partial \beta_j(t+1,\omega)}{\partial \beta_i(t,\omega)}, \quad (4.2)$$

with the final condition, $\partial^+ C/\partial\beta_i(T,\omega) = \partial C/\partial\beta_i(T,\omega)$. This relationship is the TDRBP.

The formulas (4.1) and (4.2) constitute a very efficient technique for evaluating the gradient dC/dw. For the numerical examples in the next section, the RMLP's are trained by first applying TDRBP and then optimizing by the conjugate gradient method.

V. Numerical Examples

Four examples are worked out. The first two show that both a neural filter with fully-interconnected neurons (NFFN) and that with ring-connected neurons (NFRN) outperform the extended Kalman filter (EKF) and the iterated extended Kalman filter (IEKF) for two nonlinear systems described in the form of (1.1) and (1.2). It is also shown that the NFFN's converge rather fast for the example systems. By the main theorem, they converge to the minimum variance filter. The third shows that the performance of even an NFRN is indistinguishable from that of the Kalman filter for a linear system. In the last example, neither the signals nor the measurements can be described by (1.1) and (1.2). Nevertheless, the neural filters of both types work satisfactorily.

In describing the signal/sensor systems in the following examples, the symbols $w(t)$ and $v(t)$ denote statistically independent standard white Gaussian sequences with mean $E[w(t)] = E[v(t)] = 0$ and variance $E[w(t)^2] = E[v(t)^2] = 1$. The training data for each example are obtained by simulating the system and consist of batches of 200 realizations of 100 consecutive measurements and signals. A batch is used in a training session of 100 epochs (or sweeps). If the convergence of the weights is not reached, a different batch is used in

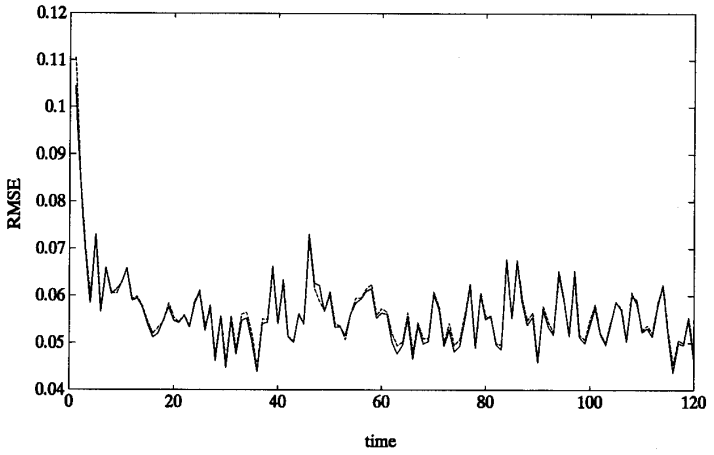


Fig. 10. RMSE's versus time of NFFN (—) and NFRN (---) for Example 4. The RMSE's over 120 time points are 0.0567 and 0.0572, respectively.

another training session. Continuing in this manner, we stop the training whenever the convergence is reached, i.e., the change in the mean square error $C(w)$ is less than $10^{-10}C(w)$.

The training method of first applying TDRBP to evaluate the gradient dC/dw and optimize C by the conjugate gradient method was effective, albeit slow. A 486-based personal computer was used for training. On the average, it took about 16 hours to train up a neural filter on the PC. Local minima of the error function did not pose a serious problem. Only three training sessions resulted in unsatisfactory filters during our entire training experience. Restarting with a new set of random weights always resolved the problem.

After training, 500 Monte Carlo test runs were performed for the EKF, IEKF, NFFN, and/or NFRN that are considered in each example. The RMSE of the estimates $\hat{x}(t,\omega)$ produced by a filter at time t for the 500 Monte Carlo runs $\omega = 1, \cdots, 500$ is defined by

$$[\frac{1}{500}\sum_{\omega=1}^{500}(x(t,\omega)-\hat{x}(t,\omega))^2]^{1/2}.$$

Therefore the RMSE for a filter is a function of time. The RMSE of each filter considered versus time is plotted for 120 time points. In the last example, the true signal and the estimates produced by NFFN and NFRN for a single run are also plotted versus time. The last 20 time points are included to demonstrate the generalization ability of neural filters.

In all the examples, both the NFFN and NFRN have 7 hidden neurons. Notice that while there are 71 weights in the NFFN, there are only 36 weights in the NFRN.

Example 1: The signal/sensor system is

$$x(t+1) = 1.1\exp(-2x^2(t)) - 1 + 0.5w(t),$$
$$y(t) = x^3(t) + 0.1v(t),$$

where $x(0)$ is Gaussian with mean -0.5 and variance 0.1^2. Note that $x(t+1) = 1.1\exp(-2x^2(t)) - 1$ has a global attractor at $x(t) = 0.0844$.

The EKF fails badly. The RMSE's versus time for the IEKF, NFFN, and NFRN are shown in Fig. 4. They are plotted by different lines as specified in the caption of the figure. We note that the RMSE's for the NFFN and NFRN are virtually the same. In fact the RMSE's of the NFRN are only worse than those of the NFFN by 0.1% over the 120 time points. The RMSE's for NFFN's with 2, 4, and 8 neurons are plotted versus time in Fig. 5. We notice that they converge rapidly as the number of neurons increases.

Example 2: The signal/sensor system is

$$x(t+1) = 1.7\exp(-2x^2(t)) - 1 + 0.1w(t),$$
$$y(t) = x^3(t) + 0.1v(t),$$

where $x(0)$ is Gaussian with mean zero and variance 0.5^2. Note that $x(t+1) = 1.7\exp(-2x^2(t)) - 1$ is a chaotic process.

The RMSE's versus time for the EKF, IEKF, NFFN, and NFRN are shown in Fig. 6. We note that the RMSE's of the NFRN are slightly worse than those of the NFFN by 1.42%. The RMSE's for NFFN's with 2, 4 and 8 neurons are plotted versus time in Fig. 7. Again, they converge rapidly as the number of neurons increases.

Example 3: The signal/sensor system is

$$x(t+1) = 0.9x(t) + 0.2w(t),$$
$$y(t) = x(t) + v(t),$$

where $x(0)$ is Gaussian with mean zero and variance 1^2.

The RMSE's versus time for the Kalman filter and NFRN are shown in Fig. 8. We note that the two lines are virtually the same.

Example 4: The signal/sensor system is

$$x(t+1) = 0.5x(t) + 0.5\tanh(x(t) + 0.5w(t)),$$
$$y(t) = x(t) + 0.5x^3(t)v(t),$$

where $x(0)$ is Gaussian with mean zero and variance 0.5^2. Note that neither the signal nor the measurement process can be transformed into (1.1) or (1.2).

The true signal and its estimates produced by the NFFN and NFRN for a single run are shown in Fig. 9. The RMSE's versus time for the NFFN and NFRN are plotted in Fig. 10. We note that the RMSE's of the NFRN are worse than those of the NFFN by 0.84%.

VI. Conclusion

A synthetic approach to optimal filtering is proposed, which has the following advantages:

1) No such assumption as the Markov property, Gaussian distribution, additive measurement noise is necessary;
2) It applies, even if a mathematical model of the signal and measurement processes is not available;
3) The resulting neural filter has the least error variance for the given structure;
4) The neural filter converges to the minimum-variance filter as the number of hidden neurons increases;
5) The neural filter is well suited for real-time processing due to its massively parallel nature of computing.

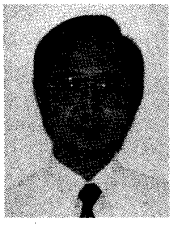
Although a large number of RMLP architectures can be used for neural filtering, only the RMLP with a single hidden layer of fully-interconnected neurons and that with a single hidden layer of ring-connected neurons are tested in the numerical examples. The performance of the latter is remarkable in view of its frugal use of interconnections. Our main theorem does not even apply to it. This only indicates that much remains to be done to further understand the structures of the various RMLP's for dynamical signal processing.

Acknowledgment

The author would like to thank Dr. Lei Yu from the Maryland Technology Corporation for training and simulating the neural networks, and Mr. Daniel J. Gentile and Dr. Vincent Vanicola from the Rome Laboratory for their support in many forms.

References

[1] R. S. Bucy and P. D. Joseph, *Filtering for Stochastic Processes with Applications to Guidance, Second Edition.* New York: Chelsea, 1987.
[2] G. Cybenko, "Continuous valued neural networks with two hidden layers are sufficient," Technical report, Department of Computer Science, Tufts University, Medford, MA, 1988.
[3] J. L. Elman, "Finding structure in time," Technical Report CRL 8801, Center for Research in Language, University of California, San Diego, 1988.
[4] K. Funahashi, "On the approximate realiztion of continuous mappings by neural networks," *Neural Networks,* vol. 2, pp. 183–192, 1989.
[5] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks,* vol. 2, pp. 359–366, 1989.
[6] M. I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine," in *proceedings of the Eighth Annual Conference of the Congnitive Science Society,* Amherst, MA: ErlBaum, 1986, pages 531–456.
[7] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng., Trans. ASME,* vol. D82-1, pp. 35–45, 1960.
[8] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *J. Basic Eng., Trans. ASME,* vol. D83-3, pp. 95–108, 1961.
[9] R. S. Liptser and A. N. Shiryayev. *Statistics of Random Processes I: general theory.* New York: Springer-Verlag, 1977.
[10] D. B. Parker, "Learning logic," Technical Report TR-47, Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA, 1985.
[11] B. A. Pearlmutter, "Learning space trajectories in recurrent neural networks," *Neural Computation,* vol. 1, pp. 263–269, 1989.
[12] G. V. Puskorious and L. A. Feldkamp, "Recurrent network training with the decoupled extended Kalman filter algorithm," in *Proceedings of the 1992 SPIE Conference on the Science of Artificial Neural Networks,* Orlando, Florida, 1992.
[13] D. E. Rumelhart, G. E. Hinton, and P. J. Williams, "Learning internal representation by error propagation," in *Parallel Distributed Processing,* D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: MIT Press, 1986.
[14] P. J. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral sciences," Ph.D. thesis, Harvard University, Cambridge, MA, 1974.
[15] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks,* vol. 1, pp. 339-356, 1988.



James Ting-Ho Lo (M'73) received the B.S. degree in mechanical engineering from the National Taiwan University in 1964 and the Ph.D. degree in aerospace engineering from the University of Southern California in 1969. From 1968 to 1972 he was consecutively a Research Fellow at Stanford University and then Harvard University. Since 1972 he has been on the faculty of the University of Maryland Baltimore County, Baltimore, where he is a Professor in the Department of Mathematics and Statistics. He has been a consultant with numerous companies, including Maryland Technology Corporation, Martin-Marietta Aero and Naval Systems, and AAI Corporation. His current research interests include artificial neural networks, optimal filtering and detection, active noise cancellation, image fusion and enhancement, digital signal processing, and system identification and control.

In November 1992, he won the Best Paper Award in a research paper contest organized by the 1992 Workshop on Neural Networks (WNN-92/Houston). The winning paper is entitled "Synthetic Approach to Optimal Filtering."