A Deterministic Analysis of an Online Convex Mixture of Experts Algorithm

Huseyin Ozkan, Mehmet A. Donmez, Sait Tunc, and Suleyman S. Kozat

***Abstract*— We analyze an online learning algorithm that adaptively combines outputs of two constituent algorithms (or the experts) running in parallel to estimate an unknown desired signal. This online learning algorithm is shown to achieve and in some cases outperform the mean-square error (MSE) performance of the best constituent algorithm in the steady state. However, the MSE analysis of this algorithm in the literature uses approximations and relies on statistical models on the underlying signals. Hence, such an analysis may not be useful or valid for signals generated by various real-life systems that show high degrees of nonstationarity, limit cycles and that are even chaotic in many cases. In this brief, we produce results in an individual sequence manner. In particular, we relate the time-accumulated squared estimation error of this online algorithm at any time over any interval to the one of the optimal convex mixture of the constituent algorithms directly tuned to the underlying signal in a deterministic sense without any statistical assumptions. In this sense, our analysis provides the transient, steady-state, and tracking behavior of this algorithm in a strong sense without any approximations in the derivations or statistical assumptions on the underlying signals such that our results are guaranteed to hold. We illustrate the introduced results through examples.**

***Index Terms*— Convexly constrained, deterministic, learning algorithms, mixture of experts, steady-state, tracking, transient.**

I. Introduction

The problem of estimating or learning an unknown desired signal is heavily investigated in online learning [1]–[7] and adaptive signal processing literature [8]–[13]. However, in various applications, certain difficulties arise in the estimation process due to the lack of structural and statistical information about the data model. To resolve this issue, mixture approaches that adaptively combine outputs of multiple constituent algorithms performing the same task are proposed in the online learning literature under the mixture of experts framework [5]–[7] and adaptive signal processing literature under the adaptive mixture methods framework [8], [9]. Along these lines, an online convexly constrained mixture of experts method that combines outputs of two learning algorithms is introduced in [8]. We point out that the mixture of experts framework refers to a different model in another context [14], where the input space is divided into regions in a nested fashion to each of which an expert corresponds. The partitioning of the input space and the corresponding experts are learned jointly and combined such that a mixture of experts method is obtained. On the other hand, the mixture method in [8] adaptively combines the outputs of the constituent algorithms that run in parallel on the same task under a convex constraint to minimize the final mean-square error (MSE). This adaptive mixture is shown to be universal with respect to the input algorithms in a certain stochastic sense such that this mixture achieves and in some cases outperforms the MSE performance of the best constituent algorithm in the steady state [8]. However, the MSE analysis of this adaptive mixture in the transient and steady states uses approximations, such as the separation assumptions, and relies on strict statistical models on the signals, e.g., stationary data models [8], [9]. In this brief, we study this algorithm [8] from the perspective of online learning and produce results in an individual sequence manner such that our results are guaranteed to hold for any bounded arbitrary signal.

Nevertheless, signals produced by various real-life systems, such as in underwater acoustic communication applications, show high degrees of nonstationarity, limit cycles and, in many cases, are even chaotic so that they hardly fit to assumed statistical models [15]. Hence, an analysis based on certain statistical assumptions or approximations may not be useful or adequate under these conditions. To this end, we refrain from making any statistical assumptions on the underlying signals and present an analysis that is guaranteed to hold for any bounded arbitrary signal without any approximations. In particular, we relate the performance of this learning algorithm that adaptively combines outputs of two constituent algorithms to the performance of the optimal convex combination that is directly tuned to the underlying signal in a deterministic sense. Naturally, this optimal convex combination can only be chosen in hindsight after observing the whole signal *a priori* (before we even start processing the data). Since we compare the performance of this algorithm with respect to the best convex combination of the constituent filters in a deterministic sense over any time interval, our analysis provides, without any assumptions, the transient, the tracking, and the steady-state behaviors together [5]–[7]. In particular, if the analysis window starts from $t = 1$, then we obtain the transient behavior; if the window length goes to infinity, then we obtain the steady-state behavior; and finally, if the analyze window is selected arbitrary, then we get the tracking behavior as explained in detail in Section III. The corresponding bounds may also hold for unbounded signals such as with Gaussian and Laplacian distributions, if one can define reasonable bounds such that a sample stays in the defined interval with high probability.

After we provide a brief problem description in Section II, we present a deterministic analysis of the convexly constrained mixture algorithm in Section III, where the performance bounds are given as a theorem and a corresponding corollary. We illustrate the introduced results through examples in Section IV. This brief concludes with certain remarks in Section V.

II. Problem Description

In this framework, we have a desired signal $\{y_t\}_{t\geq 1} \subset \mathbb{R}$, where t is the time index, and two constituent algorithms running

Manuscript received July 22, 2012; revised March 25, 2014; accepted July 31, 2014. Date of publication August 28, 2014; date of current version June 16, 2015. This work was supported in part by IBM Faculty Award and in part by the Outstanding Young Scientist Award Program, Turkish Academy of Sciences, Ankara, Turkey.

H. Ozkan is with the Department of Electrical and Electronics Engineering, Bilkent University, Ankara 06800, Turkey, and also with the MGEO Division, Image Processing Department, Aselsan Inc., Ankara 06370, Turkey (e-mail: huseyin@ee.bilkent.edu.tr).

M. A. Donmez is with the Coordinated Science Laboratory, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Champaign, IL 61801 USA (e-mail: donmez2@illinois.edu).

S. Tunc is with the Competitive Signal Processing Laboratory, Koc University, Istanbul 34450, Turkey (e-mail: saittunc@ku.edu.tr).

S. S. Kozat is with the Department of Electrical and Electronics Engineering, Bilkent University, Ankara 06800, Turkey (e-mail: kozat@ee.bilkent.edu.tr).

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

Digital Object Identifier 10.1109/TNNLS.2014.2346832

TABLE I
LEARNING ALGORITHM THAT ADAPTIVELY COMBINES OUTPUTS OF TWO ALGORITHMS

```
The Convexly Constrained Algorithm:
  Parameters:
    $\mu > 0$: learning rate.
    $\lambda^+ \in [0, \frac{1}{2}]$: clipping parameter.
  Inputs:
    $y_t$: desired signal.
    $\hat{y}_{1,t}, \hat{y}_{2,t}$: constituent learning algorithms.
  Outputs:
    $\hat{y}_t$: estimate of the desired signal.
  Initialization: Set the initial weights $\lambda_1 = 1/2$ and $\rho_1 = 0$.
  for $t = 1 : \ldots : n$,
    % receive the constituent algorithm outputs $\hat{y}_{1,t}$ and $\hat{y}_{2,t}$ and
    % estimate the desired signal
    $\hat{y}_t = \lambda_t \hat{y}_{1,t} + (1 - \lambda_t)\hat{y}_{2,t}$
    % Upon receiving $y_t$, update the weight according to the rule:
    $\rho_{t+1} = \rho_t + \mu e_t \lambda_t (1 - \lambda_t)[\hat{y}_{1,t} - \hat{y}_{2,t}]$
    $\lambda_{t+1} = \frac{1}{1+e^{-\rho_{t+1}}}$
    $\lambda_{t+1} \leftarrow \lambda^+$ and $\rho_{t+1} = \ln \frac{\lambda_{t+1}}{1-\lambda_{t+1}}$, if $\lambda_{t+1} < \lambda^+$
    $\lambda_{t+1} \leftarrow 1 - \lambda^+$ and $\rho_{t+1} = \ln \frac{\lambda_{t+1}}{1-\lambda_{t+1}}$, if $\lambda_{t+1} > 1 - \lambda^+$
  endfor
```

in parallel producing $\{\hat{y}_{1,t}\}_{t\geq 1}$ and $\{\hat{y}_{2,t}\}_{t\geq 1}$, respectively, as the estimations (or predictions) of the desired signal. We assume that the desired signal $\{y_t\}_{t\geq 1}$ is finite and bounded by a known constant Y, i.e., $|y_t| \leq Y < \infty$. Here, we have no restrictions on $\hat{y}_{1,t}$ or $\hat{y}_{2,t}$, e.g., these outputs are not required to be causal, however, without loss of generality, we assume $|\hat{y}_{1,t}| \leq Y$ and $|\hat{y}_{2,t}| \leq Y$, i.e., these outputs can be clipped to the range $[-Y, Y]$ without sacrificing performance under the squared error. As an example, the desired signal and outputs of the constituent learning algorithms can be single realizations generated under the framework of [8]. At each time t, the convexly constrained algorithm receives an input vector $\boldsymbol{x}_t \overset{\triangle}{=} [\hat{y}_{1,t}\ \hat{y}_{2,t}]^T$ and outputs

$$\hat{y}_t = \lambda_t \hat{y}_{1,t} + (1 - \lambda_t)\hat{y}_{2,t} = \boldsymbol{w}_t^T \boldsymbol{x}_t$$

where $\boldsymbol{w}_t \overset{\triangle}{=} [\lambda_t\ (1 - \lambda_t)]^T$, $0 \leq \lambda_t \leq 1$, as the final estimate. The final estimation error is given by $e_t = y_t - \hat{y}_t$. The combination weight λ_t is trained through an auxiliary variable ρ_t using a stochastic gradient update to minimize the squared final estimation error as

$$\lambda_t = \frac{1}{1 + e^{-\rho_t}} \tag{1}$$

$$\rho_{t+1} = \rho_t - \mu \nabla_{\rho_t} e_t^2 = \rho_t + \mu e_t \lambda_t (1 - \lambda_t)[\hat{y}_{1,t} - \hat{y}_{2,t}] \tag{2}$$

where $\mu > 0$ is the learning rate. The combination parameter λ_t in (1) is constrained to lie in $[\lambda^+, (1 - \lambda^+)]$, $0 < \lambda^+ < 1/2$ in [8], since the update in (2) may slow down when λ_t is too close to the boundaries. We follow the same restriction and analyze (2) under this constraint. The algorithm, presented in Table I, consists of two steps: 1) the update step of the parameter ρ: $\rho_{t+1} = \rho_t + \mu e_t \lambda_t (1 - \lambda_t)[\hat{y}_{1,t} - \hat{y}_{2,t}]$ and 2) the mapping of ρ back to the corresponding combination parameter λ: $\lambda_{t+1} = 1/1 + e^{-\rho_{t+1}}$. At every time, the update step requires six multiplications and three additions (two multiplications and one addition for calculating e_t); the mapping of ρ simply requires one division and two additions (taking the exponent is only a look-up). This is per time, i.e., the computational complexity does not increase with time. As for the multidimensional case, the corresponding complexity scales linearly with the input dimensionality. Hence, the complexity is $O(d)$, where d is the dimension of the input regressor.

Under the deterministic analysis framework, the performance of the algorithm is determined by the time-accumulated squared error [5], [7], [16]. When applied to any sequence $\{y_t\}_{t\geq 1}$, the algorithm of (1) yields the total accumulated loss

$$L_n(\hat{y}, y) = L_n(\boldsymbol{w}_t^T \boldsymbol{x}_t, y) \overset{\triangle}{=} \sum_{t=1}^{n} (y_t - \hat{y}_t)^2 \tag{3}$$

for any n. We emphasize that for unbounded signals, such as Gaussian and Laplacian distributions, we can define a suitable Y such that the samples of y_t are inside of the interval $[-Y, Y]$ with high probability.

Next, we provide deterministic bounds on $L_n(\hat{y}, y)$ with respect to the best convex combination $\min_{\beta\in[\lambda^+, 1-\lambda^+]} L_n(\hat{y}_\beta, y)$, where

$$L_n(\hat{y}_\beta, y) = L_n(\boldsymbol{u}^T \boldsymbol{x}_t, y) = \sum_{t=1}^{n} (y_t - \hat{y}_{\beta,t})^2$$

and $\hat{y}_{\beta,t} \overset{\triangle}{=} \beta \hat{y}_{1,t} + (1 - \beta)\hat{y}_{2,t} = \boldsymbol{u}^T \boldsymbol{x}_t$, $\boldsymbol{u} \overset{\triangle}{=} [\beta\ 1 - \beta]^T$ that holds uniformly in an individual sequence manner without any stochastic assumptions on y_t, $\hat{y}_{1,t}$, $\hat{y}_{2,t}$, or n. Note that the best fixed convex combination parameter

$$\beta_o = \arg \min_{\beta\in[\lambda^+, 1-\lambda^+]} L_n(\hat{y}_\beta, y)$$

and the corresponding estimator

$$\hat{y}_{\beta_o,t} = \beta_o \hat{y}_{1,t} + (1 - \beta_o)\hat{y}_{2,t}$$

which we compare the performance against, can only be determined after observing the entire sequences, i.e., $\{y_t\}$, $\{\hat{y}_{1,t}\}$, and $\{\hat{y}_{2,t}\}$, in advance for all n.

III. DETERMINISTIC ANALYSIS

In this section, we first relate the accumulated loss of the mixture to the accumulated loss of the best convex combination that minimizes the accumulated loss in the following theorem. Then, we discuss the implications of the our theorem in a corollary to compare the adaptive mixture [8] with the exponentiated gradient algorithm [6]. The use of the Kullback–Leibler (KL) divergence as a distance measure for obtaining worst case loss bounds was pioneered in [17], and later adopted extensively in the online learning literature [6], [7], [18]. We emphasize that although the steady-state and transient MSE performances of the convexly constrained mixture algorithm are analyzed with respect to the constituent learning algorithms [8], [9], we perform the steady-state, transient, and tracking analysis without any stochastic assumptions or use any approximations in the following theorem.

Theorem 1: The algorithm given in (2), when applied to any sequence $\{y_t\}_{t\geq 1}$, with $|y_t| \leq Y < \infty$, $0 < \lambda^+ < 1/2$, and $\beta \in [\lambda^+, 1 - \lambda^+]$, yields for any n and $\epsilon > 0$

$$L_n(\hat{y}, y) - \left(\frac{2\epsilon + 1}{1 - z^2}\right) \min_\beta\{L_n(\hat{y}_\beta, y)\} \leq \frac{(2\epsilon + 1)Y^2}{\epsilon(1 - z^2)} \ln 2 \leq O\left(\frac{1}{\epsilon}\right) \tag{4}$$

where $O(.)$ is the order notation, $\hat{y}_{\beta,t} = \beta \hat{y}_{1,t} + (1 - \beta)\hat{y}_{2,t}$, $z \overset{\triangle}{=} 1 - 4\lambda^+(1 - \lambda^+)/1 + 4\lambda^+(1 - \lambda^+) < 1$, and step size $\mu = (4\epsilon/2\epsilon + 1)(2 + 2z/Y^2)$.

This theorem provides a regret bound for the algorithm (2) showing that the cumulative loss of the convexly constrained algorithm is close to a factor times the cumulative loss of the algorithm with the best weight chosen in hindsight. If we define the regret

$$R_n \overset{\triangle}{=} L_n(\hat{y}, y) - \left(\frac{2\epsilon + 1}{1 - z^2}\right) \min_\beta\{L_n(\hat{y}_\beta, y)\} \tag{5}$$

then (4) implies that time-normalized regret

$$\frac{R_n}{n} \triangleq \frac{L_n(\hat{y}, y)}{n} - \left(\frac{2\epsilon + 1}{1 - z^2}\right) \min_{\beta} \left\{\frac{L_n(\hat{y}_\beta, y)}{n}\right\}$$

converges to zero at a rate $O(1/n\epsilon)$ uniformly over the desired signal and the outputs of constituent algorithms. Moreover, (4) provides the exact tradeoff between the transient and steady-state performances of the convex mixture in a deterministic sense without any assumptions or approximations. Note that (4) is guaranteed to hold independent of the initial condition of the combination weight λ_t for any time interval in an individual sequence manner. Hence, (4) also provides the tracking performance of the convexly constrained algorithm in a deterministic sense.

From (4), we observe that the convergence rate of the right-hand side, i.e., the bound, is $O(1/n\epsilon)$, and, as in the stochastic case [9], to get a tighter asymptotic bound with respect to the optimal convex combination of the learning algorithms, we require a smaller ϵ, i.e., smaller learning rate μ, which increases the right-hand side of (4). Although this result is well-known in the adaptive filtering literature and appears widely in stochastic contexts, however, this tradeoff is guaranteed to hold in here without any statistical assumptions or approximations. Note that the optimal convex combination in (4) depends on the entire signal and outputs of the constituent algorithms for all n and hence it can only be determined in hindsight.

Proof: To prove the theorem, we initially assume that clipping never happens in the course of the algorithm, i.e., it is either not required or the allowed range is never violated by λ_t. Then, the extension to the case of the clipping will be straightforward. In the following, we use the approach introduced in [7] (and later used in [6]) based on measuring progress of a mixture algorithm using certain distance measures.

We first convert (2) to a direct update on λ_t and use this direct update in the proof. Using $e^{-\rho_t} = 1 - \lambda_t/\lambda_t$, the update in (2) can be written as

$$\lambda_{t+1} = \frac{\lambda_t e^{\mu e_t \lambda_t (1-\lambda_t)\hat{y}_{1,t}}}{\lambda_t e^{\mu e_t \lambda_t (1-\lambda_t)\hat{y}_{1,t}} + (1-\lambda_t) e^{\mu e_t \lambda_t (1-\lambda_t)\hat{y}_{2,t}}}. \qquad (6)$$

Unlike [6, Lemma 5.8], our update in (6) has, in a certain sense, an adaptive learning rate $\mu\lambda_t(1 - \lambda_t)$, which requires different formulation, however, follows similar lines of [6] in certain parts.

Here, for a fixed β, we define an estimator

$$\hat{y}_{\beta,t} \triangleq \beta \hat{y}_{1,t} + (1-\beta)\hat{y}_{2,t} = \boldsymbol{u}^T \boldsymbol{x}_t$$

where $\beta \in [\lambda^+, 1 - \lambda^+]$ and $\boldsymbol{u} \triangleq [\beta \quad 1 - \beta]^T$. Defining $\zeta_t = e^{\mu e_t \lambda_t (1-\lambda_t)}$, we have from (6)

$$\beta \ln\left(\frac{\lambda_{t+1}}{\lambda_t}\right) + (1-\beta)\ln\left(\frac{1-\lambda_{t+1}}{1-\lambda_t}\right) = \hat{y}_{\beta,t} \ln \zeta_t - \ln\left(\lambda_t \zeta_t^{\hat{y}_{1,t}} + (1-\lambda_t)\zeta_t^{\hat{y}_{2,t}}\right). \qquad (7)$$

Using the inequality $\alpha^x \le 1 - x(1-\alpha)$ for $\alpha \ge 0$ and $x \in [0, 1]$ from [7], we have

$$\zeta_t^{\hat{y}_{1,t}} = \left(\zeta_t^{2Y}\right)^{\frac{\hat{y}_{1,t}+Y}{2Y}} \zeta_t^{-Y} \le \zeta_t^{-Y}\left(1 - \frac{\hat{y}_{1,t}+Y}{2Y}\left(1 - \zeta_t^{2Y}\right)\right)$$

which implies in (7)

$$\ln\left(\lambda_t \zeta_t^{\hat{y}_{1,t}} + (1-\lambda_t)\zeta_t^{\hat{y}_{2,t}}\right) \le -Y \ln \zeta_t + \ln\left(1 - \frac{\hat{y}_t + Y}{2Y}\left(1 - \zeta_t^{2Y}\right)\right) \qquad (8)$$

where $\hat{y}_t = \lambda_t \hat{y}_{1,t} + (1-\lambda_t)\hat{y}_{2,t}$. As in [6], one can further bound (8) using $\ln(1 - q(1 - e^p)) \le pq + (p^2/8)$ for $0 \le q < 1$

$$\ln\left(\lambda_t \zeta_t^{\hat{y}_{1,t}} + (1-\lambda_t)\zeta_t^{\hat{y}_{2,t}}\right) \le -Y \ln \zeta_t + (\hat{y}_t + Y)\ln \zeta_t + \frac{Y^2 (\ln \zeta_t)^2}{2}. \qquad (9)$$

Using (9) in (7) yields

$$\beta \ln\left(\frac{\lambda_{t+1}}{\lambda_t}\right) + (1-\beta)\ln\left(\frac{1-\lambda_{t+1}}{1-\lambda_t}\right) \ge (\hat{y}_{\beta,t} + Y)\ln \zeta_t - (\hat{y}_t + Y)\ln \zeta_t - \frac{Y^2(\ln \zeta_t)^2}{2}. \qquad (10)$$

Now for the case of clipping, let us suppose without the loss of generality $\lambda_{t+1} = \lambda^+ - \alpha$, where $\lambda^+ > \alpha > 0$ so that it is set back to λ^+. We claim that the left-hand side of (10) can only increase by clipping, and hence, (10) stays valid after clipping. Since the derivative of $\ln x$ is monotonically decreasing with x and always positive, $\ln(\lambda_{t+1})$ must increase not less than α/λ^+ after clipping. On the other hand, $\ln(1-\lambda_{t+1})$ can decrease not more than $\alpha/1 - \lambda^+$ after clipping. As a result, $\beta \ln(\lambda_{t+1}) + (1-\beta)\ln(1 - \lambda_{t+1})$ must increase not less than $\delta = \beta\alpha/\lambda^+ - (1-\beta)\alpha/1 - \lambda^+$ after clipping. Since $\beta \in [\lambda^+, 1 - \lambda^+]$, $\delta \ge 0$. Hence, (10) is valid even after clipping.

At each adaptation, the progress made by the algorithm toward $\boldsymbol{u}$ at time t is measured as $D(\boldsymbol{u}||\boldsymbol{w}_t) - D(\boldsymbol{u}||\boldsymbol{w}_{t+1})$, where $\boldsymbol{w}_t \triangleq [\lambda_t \ (1-\lambda_t)]^T$ and

$$D(\boldsymbol{u}||\boldsymbol{w}) \triangleq \sum_{i=1}^{2} u_i \ln(u_i/w_i)$$

is the KL divergence [7], $\boldsymbol{u} \in [0, 1]^2$, and $\boldsymbol{w} \in [0, 1]^2$. We require that this progress is at least $a(y_t - \hat{y}_t)^2 - b(y_t - \hat{y}_{\beta,t})^2$ for certain a, b, and μ [6], [7]

$$a(y_t - \hat{y}_t)^2 - b(y_t - \hat{y}_{\beta,t})^2 \le D(\boldsymbol{u}||\boldsymbol{w}_t) - D(\boldsymbol{u}||\boldsymbol{w}_{t+1}) = \beta \ln\left(\frac{\lambda_{t+1}}{\lambda_t}\right) + (1-\beta)\ln\left(\frac{1-\lambda_{t+1}}{1-\lambda_t}\right) \qquad (11)$$

which yields the desired deterministic bound in (4) after telescoping. In information theory and probability theory, the KL divergence, which is also known as the relative entropy, is empirically shown to be an efficient measure of the distance between two probability vectors [6], [7]. Here, the vectors $\boldsymbol{u}$ and $\boldsymbol{w}_t$ are probability vectors, i.e., $\boldsymbol{u}, \boldsymbol{w}_t \in [0, 1]^2$, and $\boldsymbol{u}^T \boldsymbol{1} = \boldsymbol{w}_t^T \boldsymbol{1} = 1$, where $\boldsymbol{1} \triangleq [1 \ 1]^T$. This use of KL divergence as a distance measure between weight vectors is widespread in the online learning literature [6], [18].

We observe from (10) and (11) that to prove the theorem, it is sufficient to show that $G(y_t, \hat{y}_t, \hat{y}_{\beta,t}, \zeta_t) \le 0$, where

$$G(y_t, \hat{y}_t, \hat{y}_{\beta,t}, \zeta_t) \triangleq -(\hat{y}_{\beta,t} + Y)\ln \zeta_t + (\hat{y}_t + Y)\ln \zeta_t + \frac{Y^2(\ln \zeta_t)^2}{2} + a(y_t - \hat{y}_t)^2 - b(y_t - \hat{y}_{\beta,t})^2. \qquad (12)$$

For fixed y_t, $\hat{y}_t$, and ζ_t, $G(y_t, \hat{y}_t, \hat{y}_{\beta,t}, \zeta_t)$ is maximized when $\partial G/\partial \hat{y}_{\beta,t} = 0$, i.e., $\hat{y}_{\beta,t} - y_t + (\ln \zeta_t/2b) = 0$, since $\partial^2 G/\partial \hat{y}_{\beta,t}^2 = -2b < 0$, yielding $\hat{y}_{\beta,t}^* = y_t - (\ln \zeta_t/2b)$. Note that while taking the partial derivative of $G(\cdot)$ with respect to $\hat{y}_{\beta,t}$ and finding $\hat{y}_{\beta,t}^*$, we assume that all y_t, $\hat{y}_t$, ζ_t are fixed. This yields an upper bound on $G(\cdot)$ in terms of $\hat{y}_{\beta,t}$. Hence, it is sufficient to show $G(y_t, \hat{y}_t, \hat{y}_{\beta,t}^*, \zeta_t) \le 0$, where, after some algebra [6]

$$G(y_t, \hat{y}, \hat{y}^*_{\beta,t}, \zeta_t) = (y_t - \hat{y}_t)^2 \times \left[a - \mu\lambda_t(1-\lambda_t) + \frac{\mu^2{\lambda_t}^2(1-\lambda_t)^2}{4b} + \frac{Y^2\mu^2{\lambda_t}^2(1-\lambda_t)^2}{2}\right]. \quad (13)$$

For (13) to be negative, defining $k \triangleq \lambda_t(1-\lambda_t)$ and

$$H(k) \triangleq k^2\mu^2\left(\frac{Y^2}{2} + \frac{1}{4b}\right) - \mu k + a$$

it is sufficient to show that $H(k) \le 0$ for $k \in [\lambda^+(1-\lambda^+), 1/4]$, i.e., $k \in [\lambda^+(1-\lambda^+), 1/4]$ when $\lambda_t \in [\lambda^+, (1-\lambda^+)]$, since $H(k)$ is a convex quadratic function of k, i.e., $\partial^2 H/\partial k^2 > 0$. Hence, we require the interval where the function $H(\cdot)$ is negative should include $[\lambda^+(1-\lambda^+), 1/4]$, i.e., the roots k_1 and k_2 (where $k_2 \le k_1$) of $H(\cdot)$ should satisfy

$$k_1 \ge \frac{1}{4}, \quad k_2 \le \lambda^+(1-\lambda^+)$$

where

$$k_1 = \frac{\mu + \sqrt{\mu^2 - 4\mu^2 a\left(\frac{Y^2}{2} + \frac{1}{4b}\right)}}{2\mu^2\left(\frac{Y^2}{2} + \frac{1}{4b}\right)} = \frac{1 + \sqrt{1-4as}}{2\mu s} \quad (14)$$

$$k_2 = \frac{\mu - \sqrt{\mu^2 - 4\mu^2 a\left(\frac{Y^2}{2} + \frac{1}{4b}\right)}}{2\mu^2\left(\frac{Y^2}{2} + \frac{1}{4b}\right)} = \frac{1 - \sqrt{1-4as}}{2\mu s} \quad (15)$$

$$s \triangleq \left(\frac{Y^2}{2} + \frac{1}{4b}\right).$$

To satisfy $k_1 \ge 1/4$, we straightforwardly require from (14)

$$\frac{2 + 2\sqrt{1-4as}}{s} \ge \mu.$$

To get the tightest upper bound for (14), we set

$$\mu = \frac{2 + 2\sqrt{1-4as}}{s}$$

that is, the largest allowable learning rate.

To have $k_2 \le \lambda^+(1-\lambda^+)$ with $\mu = 2 + 2\sqrt{1-4as}/s$, from (15), we require

$$\frac{1 - \sqrt{1-4as}}{4(1+\sqrt{1-4as})} \le \lambda^+(1-\lambda^+). \quad (16)$$

Equation (16) yields

$$as = a\left(\frac{Y^2}{2} + \frac{1}{4b}\right) \le \frac{1-z^2}{4} \quad (17)$$

where

$$z \triangleq \frac{1 - 4\lambda^+(1-\lambda^+)}{1 + 4\lambda^+(1-\lambda^+)}$$

and $z < 1$ after some algebra.

To satisfy (17), we set $b = \epsilon/Y^2$ for any (or arbitrarily small) $\epsilon > 0$ that results

$$a \le \frac{(1-z^2)\epsilon}{Y^2(2\epsilon+1)}. \quad (18)$$

To get the tightest bound in (11), we select

$$a = \frac{(1-z^2)\epsilon}{Y^2(2\epsilon+1)}$$

in (18). Such selection of a, b, and μ results in (11)

$$\left(\frac{(1-z^2)\epsilon}{Y^2(2\epsilon+1)}\right)(y_t - \hat{y}_t)^2 - \left(\frac{\epsilon}{Y^2}\right)(y_t - \hat{y}_{\beta,t})^2 \le \beta \ln\left(\frac{\lambda_{t+1}}{\lambda_t}\right) + (1-\beta)\ln\left(\frac{1-\lambda_{t+1}}{1-\lambda_t}\right). \quad (19)$$

After telescoping, i.e., summation over t, $\sum_{t=1}^n$, (19) yields

$$aL_n(\hat{y}, y) - b\min_\beta\{L_n(\hat{y}_\beta, y)\} \le \beta\ln\left(\frac{\lambda_{n+1}}{\lambda_1}\right) + (1-\beta)\ln\left(\frac{1-\lambda_{n+1}}{1-\lambda_1}\right) \le \ln 2 \le O(1) \quad (20)$$

where $\beta\ln(\lambda_{n+1}/\lambda_1) + (1-\beta)\ln(1-\lambda_{n+1}/1-\lambda_1) \le \ln 2$ since we initialize the algorithm with $\lambda_1 = 1/2$. Note for a random initialization that this bound would correspond to in general $\beta\ln(\lambda_{n+1}/\lambda_1) + (1-\beta)\ln(1-\lambda_{n+1}/1-\lambda_1) \le -((1-\lambda^+)\ln\lambda^+ + \lambda^+\ln(1-\lambda^+)) = O(1)$. Hence

$$\left(\frac{(1-z^2)\epsilon}{Y^2(2\epsilon+1)}\right)L_n(\hat{y}, y) - \left(\frac{\epsilon}{Y^2}\right)\min_\beta\{L_n(\hat{y}_\beta, y)\} \le \ln 2 \le O(1). \quad (21)$$

Then, it follows that:

$$L_n(\hat{y}, y) - \left(\frac{2\epsilon+1}{1-z^2}\right)\min_\beta\{L_n(\hat{y}_\beta, y)\} \quad (22)$$

$$\le \frac{(2\epsilon+1)Y^2}{\epsilon(1-z^2)}\ln 2 \le O\left(\frac{1}{\epsilon}\right) \quad (23)$$

which is the desired bound.

Note that using

$$b = \frac{\epsilon}{Y^2}, \quad a = \frac{(1-z^2)\epsilon}{Y^2(2\epsilon+1)}, \quad s = \left(\frac{Y^2}{2} + \frac{1}{4b}\right)$$

we get

$$\mu = \frac{2 + 2\sqrt{1-4as}}{s} = \frac{4\epsilon}{2\epsilon+1}\frac{2+2z}{Y^2}$$

after some algebra, as in the statement of the theorem. □

Finally, we also define a time-normalized regret as in [6] to have a comparison between the exponentiated gradient algorithm and the adaptive mixture given in (2). Let us define the regret R_n^* as

$$R_n^* \triangleq L_n(\hat{y}, y) - \min_\beta\{L_n(\hat{y}_\beta, y)\} \quad (24)$$

then in the following corollary, we show that the time normalized regret $1/nR_n^*$ for the algorithm proposed originally in [8] and given in (2) improves, i.e., decreases, with $O(n^{-1/2})$ in a similar manner to the exponentiated gradient algorithm [6], except that the time-normalized regret $1/nR_n^*$ is always above an error floor, i.e., it is a linear regret with n and hence, it does not converge to 0.

Corollary 1: The algorithm given in (2), when applied to any sequence $\{y_t\}_{t\ge1}$, with $|y_t| \le Y < \infty$, $0 < \lambda^+ < 1/2$, and $\beta \in [\lambda^+, 1-\lambda^+]$, yields for any n

$$\frac{1}{n}R_n^* \le \frac{4Y\epsilon}{1-z^2} + \frac{2Yz^2}{1-z^2} + \frac{Y^2\ln 2}{n(1-z^2)}\left(2 + \frac{1}{\epsilon}\right) \le O\left(n^{\frac{-1}{2}}\right) + O(1) \quad (25)$$

where $O(.)$ is the order notation, R_n^* is defined in (24), $z \triangleq (1-4\lambda^+(1-\lambda^+))/(1+4\lambda^+(1-\lambda^+)) < 1$, $\epsilon = \sqrt{Y\ln 2/4n}$, and step size $\mu = (4\epsilon/2\epsilon+1)(2+2z/Y^2)$.

Proof: We first note that $\beta \ln(\lambda_{n+1}/\lambda_1) + (1-\beta)\ln(1-\lambda_{n+1}/1-\lambda_1) \leq \ln 2$ for $\beta, \lambda_n \in [0,1], \forall n$ since $\lambda_1 = 1/2$ and $L_n(\hat{y}_\beta, y) \leq 2Yn, \forall\beta$. Then, from (19) and (23)

$$L_n(\hat{y}, y) - L_n(\hat{y}_\beta, y) \leq \gamma(\epsilon) \quad \forall\beta$$

and

$$\gamma(\epsilon) = \frac{4Y\epsilon n}{1-z^2} + \frac{2Yz^2 n}{1-z^2} + \frac{Y^2 \ln 2}{1-z^2}\left(2 + \frac{1}{\epsilon}\right)$$

where

$$\gamma'(\epsilon) = \frac{4Yn}{1-z^2} - \frac{Y^2 \ln 2}{1-z^2}\frac{1}{\epsilon^2} = 0 \Rightarrow \epsilon^* = \sqrt{\frac{Y \ln 2}{4n}}$$

is chosen to get the tightest bound since $\gamma''(\epsilon) = (2Y^2 \ln 2/1-z^2)(1/\epsilon^3) > 0, \forall\epsilon > 0$. Hence, the statement in the corollary follows. □

We note that the algorithm given in (2), as shown in the corollary, has an error floor $2Yz^2/1-z^2$, which bounds the limit of the time-normalized regret $\lim_{n\to}\infty 1/nR_n^*$ as follows. This result is due to the nonconvexity of the loss function that uses the sigmoid function in parameterization of λ_t. On the other hand, we have a certain control over this error floor, which is given here as a function of $0 < z = (1-4\lambda^+(1-\lambda^+))/(1+4\lambda^+(1-\lambda^+)) < 1$. Since $\lim_{\lambda^+\to 1/2} z = 0$, and $\lim_{\lambda^+\to 0 \text{ or } 1} z = 1$, z controls the size of the competition class $\{\beta\}$, where $\beta \in [\lambda^+, 1-\lambda^+]$. As this class grows, the studied algorithm in this brief is affected by a larger error floor induced on the time-normalized regret $1/nR_n^*$. Therefore, the algorithm given in (2) does not guarantee a diminishing time normalized regret and the bound it promises is weak when compared with the, for example, exponentiated gradient algorithm [6], whose time normalized regret is $O(n^{-1/2})$.

IV. Simulations

In this section, we illustrate the performance of the learning algorithm (2) and the introduced results through examples. We demonstrate that the upper bound given in (4) is asymptotically tight by providing a specific sequence for the desired signal y_t and the outputs of constituent algorithms $\hat{y}_{1,t}$ and $\hat{y}_{2,t}$. We also present a performance comparison between the adaptive mixture and the corresponding best mixture component on a pair of sequences.

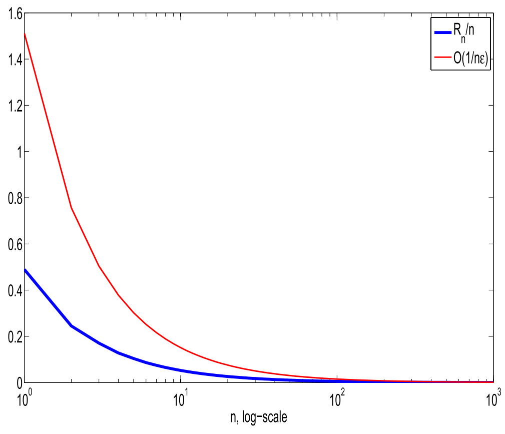
In the first example, we present the time-normalized regret $1/nR_n$ of the learning algorithm (2) defined in (5) and the corresponding upper bound given in (4). We first set $Y = 1$, $\lambda^+ = 0.15$, and $\epsilon = 1$. Here, for $t = 1, \ldots, 1000$, the desired signal y_t and the sequences $\hat{y}_{1,t}, \hat{y}_{2,t}$, which the parallel running constituent algorithms produce are given by

$$\hat{y}_{1,t} = Y; \quad \hat{y}_{2,t} = (-1)^t Y; \quad \text{and } y_t = 0.15\hat{y}_{1,t} + 0.85\hat{y}_{2,t}.$$

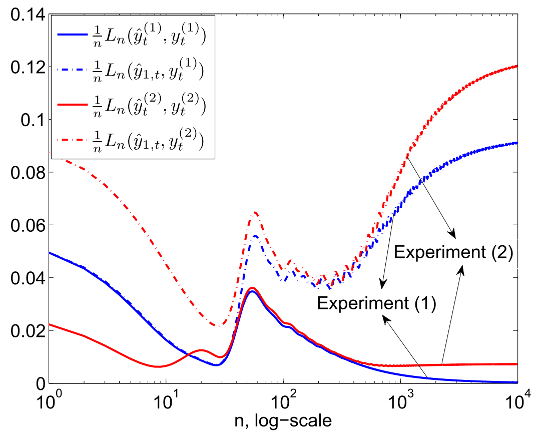
Note that, in this case, the best convex combination weight is $\beta_o = 0.15$. In Fig. 1(a), we plot the time-normalized regret of the learning algorithm (2) $1/nR_n$ and the upper bound given in (4) $O(1/(n\epsilon))$. From Fig. 1(a), we observe that the bound introduced in (4) is asymptotically tight, i.e., as n gets larger, the gap between the upper bound and the time-normalized regret gets smaller.

In the second example, we demonstrate the effectiveness of the mixture of experts algorithm (2) through a comparison between the time-normalized accumulated loss (3) of the learned mixture and the one of the best constituent expert. To this end, we design two experiments with $t = 1, \ldots, 10\,000$, $\lambda^+ = 0.01$, $\epsilon = 0.1$, and $Y = e$, where

$$\hat{y}_{1,t} = 2e^{-0.005t} - 1, \quad \hat{y}_{2,t} = \sin(0.1t)$$



(a)



(b)

Fig. 1. (a) Regret bound derived in Theorem 1. (b) Comparison of the adaptive mixture (2) with respect to the best expert.

are chosen as the experts in both of the experiments. In the first experiment, we choose the desired signal as the linear combination $y_t^{(1)} = 0.75\hat{y}_{1,t} + 0.25\hat{y}_{2,t}$, where $\beta_0 = 0.75$. In the second experiment, we choose the desired signal as the nonlinear function of the outputs of the both experts as $y_t^{(2)} = \sin(0.75\hat{y}_{1,t} + 0.25\hat{y}_{2,t})$. Note that the first expert provides a better time-normalized accumulated loss in both cases, i.e., $1/nL_n(\hat{y}_{1,t}, y_t^{(i)}) < (1/n)L_n(\hat{y}_{2,t}, y_t^{(i)})$. In Fig. 1(b), we plot the time-normalized accumulated loss of the best (first) expert as well as the one of the mixture returned by the learning algorithm. From Fig. 1(b), we observe that the adaptive mixture outperforms the best mixture component, i.e., expert one in these examples, in both of the cases. Furthermore, the adaptive mixture optimally tunes to the best linear combination in the first case, which is expected since the generation of the desired output is through a linear combination. On the other hand, the adaptive mixture suffers from an error floor, i.e., the time-normalized accumulated loss does not converge to 0, in the second case, since the generation of the desired signal is through a nonlinear transformation.

In this section, we illustrated our theoretical results and the performance of the learning algorithm (2) through examples. We observed through an example that the upper bound given in (4) is asymptotically tight. We also illustrated the effectiveness of the adaptive mixture on another example by a performance comparison between the mixture and its best component.

V. Conclusion

In this brief, we analyze a learning algorithm [8] that adaptively combines outputs of two constituent algorithms running in parallel to model an unknown desired signal from the perspective of the online learning theory and produce results in an individual sequence manner such that our results are guaranteed to hold for any bounded arbitrary signal. We relate the time-accumulated squared estimation error of this algorithm at any time to the time-accumulated squared estimation error of the optimal convex combination of the constituent algorithms that can only be chosen in hindsight. We refrain from making statistical assumptions on the underlying signals and our results are guaranteed to hold in an individual sequence manner. To this end, we provide the transient, steady state, and tracking analysis of this mixture in a deterministic sense without any assumptions on the underlying signals or without any approximations in the derivations. We illustrate the introduced results through examples.

REFERENCES

[1] S. Wan and L. E. Banta, "Parameter incremental learning algorithm for neural networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1424–1438, Nov. 2006.

[2] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1411–1423, Nov. 2006.

[3] W. Wan, "Implementing online natural gradient learning: Problems and solutions," *IEEE Trans. Neural Netw.*, vol. 17, no. 2, pp. 317–329, Mar. 2006.

[4] T. C. Silva and L. Zhao, "Stochastic competitive learning in complex networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 3, pp. 385–398, Mar. 2012.

[5] N. Cesa-Bianchi and L. Lugosi, *Prediction, Learning and Games*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[6] J. Kivinen and M. K. Warmuth, "Exponentiated gradient versus gradient descent for linear predictors," *J. Inf. Comput.*, vol. 132, no. 1, pp. 1–62, Jan. 1997.

[7] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth, "How to use expert advice," *J. ACM*, vol. 44, no. 3, pp. 427–485, May 1997.

[8] J. Arenas-Garcia, A. R. Figueiras-Vidal, and A. H. Sayed, "Mean-square performance of a convex combination of two adaptive filters," *IEEE Trans. Signal Process.*, vol. 54, no. 3, pp. 1078–1090, Mar. 2006.

[9] S. S. Kozat, A. T. Erdogan, A. C. Singer, and A. H. Sayed, "Steady-state MSE performance analysis of mixture approaches to adaptive filtering," *IEEE Trans. Signal Process.*, vol. 58, no. 8, pp. 4050–4063, Aug. 2010.

[10] A. H. Sayed, *Fundamentals of Adaptive Filtering*. New York, NY, USA: Wiley, 2003.

[11] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[12] S. G. Osgouei and M. Geravanchizadeh, "Speech enhancement using convex combination of fractional least-mean-squares algorithm," in *Proc. 5th Int. Symp. Telecommun.*, Dec. 2010, pp. 869–872.

[13] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 58, no. 9, pp. 4795–4810, Sep. 2010.

[14] S. E. Yuksel, J. E. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1177–1193, Aug. 2012.

[15] S. S. Kozat, "Competitive signal processing," Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Illinois Urbana-Champaign, Urbana, IL, USA, 2004.

[16] A. Gyorgy, T. Linder, and G. Lugosi, "Tracking the best quantizer," *IEEE Trans. Inf. Theory*, vol. 54, no. 4, pp. 1604–1625, Apr. 2008.

[17] N. Littlestone, "Mistake bounds and logarithmic linear-threshold learning algorithms," Ph.D. dissertation, Dept. Comput. Res. Lab., Univ. California, Santa Cruz, CA, USA, 1989.

[18] N. Cesa-Bianchi, P. M. Long, and M. K. Warmuth, "Worst-case quadratic loss bounds for prediction using linear functions and gradient descent," *IEEE Trans. Neural Netw.*, vol. 7, no. 3, pp. 604–619, May 1996.