A Comparison of Forward-Only and Bi-Directional Fixed-Lag Adaptive SISOs

Jun Heo, Keith M. Chugg, and Achilleas Anastasopoulos

Abstract—Several structures for fixed-lag (FL) soft-in/soft-out (SISO) algorithms in the case of a perfectly known channel are well-known. These forward-only and bi-directional fixed-lag SISOs have recently been described with the bi-directional version shown to be preferred. Adaptive iterative detection using adaptive SISOs (A-SISOs) have also recently been demonstrated to provide significant performance gains for time-varying channels. However, these impressive results have been obtained with fixed-interval, bi-directional A-SISOs and training signals at both ends of the data packet. We combine these recent results to develop and compare various adaptive, fixed-lag SISOs. Among several reasonable options considered, the preferred A-SISO algorithm is found to be bi-directional with forward-only channel estimation.

I. INTRODUCTION

The mobile channel is characterized by time varying multipath propagation which can result in severe inter symbol interference (ISI). A combination of Trellis Coded Modulation (TCM) and interleaving is often used to combat time-varying, ISI. Three receiver structures are: (i) hard decision decoding of the code with sequence detection on the inner ISI channel, (ii) soft-output “equalization” of the ISI channel with soft-decision decoding of the code, and (iii) iterative equalization-decoding using a soft-in/soft-out (SISO) module for each the decoder and the “equalizer”. These approaches are listed in the order of improving performance with (i) a special case of (ii) (i.e., no iteration). In this paper we are interested in performing method (iii) when perfect knowledge of the channel is not available at the receiver.

Several soft-output algorithms or SISOs have been suggested for the approaches (ii) and (iii), respectively, when the channel is known. Of particular interest in this paper are fixed-lag (FL) algorithms which produce soft-output information of a quantity \( u_k \) based on the soft-in information from time 0 to time \( k + D \), where \( D \) is the smoothing lag. There are several SISOs that compute the same (desired) soft-information with different structures. The Abend and Fritchman [6] and the Lee [1] and Li, Vucetic and Sato [2] (L^2 VS) algorithms produce the same soft-information using forward-only recursive processing. The complexity increases exponentially with \( D \) for the Abend-Fritchman algorithm and linearly in \( D \) for the L^2VS version. Recently, it has been shown that a bi-directional FL SISO [3], [8] produces the same soft-outputs with complexity that is less than that of the L^2VS [3], [9] with the same latency. Furthermore, these FL-SISOs have been shown to be generally applicable to iterative detection [8] in both sum-product or minimum form, yielding soft-inverse of a finite state machine (FSM) based on the A-Posteriori Probability (APP) or Minimum Sequence Metric (MSM) [9], respectively. Thus, the bi-directional FL-algorithm is the desired structure when all system parameters are known.

When the channel is unknown and/or time variant, joint channel and data estimation may be a reasonable approach. In the hard-decision case, approaches vary from simple single channel estimator structures to the more advanced multiple estimator structures, based on the Per Survivor Processing (PSP) concept [10]. In [11] the L^2VS algorithm was augmented with a PSP-like estimation for the special case \( L = D \), where \( L \) is the memory size of a channel and \( D \) is the lag of the L^2VS algorithm. For the general lag size \( D \), a fixed-lag, forward-only, adaptive soft output algorithm was introduced in [12], based on the Abend and Fritchman [6] approach. However, the exponential complexity of this algorithm with the lag \( D \) precludes its use for many applications of interest. In [4], a new bi-directional fixed interval A-SISO was introduced, having linear complexity in the observation record. Nevertheless, in the above algorithm it is assumed that both a forward and backward channel training sequences are present. In many realistic situations, however, there is only a forward training sequence available. In addition there may be a necessity to process the received data without large delay. In these cases, a fixed-lag soft output algorithm is required. Thus, two interesting questions arise: (i) Can the fixed-interval A-SISO [4] be modified to a (bi-directional) fixed-lag A-SISO maintaining the significant performance gains? and (ii) If so, which fixed-lag soft output algorithm architecture (i.e., L^2VS vs. bi-direction) is more effective in terms of complexity and performance when modified to be adaptive?

In this paper we propose new Fixed-Lag A-SISO (FL-A-SISO) architectures. In particular the FL-A-SISO is realized both in the bi-directional form [3], as well as in the L^2VS forward-only form, by synthesizing the PSP concept with the recent results from [9], [4]. The resulting FL-A-SISO algorithms have linear complexity in \( D \) and estimate the channel without a backward channel training sequence. The simulation results show that the suggested FL-A-SISO architectures maintain most of the performance gains demonstrated for the fixed-interval case [4] and have almost the same performance as the algorithm in [12] with significantly less complexity. Among several reasonable options considered, the preferred FL-A-SISO algorithm is found to be bi-directional with forward-only channel estimation.

II. SYSTEM DESCRIPTION

A typical TDMA cellular transmission system with trellis-coded modulation (TCM), interleaving, and frequency selective
fading channel is considered. After the source symbol $b_k$ is trellis coded and modulated, the modulated symbols are interleaved. The interleaved symbols $a_k$ are transmitted through a frequency selective fading channel with $L + 1$ taps. The signals are observed in additive white Gaussian noise (AWGN). A simple discrete-time model for the received signal is

$$z_k = \sum_{i=0}^{L} a_{k-i} f_k(i) + n_k = x(t_k; f_k) + n_k$$  

(1)

where, $a_k$ is the coded symbol, $t_k \{a_{k-i}\}_{i=0:L}$ is the state transitions, $f_k(i)$ is the coefficient of the frequency selective fading channel, and $n_k$ is a white circular complex Gaussian noise. The structure of overall system is considered in Fig. 1.

![Fig. 1. The block diagram of TCM in interleaved frequency selective fading channel](image)

The receiver consists of two SISOs [8] which are counterparts of the two FSMs in the transmitter. The two SISOs are connected by an interleaver and a deinterleaver to exchange soft information iteratively. In [8], the architecture of standard iterative detection with SISOs was described. Iterative detection was applied to systems with unknown parameters in [4] where the inner SISO was adaptive and re-estimated the ISI channel during each activation (i.e., each iteration). The fixed-lag A-SISO architectures of this paper are based on the development of the A-SISOs in [4] and the relation between the L²VS and bi-directional architectures established in [9]. The adaptive iterative detection algorithm is the same as that developed in [4] with the inner A-SISO changed from the FI version to various fixed-lag A-SISO algorithms.

III. SISO ALGORITHMS

A SISO algorithm can be classified by the type of soft output as the APP and the MSM, with the latter defined by

$$MSM_{k_1}^{k_2}(u_k) = -\ln \max_{l_{k_1}:u_k} P(z_k|b_k, l_{k_1}) P(t_k|l_{k_1})$$  

(2)

where $k$ is the time index between $k_1$ and $k_2$, $t_k = (s_k; a_k; s_{k+1})$ is the transition, $u_k$ is an arbitrary quantity related to the FSM (e.g., $a_k$ in the system of Fig. 1), and $l_{k_1}:u_k$ denotes all transitions consistent with $u_k$. It is noted that the MSM is the negative log of a generalized APP. The two types of soft output are interchangeable by replacing min-sum operation by sum-product operations. Therefore we will concentrate on the MSM-based algorithms in this development. Based on the MSM soft output, the basic forward-Add Compare Select(ACS) and backward-ACS recursions are

$$MSM_{k_1}^{k_2}(s_{k+1}) = \min_{t_{k+1} : s_{k+1}} [MSM_{k_1}^{k-1}(s_k) + M_k(t_k)]$$  

(3)

$$MSM_{k_1}^{k_2}(s_k) = \min_{t_{k+1} : s_{k+1}} [MSM_{k_1}^{k-1}(s_{k+1}) + M_k(t_k)]$$  

(4)

where $M_k(t_k)$ is a transition metric in terms of $t_k$, specifically

$$M_k(t_k) = \|z_k - x(t_k, f_k)\|^2 + SI(a_k)$$  

(5)

The extrinsic information is computed by the following completion step.

$$SO_{k_1}^{k_2}(u_k) = \min_{l_{k_1}:u_k} [MSM_{k_1}^{k-1}(s_k) + M_k(t_k) +$$

$$MSM_{k_1}^{k_2}(s_{k+1})] - SI(u_k)$$  

(6)

where $SI(u_k)$ is the input soft-information on $u_k$ and $SO_{k_1}^{k_2}(u_k)$ is the output soft-information on $u_k$ based on input soft information between times $k_1$ and $k_2$.

Since we consider the time variant channel, the ACS algorithm is combined with a channel estimation algorithm. We will refer to this as an adaptive-ACS. In the time variant channel, to get a practical algorithm the sequence tree must be folded into a trellis, yielding recursions analogous to those in (3),(4)

$$F_{k_1}^{k_2}(s_{k+1}) = \min_{t_{k+1}:s_{k+1}} [F_{k_1}^{k_2-1}(s_k) + M_k(t_k)]$$  

(7)

$$B_{k_1}^{k_2}(s_k) = \min_{t_{k+1}:s_{k+1}} [B_{k_1}^{k_2-1}(s_{k+1}) + M_k(t_k)]$$  

(8)

where $M_k(t_k) (M_k^b(t_k))$ is obtained by inserting the forward (backward) channel estimate $\hat{f}(s_k)$ ($\hat{f}(s_{k+1})$) in place of $f_k$ in (5). The state metrics $F_{k_1}^{k_2}(s_{k+1})$ and $B_{k_1}^{k_2}(s_k)$ represent approximations of the MSMs with averaging over the channel statistics [4]. Specifically, the adaptive-ACS operations in (7)-(8) are suboptimal due to a forced-folding of forward and backward sequence trees onto a trellis. Similarly, the completion equation (6) is also changed as

$$SO_{k_1}^{k_2}(u_k) = \min_{l_{k_1}:u_k} [F_{k_1}^{k_2-1}(s_k) + M_k(t_k) + B_{k_1}^{k_2-1}(s_{k+1}) +$$

$$b(\hat{f}(s_{k+1}), \hat{f}(s_{k+1})) - SI(u_k)$$  

(9)

where $\hat{f}(s_k) = \{\hat{f}_k(s_k, i)\}_{i=0}^1$ is the forward estimate of the unknown vector (i.e., the channel coefficients in this context) and similarly $\hat{f}(s_{k+1})$ is the backward estimate of the channel. Note that there is an additional term $b(\hat{f}(s_{k+1}), \hat{f}(s_{k+1}))$ in (9) which is called as a binding term in [4]. The binding term was derived for specific channel models in [4] with these expressions used to motivate binding terms for other channel estimation strategies.
In this development, the Least Mean Square (LMS) algorithm is used for both forward and backward channel estimation in a PSP manner. In PSP, the code sequence of each survivor is used for decision feedback for the per-survivor estimate of the unknown parameters [10]. The forward and backward LMS algorithm are respectively given by

\[ \tilde{r}(s_k) = \tilde{r}'(s_{k-1}) + \beta [z_{k-1} - \tilde{r}'(s_{k-1})] \tilde{a}'(s_k) \tilde{a}'(s_k) \] (10)

\[ \tilde{r}'(s_k) = \tilde{r}'(s_{k+1}) + \beta [z_k - \tilde{r}'(s_{k+1})] \tilde{a}'(s_k) \tilde{a}'(s_k) \] (11)

where \( \tilde{a}'(s_k) = \{ \tilde{a}_{k-1} \}_{l=1}^{L+1} \), \( \tilde{a}'(s_k) = \{ \tilde{a}_{k-l} \}_{l=0}^{L} \) and \( \beta \) is a step-size of the LMS algorithm. The best choice of \( \beta \) depends on the \( E_b/N_0 \) and the channel dynamics. The forward channel estimate \( \tilde{r}(s_0) \) is initialized by running RLS algorithm on a training sequence at the left edge. The backward channel initial estimates \( \tilde{r}(s_{N+1}) \) are computed in different ways corresponding to the algorithms described in the following subsections. The binding term suggested for the LMS channel estimation algorithm in [4] is given by

\[ b(\tilde{r}'(s_k), \tilde{r}'(s_k)) = \frac{\rho}{1 - \beta^2} ||\tilde{r}'(s_k) - \tilde{r}'(s_k)||^2 \] (12)

where \( \rho = \frac{1 - L^2}{1 - L^2 \beta} \). Larger disagreement between forward and backward channel estimates for a given transition yields a larger binding terms, which in turn reduces the reliability measure via the completion step in (11) (i.e., larger sequence metric). In the following subsections the details of the various algorithms are described.

A. FL-BD-A-SISO Algorithm

A FL-BD-A-SISO algorithm is realized by the forward adaptive-ACS in (7),(10), the backward adaptive-ACS in (8),(11), and the completion with the binding term in (9),(12) with \( k_1 = 0 \) and \( k_2 = N - 1 \) respectively. The FL algorithm is obtained based on the assumption that both forward and backward training sequences are available. Therefore the forward channel initial estimate \( \tilde{r}(s_0) \) and backward channel initial estimate \( \tilde{r}(s_{N+1}) \) are obtained using the RLS algorithm and these training sequences [4].

B. FL-BD-A-SISO Algorithm

A FL-BD-A-SISO algorithm is realized by the same procedure as the FL-BD-A-SISO algorithm, but the backward adaptive recursion is started with \( k_2 = k + D \) to obtain the \( SOO_0^{k+D}(u_k) \) where \( k \) denotes the time index of the current soft output. In Fig. 2(a), the backward recursion and the completion are illustrated. Unlike the FL algorithm, the FL algorithm does not require a backward training sequence. This may more accurately reflect the training methods used in current systems. A challenge is to initialize the backward channel estimates. Among the various possible solutions, the simplest method is to use the latest updated forward channel estimate \( \tilde{r}'(s_{k+D+1}) \) as the backward channel initial coefficients \( \tilde{r}'(s_{k+D+1}) \). We refer to this as FL-BD-BA (Bi-directional Estimation). Two completion methods in (9) with \( k_2 = k + D \) are considered with the binding term in (12) and without the binding term. A variation on this backward initialization is also considered where the backward adaptive recursion is processed up to \( k + D + d \) instead of \( k + D \). When \( d \) is large enough, survivor merging will occur between times \( (k + D) \) with \( k + D + d \) with high probability. By storing all channel estimates for this forward recursion, one can traceback from time \( k + D + d \) to find the channel estimate associated with the best state at time \( k + D \). This estimate associated with the best survivor can then be used to initialize the backward channel estimates for all states.

Another variation is to store the forward channel estimates of every state during the forward adaptive-ACS processing. Specifically, store \( \tilde{r}'(s_k) \) for each \( s_k \) and each time \( k \). The backward ACS is then run using these stored forward channel estimates in a PSP manner without channel adaptation. Specifically, in place of \( \tilde{r}'(s_{k+1}) \) for the transition metric computation in (8), the stored values for \( \tilde{r}'(s_k) \) are used. Since backward channel estimation is not performed in this approach, the issue of backward estimator initialization is avoided. Similarly, no binding is required in this approach. We refer to this algorithm as FL-BD-FE (Forward-only Estimation).

C. FL-L²VS-A-SISO Algorithm

The FL-L²VS algorithm is based on a constrained forward ACS, defined by

\[ F_k^{t_k}(u_{k-d}, s_k+1) = \min_{t_k: (k+1); u_{k-d}} \left[ F_k^{t_k-1}(u_{k-d}, s_k) + M_k(t_k) \right] \] (13)

Note that in the constrained ACS all transitions \( t_k-d \) that are inconsistent with the conditional quantity \( u_{k-d} \) are terminated [9], [2]. In the case of perfect channel knowledge, it can be shown that \( F_k^{t_k}(u_{k-d}, s_k) \) is the MSM of the quantity \( u_{k-d}, s_k \) based on the metrics from \( k_1 \) to \( k \). More precisely, with the metric of (5) used, the constrained ACS in (13) provides a recursive update for this joint MSM. The L²VS algorithm uses a single unconstrained forward ACS as in (3) and one constrained ACS as in (13) for each value of \( u_{k-d} \) for \( d = 0, 1, \ldots D \) [9]. Finally, the desired FL soft output is obtained by minimizing the MSM of \( (u_{k-D}, s_{k+1}) \) over all \( s_{k+1} \) [9]. From this interpretation, it is clear that the L²VS is more complex than the bi-direction architecture by a factor equal to the cardinality of \( u_k \).

In the adaptive context, we consider two variations of the FL-L²VS-A-SISO algorithm. In Fig. 2 the two FL-L²VS-A-SISO algorithms are illustrated. In Fig. 2(b), both the constrained forward ACS and the unconstrained forward ACS recursions are adaptively implemented in a PSP manner. Specifically, the metric used in (13) is the \( M_k(t_k) \) associated with the constrained or unconstrained adaptive ACS. We refer to this as FL-L²VS-CE (i.e., constrained estimation).

The other variation is shown in Fig. 2(c). When the unconstrained forward adaptive-ACS recursion is running, the chan-
block interleaver. Each column of block interleaved symbols with additional training sequence information is referred to as a burst. A burst is the processing unit of inner adaptive SISO. The interleaved symbols were sent over a 3-tap i.i.d. Rayleigh fading channel with normalized Doppler spread \(v_d=0.005\). All of the simulations were executed under the wide sense stationary, uncorrelated scatter (WSSUS) model. Each tap of the channel was generated as an independent Gaussian process with the Clarke spectrum [13]. For each \(E_b/N_0\) we applied the optimal step-size (\(\beta\)) for LMS channel estimation (i.e., determined empirically). In this section, each algorithm was identified by the following label

- the type of observation (FI or FL)
- the type of FL structure (BD or L)
- the type of binding (B:Binding or NB:No Binding)
- the type of observation (FI or FL)
- the type of bi-directional channel adaptation (BE:Bi-directional Estimation or FE:Forward-only Estimation)

In Fig. 3 the performance of the adaptive iterative detector is shown for the first and the fifth iteration. For comparison, two FL methods (BD, L) were shown with the FI algorithm. As the FL-BD-FE and FL-L VS-UE algorithms generate the same soft output, they show the same performance in Fig. 3. The performance of the FL-BD-FE/FL-L VS-UE algorithm is about 5dB in \(E_b/N_0\) worse than that of the FI algorithm at a BER of \(10^{-3}\) for the first iteration but is only 2dB worse for the fifth iteration. Note that the FL algorithm shows a larger iteration gain compared with the FI algorithm. For the FL-BD-FE algorithm, the backward ACS recursion was simulated with the previously stored forward channel estimates as described in the previous section. For the FL-BD-BE algorithm we simulated with the simplest backward initialization method which was to initialize backward channel estimator with the latest estimated

![Fig. 2. Architectures for the FL algorithms: (a)FL-BD-BE, (b)FL-BD-FE, (c)FL-L VS-CE, and (d)FL-L VS-UE](image)

Fig. 2. Architectures for the FL algorithms: (a)FL-BD-BE, (b)FL-BD-FE, (c)FL-L VS-CE, and (d)FL-L VS-UE

![Fig. 3. The performance of the fixed-interval and fixed-lag adaptive SISO algorithms, \(D=6\)](image)

Fig. 3. The performance of the fixed-interval and fixed-lag adaptive SISO algorithms, \(D=6\)

IV. NUMERICAL RESULTS

The simulations were run based on the system described in Section II. The convolutional code rate was 1/2 and the output of convolutional code was modulated by QPSK via the Gray mapping. The coded symbols were interleaved using a 57x30
value of the forward channel estimator. As a result, the backward estimate has a strong dependency on the forward estimate. Therefore, we expect little inconsistency between the forward and backward channel estimates, and thus a small binding gain for the FL-BD-BE algorithm. This is in contrast to the FL algorithm, which has a large binding gain (i.e., 2dB). Although we do not present the results here, the variation of the FL-BD algorithm using channel estimates after merging was also simulated with no significant improvement observed. In conclusion, attempts to use backward channel estimation without a backward channel training sequence were not as successful as the novel FL-A-SISO algorithms introduced herein with only forward channel estimation.

In Fig. 3, the two different channel adaptation algorithms for the FL-L^2VS are also compared. The FL-L^2VS-UE performs better than the FL-L^2VS-CE. The channel coefficients are estimated based on the symbol estimates in the PSP manner, and the symbol estimates come from the ACS recursion. Therefore, the constrained ACS recursion provides a constraint on the channel estimates. This constraint on the channel estimate apparently degrades the performance in the FL-L^2VS-CE. Although the FL-L^2VS-UE requires additional storage for the previous channel estimates, it yields a significant reduction in computational complexity relative to the FL-L^2VS-CE algorithm. However, the FL-L^2VS-UE is more complex by a factor of N_a which is the alphabet size of the symbol a_k (e.g., N_a=4 for QPSK) compared to the FL-BD-FE algorithm. As the alphabet size N_a increases, the FL-BD-FE algorithm is much less complex than the FL-L^2VS-UE algorithms while they produce exactly the same soft-information. In Fig. 4 the performance of the proposed FL algorithms was compared with ZFG’s algorithm [12] for different lag sizes. The performance of the FL-BD-FE/FL-L^2VS-UE algorithms and ZFG’s algorithm were found to be similar despite a significant difference in complexity (i.e., with D=2,6, ZFG’s algorithm has 16 and 4096 states, respectively, as compared to 16 states for the FL-BD-FE/FL-L^2VS-UE). Moreover, for the FL-BD-BE algorithm, the performance was not improved noticeably when the lag size was larger than 3. However, for the FL-BD-FE/FL-L^2VS-UE algorithms the performance gradually increases with the lag size.

V. CONCLUSION

The FL-A-SISO algorithms were proposed and the performance were compared for TCM in interleaved frequency selective fading channel. It was shown that both FL algorithm perform slightly worse than the FI version while maintaining the iteration gain of the FI algorithm. Without the necessity of backward channel estimation, the FL-BD-FE and FL-L^2VS-UE algorithms performed better than any proposed variations of the fixed-lag adaptive structure. It was noted that the complexity of the FL-BD-FE is lower than that of the FL-L^2VS-UE algorithm while they produce exactly the same soft-output and hence yield the same performance.

REFERENCES


Note that for this non-recursive FSM, one can use “L-early completion” – i.e., the backward recursion for the BD algorithm or the constrained forward recursions of the L^2VS approach can be shortened by L steps. This was not done in the simulations, but a practical D = L (i.e., D = 2 in this case) version of the BD, L^2VS, and ZFG will all be the same as given in [11].