# A VARIABLE TIME LAG ALGORITHM FOR THE FIR EQUALIZER

*Y. Gong and C. F. N. Cowan*

Department of Electrical & Electronics Engineering,
The Queen's University of Belfast
Stranmillis Road, Belfast BT9 5AH, United Kingdom
phone: +44+(0)28 90 274049, fax: +44+ (0)28 90 667023
email: y.gong, c.f.n.cowan@ee.qub.ac.uk

## ABSTRACT

In linear equalization, time lag is an important parameter that significantly influences the performance. Only with the optimum time lag that minimizes the minimum-mean-square-error (MMSE), can we have best use of the available resources. Many designs, however, choose the time lag either based on pre-assumption of the channel or simply based on average experience. In this paper, we propose a novel variable time lag algorithm with the concept of pseudo fractional time lag. The proposed algorithm can converge to the optimum time lag in the mean and is verified by the numerical simulations provided in this paper.

## 1. INTRODUCTION

The finite impulse response (FIR) equalizer is widely used in digital communications to combat intersymbol interference (ISI). The basic structure of the FIR equalizer is shown in Figure 1, where $x(n)$ is the information signal, $H(z)$ is the channel transfer function, $\eta(n)$ is the channel noise, $y(n)$ is the received signal, $\Delta$ is the time lag, $d(n) = x(n-\Delta)$ is the reference signal, $z(n)$ is the equalization output, $e(n)$ is the error signal, and $W(z)$ is the transfer function of the equalizer. The FIR equalizer is usually implemented on a tapped-delay-line (TDL) structure with tap coefficients updated by adaptive algorithms such as least mean square (LMS) or recursive least square (RLS) algorithms [1].

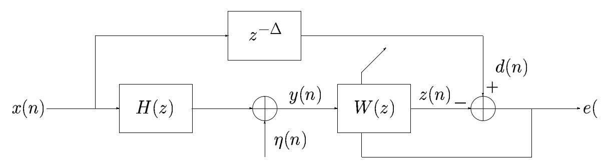



Figure 1: The adaptive MSE equalizer.

To preserve the causality of the FIR equalizer, the time lag, or the decision delay, is introduced. The value of the time lag must be carefully chosen as it significantly influences the equalization performance. Only if the time lag is set as the optimum value that minimizes the MMSE, can we have best use of the resources once the structure of the adaptive equalizer is fixed. In many applications, however, the time lag is chosen based on either the pre-assumption of the channel or simply from experience. For instance, in an experiment shown in [1, Section 9.7], the channel impulse response was assumed to be symmetric and the time lag was set equal to half of the tap-length plus the delay of the strongest pulse of the channel impulse response. Obviously this method can not be generalized. For example, for a minimum phase channel where all zeros of the transfer function are inside the unit circle, the value of the optimum time lag is usually zero, while for a maximum phase channel where all zeros are outside the unit circle, the optimum time lag is about equal to the tap-length minus one. Generally, even with accurate channel estimation, it is still not straightforward to set an appropriate time lag. It is therefore desirable to derive an algorithm that can automatically find the optimum time lag. Unfortunately, there are only a few papers in the literature regarding this topic, a typical example of which is [2], where the time lag was included as an explicit parameter for the decision feedback equalizer (DFE) and a *brute force* searching method was proposed to determine the optimum time lag based on calculating the DFE performance for every possible time lag. However, this algorithm requires a priori knowledge of the channel and suffers from high computational complexity especially when the tap-length is long, making it inefficient to apply. Some other papers, e.g. [3, 4, 5], investigated the time delay for multichannel equalizers. The results, however, can not be applied to traditional equalizers.

This paper proposes a novel variable time lag algorithm. The proposed algorithm is based on the observation that though the closed-form for the minimum-mean-squared-error (MMSE) function with respect to the time lag is difficult, if not impossible, to obtain, the relationship between the MMSE and time lag can be revealed in an ad-hoc manner since the time lag is only a one dimensional parameter. Moreover, although the time lag must be an integer, we can apply the concept of the pseudo *fractional* lag to make instantaneous adaption possible, where the true time lag is the integer part of the fractional lag.

## 2. TIME LAG ADAPTATION

In this section, we describe the variable time lag algorithm in detail. First, the algorithm basis is introduced, followed by the description of the full algorithm. Finally some variants of the algorithm are presented.

### 2.1 Basic algorithm

The optimum time lag $\Delta_o$ is a function of the channel specification and the equalizer tap-length. As the time lag $\Delta$ is set away from $\Delta_o$, the MMSE tends to increase. Generally, the MMSE is a convex function of $\Delta$, though there may exist local minima.

This research was sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/S00217/01.

If the MMSE is known for all lags, $\Delta_o$ can be searched by traditional one-dimensional searching methods. In practice, unfortunately, the MMSE is usually not available but may be estimated by the exponential average:

$$\bar{\xi}(n) = \lambda \bar{\xi}(n-1) + (1-\lambda)e^2(n), \quad (1)$$

where $\lambda$ is a forgetting factor which is set close to one.

Defining $\delta_f$ as the pseudo time lag which can take fractional non-integer values, we have the following adaptation rule:

$$\delta_f(n+1) = \delta_f(n) + \beta \cdot \left[\bar{\xi}_{\mathrm{p}} - \bar{\xi}(n)\right], \quad (2)$$

where the true time lag $\Delta(n) = \lfloor \delta_f(n) \rfloor$, $\lfloor . \rfloor$ truncates the embraced value to the nearest integer, $\bar{\xi}_{\mathrm{p}}$ is the estimated converged MSE for the previous lag $\Delta(n-1)$, $\bar{\xi}(n)$ is obtained by (1), and $\beta$ is the step-size for $\delta_f(n)$ adaptation.

Initially we set $\bar{\xi}_{\mathrm{p}} = \mathscr{P}$ and $\Delta(0) = \delta_f(0) = \Delta_s$, where $\mathscr{P} > \xi_\Delta$ for all $\Delta$ [1] and $\Delta_s \leqslant \Delta_o$ (e.g. $\Delta_s = 0$). Then $\delta_f(n)$ starts to increase from $\Delta_s$. Only at the so-called *changing time* that $|\delta_f(n) - \Delta(n)| \geqslant 1$, the true lag is increased by one, otherwise it remains unchanged.

Starting at one changing time, the filter converges towards the MMSE corresponding to the new lag. If $\beta$ is small enough, we can have $\mathrm{E}[\bar{\xi}(\infty)] = \xi_{\lfloor \delta_f(n) \rfloor}$ before the next changing time, where $\xi_\Delta$ is the MMSE when the time lag is $\Delta$. Then taking expectations on both sides of (2) gives:

$$\mathrm{E}[\delta_f(n+1)] = \mathrm{E}[\delta_f(n)] + \beta \cdot \left[\xi_{\lfloor \delta_f(n) \rfloor - 1} - \xi_{\lfloor \delta_f(n) \rfloor}\right], \quad (3)$$

where $n$ corresponds to the converged periods of the adaptive filter. It is clear from (3) that $\mathrm{E}[\delta_f(n)]$ keeps increasing until (note $\Delta(n) = \lfloor \delta_f(n) \rfloor$):

$$\xi_{\Delta(n-1)} - \xi_{\Delta(n)} \leqslant 0. \quad (4)$$

The time lag can only be increased by (2). Similarly we can also construct a recursion to decrease the lag which is given by:

$$\delta_f(n+1) = \delta_f(n) - \beta \cdot \left[\bar{\xi}_{\mathrm{p}} - \bar{\xi}(n)\right], \quad (5)$$

where the true tap-length $\Delta(n) = \lfloor \delta_f(n) \rfloor + 1$. Initially we set $\Delta(0) = \delta_f(0) = \Delta_b$ which is an integer bigger than $\Delta_o$, and $\bar{\xi}_{\mathrm{p}} = \mathscr{P}$ which is same as that for (2). Then $\delta_f(n)$ starts to decrease from $\Delta_b$. The changing time is also defined as the time when $|\delta_f(n) - \Delta(n)| > 1$, at which we let $\bar{\xi}_{\mathrm{p}} = \bar{\xi}(n)$. Similar to the analysis for (2), we have that if $\beta$ is small enough, $\mathrm{E}[\delta_f(n)]$ keeps decreasing until:

$$\xi_{\Delta(n-1)} - \xi_{\Delta(n)} > 0. \quad (6)$$

It is clear from (4) and (6) that if there are no suboptima, both (2) and (5) can converge to $\Delta_o$ in the mean. However, some systems may have sub-optima lag. Fortunately, both (2) and (5) are adapted based on the instantaneous values of $\bar{\xi}(n)$, the variance of which can be regarded as a random disturbance to the search procedure. If such disturbance is much larger than the variation of the sub-optima, the search algorithm can escape from the sub-optima. In some applications, however, it may take too long for the search to go through the sub-optimum, if it does at all. In such cases, we may increase the adjusting step-size of the lag, i.e. adjust $\Delta$ by the value of $K$ at every changing time, where $K$ is an integer larger than one. Then the tap-length adaption converges to a biased value within a range which is centered at $\Delta_o$ and has length of $K$. We have done extensive simulations to show that unless the tap-length is very short, there are generally no sub-optima or the variation of the sub-optima is small.

[1] For example, we can set $\mathscr{P} = 1$ if the desired signal's power is normalized to one.

### 2.2 Algorithm principle

In (2) and (5), at every time that the true lag changes, there may be a sudden rise of the MSE before it eventually converges to the new MMSE. This forces us to choose a very small $\beta$ to ensure the convergence of $\delta_f(n)$, which however implies a slow search rate. To suppress such a MSE rise, we can shift the tap-vector when the time lag changes. Specifically, assuming the optimum tap-vector for the time lag $\Delta_1$ is $\mathbf{w}$, when the time lag is changed to $\Delta_2$, $\mathbf{w}$ is shifted as:

$$\mathrm{shift}(\mathbf{w}) = \begin{cases} [0, \cdots, 0, w(0), \cdots, w(N-|\delta|-1)]^{\mathrm{T}}, & \text{if } \delta > 0, \\ [w(|\delta|), \cdots, w(N-1), 0, \cdots, 0]^{\mathrm{T}}, & \text{if } \delta < 0 \end{cases} \quad (7)$$

where $\delta = \Delta_2 - \Delta_1$ and $w(i)$ is the $i$th coefficient of $\mathbf{w}$.

Another problem of (2) and (5) is that they can only search for the lag in one direction. Therefore unless $\beta$ is very small which leads to slow convergence rate of $\delta_f(n)$, the search may fail due to the inaccurate estimate of the MMSE. Fortunately, because they differ only by a sign factor in the recursion, (2) and (5) may be merged into one recursion and applied simultaneously, by which a more robust and faster tap-length adaption can be obtained.

With these observations and the analysis, we have the full variable time lag algorithm as below.

For every $n = 1, 2, 3, \cdots$
  $\bar{\xi}(n) = \lambda \bar{\xi}(n-1) + (1-\lambda)e^2(n)$
  $\delta_f(n+1) = \delta_f(n) + \beta\, \gamma \left[\bar{\xi}_{\mathrm{p}} - \bar{\xi}(n)\right]$ (8)
  $\delta_f(n+1) = |\delta_f(n+1)|_{N-1}$
  If $|\delta_f(n) - \Delta(n)| \geqslant K$
    $\Delta(n) = \langle \delta_f(n) \rangle$, $\bar{\xi}_{\mathrm{p}} = \bar{\xi}(n)$
    $\delta_\Delta = \Delta(n) - \Delta(n-1)$, $\gamma = \mathrm{sign}(\delta_\Delta)$
    shift the tap-vector $\mathbf{w}$ based on (7)
  End
  If $|\delta_f(n) - \Delta(n)| < K$
    $\Delta(n) = \Delta(n-1)$
  End

In the above procedure, the true lag $\Delta(n) = \langle \delta_f(n) \rangle$, where $\langle . \rangle$ rounds the embraced value to the nearest integer; $\delta_f(n+1) = |\delta_f(n+1)|_{N-1}$, where $|.|_{N-1}$ is a *limiter* that constrains the embraced value in the range of $[0, N-1]$ in which $\Delta_o$ lies if the tap-length $N$ is long enough; $K$ is the step-size for the lag adjustment, which is set according to the suboptimum and the system requirements; $\gamma$ is called the *direction factor*, with which (2) and (5) are merged into one recursive equation. Specifically, at the changing time $n$, if $\Delta(n) - \Delta(n-1) > 0$, we have $\gamma = 1$ and (2) is actually applied, otherwise we have $\gamma = -1$ and (5) is used. Therefore even if the lag adapts in the wrong direction due to the instantaneous values of $\bar{\xi}(n)$, the algorithm can automatically draw it back to the right direction.

Without a priori knowledge of the channel, initially we may set $\xi_P = 1$, $\gamma = 1$ and $\Delta(0) = \lfloor N/2 \rfloor$ which might be a *good* guess of the optimum time lag. Then the first change of the lag is to be increased by $K$. Furthermore, to ensure stability, initially the adaptation of the time lag should not begin until the tap-vector has converged.

### 2.3 Algorithm variants

If the tap-length is long enough, there may exists a *flat area* around $\Delta_o$ in the curve of MMSE with respect to $\Delta$, i.e. the MMSE difference are trivial for those time lags within the flat area. Obviously in practice, it is more appropriate for the algorithm to converge to the smallest lag in the flat area rather than $\Delta_o$. However, the proposed adaptive lag algorithm may converge to any value within the flat area. Using the similar philosophy to the leaky LMS algorithm, we can introduce a leaky factor to keep the lag search algorithm from entering the large values in the flat area. To be specific, the recursive equation of (8) is modified as:

$$\delta_f(n+1) = (1-\alpha)\cdot\delta_f(n) + \beta\,\gamma\left[\bar{\xi}_{\mathrm{p}} - \bar{\xi}(n)\right], \qquad (9)$$

where $\alpha$ is the so-called leaky factor which is a small positive constant less than one. To ensure stability, we should let $\alpha \ll \beta$.

In many applications, the power of the reference signal is always normalized to one, implying that the MMSE is usually smaller than one and is thus better represented in a logarithmic scale than in a linear scale. In such cases, the adaptation rule of (9) may further be modified as:

$$\delta_f(n+1) = (1-\alpha)\delta_f(n) + \beta\,\gamma\left[\log\bar{\xi}_{\mathrm{p}} - \log\bar{\xi}(n)\right]. \quad (10)$$

## 3. NUMERICAL SIMULATIONS

In this section, we apply the proposed lag algorithm to the adaptive MSE equalizer as shown in Figure 1, where we assume the transmitted signals $x(n)$ are either $+1$ or $-1$ (BPSK) and are independent of each other, the channel SNR is 20dB and the tap-length of the equalizer is set at $N = 16$. In all the simulations below, the normalized LMS [1] is used for tap adaption with step-size 0.4, the recursion rule (10) is used for the lag adaptation where $K = 1$ and initially $\Delta(0) = \lfloor N/2 \rfloor$, $\bar{\xi}_P = 1$ and $\gamma = 1$. Based on experiments, we choose $\beta = 0.3$ and $\alpha = 0.00025$ for all the simulations in this section. The adaptation of $\delta_f(n)$ begins after $n = 100$.

### 3.1 Mixed-phase channel

First, we examine the mixed-phase channel which has zeros both in and outside the unit circle. The channel vector is set as $\mathbf{h}(n) = [-0.1\ -0.3\ 0.4\ 1\ 0.4\ 0.3\ -0.1]^{\mathrm{T}}$. The curve of the MMSE with respect to the lag is shown in Figure 2, where we can clearly observe that the optimum lag $\Delta_o$ can be either 10 or 11 which corresponds to similar MMSE performance.

Before simulating the proposed lag algorithm, we firstly examine the effect of the tap-vector shift on (7). The results are shown in Figure 3, where the time lag is firstly fixed at 6 and later increased to 7 after symbol 400, and the curves are obtained by averaging over 30 independent runs. We can clearly observe that if the tap-vector $\mathbf{w}(n)$ remains unchanged when the lag changes (i.e. without $w(n)$ shift), there is a sharp rise in the MSE learning curve at symbol 400, but the shift of $\mathbf{w}(n)$ can effectively suppress that rise.

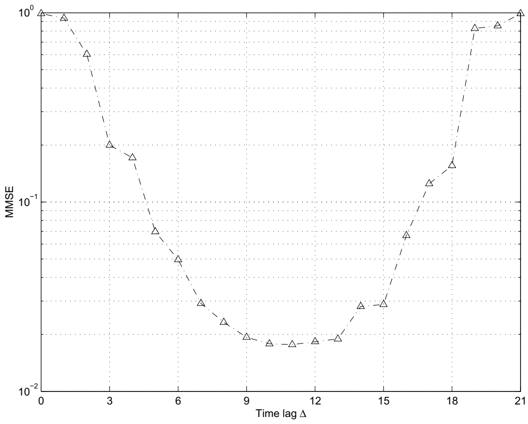



Figure 2: The function of MMSE with respect to the lag for the mixed-phased channel.

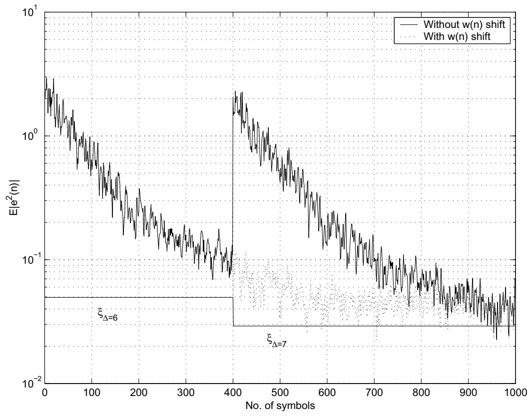



Figure 3: The function of MMSE with respect to the lag for the mixed phased channel.

The learning curves of the MSE and $\delta_f(n)$ for the proposed time lag algorithm are shown in Figure 4 (a) and (b) respectively, where both curves are for one typical simulation run, and Figure 4 (a) is obtained by averaging the MSE learning curve with a rectangular smoothing window of size 50. It is clearly shown in Figure 4 (b) that $\delta_f(n)$ converges to around the optimum time lag.

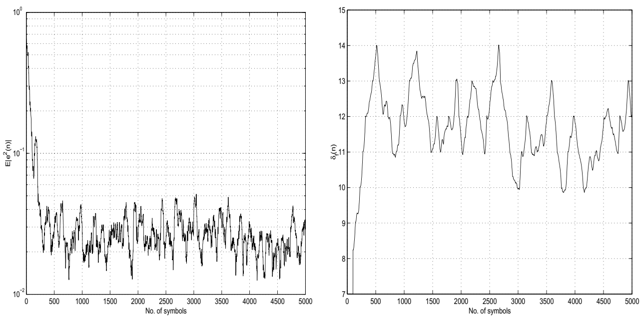
### 3.2 Minimum phase and maximum phase channels

In this example, we consider the minimum phase and maximum phase channels. To be specific, the channel vectors are set as:

$$\begin{aligned}\mathbf{h}(n) &= [1\ 0.8\ 0.6\ 0.4\ 0.3\ 0.2\ 0.1]^{\mathrm{T}} \\ \mathbf{h}(n) &= [0.1\ 0.2\ 0.3\ 0.4\ 0.6\ 0.8\ 1]^{\mathrm{T}}\end{aligned} \qquad (11)$$

respectively. The curves of the MMSE function with respect to the time lag are shown in Figure 5, where the *flat area* can be clearly observed in both curves. Although all time lags on the flat area correspond to similar MMSE, it is obvious from Figure 5 that the ideal time lags are 0 and 15 for the minimum and maximum phase channels respectively.

The learning curves of the MSE and $\delta_f(n)$ for the minimum and maximum phase channels are shown in Figure 6



(a) Learning curve of the MSE. (b) Learning curve of $\delta_f(n)$.

Figure 4: Learning curves of the MSE and time lag for the mixed phase channel.

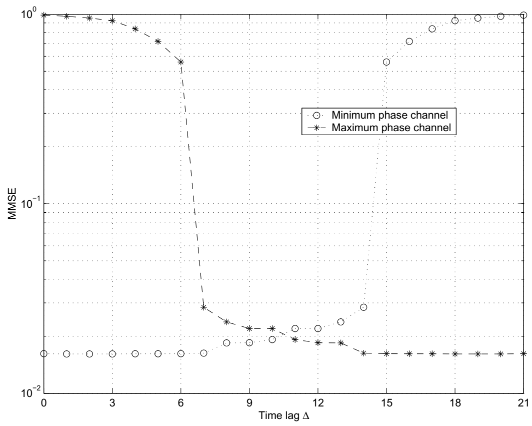
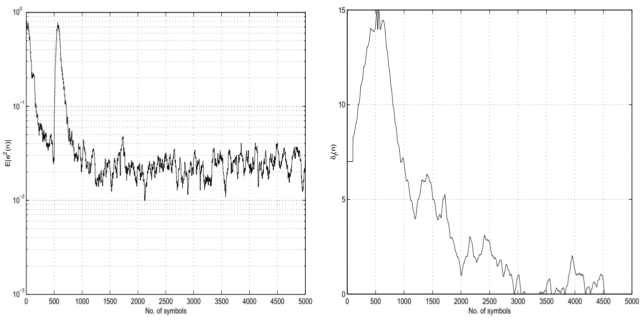



Figure 5: The functions of MMSE with respect to the lag for the minimum phase and maximum phase channels.

and 7 respectively. Similar to those in Figure 4, all curves are based on one typical simulation run, Figure 6 (a) and 7 (a) are obtained by averaging the MSE learning curve with a rectangular smoothing window of size 50. Clearly $\delta_f(n)$ for both channels converge to around the ideal time lags. Particularly in Figure 6 (a), we can observe an obvious rise around symbol 500 in the MSE learning curve. This is because the *first* change of the time lag initially drives the search in the wrong direction away from the ideal lag.

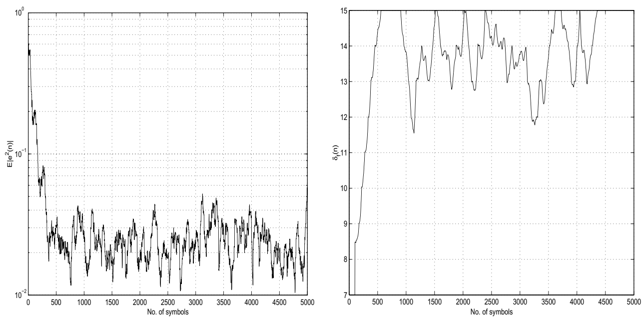
## 4. CONCLUSIONS

This paper proposed a novel variable time lag algorithm which can converge to the optimum lag in the mean. Since the proposed variable tap-length algorithm is based on gradient search, the extra complexity added to the basic LMS algorithm is low. Finally computer simulations were given to verify the proposed algorithm.



(a) Learning curve of the MSE. (b) Learning curve of $\delta_f(n)$.

Figure 6: Learning curves of the MSE and time lag for the minimum phase channel.



(a) Learning curve of the MSE. (b) Learning curve of $\delta_f(n)$.

Figure 7: Learning curves of the MSE and time lag for the maximum phase channel.

## REFERENCES

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

[2] P. A. Voois, I. Lee, and J. M. Cioffi, "The effect of decision delay in finite-length decision feedback equalization," *IEEE Trans. on Information Theory*, vol. 42, no. 2, pp. 618 – 621, March 1996.

[3] A. P. Liavas, P. A. Regalia, and J. P. Delmas, "Least-squares channel equalization performance versus equalization delay," *IEEE Trans. on Signal Processing*, vol. 48, no. 6, pp. 1832 – 1835, June 2000.

[4] H. Luo and R.-W. Liu, "Blind equalizers for multipath channels with best equalization delay," in *Proceedings of 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 15-19 March 1999, vol. 5, pp. 2511 – 2514.

[5] D. Gesbert, P. Duhamel, and S. Mayrargue, "Blind multichannel adaptive mmse equalization with controlled delay," in *Proceedings of 8th IEEE Signal Processing Workshop on Statistical Signal Array Processing (Cat. No.96TB10004)*, June 1996, pp. 172 – 175.