On the Reduced Trellis Equalization Using the M-BCJR Algorithm

Christina Fragouli
Department of Electrical Eng.
UC Los Angeles
LA, CA 90024
e-mail: christin@ee.ucla.edu

Nambi Seshadri
Broadcom Corporation
Irvine, CA 92615
e-mail: nambi@broadcom.com

William Turin
AT&T Labs-Research
Florham Park, NJ 07932
e-mail: wt@research.att.com

Abstract — **The complexity of channel equalization using the BCJR algorithm [1] grows exponentially with the channel memory. The M-BCJR algorithm [2], provides a method for reduced-trellis channel equalization. This paper examines the algorithm performance for different finite impulse response channels, identifies pathological cases, and introduces designer choices such as the decision delay, maximum and minimum phase transformation, and selection of states used by the algorithm. Simulation results demonstrate the effect of the different parameters in the algorithm's performance.**

I. Introduction

Typically, intersymbol interference (ISI) channels are modeled by discrete-time finite-state machines, which can be described by a trellis. Trellis equalization provides improved performance as compared to linear or decision feedback equalizers at the cost of increased complexity. The complexity grows exponentially with the number of symbols that are affected by the ISI. Therefore, it is important to develop suboptimal algorithms that require less computing resources without significant degradation in performance.

In this paper, we consider soft decision trellis equalizers which are more attractive than hard decision equalizers [3, 4] to applications using coding. The soft decision is achieved by computing a transmitted symbol a posteriori probability (APP) on the basis of the received sequence. For the ISI channel with independent noise, the APP can be computed using the forward-backward algorithm whose complexity depends on the number of channel states.

Suboptimal reduced-state MAP equalization algorithms, such as the M-algorithm and the T-algorithm [2], select the most important sequence of states for which the APP does not differ significantly from that of the complete state sequence. The M-algorithm uses a subset of M states at each trellis step, while the T-algorithm uses variable number of states at different steps. In this paper, we analyze various factors that affect the M-algorithm performance, and identify pathological cases. We propose a variation of the M-algorithm which offers consistent performance over different finite impulse response (FIR) channels, and is based on the implementation of the forward-backward algorithm in the log-domain. Moreover, we demonstrate that, for the same number of reduced states, the performance can be improved by using the minimum-phase or maximum phase transformation of the received sequence. Finally, we examine the effect of the decision delay, and the distribution of states between the forward and backward recursions.

The paper is organized as follows. Section II presents the channel model and reviews the BCJR algorithm. Section III discusses the reduced-state M-BCJR algorithm, which motivates our proposed implementation in the log domain. Various implementations are introduced and discussed. Section IV uses all-pass filtered channel observations to enhance the algorithm's performance. Section V presents the simulation results, and Section VI concludes the paper.

II. System model

The transmitter sends discrete time information symbols $\{x_k\}$ through a band-limited channel, at a rate $1/T$ *symbols/sec* higher than the Nyquist rate. The cascade of the analog filter at the transmitter, the channel, the matched filter at the receiver, the sampler operating at the symbol rate $1/T$, and the discrete-time noise-whitening filter, is represented as a discrete-time equivalent channel with impulse response h_k [5]. We assume that the intersymbol interference (ISI) affects a finite number K of transmitted symbols $(x_{t-K+1} \dots x_t)$. Thus the output Y_t at time t (Fig. 1) can be expressed as:

$$Y_t = \sum_{k=0}^{K-1} h_k x_{t-k} + n_t \tag{1}$$

where $\{n_k\}$ is the discrete-time Additive White Gaussian Noise (AWGN).

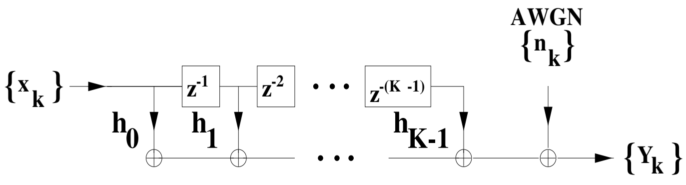

Fig. 1: Equivalent discrete-time model of channel with intersymbol interference.

The equivalent discrete time channel is a finite-state machine whose state at time t is defined by the $K-1$ most recent inputs $s_t = (x_{t-K+2} \dots x_t) = (x_{t-K+2}^t)$. Thus the channel can be described by a trellis with C^{K-1} states, where C is the constellation size.

The maximum a posteriori (MAP) symbol decoder maximizes the symbol x_t a posteriori probability (APP) after receiving a sequence Y_1^N which is proportional to the so-called unnormalized APP $Pr(x_t, Y_1^N)$. To calculate this probability we define an input-output state-transition probability matrix $\mathbf{P}(x_t, Y_t)$ whose elements are conditional probabilities of transferring from state s_{t-1} to state s_t, and having x_t as an input and Y_t as the corresponding output ([6]). Using this matrix we can express the APP in the following matrix form

$$Pr(x_t, Y_1^N) = \boldsymbol{\alpha}(Y_1^{t-1})\mathbf{P}(x_t, Y_t)\boldsymbol{\beta}(Y_{t+1}^N) \quad (2)$$

where the forward and backward probability vectors are defined as

$$\boldsymbol{\alpha}(Y_1^{t-1}) = \boldsymbol{\pi}\mathbf{P}(Y_1)\mathbf{P}(Y_2)\ldots\mathbf{P}(Y_{t-1}) \quad (3)$$

$$\boldsymbol{\beta}(Y_{t+1}^N) = \mathbf{P}(Y_{t+1})\mathbf{P}(Y_{t+2})\ldots\mathbf{P}(Y_T)\mathbf{1} \quad (4)$$

$\boldsymbol{\pi}$ is a vector of the state initial probabilities,

$$\mathbf{P}(Y_t) = \sum_x \mathbf{P}(x, Y_t)$$

and $\mathbf{1}$ is a column vector whose all elements are ones. The forward and backward probability vectors can be computed recursively as

$$\boldsymbol{\alpha}(Y_1) = \boldsymbol{\pi}\mathbf{P}(Y_1) \quad \boldsymbol{\alpha}(Y_1^t) = \boldsymbol{\alpha}(Y_1^{t-1})\mathbf{P}(Y_t)$$

$$\boldsymbol{\beta}(Y_N) = \mathbf{P}(Y_N)\mathbf{1} \quad \boldsymbol{\beta}(Y_t^N) = \mathbf{P}(Y_t)\boldsymbol{\beta}(Y_{t+1}^N)$$

The previous equations for computing the unnormalized APP represent the so-called forward-backward algorithm. The APPs serve as soft decision output of the BCJR algorithm. Because of it and the utility of trellis diagram, we will often call them soft decisions at a trellis node. As we can see, the algorithm's complexity grows exponentially with the ISI length K. For complexity and precision reasons, these equations are usually implemented in the logarithmic domain [7].

III. Reduced-state M-BCJR algorithm

The M-BCJR algorithm is a reduced state-trellis decoding technique [2]. At each trellis step it keeps only M active states associated with the highest metrics and disregards the rest. In other words, the forward vectors $\boldsymbol{\alpha}(Y_1^{t-1})$ and backward vectors $\boldsymbol{\beta}(Y_{t+1}^N)$ retain the M largest value components, and zero the rest thus prohibiting path propagation from them.

Let $\mathbf{A_t^M}$ denote the set of M nodes selected by the forward recursion at time t, and $\mathbf{B_t^M}$ the corresponding set for the backward recursion. The edges (i, j) between active nodes $(i \in \mathbf{A_{t-1}^M},\, j \in \mathbf{B_t^M})$ are the only ones that take part in the APP calculation. It is difficult to determine those edges, so typically the forward recursion is allowed to choose the tree of active nodes $\mathbf{A} = \bigcup_t \mathbf{A_t^M}$, while the backward recursion may only follow a subtree of the chosen active nodes, i.e. $\mathbf{B_t^M} \subseteq \mathbf{A_t^M}$ for every t [2]. Decisions are taken using the edges in the common subtree.

This approach simplifies the problem of suitably combining the $\boldsymbol{\alpha}(Y_1^{t-1})$ and $\boldsymbol{\beta}(Y_{t+1}^N)$ vectors to form soft decision products, but it biases the probability of error upon the correct selection of active nodes by the forward recursion. This bias leads to significant degradation of performance if the channel has low energy in the first taps, as is the case for many channels. Fig. 2 shows, as an example, the difference in performance of the M-Algorithm between the linear phase channel: [0.3313 0.8835 0.3313] and the corresponding min-phase: [0.7339 0.6623 0.1496] and max-phase: [0.1496 0.6623 0.7339] channels.

One way to explain this difference is to examine when the input path $X_{corr} = \{x_0^{N-1}\}$, corresponding to the sequence of the actually transmitted channel symbols, is included in the "forward" active tree $\mathbf{A}$. From 1 the output

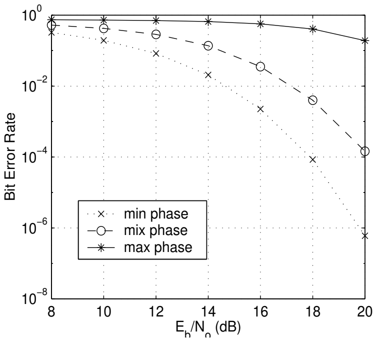

Fig. 2: Performance of the M-Algorithm for the different phases of the 3-tap channel: [0.3313 0.8835 0.3313], and M=4. The simulations employ 8-PSK modulation $\Rightarrow$ total number of states=64.

at time t is $Y_t = \sum_{k=0}^{K-1} h_k x_{t-k} + n_t$. Consider the previous state that corresponds to the correct transmitted data: $s_{t-1} = (x_{t-K+1}\ldots x_{t-1}) \subset X_{corr}$, and assume it is among the active ones at time $t-1$: $s_{t-1} \in \mathbf{A_{t-1}^M}$.

The metric associated with each of the next states of the form: $s_t = (x_{t-K+2}\ldots x_{t-1}\ u)$, for each possible input u, depends on the distance:

$$||Y_t - (h_k x_0 + \sum_{k=1}^{K-1} h_k x_{t-k})||_2 = ||h_0(x_t - u) + n_t||_2$$

where $||.||_2$ denotes the Euclidean norm. This metric will determine if the correct state will be active in the next step. Thus the "free distance" $d_A \triangleq ||h_0 d_C||_2$ of our decision is the constellation free distance $d_C \triangleq min_{i,j}||i-j||_2$ for any i, j constellation points, weighted by the coefficient of the first tap. The smaller this coefficient is, the higher is the probability of not including the correct path X_{corr} in the active tree $\mathbf{A}$, and hence the deterioration of the performance when the channel has low energy in its first taps.

The backward recursion sees a time-reversed version of the channel, because of its filters anticausallity. Thus, the corresponding free distance d_B can be defined as: $d_B = ||h_{K-1} d_C||_2$, where h_{K-1} is the last tap of the FIR channel of length K.

Another problem that the M-algorithm might have, is that since the backward recursion cannot follow paths that are disrupted at an intermediate trellis step sometimes the forward recursion may allow only one possible path and thus the decoding approximates the Viterbi Algorithm as in the example that Fig. 3 depicts. Moreover, we may even not have soft metrics for all inputs at every step. All those effects become more pronounced as M decreases (Fig. 4).

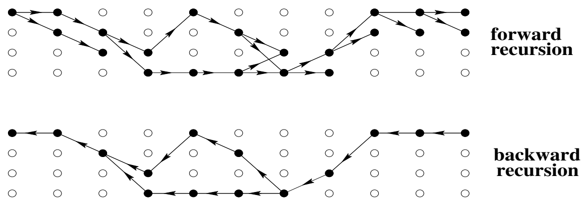

Fig. 3: Example of M-Algorithm, where M=2, and total number of states=4.

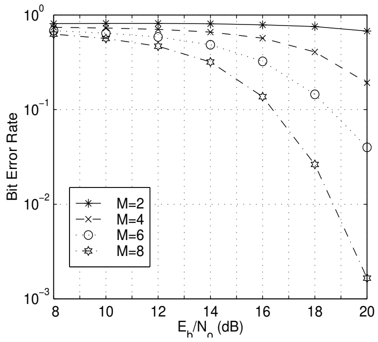

Fig. 4: Effect of parameter M in the M-Algorithm performance over the max-phase channel: [0.1496 0.6623 0.7339]. The simulations employ 8-PSK modulation.

A. Proposed Implementation

MAP is commonly implemented in the logarithmic domain to improve the precision of computations and reduce complexity. To counteract the bias that the selection of active nodes by only one of the recursions incurs, we can take advantage of the implementation of the MAP algorithm in the log-domain.

Let the forward and backward recursions independently select M nodes at each trellis step and propagate metrics only from them. That is, build independently the trees $\mathbf{A} = \bigcup_t \mathbf{A_t^M}$ and $\mathbf{B} = \bigcup_t \mathbf{B_t^M}$ without restricting one to be a subtree of the other. To form soft decisions at time t, use all the trellis edges or all the edges with at least one active node. As metrics for the not active nodes, use a negative number that expresses the reliability of the selection of M nodes by the forward or the backward recursions (instead of minus infinity which corresponds to probability zero in the log domain). For example, if at time t the forward recursion is reliable and the backward unreliable, use a negative number I_A with a large absolute value for $\alpha_i(Y_1^{t-1})$, $i \notin \mathbf{A_{t-1}^M}$, and a negative value I_B close to zero for $\beta_j(Y_{t+1}^N)$, $j \notin \mathbf{B_t^M}$. In other words try to approximate soft instead of hard decisions for the choice of the sets $\mathbf{A_t^M}$ and $\mathbf{B_t^M}$ by using appropriate values of I_A and I_B.

The values I_A and I_B depend upon the noise power, the signal energy and its distribution among the channel taps. A possible choice for I_A, where i denotes a not active state, is:

$$I_A = logPr(\{i \notin \mathbf{A_t^M}\} | \{i \in X_{corr}\})$$

which can be approximated as:

$$IA \backsimeq logPr(\{\{i \notin \mathbf{A_t^M}\} | \{j \in \mathbf{A_{t-1}^M},\ j, i \in X_{corr}\})$$

and for the AWGN we can further use the "decision free distance" d_A to calculate I_A as:

$$I_A = logQ(d_A/\sigma\sqrt{2}), \quad d_A = ||h_0 d_C||_2 \tag{5}$$

where $Q(.)$ is the Q-function. Similarly for I_B we may use:

$$I_B = logQ(d_B/\sigma\sqrt{2}), \quad d_B = ||h_{K-1} d_C||_2 \tag{6}$$

B. Delayed Decisions

The nonlinearity introduced by deciding on the sets $\mathbf{A_t^M}$ and $\mathbf{B_t^M}$ has similarities to the decision feedback equalizer (DFE) nonlinear hard decisions. Indeed the M-algorithm with $M_A = 1$ and $M_B = 0$ reduces to a DFE structure with no feedforward filter. Also, similarly to DFE, the M-algorithm suffers from error propagation.

In the minimum mean square error (MMSE) DFE framework making decisions with a delay $\Delta > 0$ can result in significant performance improvement [8]. Similarly, during the MAP decoding for a channel with K taps at time instance t we can take soft decisions about any $x_{t-\Delta}$ with $\Delta = 0 \dots K-1$ by summing over $x_{t-K+1}^{t-\Delta-1}$ and $x_{t-\Delta+1}^{t}$ the probabilities

$$\begin{aligned} Pr(x_{t-K+1}^t, Y_1^N) &= Pr(s_{t-1} = i, s_t = j, Y_1^N) \\ &= \alpha_i(Y_1^{t-1}) Pr(Y_t, j|i) \beta_j(Y_{t+1}^N) \end{aligned}$$

of trellis edges connecting nodes $s_{t-1} = x_{t-K+1}^{t-1}$ and $s_t = x_{t-K+2}^t$. In this equation, $\alpha_i(Y_1^{t-1})$ is the i element of the forward probability vector $\boldsymbol{\alpha}(Y_1^{t-1})$, $Pr(Y_t, j|i)$ is the (i,j) element of the output transition probability matrix $\mathbf{P}(Y_t)$ and $\beta_j(Y_{t+1}^N)$ is the j element of the backward probability vector $\boldsymbol{\beta}(Y_{t+1}^N)$. Fig. 5 illustrates the delayed decisions for $\Delta = 2$. The optimal delay Δ depends on the reliability of the α and β metrics and the distribution of a channels energy to its taps.

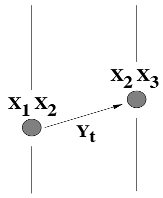

Fig. 5: Decisions with delay: at time t we can take soft decisions for any of the input symbols x_1, x_2 and x_3 by summing the probabilities of the appropriate in each case trellis edges.

IV. All-pass filtered M-BCJR algorithm

Equations (5) and (6) imply that the accuracy of the forward recursion is better if most of a channels energy is concentrated in its first taps, while the backward recursion accuracy benefits if the energy is concentrated in the last taps.

Among all the systems having the same frequency response magnitude and the same total energy, the minimum-phase system has the largest partial energy

$$E(n) = \sum_{k=0}^{n} |h_k|^2, \text{ for every } n \geq 0$$

and the maximum-phase system has the smallest partial energy [9]. The minimum phase system has all zeros inside the unit circle, while the maximum phase has all its zeros outside the unit circle. If the system has some zeros inside and some outside the unit circle it is called mixed phase.

We propose to use the minimum phase channel for the forward recursion and the maximum phase channel for the backward recursion. In other words use a different suitable output observed sequence for each recursion. The architecture is depicted in Fig. 6.

Consider a mixed phase channel with zeros: $z_1, z_2, \dots z_K$ such that $z_1, z_2, \dots, z_M$ are inside the unit circle while the remaining zeros $z_{M+1}, z_{M+2} \dots, z_K$ are outside the unit circle. To create the equivalent minimum phase channel, the received

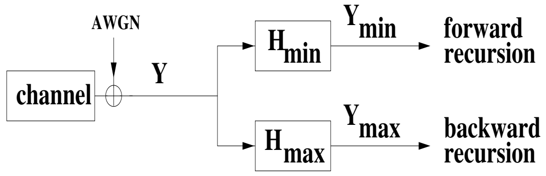

Fig. 6: Transform a mix-phase output sequence to the min and max-phase output sequences.

(mixed phase channel) output signal Y is passed through the anti-causal stable filter whose z-transform is

$$H_{min}(z) = \frac{(z - \frac{1}{z_{M+1}})}{(z - z_{M+1})} \frac{(z - \frac{1}{z_{M+2}})}{(z - z_{M+2})} \cdots \frac{(z - \frac{1}{z_K})}{(z - z_K)} \qquad (7)$$

Similarly, to create the equivalent maximum phase channel, the received signal is passed through the causal stable filter:

$$H_{max}(z) = \frac{(z - \frac{1}{z_1})}{(z - z_1)} \frac{(z - \frac{1}{z_2})}{(z - z_2)} \cdots \frac{(z - \frac{1}{z_M})}{(z - z_M)} \qquad (8)$$

The inverse filter $H_{min}^{-1}(z)$ corresponds to a causal stable system and $H_{max}^{-1}(z)$ corresponds to an anti-causal stable system. Since H_{min} and H_{max} are all-pass filters, the noise at their output is not colored.

Let $Y_{min,t}$ and $Y_{max,t}$ be the $H_{min}(z)$ filter and $H_{max}(z)$ filter outputs, respectively (Fig. 6). Based on the previous analysis, we can use $Y_{min,t}$ for computing the forward probability vectors (3) and $Y_{max,t}$ for computing the backward probability vectors (4). Since $H_{min}^{-1}(z)$ is strictly causal and $H_{max}^{-1}(z)$ strictly anti-causal, the sequence

$$\{Y_{min,1}, Y_{min,2} \ldots Y_{min,(t-1)}, Y_t, Y_{max,(t+1)} \ldots Y_{max,N}\}$$

uniquely identifies the original channel output sequence Y_1^N, and thus incurs no loss of information.

V. Simulation Results

The simulation results investigate three different things: The advantage of using both the forward and backward trees and their combining with the reliability metrics I_A and I_B; the effect of delayed decisions on the performance of the M-algorithm; and the gain of exploiting the phase characteristics of a channel.

For the simulations, we consider two channels employing 8-PSK modulation. The first channel C_1 has three taps and significantly different mixed, minimum and maximum phase impulse responses. The second channel C_2 with three taps [5] has zeros very close to the unit circle. As a result, the mixed, minimum and maximum phase channel taps are almost the same. The tap coefficients for both channels are summarized in Table 1. For the simulations, we assume that the receiver has perfect knowledge of the channel. We denote as M_A the number of active states for the forward recursion, and M_B - the number of active states for the backward recursion.

A. Channel C_1.

Fig. 2 and Fig. 4 presented simulation results for channel C_1 using the standard M-algorithm. We observed that the M-algorithm does not offer consistent performance over the different channel phases, and that as M increases the difference in performance decreases. The simulation results were for 10^7 symbols (20,000 blocks of length 500 each).

Tab. 1: Simulations Channels Coefficients

Channels	mix-phase
C_1	{0.33 0.88 0.33}
C_2	{0.4079 0.8168 0.4079}
	min-phase
C_1	{0.73 0.66 0.15}
C_2	{0.4286 0.8158 0.3882}
	max-phase
C_1	{0.15 0.66 0.73}
C_2	{0.3882 0.8158 0.4286}

The results of building independently both the forward and backward recursion trees for the M-Algorithm over the same channel are presented in Figures 7 and 8 for the max and mix-phase channel respectively. I_A and I_B are calculated from Equations (5) and (6). The performance significantly improves for the max-phase channel, but stays the same for the mix-phase channel. The backward recursion takes advantage of the max-phase channel and offers increased reliability (which is not the case for the mix-phase channel). Both plots are for $M_A = M_B = 2$ active states out of 64 total trellis states.

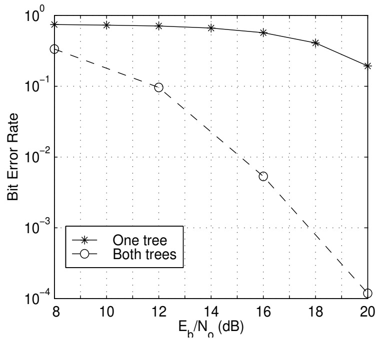

Fig. 7: Combining both forward and backward recursion trees using I_A and I_B for max phase channel.

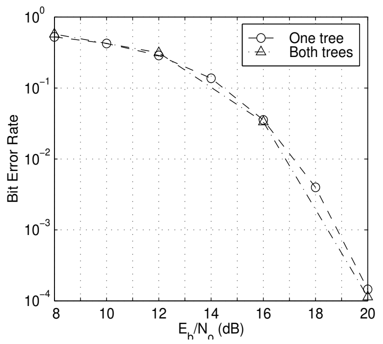

Fig. 8: Combining both forward and backward recursion trees using I_A and I_B for mix phase channel.

The previous plots examined time-invariant channels. Consider a Rayleigh time-varying channel with tap gains equal to the mix channel coefficients. For the constant mix-phase channel introduction of both trees did not offer performance improvement. But when the channel becomes time-variant, the relative tap values change, and introduction of both trees improves the performance as Fig. 9 demonstrates. For the simulations we assumed that the channel changes independently for every block of 100 symbols, and we simulated 10^4 blocks (10^6 symbols).

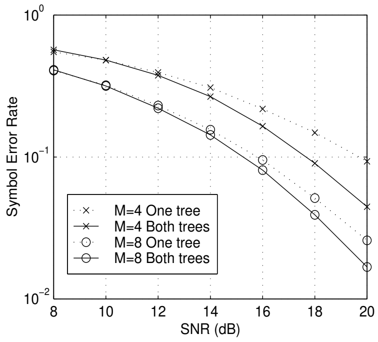

Fig. 9: Rayleigh channel with tap gains: [0.33 0.88 0.33].

Note that the performance of the standard M-algorithm (one tree) is better over the min-phase channel when we use one tree, than when we use both trees. Fig. 10 shows the performance when we use both trees, with different number of active nodes M_A and M_B for every tree, and total number of active nodes is constant: $M_A + M_B = 8$. The simulation results are for 10^6 symbols.

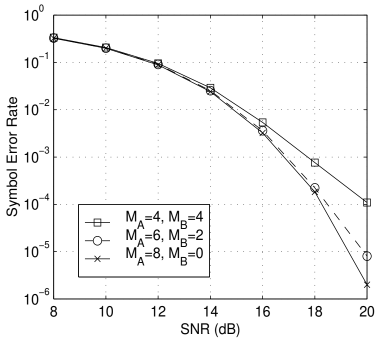

Fig. 10: Both trees, performance for different values of M_A and M_B, $\mathrm{M}_A + \mathrm{M}_B = 8$.

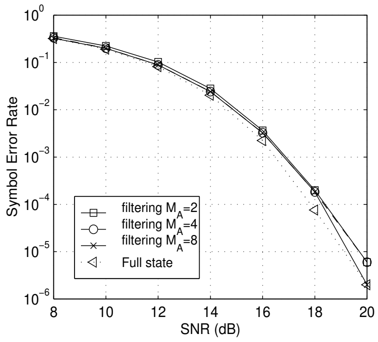

Fig. 11: Filtering performance for different values of M_A.

For this channel, indeed filtering ($\mathrm{M}_B = 0$), that is closest to the one-tree M-algorithm, gives the best performance. For a known channel, and a constant required complexity $M_A + M_B$, one could distribute the available number of active nodes between the forward and backward recursions to optimize the performance.

Fig. 11 shows the performance of filtering for different values of M_A. In the same plot, the dotted curve gives the performance of the full MAP decoding (using all 64 states). Increasing M_A offers a difference in performance only below 10^{-4}.

Fig. 12 and 13 demonstrate the effect of the decision delay. The simulations use both trees for the M-algorithm. The optimal delay Δ for the min-phase channel is $\Delta = 2$, and for the max-phase channel is $\Delta = 0$.

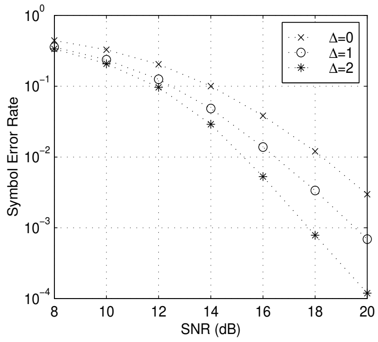

Fig. 12: Taking decisions with delay Δ for the min-phase channel.

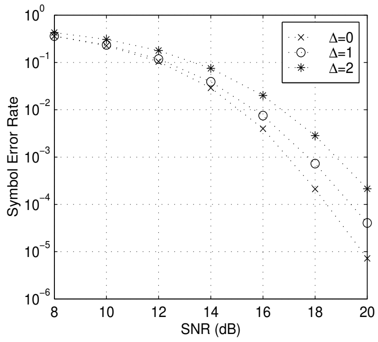

Fig. 13: Taking decisions with delay Δ for the max-phase channel.

This might at first seem counter-intuitive, because it's as if making decisions based on the tap with the lowest value. It is important to keep in mind though which trellis-edge metrics take part in the soft decisions. For the min-phase channel, to make correct decisions with delay $\Delta = 2$ (i.e. for X_1 in Fig. 5) it is sufficient that the highest metrics are associated with edges that have correct starting states: the ending state does not depend upon X_1 and will not affect the decision. The min-phase channel C_1 has a reliable forward recursion, which helps to choose the edges with the correct starting states. With the same token, the max-phase channel C_1 has a reliable backward recursion, which helps to choose the edges with the correct ending states, and therefore the optimal delay is $\Delta = 0$.

Fig. 14 plots the performance when all-pass filtered observations are used for the equalization, as is described in Section IV. Compared to using both trees (which is again plotted for comparison), this procedure offers performance improvement for SER below 10^{-3}. All the plots are for $M_A = M_B = 4$ states.

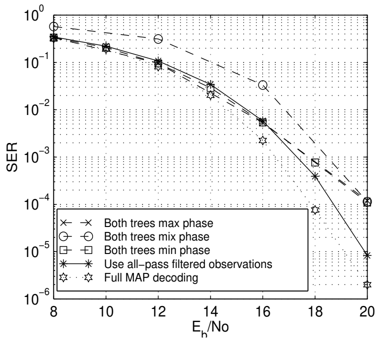

Fig. 14: All-pass filtered performance.

B. Channel C_2.

This subsection presents simulation results for channel C_2, which, unlike the previously examined channel, has almost identical max min and mix-phase responses. Fig. 15 and 16 plot the traditional M algorithm, the effect of using both trees ($\Delta = 0$), and the effect of using both trees and decision based on center tap ($\Delta = 1$), for different values of M.

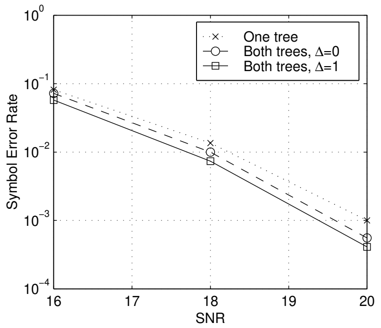

Fig. 15: Using both trees and taking delayed decisions with $M = 3$.

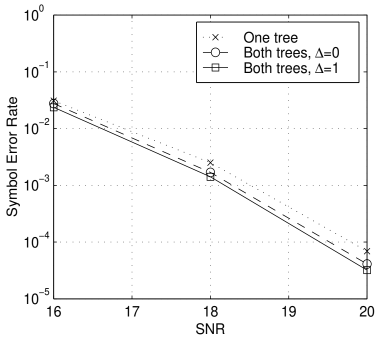

Fig. 16: Using both trees and taking delayed decisions with $M = 4$.

Fig. 17 compares the performance of the M-algorithm over the min-phase channel C_2, with the performance when we use the min-phase channel observations for the forward recursion. the max-phase channel observations for the backward recursion, and the mix-phase channel observations with delay $\Delta = 1$.

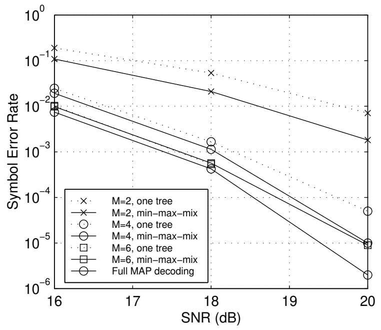

Fig. 17: Performance when using all the channel phases.

VI. Conclusions

The performance of the M-algorithm varies considerably with the distribution of a channel's energy between its taps. This effect is more pronounced for small values of M. In this paper, we proposed an approach, based on the implementation of the forward-backward algorithm in the log domain, that offers more consistent performance over different channels. Several designer choices were discussed and investigated through simulation results, such as the reliability values I_A and I_B, the decision delay, which among the min, max, and mix observation to use for the forward and backward recursion, and the effect of using different number of states M_A and M_B.

References

[1] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Info. Theory*, vol. 20, pp. 284–287, Mar. 1974.

[2] V. Franz and J. Anderson, "Concatenated decoding with a reduced-search BCJR algorithm," *IEEE Journal on selected Areas in Communications*, vol. 16, no. 2 pp. 186–195, Feb. 1998.

[3] A. Duel-Hallen and C. Heegard, "Delayed decision-feedback sequence estimation," *IEEE Transactions on Communications*, vol. 37, no. 5, pp. 436–438, May 1989.

[4] M. Eyuboglu and S. Qureshi, "Reduced-state sequence estimation with set partitioning and decision feedback," *IEEE Transactions on Communications*, vol. 35, no. 4, pp. 13–20, Apr. 1998.

[5] J. Proakis, "Digital Communications," *McGraw-Hill*, Third Edition, 1995.

[6] W. Turin, "Digital Transmission Systems: Performance," *McGraw-Hill*, 1999.

[7] S. Benedetto, D. Divsalar, G. Montorsi and F. Pollara, "A soft-input soft-output APP module for the iterative decoding of concatenated codes," *IEEE Communications Letters*, vol. 1, no. 1, Jan. 1977.

[8] A. Al-Dahir and J. Cioffi, "MMSE decision-feedback equalizers: finite-length results," *IEEE Transactions on Info. Theory*, vol. 41, no. 4, pp. 961–975, July. 1995.

[9] J. Proakis and D.Manolakis, "Digital Signal Processing," *Prentice Hall* , Third Edition, 1996.