Minimum-Bit-Error Rate Tuning for PDNP Detection

Shanwei Shi and John R. Barry

School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA

A dominant impediment in magnetic recording is pattern-dependent media noise, and its impact will only grow more severe as areal densities increase. A widely used strategy for mitigating media noise in a trellis-based detector is pattern-dependent noise prediction (PDNP); in this approach, each bit pattern (which determines a trellis branch) will have its own set of branch metric parameters (including the signal levels, noise predictor coefficients, and residual variances). Because the number of states grows exponentially with the number of tracks being detected, a multitrack detector has far more parameters than a single-track detector. In this article, we propose the adaptive minimum-bit-error rate (AMBER) algorithm for adapting these pattern-dependent multitrack detector parameters with the aim of minimizing BER. Numerical results for a 2-D-PDNP multitrack detector based on a quasi-micromagnetic simulated channel show that, when compared to a conventional MMSE criterion, the AMBER algorithm decreases the BER by 17%.

***Index Terms*— Data-dependent noise prediction (DDNP), minimum-bit-error rate (MBER), multiple-input multiple-output (MIMO), two-dimensional magnetic recording (TDMR).**

I. Introduction

A KEY feature of media noise in magnetic recording is its dependence on the pattern of bits being written [1]. The pattern-dependent noise prediction (PDNP) algorithm [2], [3], widely used as an effective strategy for mitigating pattern-dependent media noise in single-track detection, has recently been extended to the multitrack scenario [4]; it uses 2-D patterns (spanning multiple tracks) to mitigate both downtrack and crosstrack pattern-dependent noise.

Moving from 1-D to 2-D PDNP detection results in an explosion in the number of detector parameters, primarily because the number of patterns grows exponentially in the number of tracks being detected, and further because some of the parameters (like equalizer coefficients and predictor coefficients) become matrix-valued instead of scalars. This paper examines the question of how best to choose the parameters of a PDNP detector.

Traditionally, the PDNP parameters are chosen to minimize some form of a minimum-mean-squared-error (MMSE) criterion, due to its quadratic form which ensures a closed-form solution and a single local minimum. however, the MMSE parameters do not necessarily minimize the bit-error rate (BER), which is the more relevant performance metric from the perspective of the end user interested in maximizing areal density. Furthermore, a closed-form solution for the MMSE parameters requires full knowledge of the noise second-order statistics for each pattern, which can be impractical for a multitrack detector, where the number of patterns is large and may exceed the amount of training data available.

Adaptive strategies aiming to minimize BER instead of MSE have been widely studied in a variety of applications. Yeh and Barry [5] derive the exact MBER full-response equalizer, and propose an adaptive MBER (AMBER) algorithm. An extension to partial-response equalization called the near minimum-BER (NMBER) algorithm was proposed in [6], which yielded a 1.2 dB SNR gain with respect to the least mean squares (LMS) algorithm for an optical storage system. In [7] and [8] a deep neutral network is used to estimate detector parameters that reduce BER and computation time. An AMBER algorithm for tuning the parameters of a single-track PDNP detector was recently proposed in [9].

In this paper, we present for the first time an adaptive algorithm for tuning the parameters of a multitrack PDNP detector (namely the 2-D-PDNP detector of [4]) so as to minimize BER. We test the proposed algorithm on a set of simulated channel waveforms and compare its performance to traditional MMSE parameters.

This paper is organized as follows. In Section II, we introduce the AMBER algorithm in the context of single-track detection, extend it to multitrack detection, and explore its properties. In Section III, we evaluate the performance of the proposed algorithm using simulated waveforms, and in Section IV, we summarize this paper.

II. AMBER Algorithm

We consider the problem of adapting the parameters of a typical magnetic recording read channel, such as those illustrated in Fig. 1. For example, consider the 1-D scenario depicted in Fig. 1(a), where the readback waveform from a single reader is sampled, equalized, and fed to a 1-D trellis-based detector, which ultimately results in a sequence of decisions $\hat{a}_k$ about the written bits a_k. In this case, the parameters to be adapted are the coefficients of the equalizer along with any parameters within the detector. As another example, we consider the multitrack scenario depicted in Fig. 1(b), where multiple readback waveforms are sampled and equalized by a multiple-input multiple-output equalizer before being fed to a trellis-based multitrack detector, which produces decisions $\hat{\mathbf{a}}_k$ about the bits written on the multiple tracks. Throughout this article, we will use $\boldsymbol{\Theta}$ to denote the set

Manuscript received August 3, 2020; revised September 6, 2020; accepted September 18, 2020. Date of publication September 25, 2020; date of current version February 18, 2021. Corresponding author: S. Shi (e-mail: shanweishi@gatech.edu).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMAG.2020.3026979

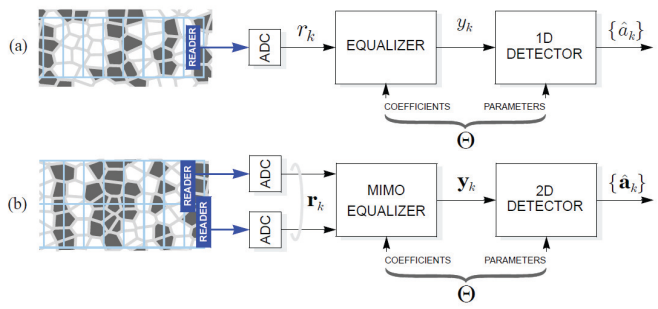

Fig. 1. Illustration of the parameters to be optimized, for two scenarios. (a) For single-track detection using a single reader. (b) For multitrack detection using multiple readers.

of read channel parameters that are to be optimized. In the case of a PDNP detector, $\mathbf{\Theta}$ includes the equalizer coefficients, the pattern-dependent signal levels, the pattern-dependent predictor coefficients, and the pattern-dependent residual variances.

A. AMBER Update Equation

While the aim of the AMBER algorithm proposed in [9] for single-track detection is to minimize BER, it is driven not by a direct measure of BER but instead by a closely related performance metric known as the *path metric margin*. Roughly, the path metric margin at time k measures the gap between the metric of a competing path and that of the correct path. More precisely, assume that the written bits a_0 through a_k are known, as would arise during a training phase. In this case, if θ_k denotes the state at time k of a Viterbi detector [10], this implies that the correct path with state sequence $\{\theta_0, \ldots, \theta_k\}$ leading to the correct state θ_k is known. We define the *competing* path at time k as the "best of the rest" of the paths that lead to state θ_k at time k; in particular, when the Viterbi algorithm aims to find the path with minimum metric, the competing path at time k is the partial path that leads to the correct state at time k, excluding the correct path, with minimal metric. The competing path can be traced back from θ_k until it merges with the correct path, defining the competing path sequence $\{\hat{\theta}_{k-\ell_k}, \ldots, \hat{\theta}_k\}$, where ℓ_k denotes the length of the separation between the correct and competing paths. Because the competing path starts and ends on the correct path, we have $\hat{\theta}_{k-\ell_k} = \theta_{k-\ell_k}$ and $\hat{\theta}_k = \theta_k$. The path metric margin is then simply the difference between the competing and correct path metrics

$$M_k = \sum_{i=0}^{\ell_k-1} \gamma\left(\hat{\theta}_{k-i}, \hat{\theta}_{k-i-1}; \mathbf{\Theta}\right) - \sum_{i=0}^{\ell_k-1} \gamma\left(\theta_{k-i}, \theta_{k-i-1}; \mathbf{\Theta}\right) \tag{1}$$

where $\gamma(\theta_k, \theta_{k-1})$ is the branch metric from state θ_{k-1} to θ_k.

A large margin implies that the correct path is easily distinguishable from the incorrect path, while a small positive margin implies that the correct path is barely preferred over the incorrect path. A negative margin (when $M_k < 0$) implies that a Viterbi detector that ignores the training information would make bit errors, either at or near time k. Note that one bit error can result in multiple negative margins. Based on this observation one might be tempted to choose the parameters $\mathbf{\Theta}$ to minimize the probability that M_k is negative, or (in terms of the unit-step function) so as to minimize the following cost function:

$$J(\mathbf{\Theta}) = E(u(-M_k)) \tag{2}$$

where $E(\cdot)$ is the expectation and $u(\ \cdot\ )$ is the unit step function. However, this function is not differentiable and is difficult to minimize directly. Therefore, we instead propose to choose $\mathbf{\Theta}$ so as to minimize the following cost function:

$$J_\tau(\mathbf{\Theta}) = E((\tau - M_k)u(\tau - M_k)) \tag{3}$$

where τ is a small positive threshold. Not only is $J_\tau(\mathbf{\Theta})$ differentiable, but under certain circumstances (explained in the next section) minimizing $J_\tau(\mathbf{\Theta})$ is equivalent to minimizing $J(\mathbf{\Theta})$.

Applying the stochastic gradient algorithm to $J_\tau(\mathbf{\Theta})$ leads to the AMBER algorithm for adapting the parameters $\mathbf{\Theta}$

$$\mathbf{\Theta}_{k+1} = \mathbf{\Theta}_k + \lambda u(\tau - M_k)\nabla_{\mathbf{\Theta}} M_k \tag{4}$$

where λ is the step size. The unit step factor in (4) acts as an indicator function: it ensures that the parameters only update when $M_k < \tau$. In other words, the parameters update only when the margin is dangerously small; otherwise, the parameters do not change. This feature is in stark contrast to MMSE algorithms like LMS and recursive least squares (RLS), which would continually update the parameters.

The AMBER algorithm of (4) is general and can be applied to a wide range of detector architectures, in any specific application one must first find an expression for the path metric margin in terms of the detector parameters, so that the gradient in (4) can be computed explicitly. We close this section with three specific examples.

1) AMBER for Viterbi Without PDNP: We first consider a 2^μ-state Viterbi detector without PDNP, where μ is the channel memory. In this case, the only parameters to optimize are the equalizer coefficients and the *signal levels* associated with each state transition. The signal levels can be viewed as the entries of a look-up table, one signal level for each state transition. Each state transition can be represented by a vector or *pattern* $\mathbf{a}$ of $\mu + 1$ bits (containing the current input bit as well as the μ previous input bits). We use the notation $s(\mathbf{a})$ to denote the signal level associated with the bit pattern $\mathbf{a}$. In this case the path metric margin of (1) reduces to

$$\begin{aligned} M_k &= \sum_{i=0}^{\ell_k-1} (y_{k-i} - s(\hat{\mathbf{a}}_{k-i}))^2 - (y_{k-i} - s(\mathbf{a}_{k-i}))^2 \\ &= \sum_{i=0}^{\ell_k-1} 2\mathbf{c}^T \mathbf{r}_{k-i}(s(\mathbf{a}_{k-i}) - s(\hat{\mathbf{a}}_{k-i})) + s^2(\hat{\mathbf{a}}_{k-i}) \\ &\quad - s^2(\mathbf{a}_{k-i}) \end{aligned} \tag{5}$$

where $\mathbf{a}_k = [a_k, \ldots, a_{k-\mu}]^T$ is a vector of bits for the correct path, where $\hat{\mathbf{a}}_k$ are the corresponding bits for the competing path, and where $\mathbf{c}$ is the vector of equalizer coefficients and $\mathbf{r}_k$ is the vector of relevant waveform samples at time k. Differentiating M_k with respect to $\mathbf{c}$ and $\mathbf{s}$ and substituting into (4) leads to explicit AMBER update equations for $\mathbf{c}$ and $\mathbf{s}$.

2) AMBER for 1-D PDNP: As a second example, consider the AMBER algorithm applied to a single-track PDNP detector, so that the parameters $\mathbf{\Theta}$ to be optimized are the equalizer $\mathbf{c}$, the signal levels $s(\mathbf{a})$, the residual prediction-error variances $v(\mathbf{a})$, and the $N_p \times 1$ predictor coefficient vectors $\mathbf{p}(\mathbf{a})$ associated with each bit pattern $\mathbf{a}$. In this case, the path metric margin of (1) reduces to

$$M_k = \sum_{i=0}^{l_k-1} \Big\{ \ln v(\hat{\mathbf{a}}_{k-i}) + \frac{1}{v(\hat{\mathbf{a}}_{k-i})} \big([1, \ -\mathbf{p}^T(\hat{\mathbf{a}}_{k-i})]\big(\mathbf{y}_{k-i}^{k-i-N_p} - \mathbf{s}\big(\hat{\mathbf{a}}_{k-i}^{k-i-N_p}\big)\big)\big)^2 - \ln v(\mathbf{a}_{k-i}) - \frac{1}{v(\mathbf{a}_{k-i})} \times \big([1, -\mathbf{p}^T(\mathbf{a}_{k-i})]\big(\mathbf{y}_{k-i}^{k-i-N_p} - \mathbf{s}\big(\mathbf{a}_{k-i}^{k-i-N_p}\big)\big)\big)^2 \Big\}. \tag{6}$$

Differentiating and substituting into (4) leads to explicit update equations for the detector parameters.

3) AMBER for 2-D-PDNP: Here, we show how the AMBER algorithm can be applied to a multitrack detector that uses 2-D-PDNP [4], where two readers spanning a pair of neighboring tracks are used to jointly detect the bits on the two tracks. Synchronous samples of the two readback waveforms are filtered by a two-input two-output equalizer with N_c coefficients (each a 2×2 matrix), represented by a $2 \times 2N_c$ matrix $\mathbf{C}$, resulting in the vector output y_k at time k. The equalizer outputs are then passed to a 2-D-PDNP multitrack detector [4]. The 2-D bit pattern $\mathbf{A}$ is a matrix of bits with two rows, one for each track. Associated with each 2-D bit pattern is a signal level vector $\mathbf{s}$, a standard deviation diagonal matrix $\mathbf{\Lambda}$, and a set of matrix-valued predictor coefficients $\{\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_{N_p-1}\}$. The number of parameters is $2^b(5+4(N_p-1))+4N_c$ (for each of the 2^b patterns there are two signal levels, two residual variances, and $4(N_p-1)+1$ predictor coefficients), a number which can easily reach into the hundreds, depending on the number of bits b in each 2-D pattern, the number N_p of matrix-valued predictor coefficients, and the number N_c of matrix-valued equalizer coefficients.

The branch metric for an edge e in the 2-D-PDNP Viterbi detector is [4]

$$\gamma_k(e) = \log\big(\sigma_1^2(\mathbf{A}_k)\sigma_2^2(\mathbf{A}_k)\big) + \left\| \mathbf{\Lambda}^{-1}(\mathbf{A}_k) \sum_{i=0}^{N_p} \mathbf{B}_i(\mathbf{A}_k)(\mathbf{y}_{k-i} - \mathbf{s}(\mathbf{A}_{k-i})) \right\|^2 \tag{7}$$

where $\mathbf{A}_k$ is that pattern at time k associated with e, where σ_1 and σ_2 are the diagonal components of $\mathbf{\Lambda}$, and $\mathbf{B}_i = \delta_i \mathbf{I} - \mathbf{P}_i$ are the coefficients of a prediction-error filter, defined in terms of the unit impulse δ_i (satisfying $\delta_0 = 1$, $\delta_{i\neq 0} = 0$).

By plugging (7) into (1) and applying the AMBER algorithm, we arrive at the update equations for $\mathbf{C}$, $\mathbf{\Lambda}$, $\mathbf{s}$, and $\mathbf{P}_i$.

B. Exponential Assumption and Threshold Optimization

The threshold parameter τ of the AMBER algorithm is not arbitrary. Instead, it must be chosen carefully to ensure good performance and avoid trivial solutions with bad performance. In this section, explore the role of τ and propose a strategy for its optimization.

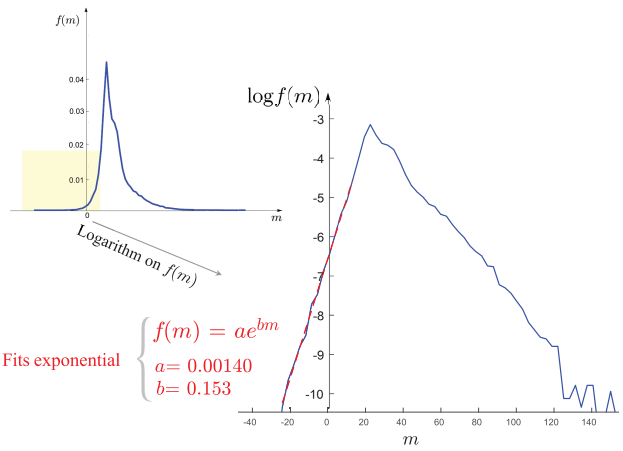

Fig. 2. Margin pdf and its tail fit to an exponential function.

The basis for our analysis is the observation that the *tail* of the probability density function (pdf) for the path margin M_k often appears to have an exponential shape. For example, consider the upper left inset of Fig. 2, which shows an experimentally measured pdf for the margin M_k in a single-track PDNP Viterbi detector operating on a quasi-micromagnetic simulated channel with a track pitch of 22.1 nm and a 70% centered reader (see Section III for clarification). The bottom of Fig. 2 shows a close fit between the tail of the pdf and an exponential distribution (the red dashed curve).

Because we observed similar good fits to an exponential distribution over a wide range of channel conditions and detector parameters, we were encouraged to adopt the exponential model for the pdf tail described below, to facilitate analysis. It should be noted that the tail is not strictly speaking exponentially distributed, and that ultimately the value of the AMBER algorithm rests not on the exponential assumption that facilitates some of its analysis but on the good experimental results of the algorithm itself (see Section III).

When the *tail* of the pdf for M_k follows an exponential distribution:

$$f(m) = a(\mathbf{\Theta})e^{b(\mathbf{\Theta})m}, \quad m < \tau \tag{8}$$

where a and b are parameters that depend on $\mathbf{\Theta}$, then the cost functions (2) and (3) reduce to

$$J(\mathbf{\Theta}) = \frac{a(\mathbf{\Theta})}{b(\mathbf{\Theta})}, \tag{9}$$

$$J_\tau(\mathbf{\Theta}) = \frac{a(\mathbf{\Theta})}{b^2(\mathbf{\Theta})} e^{b(\mathbf{\Theta})\tau}. \tag{10}$$

Straightforward differentiation of (9) and (10) results in

$$\nabla_{\mathbf{\Theta}} J(\mathbf{\Theta}) = \frac{1}{b^2(\mathbf{\Theta})}(\nabla a(\mathbf{\Theta})b(\mathbf{\Theta}) - a(\mathbf{\Theta})\nabla b(\mathbf{\Theta})) \tag{11}$$

$$\nabla_{\mathbf{\Theta}} J_\tau(\mathbf{\Theta}) = \frac{e^{b(\mathbf{\Theta})\tau}}{b^3(\mathbf{\Theta})}(\nabla a(\mathbf{\Theta})b(\mathbf{\Theta}) - (2 - b(\mathbf{\Theta})\tau) \times a(\mathbf{\Theta})\nabla b(\mathbf{\Theta})). \tag{12}$$

Let $\mathbf{\Theta}^*$ denote the set of parameters that minimizes $J(\mathbf{\Theta})$, so that substituting $\mathbf{\Theta}^*$ into (11) yields $\nabla_{\mathbf{\Theta}} J(\mathbf{\Theta}^*) = \mathbf{0}$. Let $\tau* = 1/b(\mathbf{\Theta}^*)$. Substituting $\tau = \tau*$ into (12) reveals that

Algorithm 1 Iterative Algorithm to Ensure Optimal τ

Input: Initial threshold τ_0, stop criterion ϵ
Output: Optimal τ and $\mathbf{\Theta}$.

$i = 0$
repeat
  Run AMBER to get $\mathbf{\Theta}(\tau_i)$ that minimizes $J_{\tau_i}(\mathbf{\Theta})$
  Fit steady-state margin tail pdf (for $m < \tau$) to an exponential distribution, and estimate its $b(\mathbf{\Theta}(\tau_i))$
  Set $\tau_{i+1} = \frac{1}{b(\mathbf{\Theta}(\tau_i))}$
  $i = i + 1$
until $|\tau_i - \tau_{i-1}| < \epsilon$

the same $\mathbf{\Theta}^*$ that minimizes $J(\mathbf{\Theta})$ also minimizes $J_{\tau*}(\mathbf{\Theta})$. The implication of this observation is that, when the margin tail is exponential, and when the AMBER threshold is chosen carefully (according to $\tau = 1/b(\mathbf{\Theta}^*)$), the cost function $J(\mathbf{\Theta})$ can be minimized by the AMBER algorithm.

As stated there is a circular flaw: the optimal value for τ depends on the optimal value $\mathbf{\Theta}^*$ for the parameter set. Clearly, there would be no need for AMBER or its threshold if $\mathbf{\Theta}^*$ were already known. To help break this cycle, we make use of the following fixed-point relationship.

Corollary 1: Let $\mathbf{\Theta}(\tau)$ denote the parameter set that minimizes $J_\tau(\mathbf{\Theta})$. If τ satisfies the fixed-point relationship $\tau = (1/(b(\mathbf{\Theta}(\tau))))$, then $\tau = \tau*$ and $\mathbf{\Theta}(\tau) = \mathbf{\Theta}^*$.

Inspired by this corollary, we propose a fixed-point iterative strategy for automatically finding the best threshold. Starting with an arbitrary threshold, we run the AMBER algorithm until it converges, fit the margin tail that results to an exponential shape with parameter b, and then set the new threshold to $1/b$. This process is repeated until the threshold converges. The pseudocode of the proposed iterative algorithm is shown in Algorithm 1.

As an illustration of how Algorithm 1 works, consider the example of a conventional ten-coefficient equalizer followed by a single-track two-state Viterbi detector without PDNP, so that the parameter set $\mathbf{\Theta}$ consists of ten equalizer coefficients and four signal levels. To ensure that the optimal $\mathbf{\Theta}$ is unique, we constrain both the equalizer energy and the signal level energy. The input to the detector is the simulated channel described in [11], based on a 70% centered reader and a track pitch of 24.1 nm.

We sweep the threshold τ from 0 to 3, and for each value we use the AMBER algorithm to find the $\mathbf{\Theta}(\tau)$ that minimizes $J_\tau(\mathbf{\Theta})$. We then estimate the path metric margin pdf and fit its tail (for $M_k < \tau$) to an exponential shape, resulting in a b parameter we denote $b(\mathbf{\Theta}(\tau))$. In Fig. 3, we plot $1/b(\mathbf{\Theta}(\tau))$ versus τ in red (right-hand scale). The goal of the iterative algorithm is to find the value of τ where the fixed-point relationship is satisfied, namely where the dashed line (representing τ) intersects the red curve [representing $1/b(\mathbf{\Theta}(\tau))$]. Starting with an arbitrary initial value of $\tau_0 = 2$, Algorithm 1 produces $\tau_1 = 1.1$, $\tau_2 = 0.75$, and $\tau_3 = 0.75$, converging quickly after only three iterations. The blue trajectory graphically illustrates how the algorithm bounces between the red and dashed curves until it converges to their intersection. Finally, overlaid on the same graph, we also plot $J(\mathbf{\Theta}(\tau))$ from (2) as a function of τ in black (left scale). Observe that the value of τ that minimizes this cost function coincides with the value of τ that satisfies the fixed-point relationship.

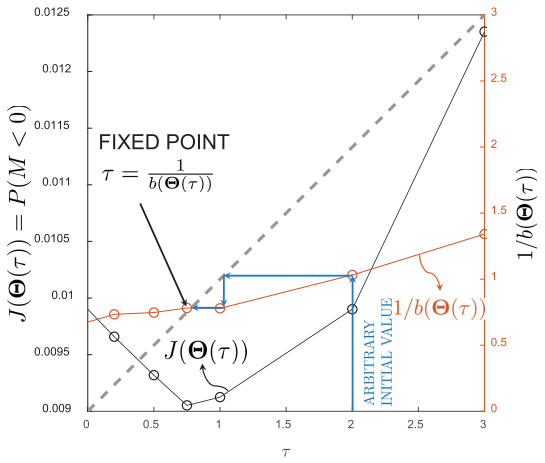

Fig. 3. Relationship between τ and $1/b(\mathbf{\Theta}(\tau))$. The simulation is run in training mode, with a 70% centered reader over a 24.1 nm track pitch.

C. Hamming-Weighted Step Size

As written, when the margin tail is exponential and the threshold τ is optimized, the AMBER algorithm of (4) minimizes $J(\mathbf{\Theta}) = P(M_k < 0)$. If each instance of the error event $M_k < 0$ led to a single bit error, then this would be equivalent to minimizing BER. However, because some error events cause more bit errors than others, the AMBER algorithm as written does not minimize BER. We can improve the BER performance of the AMBER algorithm by a simple modification of the step size λ in (4), namely by introducing an adaptive Hamming-weighted step size $\lambda = \lambda_0 w_H$, where λ_0 is a fixed nominal step size, and w_H is shorthand for the number of message bits that differ between the correct subpath $\{\theta_{k-\ell_k}, \ldots, \theta_k\}$ and the competing subpath $\{\hat{\theta}_{k-\ell_k}, \ldots, \hat{\theta}_k\}$. The Hamming-weight factor ensures that a large burst of bit errors will cause a bigger change in parameters than a small burst of errors.

To illustrate the benefit of the Hamming weight factor, consider again the example from the previous section (a ten-coefficient equalizer followed by a two-state Viterbi detector without PDNP, based on a 70% centered reader and a track pitch of 24.1 nm from the simulated channel in [11]). Let us fix the equalizer to be MMSE, and further constrain the signal levels to be symmetric: either $\pm s_1$ when two consecutive bits are the same, or $\pm s_2$ when two consecutive bits are different. There are thus only two parameters to optimize: $\mathbf{\Theta} = \{s_1, s_2\}$.

In Fig. 4, we show a contour plot of BER as a function of the parameters s_1 and s_2. Overlaid on the same graph are a pair of trajectories for the AMBER algorithm, both starting from the same arbitrary initial condition $(-1.1,\ 0.9)$. The red curve has no Hamming-weight factor, while the blue curve includes the Hamming-weight factor. In both cases, the initial λ_0 is 10^{-4}, and it decays with a half-life of 25. After convergence

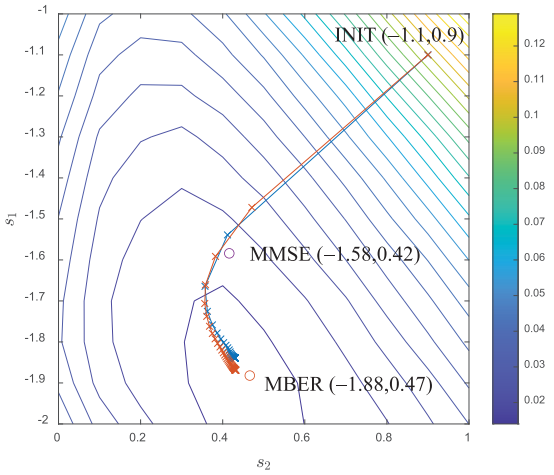

Fig. 4. BER contour with trajectories of AMBER parameters.

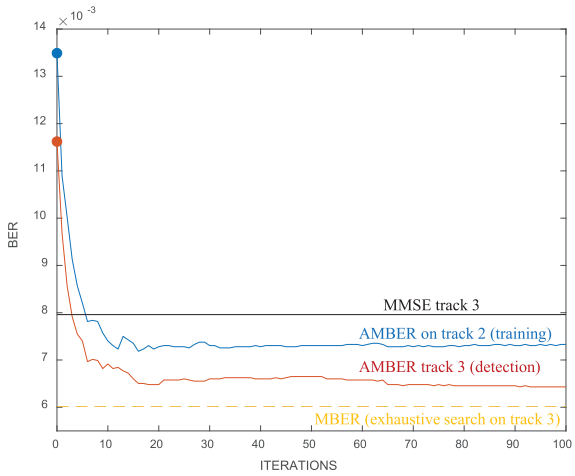

Fig. 5. Learning curves for the AMBER PDNP algorithm.

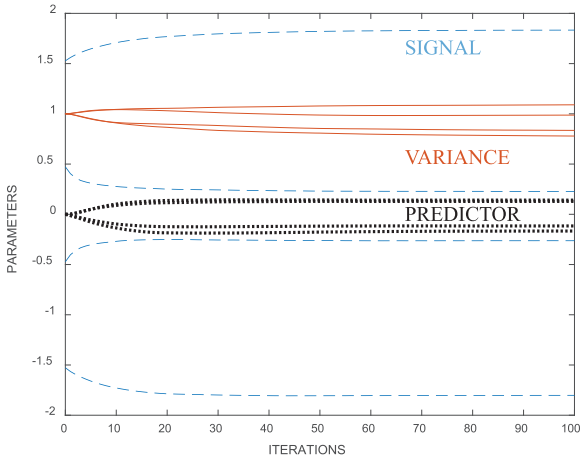

Fig. 6. Parameter evolution versus iterations for AMBER PDNP.

the BER with the Hamming factor is 0.0130, while the BER without the Hamming factor is 0.0133. Also shown in the figure is the MBER point, found by exhaustive search, which achieves BER = 0.0120, along with the MMSE point, which achieves BER = 0.0164. This example illustrates not only the benefit of the Hamming weight factor in the step size, but also the general suboptimality of MMSE with respect to BER, and further the effectiveness of the AMBER algorithm to seek out the minimum of the BER surface.

III. Quantitative Results

We tested our algorithm on simulated waveforms provided by Ehime University, which were produced by realistic head fields and a Voronoi medium with Stoner–Wohlfarth switching [11]. Five consecutive tracks were written in a shingled fashion. In each track, there are 40 950 independent pseudorandom bits sequence and 128 bits preamble and postamble, respectively. A total of 900 readback waveforms were produced with a fixed bit length of 7.3 nm, track pitches from 16.1 to 26.1 nm in 2 nm increments, reader widths from 70% to 145% in 15% increments (relative to a nominal reader width, whose full-width at half-maximum (FWHM) of the reader sensitivity function is 20.8 nm), and 25 reader positions in one-eighth of a track pitch increments, spanning from the second to the fourth tracks. The readback waveforms from different tracks were perfectly synchronized and no modulation coding or error-control coding was applied. No electronic noise was added.

A. Performance of AMBER PDNP Detection

Unlike the genie-aided tests performed in [9], we use the second track for training parameters, and the third (middle) track to detect in the following simulation. We now present numerical results for the AMBER algorithm presented in Section II, which applies to the case of a fixed MMSE equalizer followed by a PDNP Viterbi detector. The parameters to be adapted are the parameters of the PDNP detector, namely the pattern-dependent signal levels, noise variances, and predictor coefficients. We consider two-bit patterns, so that the number of patterns is four, and a single predictor coefficient for each pattern. The total number of parameters being adapted is thus 12: four signal levels, four variances, and four predictor coefficients.

To start with, we investigate the convergence properties by plotting BER versus iterations in Fig. 5. The AMBER algorithm was tested on waveforms of track pitch 24.1 nm with a 70% centered reader. A decaying step size 10^{-4} with a half-life of 100 iterations was employed, and the parameters are initialized to those of a non-PDNP Viterbi detector. We trained the parameters on track 2 by AMBER algorithm and plot the training BER curve plotted in blue versus iterations. The red curve is the testing BER achieved by detecting track 3 with the training parameters after each iteration. The figure shows that the AMBER algorithm converges fast and the BER gets close to the MBER line found via an exhaustive search on track 3, and becomes stable after 60 iterations, achieving a 20% reduction in BER compared to MMSE line plotted in black. We also plot the parameter evolution in Fig. 6. The signal levels (blue dashed curves), variances (red solid curves) and predictor coefficients (black dotted curves) converge fast.

B. Performance of AMBER 2-D-PDNP Detection

In this simulation, we consider a scenario where two readers detect two tracks. We use tracks 2 and 3 to train the

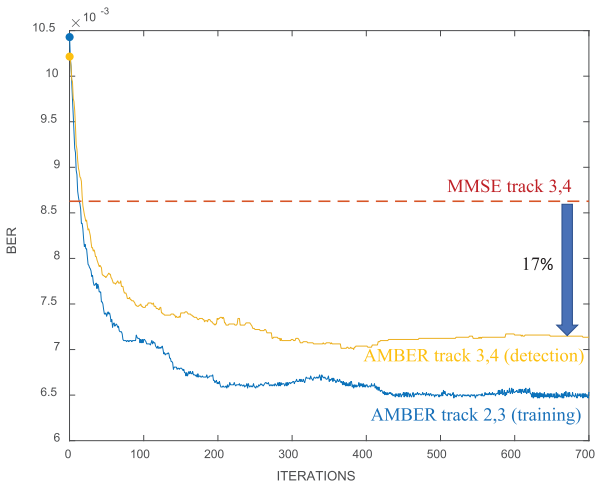

Fig. 7. Learning curve for the AMBER 2-D-PDNP algorithm.

parameters, and we use the resulting parameters to detect the bits written on track 3 and 4, with 11 matrix-valued equalizer coefficients. The detector has 16 2-D patterns, and there are only two prediction matrices $\mathbf{P}_0$ and $\mathbf{P}_1$ ($N_p = 1$), so that the number of states with 2-D-PDNP is 16. The total number of parameters is 188. We do not consider longer pattern lengths because there are not sufficient training data for each pattern.

We investigate the convergence properties of the AMBER algorithm by plotting BER (averaged over the two tracks) versus iterations. In this experiment, there are two 130% readers one-eighth track offset inside the two tracks of interest, and the track pitch is 26.1 nm. The step size is initialized to 10^{-5} and decreases exponentially with a half-life of 300 iterations, and the threshold is $\tau = 1$. Fig. 7 shows the training curve for the AMBER algorithm (in blue), where we see its convergence after about 450 iterations. The detection curve converges 50 iterations faster. Compared with 2-D-MMSE parameters, 2-D-AMBER parameters achieve a 17% decrease in BER.

IV. Conclusion

In this article, we proposed the AMBER algorithm for adapting the parameters of a multitrack detector known as the 2-D-PDNP detector. We proposed a fixed-point iterative strategy for optimizing the AMBER threshold. Simulated waveforms demonstrate that the AMBER 2-D-PDNP parameters measurably outperform conventional MMSE parameters. We expect further gains in performance by increasing the length of the training sequences, pattern lengths, and the number of prediction coefficients.

Acknowledgment

This work was supported by IDEMA ASRC. The authors would like to thank Yasuaki Nakamura and Yoshihiro Okamoto from Ehime University, Japan, for providing the simulated waveforms, and also W. Radich, and F. Erden for their guidance and support.

References

[1] B. Vasić and E. M. Kurtas, *Coding and Signal Processing for Magnetic Recording Systems*. Boca Raton, FL, USA: CRC Press, 2004.
[2] A. Kavcic and J. M. F. Moura, "The Viterbi algorithm and Markov noise memory," *IEEE Trans. Inf. Theory*, vol. 46, no. 1, pp. 291–301, Jan. 2000.
[3] J. Moon and J. Park, "Pattern-dependent noise prediction in signal-dependent noise," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 4, pp. 730–743, Apr. 2001.
[4] S. Shi and J. R. Barry, "Multitrack detection with 2D pattern-dependent noise prediction," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.
[5] C.-C. Yeh and J. R. Barry, "Approximate minimum bit-error rate equalization for binary signaling," in *Proc. ICC-Int. Conf. Commun.*, vol. 2, Jun. 1997, pp. 1095–1099.
[6] J. Riani, S. van Beneden, J. W. M. Bergmans, and A. H. J. Immink, "Near-minimum bit-error rate equalizer adaptation for PRML systems," *IEEE Trans. Commun.*, vol. 55, no. 12, pp. 2316–2327, Dec. 2007.
[7] A. Sayyafan, B. J. Belzer, K. Sivakumar, J. Shen, K. S. Chan, and A. James, "Deep neural network based media noise predictors for use in high-density magnetic recording turbo-detectors," *IEEE Trans. Magn.*, vol. 55, no. 12, pp. 1–6, Dec. 2019.
[8] J. Shen, A. Aboutaleb, K. Sivakumar, B. J. Belzer, K. S. Chan, and A. James, "Deep neural network *a posteriori* probability detector for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 56, no. 6, pp. 1–12, Jun. 2020.
[9] S. Shi and J. R. Barry, "Adaptive minimum-bit-error rate PDNP detection for magnetic recording," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.
[10] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.
[11] J. R. Barry *et al.*, "Optimization of bit geometry and multi-reader geometry for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 52, no. 2, pp. 1–7, Feb. 2016.