An LMS Style Variable Tap-Length Algorithm for Structure Adaptation

Yu Gong and Colin F. N. Cowan, *Senior Member, IEEE*

***Abstract*—Searching for the optimum tap-length that best balances the complexity and steady-state performance of an adaptive filter has attracted attention recently. Among existing algorithms that can be found in the literature, two of which, namely the *segmented filter* (SF) and *gradient descent* (GD) algorithms, are of particular interest as they can search for the optimum tap-length quickly. In this paper, at first, we carefully compare the SF and GD algorithms and show that the two algorithms are equivalent in performance under some constraints, but each has advantages/disadvanges relative to the other. Then, we propose an improved variable tap-length algorithm using the concept of the pseudo *fractional tap-length* (FT). Updating the tap-length with instantaneous errors in a style similar to that used in the stochastic gradient [or least mean squares (LMS)] algorithm, the proposed FT algorithm not only retains the advantages from both the SF and the GD algorithms but also has significantly less complexity than existing algorithms. Both performance analysis and numerical simulations are given to verify the new proposed algorithm.**

***Index Terms*—Adaptive filters, filter length, tap-length variation.**

I. Introduction

THE tap-length, or the number of the tap coefficients of a linear filter, is an important parameter that significantly influences the performance of a minimum mean squared error (MMSE) adaptive filter. It has been known that adjusting the filter length can improve convergence of the least mean square (LMS) algorithm (e.g., [1], [2]). This approach, however, does not attract much attention because the resulting convergence improvements are not very significant. Recently, the topic of the length adaptation was reproposed, but from a structure adaptation point of view, and will be investigated in this paper.

In principle, the MMSE is a monotonic nonincreasing function of the tap-length, but the decrease of the MMSE due to the tap-length increase always becomes trivial when the tap-length is long enough. Obviously, it is not suitable to have a "too" long filter, as it not only unnecessarily increases the complexity but also introduces more adaptation noise. Therefore, there exists an *optimum* tap-length that best balances the steady-state performance and complexity. In many applications, moreover, such optimum tap-length may vary with time. For example, in a multiuser echo cancellation system, the length of the echo may keep changing as users keep entering or leaving the system. In most designs, unfortunately, the tap-length is usually fixed at some compromise value, implying that often, the filter is too long and sometimes inadequate for the severity of conditions. Therefore, it is desirable to derive algorithms that can automatically find the optimum tap-length. This is how the term *structure adaptation* is derived.

Among existing variable tap-length algorithms, three are of particular interest in this paper as others either aim at improving the convergence behavior (e.g., [1]–[3]) or can only be implemented in a limited number of applications (e.g., [4]). The first is called the *Segmented Filter* (SF) algorithm, which was described in [5], where the filter is partitioned into several segments, and the tap-length is adjusted by one segment being either added to, or removed from, the filter according to the difference of the output error levels from the last two segments. In [6], a variable tap-length algorithm based on gradient descent (GD) was proposed. Expressing the tap-length adaptation in an explicit adaptation rule, the GD algorithm is more flexible to implement than the SF algorithm. Recently, the optimum tap-length has been defined quantitatively in [7] and [8], in which a novel variable tap-length algorithm that can converge to the optimum tap-length was also proposed. The proposed algorithm, however, suffers from slow tap-length convergence under some scenarios.

In this paper, in Section II, we will first describe two cost functions that can be used to search for the optimum tap-length, where the first arises directly from the optimum tap-length definition in [7] and [8], and the second may give a biased solution to the optimum tap-length but is more robust to implement. We will further point out that the algorithm proposed in [7] and [8] is based on the first cost function, and both SF and GD algorithms are based on the second cost function. In Section III, we will thoroughly compare the SF and GD algorithm to show that the two algorithms are equivalent under specific constraints, and each has advantages/disadvantages relative to the other. In Section IV, we will propose an improved variable tap-length algorithm based on the second cost function. Updating the tap-length with instantaneous errors in a manner similar to LMS, the proposed algorithm not only retains the advantages from both the SF and GD algorithms without having their drawbacks but also has significantly less complexity than the existing algorithms. In Section V, numerical simulations will be given to verify the analysis. Finally, Section VI will summarize the paper and point out that the proposed algorithm may also be used for channel order estimation.

The analysis and simulations in this paper are mainly based on the LMS algorithm or its variants since they have the widest applications. However, as will be pointed out later in this paper,

Manuscript received April 20, 2003; revised July 29, 2004. This work was supported by the UK Engineering and Physical Sciences Research Council under Grant GR/S00217/01. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ercan E. Kuruoglu.

The authors are with Department of Electrical and Electronics Engineering, Queen's University of Belfast, Belfast BT9 5AH, U.K. (e-mail: y.gong@ee.qub.ac.uk; c.f.n.cowan@ee.qub.ac.uk).

Digital Object Identifier 10.1109/TSP.2005.849170

the adaptation rules for the tap-vector and tap-length are *decoupled*, that is, the choice of one does not depend on the other. Therefore, the results in this paper can be extended to other linear adaptive algorithms such as the recursive least square (RLS) algorithm.

II. Optimum Tap-Length

In [7] and [8], the optimum tap-length is defined as the smallest integer N_o that satisfies

$$\xi_{N-1} - \xi_N \leqslant \mathcal{E}, \qquad \text{for all} \quad N \geqslant N_o \tag{1}$$

where ξ_N is the steady-state mean square error (MSE) when the tap-length is N, and $\mathcal{E}$ is a small positive value that is predetermined according to system requirements. The steady-state MSE, to which the adaptive algorithm converges, is basically a convex function of N, although there may exist suboptima [7], [8]. Due to adaptation noise, the steady-state MSE is always higher than the MMSE, and we may have $\xi_{N-1}-\xi_N < 0$ sometimes, although the MMSE is a strict nonincreasing function of N. The suboptimum tap-length, which is also defined in [7] and [8], equals an integer M, which is smaller than N_o but satisfies $\xi_{M-1} - \xi_M \leqslant \mathcal{E}$. The *width* of the suboptimum is defined as the number of successive suboptimum tap-lengths.

It is immediately clear from (1) that a cost function for searching N_o may be obtained as

$$\min\{N \mid \xi_{N-\Delta} - \xi_N \leqslant \mathcal{E}\} \tag{2}$$

which means that the minimum N that satisfies $\xi_{N-\Delta}-\xi_N \leqslant \mathcal{E}$, where Δ is a positive integer. With the appropriate choice of $\mathcal{E}$, we can simultaneously have $\xi_{N-1}-\xi_N \leqslant \mathcal{E}$ and $\xi_{N-\Delta}-\xi_N \leqslant \mathcal{E}$ for $N \geqslant N_o$. Therefore, if Δ is larger than the maximum *width* of the suboptimum, the solution of (2) can escape from local minima and gives the optimum tap-length N_o. In practice, $\mathcal{E}$ may not be chosen absolutely properly, but as was pointed out in [8], when N is large enough, all ξ_N are within a narrow range because the adaption noise is usually very small. Thus, with a small $\mathcal{E}$, we can always make it insignificant for the difference between the solution of (2) and the optimum tap-length defined in (1). Therefore, in this paper, we will assume that the optimum tap-length is equal to the solution of (2).

In [7] and [8], a variable tap-length algorithm based on the cost function of (2) was proposed. Although it can converge to the optimum tap-length in the mean, the proposed algorithm suffers from slow convergence under some scenarios. This is because, on average, the algorithm adjusts the tap-length by comparing $\xi_{N(n)}$ and $\xi_{N(n-1)}$, which are the steady-state MSEs corresponding to the tap-length at time instant n and $n-1$, respectively. At the start (i.e., $n = 0$), however, $\xi_{N(0)}$ is not available and must be initialized to some arbitrary value. If this initialization is not appropriate, the tap-length adaptation may be initially driven away from the optimum tap-length due to the transient behavior of the tap-coefficient adaptation.

Below, we introduce an alternative cost function to circumvent the random initialization problem.

Assuming the tap-length is N, and $\mathbf{w}_N$ and $\mathbf{x}_N$ are the corresponding steady-state tap-vector and input-vector, respectively, we define the segmented steady-state error as

$$e_M^{(N)} \triangleq d(n) - \mathbf{w}_N^{\mathrm{T}}(1:M)\cdot \mathbf{x}_N(1:M) \tag{3}$$

where $d(n)$ is the desired signal, $1 \leqslant M \leqslant N$, $\mathbf{w}_N(1:M)$, and $\mathbf{x}_N(1:M)$ are vectors consisting of the first M coefficients of $\mathbf{w}_N$ and $\mathbf{x}_N$, respectively. We further define the segmented steady-state MSE as $\xi_M^{(N)} \triangleq \mathrm{E}|e_M^{(N)}|^2$. Ideally, we always have $\xi_{N-\Delta}^{(N)} \geqslant \xi_N^{(N)}$, and the difference between them becomes smaller as N becomes larger. Owing to the adaptation noise, however, this inequality may not hold, especially when N is large enough. Then, we may construct the following cost function to search for the optimum tap-length:

$$\min\{N \mid \xi_{N-\Delta}^{(N)} - \xi_N^{(N)} \leqslant \mathcal{E}'\}. \tag{4}$$

To avoid confusion, (2) and (4) are called the *cost function 1* and *cost function 2*, respectively, and the solutions to (2) and (4) are denoted as N_o and N_o', respectively.

Without the adaptation noise, ξ_N and $\xi_{N-\Delta}$ become the MMSEs corresponding to tap-length N and $N-\Delta$, respectively, and then, we have $\xi_N^{(N)} = \xi_N$ and $\xi_{N-\Delta}^{(N)} \geqslant \xi_{N-\Delta}$, which means

$$\xi_{N-\Delta}^{(N)} - \xi_N^{(N)} \geqslant \xi_{N-\Delta} - \xi_N \tag{5}$$

where "=" holds if and only if $\xi_{N-\Delta} = \xi_N$. Therefore, if we let $\mathcal{E}' = \mathcal{E}$, the optimum tap-length from cost function 2 may be overestimated, i.e., $N_o' \geqslant N_o$. On the other hand, if ξ_N and $\xi_M^{(N)}$ are known, we can have $N_o' = N_o$ by choosing particular values of $\mathcal{E}'$ and $\mathcal{E}$. In practice, however, ξ_N and $\xi_M^{(N)}$ are *a priori* unknown. Thus, cost function 2 generally gives a biased solution to that of cost function 1 (i.e., $N_o' \neq N_o$), but the difference is normally not significant as both $\mathcal{E}$ and $\mathcal{E}'$ are small.

The advantage of using cost function 2 is that it leads to *memoryless* variable tap-length algorithms that require no information about the steady-state MSE for the previous tap-length $N(n-1)$. We will show in the next section that both the SF and GD algorithms are based on cost function 2. Hence, the SF and GD algorithms do not have the random initialization problem and usually converge faster than the algorithm proposed in [7] and [8]. In the following, we will only consider cost function 2.

III. Comparison Between SF and GD Algorithms

In the SF algorithm [5], the filter is divided into L segments, each with Δ coefficients. The tap-length is given by $N = L\Delta$. An exponential smoothing window with forgetting factor β is used to track the so-called *accumulated squared error* (ASE) of each segment

$$\mathrm{ASE}_k(n) \triangleq \sum_{i=1}^{n} \beta^{n-i}\left(e_{k\Delta}^{(N)}(i)\right)^2 \tag{6}$$

where k is the segment index and $e_{k\Delta}^{(N)}(i)$ is obtained from (3) with $M = k\Delta$, which is the output error from the kth segment at time instant i. Suppose at time instant n, there are $L(n)$ segments. At instant $n+1$, if $\mathrm{ASE_L}(n) \leqslant \alpha_{\mathrm{up}}\mathrm{ASE_{L-1}}(n)$, then $L(n+1) = L(n)+1$; else if $\mathrm{ASE_L}(n) \geqslant \alpha_{\mathrm{dw}}\mathrm{ASE_{L-1}}(n)$, then $L(n+1) = L(n)-1$, where $0 < \alpha_{\mathrm{up}} \leqslant \alpha_{\mathrm{dw}} \leqslant 1$. It is clear that, in the average sense, the SF algorithm adjusts the tap-length by comparing the segmented steady-state MSE of $\xi_N^{(N)}$ and $\xi_{N-\Delta}^{(N)}$, which is obviously based on cost function 2, where the threshold $\mathcal{E}'$ depends on the values of α_{up} and α_{dw}.

Below, we show that the SF and GD algorithms are equivalent in performance under specific constraints.

If we let $\alpha_{\mathrm{up}} = \alpha_{\mathrm{dw}} = 1$, the tap-length adaptation rule for the SF algorithm can be expressed as

$$N(n+1) = N(n) - \Delta \cdot \mathrm{sign}(\mathrm{ASE_L}(n) - \mathrm{ASE_{L-1}}(n)), \quad (7)$$

where $N(n) = L(n)\Delta$. However, from (6), we have

$$\begin{aligned}
&\mathrm{ASE_L}(n) - \mathrm{ASE_{L-1}}(n)\\
&\quad= \sum_{i=1}^{n} \beta^{n-i}\left(\left(e_{N(n)}^{(N(n))}(i)\right)^2 - \left(e_{N(n)-\Delta}^{(N(n))}(i)\right)^2\right)\\
&\quad= -\sum_{i=1}^{n} \beta^{n-i}\left(e_{N(n)}^{(N(n))}(i) + e_{N(n)-\Delta}^{(N(n))}(i)\right)\cdot \mathbf{x}'^{\mathrm{T}}_{\Delta}(i)\mathbf{w}'_{\Delta}(i)\\
&\quad\approx -\sum_{i=1}^{n} 2\cdot \beta^{n-i} e_{N(n)-\lfloor \Delta/2 \rfloor}^{(N(n))}(i)\cdot \mathbf{x}'^{\mathrm{T}}_{\Delta}(i)\mathbf{w}'_{\Delta}(i) \qquad (8)
\end{aligned}$$

where we use the approximation that $e_{N(n)}^{(N(n))}(i) + e_{N(n)-\Delta}^{(N(n))}(i) \approx 2\cdot e_{N(n)-\lfloor\Delta/2\rfloor}^{(N(n))}(i)$, which equals twice the output error from the middle tap of the last segment, and $\mathbf{x}'_{\Delta}(i)$ and $\mathbf{w}'_{\Delta}(i)$ consist of coefficients of the last segment of the input-vector $\mathbf{x}_{N(n)}(i)$ and tap-vector $\mathbf{w}_{N(n)}(i)$, respectively. Note that $\lfloor . \rfloor$ rounds the embraced value to the nearest integer. Substituting (8) into (7) gives an equivalent tap-length adaptation rule for the SF algorithm:

$$\begin{aligned}
&N(n+1) = N(n)\\
&\quad+\Delta\cdot \mathrm{sign}\left(\sum_{i=1}^{n}\beta^{n-i}\cdot e_{N(n)-\lfloor\Delta/2\rfloor}^{(N(n))}(i)\cdot \mathbf{x}'^{\mathrm{T}}_{\Delta}(i)\mathbf{w}'_{\Delta}(i)\right). \qquad (9)
\end{aligned}$$

On the other hand, with slight rearrangement, the adaptation rule of the GD algorithm (see [6]) can be expressed as

$$\begin{aligned}
&N(n+1) = N(n)\\
&\quad+\delta\cdot \mathrm{sign}\left(\frac{1}{T}\sum_{i=n-T+1}^{n} e_{N(n)-\lfloor\Delta/2\rfloor}^{(N(n))}(i)\cdot \mathbf{x}'^{\mathrm{T}}_{\Delta}(i)\mathbf{w}'_{\Delta}(i)\right) \qquad (10)
\end{aligned}$$

where δ is the step-size parameter for tap-length adaptation, T is the size of the rectangular window used to obtained the smoothed gradient, and both δ and T are positive integers. Note that (10) only applies at every T samples.

It is clear that the only difference between (9) and (10) is that the two approaches use different smoothing methods to estimate the gradient. However, the smoothing methods can be replaced by each other in both algorithms. Therefore, if we let $\alpha_{\mathrm{up}} = \alpha_{\mathrm{dw}} = 1$ for the SF algorithm and let $\delta = \Delta$ for the GD algorithm, the two algorithms become identical in terms of performance.

The main advantage of the GD algorithm over the SF algorithm is that the former can freely choose the step-size parameter of δ, whereas the latter must have $\delta \equiv \Delta$, which implies that the tap-length in the SF algorithm has to be changed by Δ each time. With this one more degree of freedom, the GD algorithm is more flexible in handling local minima and can more *smoothly* adjust the tap-length than its SF counterpart. Moreover, since it does not need to divide the filter, the GD algorithm may be easier to implement than the SF algorithm.

Alternatively, with the same approximation as in (8), the adaptation rule of (10) for the GD algorithm is equivalent to

$$N(n+1) = N(n) - \delta\cdot \mathrm{sign}(\mathrm{ASE_L}(n) - \mathrm{ASE_{L-1}}(n)). \quad (11)$$

Similar to (7), (11) implies that $\alpha_{\mathrm{up}} = \alpha_{\mathrm{dw}} = 1$, with which the cost function on which the GD algorithm is based becomes

$$\min\{N \mid \xi_{N-\Delta}^{(N)} - \xi_N^{(N)} \leqslant 0\}. \quad (12)$$

It is clear by comparing (12) with (4) that the GD algorithm may overestimate the optimum tap-length, which means unnecessarily greater complexity is imposed. Moreover, we observe that the difference between $\xi_{N-\Delta}^{(N)}$ and $\xi_N^{(N)}$ usually becomes very small when $N \gg N_o$. With inaccurate estimates of the steady-state MSE, this may cause the tap-length "wandering" in the range that is larger than N_o. Thus, if the initial tap-length $N(0)$ is much larger than N_o, or if the optimum tap-length decreases due to channel variation, it may take a long time for the GD algorithm to converge. This problem is more serious when the adaptation noise is low, which may be caused by choosing a small step-size for the LMS algorithm, since then, $\xi_{N-\Delta}^{(N)}$ and $\xi_N^{(N)}$ are almost identical for $N \gg N_o$. On the contrary, the SF algorithm can overcome this problem by choosing the parameters of α_{up} and α_{dw} appropriately.

IV. Fractional Tap-Length Algorithm

In this section, a new variable tap-length algorithm is first proposed, followed by performance analysis of the proposed algorithm. Finally, some discussions and comparisons are provided.

A. Algorithm

The "wandering" problem of the GD algorithm is similar to that with the LMS algorithm when it is implemented using fixed-point parameters and data. A classical solution is to implement the leaky LMS algorithm where a leaky factor is introduced in the adaptation rule [9]. Unfortunately, because the value of the tap-length must be an integer, we cannot simply add a small leaky factor to the GD adaptation rule of (10) or (11), unless the leaky factor is an integer that, however, will be too large for the length adaptation.

To overcome this problem, in this section, we will relax the constraint that the tap-length must be an integer and introduce a concept of pseudo fractional tap-length, where the "true" tap-length is the integer part of the fractional tap-length. Then, we can apply the leaky factor to the adaptation rule, which is similar to (11), of the "fractional" tap-length. The true tap-length remains unchanged until the "change" of the fractional tap-length accumulates to some extent. The concept of the fractional tap-length was first proposed in [7] and [8] but based on cost function 1.

To be specific, we define n_f as the pseudo fractional tap-length, which can take positive real values, and construct the following adaptation rule:

$$n_f(n+1) = (n_f(n) - \alpha) - \gamma \cdot \left[\left(e_{N(n)}^{(N(n))}(n) \right)^2 - \left(e_{N(n)-\Delta}^{(N(n))}(n) \right)^2 \right] \quad (13)$$

where both α and γ are small positive numbers, and α is the leaky factor that satisfies $\alpha \ll \gamma$. Initially, we have $n_f(0) = N(0)$. Then, the "true" tap-length $N(n)$ is adjusted according to

$$N(n+1) = \begin{cases} \lfloor n_f(n) \rfloor, & |N(n) - n_f(n)| \geqslant \delta \\ N(n), & \text{otherwise} \end{cases} \quad (14)$$

where $\lfloor . \rfloor$ rounds the embraced value to the nearest integer.

To make sense of (13), we should ensure $n_f(n) \geqslant \Delta + 1$. Thus, we may set $\min\{n_f(n)\} = \Delta + 1$. Then, it can be easily verified that $\min\{N(n)\} = \min\{n_f(n)\} + \delta$. In practice, the minimum $n(f)$ may be set at a higher value, depending on the system requirements. Note that, theoretically, there is no upper bound for the tap-length.

With the current tap-length $N(n)$, the tap coefficients are then recursively updated by adaptive algorithms. As has been pointed out in [7] and [8], the normalized LMS (NLMS) algorithm [9] is more robust than the LMS algorithm in a variable tap-length scenario. With moderately greater complexity, the NLMS algorithm can automatically adjust the step-size parameter to keep the system stable when the tap-length varies [9]. On the contrary, in the LMS algorithm, the stability condition that (see [9])

$$0 < \mu < \frac{1}{3 \cdot \text{Tr}[\mathbf{R}]} \quad (15)$$

must be checked every time the tap-length changes, making it inflexible to implement, where μ is the step-size parameter, and Tr[$\mathbf{R}$] is the trace of the input correlation matrix $\mathbf{R}$. It is clear that the adaptation rules for the tap-vector and tap-length are *decoupled* since the choice of one does not depend on the other.

In this paper, the new proposed variable tap-length algorithm is called the fractional tap-length (FT) algorithm.

B. Performance Analysis

Below, we analyze the convergence of the FT algorithm. Assuming α and γ are small enough, and taking expectations of both sides of (13), we have

$$\text{E}[n_f(n+1)] = (\text{E}[n_f(n)] - \alpha) - \gamma \left(\xi_{\overline{N(n)}}^{(\overline{N(n)})}(n) - \xi_{\overline{N(n)}-\Delta}^{(\overline{N(n)})}(n) \right) \quad (16)$$

where $\overline{N(n)} = \text{E}[N(n)]$, which is the average tap-length, and $\xi_{\overline{N(n)}}^{(\overline{N(n)})}(n) \triangleq \text{E} \left| e_{\overline{N(n)}}^{(\overline{N(n)})}(n) \right|^2$. It is clear from (16) that if $N(0) = \min\{N(n)\}$ and Δ is larger than the *width* of the suboptimum tap-length, $\overline{N(n)}$ keeps increasing until

$$\xi_{\overline{N(n)}-\Delta}^{(\overline{N(n)})}(n) - \xi_{\overline{N(n)}}^{(\overline{N(n)})}(n) \leqslant \frac{\alpha}{\gamma}. \quad (17)$$

On the other hand, if $N(0)$ is very large, $\overline{N(n)}$ keeps decreasing until

$$\xi_{\overline{N(n)}-\Delta}^{(\overline{N(n)})}(n) - \xi_{\overline{N(n)}}^{(\overline{N(n)})}(n) \geqslant \frac{\alpha}{\gamma}. \quad (18)$$

It is obvious from (17) and (18) that, as $n \to \infty$, if we let $\alpha/\gamma = \mathcal{E}'$, which is defined in (4) and the step-size parameter $\delta = 1$, $\overline{N(n)}$ can converge to N_o', which is the solution of cost function 2. Generally, if $\delta \neq 1$, $\overline{N(n)}$ converges to within a range of $(N_o' - \delta, N_o' + \delta)$.

The second-order analysis of the tap-length adaption for the FT algorithm is difficult, if not impossible, to obtain because $e_M^{(N)}$ is a nonlinear function of the tap-length N and segment length M. Generally, we should have $0 < \alpha \ll \gamma$ to ensure the stability of the FT algorithm. We will show in Section V through numerical simulations that if the parameters are properly chosen, the FT algorithm can converges well in the mean square. The detail of this topic, however, is left as an open question for future research.

C. Discussion

As was shown above, the FT algorithm can converge the tap-length to within the range of $(N_o' - \delta, N_o' + \delta)$ in the mean. Obviously, the smaller the δ is, the higher the tap-length resolution we can obtain, but the slower the tap-length convergence rate we may encounter. In practice, the value of δ should be set according to the system requirements. In contrast, the tap-length of the SF algorithm converges to a fixed range of $(N_o' - \Delta, N_o' + \Delta)$, whereas that of the GD algorithm to a biased range of $(N_o'' - \delta, N_o'' + \delta)$, where N_o'', is the solution to the cost function of (12), and $N_o'' \geqslant N_o'$. Moreover, like the SF algorithm, the FT algorithm does not have the "wandering" problem, due to the leaky factor α. In general, the FT algorithm can freely choose both the step-size δ and the parameters of α and γ, which have similar effects as α_{up} and α_{dw} in the SF algorithm. Thus, it retains all the advantages of the SF and GD algorithms without their respective drawbacks.

We also observe that since the FT algorithm uses instantaneous errors rather than averaged errors for the length adaptation, it has considerably less complexity than the SF and GD algorithms and the algorithm proposed in [7] and [8]. Moreover, unlike (9) and (10), the FT adaptation rule of (13) does not have a "sign" operator, and thus, it has more freedom to handle the tap-length adaptation. Obviously, these are another two advantages of using the "fractional" tap-length. It is interesting to note that the LMS algorithm has similar properties but for tap-vector adaption. Thus, we call the proposed algorithm "LMS style".

Finally, we point out that since the minimum tap-length of the FT algorithm is $\min\{N\} = \Delta + \delta + 1$, any optimum tap-length below it cannot be differentiated. Further, we are reminded that Δ must be larger than the *width* of the suboptimum tap-length for the length adaptation to escape from local minima. Therefore, in some applications where the involved system has a sparse impulse response and the width of the suboptimum length is long, it is necessary to adjust not only the tap-length but the spacing of the taps as well (e.g., [10], [11]). This topic is, however, beyond the scope of this paper. Obviously, the SF and GD algorithm also have this problem.

V. Numerical Simulations and Discussions

Generally, different variable tap-length algorithms can be rated according to their convergence rate and steady-state

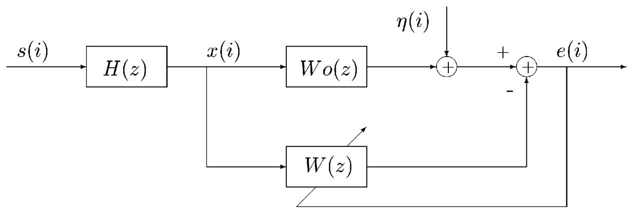


Fig. 1. Block diagram of the adaptive system modeling.

TABLE I
PARAMETERS FOR THE VARIABLE TAP-LENGTH ALGORITHMS

	Δ	δ	T	α_{up}	α_{dw}	α	γ
The SF algorithm	4	–	10	0.8	1	–	–
The GD algorithm	4	1	10	–	–	–	–
The FT algorithm	4	1	–	–	–	0.03	1

performance. From the structure adaptation point of view, an ideal variable tap-length algorithm should converge fast to the optimum tap-length with high accuracy and have the tap-length remain unchanged after the optimum length is reached. Extensive computer simulations have been performed under different scenarios, all of which show that the new proposed FT algorithm is superior to the existing algorithms in all aspects mentioned above. In this section, as an example, we will compare the FT algorithm with the SF and GD algorithms in the application of adaptive system modeling.

A. Simulation Setup

The block diagram of adaptive system modeling is shown in Fig. 1, where $W_o(z)$ is the unknown channel, $W(z)$ is the adaptive filter, $H(z)$ is the spectrum shaping filter, $s(i)$ is white Gaussian noise, $x(i)$ is the input signal, $\eta(i)$ is the additive white Gaussian channel noise, and $e(i)$ is the error signal.

We will consider two classes of the unknown systems in this section, namely, "small scale" and "large scale" systems, which will be tested in Sections V-B and C, respectively, where the small scale system means it has much shorter optimum tap-length than the large-scale system.

Unlike the simulations in [5] and [6], where white input signals were used, the input signal in this section is obtained by passing white Gaussian noise through a spectral shaping filter with a transfer function of $H(z) = 0.35 + z^{-1} + 0.35z^{-2}$. This choice of shaping filter generates a highly colored input (with an associated eigenvalue spread of 28.7) to create a "severe" scenario in which the algorithms are tested [12, Sec. 6.4.1]. The Gaussian noise added to the unknown system provides a signal-noise-ratio (SNR) of 20 dB. In all simulations, the normalized LMS (NLMS) algorithm is used for tap-vector adaptation.

For fair comparison, the rectangular window with size of $T = 10$ is used to obtain both the ASE in the SF algorithm and the smoothed gradient in the GD algorithm. Unless otherwise specified, the parameters used in the tested variable tap-length algorithms are summarized in Table I.

For clarity of exposition, all tap-length learning curves in this section are obtained based on one typical simulation run, and all MSE learning curves are averaged by passing through a rectangular window with size of 50.

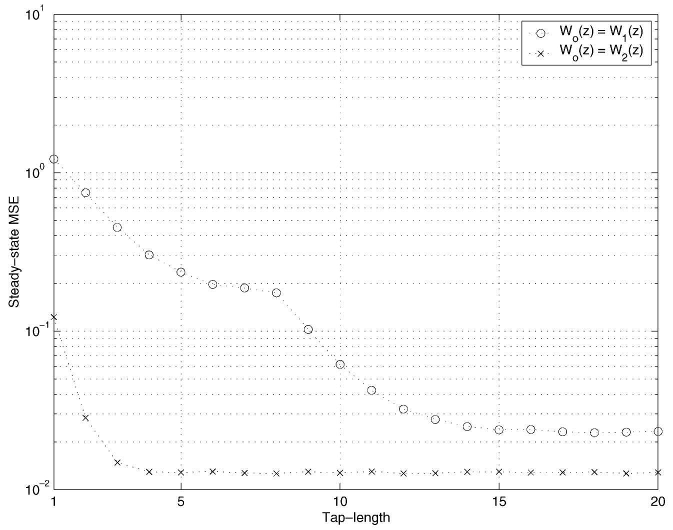


Fig. 2. Curves of the steady-state MSE with respect to the tap-length.

B. Small-Scale System

In this example, as a comparison, we use the same system as that in [7] and [8]. To be specific, two unknown systems are tested, each with transfer functions of $W_o = W_1$ and $W_o = W_2$, respectively, where

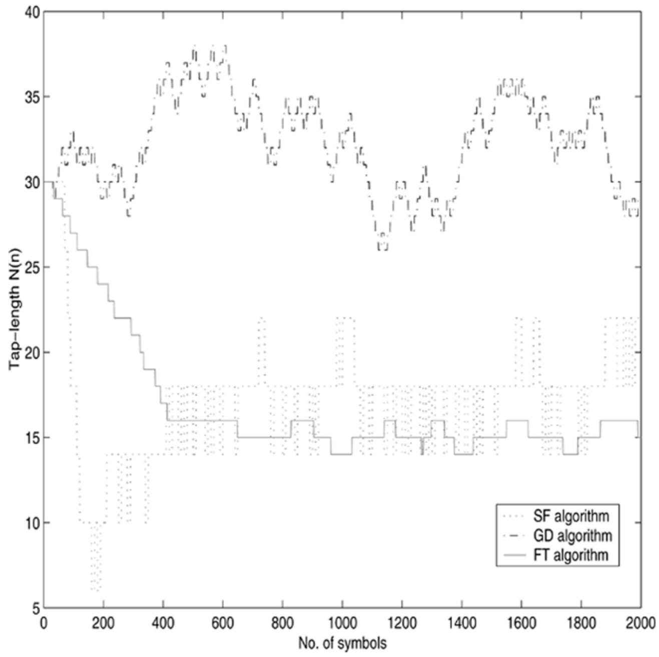
$$W_1(z) = \frac{1 + 0.2z^{-8}}{1 - 0.7z^{-1}}, \qquad W_2(z) = \frac{1}{1 - 0.3z^{-1}}. \tag{19}$$

The impulse response of $W_o(z)$ is not truncated, and thus, any special filter length has not been privileged. In this example, the step-size of the NLMS algorithm is set as $\mu = 0.2$ for all experiments except in Fig. 3, where we set $\mu = 0.05$.

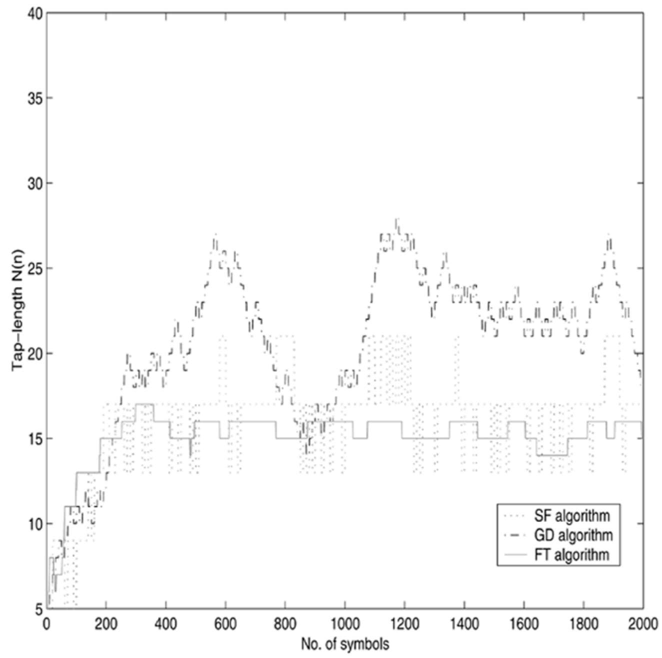
Fig. 2 shows the curves of the steady-state MSE ξ_N, with respect to the tap-length N, for $\mu = 0.2$. It is clearly shown that when $W_o(z) = W_1(z)$, the optimum tap-length is around 15, and the suboptimum tap-lengths are $\{7, 8\}$, but when $W_o(z) = W_2(z)$, the optimum tap-length is around 4, and no suboptimum tap-lengths exist.

Fig. 3 shows the tap-length learning curves with different initializations for the SF, GD, and FT algorithms, respectively. In this experiment, to magnify the "wandering" effect of the GD algorithm, we deliberately choose a small step-size of 0.05 for the NLMS algorithm to give a small adaptation noise. The curve of the steady-state MSE with respect to tap-length for $\mu = 0.05$, which is similar to that of Fig. 2, is not shown here due to space constraints. As expected, the GD algorithm overestimates the optimum tap-length, and the tap-length is "wandering" in the high value areas, especially when the initial tap-length $N(0) = 30$. On the contrary, both SF and FT algorithms converge quickly to around the optimum tap-length for either $N(0) = 30$ or $N(0) = 5$, but the FT algorithm has a much smoother learning curve than the SF algorithm because the former can freely choose the step-size parameter of δ. Note that the leaning curve of the tap-length should be as smooth as possible, especially after it reaches the steady state.

Fig. 4 shows the tap-length learning curve in a time varying scenario, where $W_o(z) = W_1(z)$ for $n < 2000$ or $n \geqslant 4000$, and $W_o(z) = W_2(z)$ for $2000 \leqslant n < 4000$. The initial tap-length $N(0) = \Delta + 1 = 5$. It is clearly shown that the



(a) $N(0) = 30$.



(b) $N(0) = 5$

Fig. 3. Learning curves of $N(n)$, where $W_o(z) = W_1(z)$ and the step-size μ for the NLMS algorithm is deliberately set as small as 0.05.

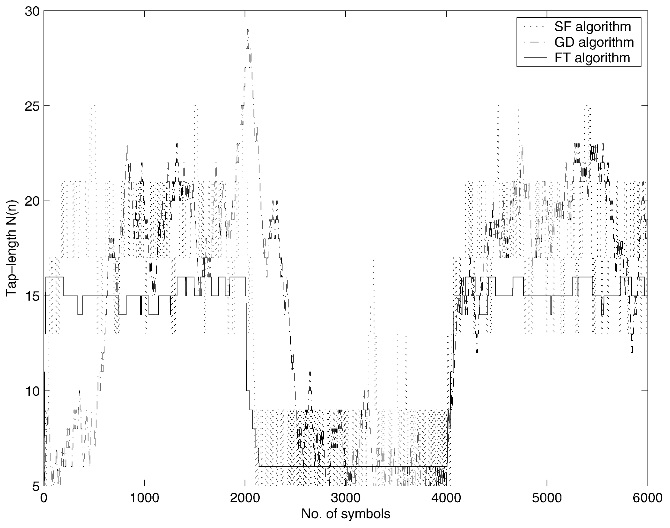


Fig. 4. Learning curves of $N(n)$, where $W_o(z) = W_1(z)$ for $n < 2000$ or $n \geqslant 4000$, and $W_o(z) = W_2(z)$ for $2000 \leqslant n < 4000$.

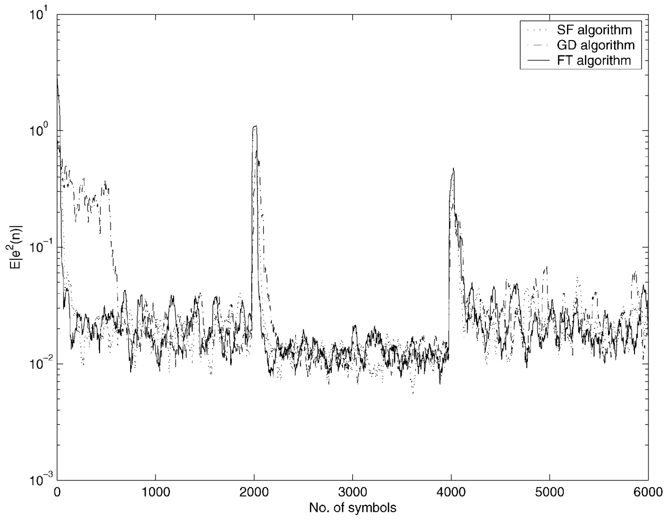


Fig. 5. MSE learning curves corresponding to Fig. 4.

SF and FT algorithms have similar transient behavior, and as expected, both converge significantly faster than the GD algorithm, especially at $n = 2000$, when the optimum tap-length is decreased from 15 to 3. It is also observed that, at the beginning, the learning curve of the GD algorithm has been trapped in the suboptimum area for around 400 symbols before it finally goes up, resulting in slow convergence initially. Although this problem can be overcome by increasing the step-size δ, this phenomenon indicates that the FT algorithm is more robust to the local minima than the GD algorithm. On the other hand, the GD algorithm has a smoother learning curve than the SF algorithm, although the FT algorithm has the smoothest learning curve of all. Moreover, it can be observed that both SF and GD algorithms overestimate the optimum tap-length. Note that unlike the GD algorithm, the SF can overcome the overestimation problem by adjusting the values of parameters $\alpha_{\rm up}$ and $\alpha_{\rm dw}$, as appropriate. Finally, as has been pointed out in Section IV-C, since we have $\min\{N(n)\} = \Delta + \delta + 1 = 6$ for the FT algorithm, any optimum tap-length below this can not be differentiated. Thus, we observe a straight line (with value of "6") in the FT learning curve when $W_o(z) = W_2(z)$.

Fig. 5 shows the MSE learning curves corresponding to Fig. 4. It is clearly shown that the steady-state performance for all algorithms are similar because all algorithm can track the optimum length variation, but their transient performances may be rated as, from best to worst, the FT algorithm, the SF algorithm, and the GD algorithm, corresponding to the transient behaviors of the respective tap-length adaptation.

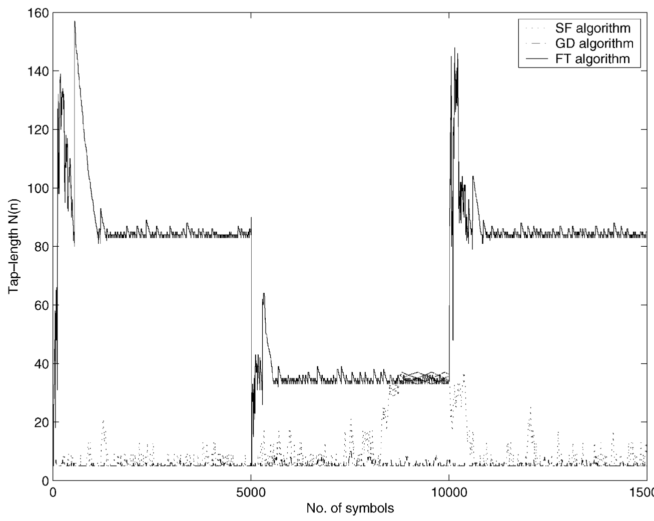


Fig. 6. Learning curves of $N(n)$ and the MSE, where all parameters are set as in Table I, $W_o(z) = W_3(z)$ for $n < 5000$ or $n \geqslant 10000$, and $W_o(z) = W_4(z)$ for $5000 \leqslant n < 10000$

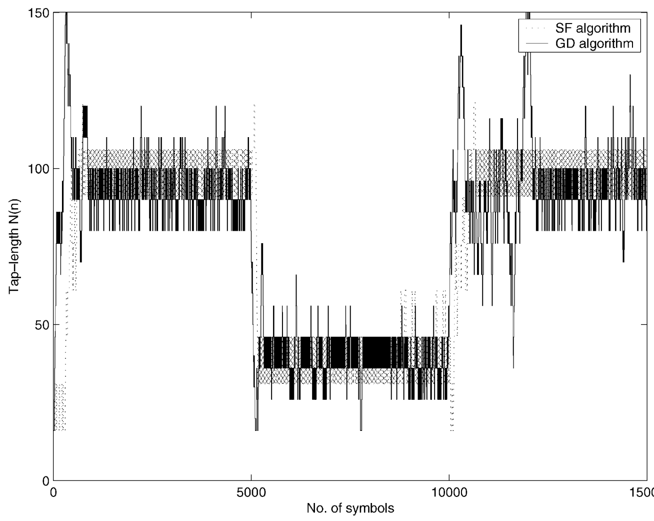


Fig. 8. Learning curves of $N(n)$, where we let $\Delta = 15$ for the SF algorithm, $\Delta = 15$, and $\delta = 10$ for the GD algorithm, $W_o(z) = W_3(z)$ for $n < 5000$ or $n \geqslant 10000$, and $W_o(z) = W_4(z)$ for $5000 \leqslant n < 10000$.

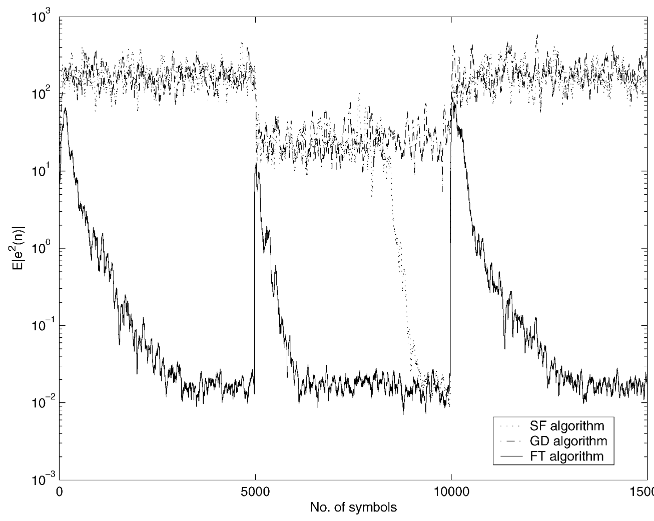


Fig. 7. MSE learning curves corresponding to Fig. 6.

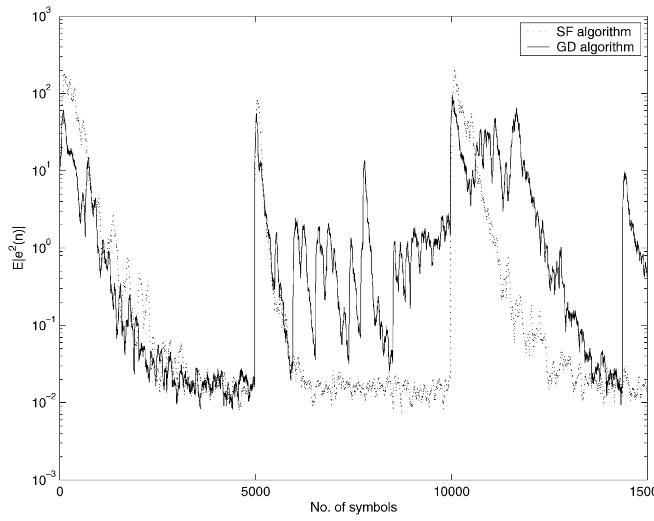


Fig. 9. MSE learning curves corresponding to Fig. 8.

C. Large-Scale System

In this example, we consider two unknown large scale systems, each with transfer functions of $W_o = W_3$ and $W_o = W_4$, respectively, where

$$W_3(z) = \sum_{k=1}^{80} a_k z^{-k}, \qquad W_4(z) = \sum_{k=1}^{30} b_k z^{-k} \tag{20}$$

where a_k and b_k are chosen from a white Gaussian random sequence with mean zero and variance one. It is obvious that the optimum tap-length for system $W_3(z)$ and $W_4(z)$ are 80 and 30, respectively. Both $W_3(z)$ and $W_4(z)$ are very "severe" artificial channels as the tails of their respective impulse responses do not die down gradually but suddenly.

For all experiments in this example, the step-size of the NLMS algorithm is set as 0.8, the initial tap-length $N(0) = \Delta + 1$, and a time varying scenario is considered, where $W_o(z) = W_3(z)$ for $n < 5000$ or $n \geqslant 10{,}000$, and $W_o(z) = W_4(z)$ for $5000 \leqslant n < 10000$.

Figs. 6 and 7 show the learning curves of the tap-length and MSE, respectively, for all three algorithms, where the parameters for all three algorithms are set as same as those for the small-scale system, which are shown in Table I. It is clearly shown that the FT algorithm can still track the channel variations well, although it slightly overestimates the optimum tap-length. However, neither the SF nor the GD algorithm works in this experiment.

Figs. 8 and 9 show the tap-length and MSE learning curves, respectively, for the SF and GD algorithms, where we let $\Delta = 15$ for the SF algorithm, $\Delta = 15$, and $\delta = 10$ for the GD algorithm, and other parameters remain unchanged. We can observe that with these new parameter settings, the SF and GD algorithm can track the channel variations. However, it is clearly shown by comparing Figs. 6 and 8 that the FT algorithm has a much better performance in tracking the tap-length variation than the other two, as the tap-lengths of the SF and GD algorithms fluctuate drastically in the steady state.

In fact, to make the SF and GD algorithms work for such "severe" large-scale systems, many parameter settings have been tested. Unfortunately, only some of the settings perform well, one of which is shown in Figs. 8 and 9. The results in this example indicate that the FT can apply for a wide range of applications with a fixed setting of parameters. On the contrary, the SF and GD algorithm may have to adjust their parameter settings for different scenarios, which, however, contradicts the basic idea of the *structure adaptation*, where the "parameters" should adapt to environmental variations. Therefore, from this point of view, the FT algorithm is more robust than the other two.

VI. Conclusion

This paper first described two cost functions that can be used to search for the optimum tap-length and then thoroughly compared the SF and GD algorithms to show that each has advantages/disadvantages relative to the other. Finally, an improved variable tap-length algorithm using the concept of the *fractional* tap-length was proposed. The proposed FT algorithm not only has better performance but also has less complexity than the existing algorithms. This provides a significant advance toward the practical implementation of flexible, structurally adaptive filters for real-time application in a number of scenarios.

The tap-length adaptation also impacts the issue of channel order estimation. It is clear the channel order is equal, or very close, to the optimum tap-length defined in (1) when the application of system modeling is considered, depending on how the threshold $\mathcal{E}$ is chosen. Although it may converge to a biased value of the optimum tap-length, the algorithm proposed in this paper provides a fast and simple method to estimate, if not accurately enough, the channel order. In contrast, most current order estimate algorithms (e.g., [13] and [14]) require complicated matrix manipulations and are difficult to implement in real time.

References

[1] Z. Pritzker and A. Feuer, "Variable length stochastic gradient algorithm," *IEEE Trans. Signal Process.*, vol. 39, no. 4, pp. 997–1001, Apr. 1991.

[2] Y. K. Won, R. H. Park, J. H. Park, and B. U. Lee, "Variable LMS algorithm using the time constant concept," *IEEE Trans. Consumer Electron.*, vol. 40, no. 3, pp. 655–661, Aug. 1994.

[3] C. Rusu and C. F. N. Cowan, "Novel stochastic gradient adaptive algorithm with variable length," in *Proc. Eur. Conf. Circuit Theory Design*, Espoo, Finland, Aug. 2001.

[4] Y. Gu, K. Tang, H. Cui, and W. Du, "Convergence analysis of a deficient-length LMS filter and optimal-length to model exponential decay impulse response," *IEEE Signal Process. Lett.*, vol. 10, no. 1, pp. 4–7, Jan. 2003.

[5] F. Riera-Palou, J. M. Noras, and D. G. M. Cruickshank, "Linear equalisers with dynamic and automatic length selection," *Electron. Lett.*, vol. 37, no. 25, pp. 1553–1554, Dec. 2001.

[6] Y. Gu, K. Tang, H. Cui, and W. Du, "LMS algorithm with gradient descent filter length," *IEEE Signal Process. Lett.*, vol. 11, no. 3, pp. 305–307, Mar. 2004.

[7] Y. Gong and C. F. N. Cowan, "A novel variable tap-length algorithm for linear adaptive fitlers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Montreal, QC, Canada, May 2004.

[8] ——, "Structure adaptation of linear MMSE adaptive filters," *Proc. Inst. Elect. Eng.—Vision, Image, Signal Process.*, vol. 151, no. 4, pp. 271–277, Aug. 2004.

[9] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[10] J. Homer, I. Mareels, R. R. Bitmead, B. Wahlberg, and F. Gustafsson, "LMS estimation via structure detection," *IEEE Trans. Signal Process.*, vol. 46, no. 10, pp. 2651–2663, Oct. 1998.

[11] S. Ariyavisitakul and N. R. Sollenberger, "Tap-selectable decision-feedback equalization," *IEEE Trans. Commun.*, vol. 45, no. 12, pp. 1497–1500, Dec. 1997.

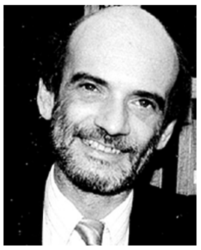
[12] B. Farhang-Boroujeny, *Adaptive Filters: Theory and Application*. New York: Wiley, 1998.

[13] A. P. Liavas, P. A. Regalia, and J. P. Delmas, "Blind channel approximation: Effective channel order estimation," *IEEE Trans. Signal Process.*, vol. 47, no. 12, pp. 3336–3344, Dec. 1999.

[14] W. H. Gerstacker and D. P. Taylor, "Blind channel order estimation based on second-order statistics," *IEEE Signal Process. Lett.*, vol. 10, no. 2, pp. 39–42, Feb. 2003.

Yu Gong received the B.Sc. and M.Eng. degrees from the University of Electronic Science and Technology of China in 1992 and 1995, respectively, and the Ph.D. degree from the National University of Singapore in 2000, all in electrical and electronic engineering.

From 2000 to 2002, he was a research engineer, and later a senior engineer, with the Institute for Infocomm Research, Singapore. Since 2003, he has been with School of Electrical and Electronic Engineering Queen's University, Belfast, U.K., as a Research Fellow. His research interests include signal processing, adaptive filtering, CDMA multiuser detection, and communications systems.



Colin F. N. Cowan (M'82–SM'91) received the B.Sc. and Ph.D. degrees in electrical and electronic engineering from the University of Edinburgh, Edinburgh, U.K., in 1977 and 1980, respectively.

He currently holds the Nortel Networks/Royal Academy of Engineering research professorship in telecommunications at the Queen's University, Belfast, U.K. He was Professor of Signal Processing at Loughborough University, Loughborough, U.K., from 1991 to 1996, and head of the Electronic and Electrical Engineering Department at Loughborough from 1994 to 1996. During his time at Loughborough, he was the head of the Communications and Signal Processing research group, within which he still holds a visiting chair. From 1989 to 1991, he was a reader in the Electrical Engineering Department, University of Edinburgh, and a lecturer from 1980 to 1989. He has published over 200 papers and articles in the area of adaptive signal processing.

Prof. Cowan has been actively involved in the organization of a number of key international conferences, including chairing the First IEE International Conference on Artificial Neural Networks in 1989 and the Seventh European Signal Processing Conference (EUSIPCO) in 1994. He has also served as an associate editor for the IEEE Transactions on Signal Processing between 1999 and 2002.