STEADY STATE AND TRANSIENT MSE ANALYSES OF ADAPTIVE MIXTURE METHODS

by

Mehmet Ali Dönmez

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Electrical & Electronics Engineering

Koç University

June, 2013

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Mehmet Ali Dönmez

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Assist. Prof. Süleyman Serdar Kozat

Assoc. Prof. Alper Tunga Erdoğan

Assist. Prof. Sinem Çöleri Ergen

Prof. Özgür Barış Akan

Assist. Prof. Emre Mengi

Date: _____

To my family

ABSTRACT

In this thesis, we analyze adaptive mixture methods that combine outputs of several adaptive filters running in parallel to model an unknown system. We first study three different convex combination methods that combine outputs of two adaptive algorithms and provide their steady-state and transient performances. We next investigate affine and linear combination methods based on Bregman divergences that combine outputs of several adaptive filters and present the mean and the mean-square transient analysis of these adaptive algorithms.

In the first part, we investigate convexly constrained mixture methods to adaptively combine outputs of two adaptive filters running in parallel to model a desired unknown system. We compare several algorithms with respect to their mean square error in the steady-state, when the underlying unknown system is nonstationary with a random walk model. We demonstrate that these algorithms are universal such that they achieve the performance of the best constituent filter in the steady-state if certain algorithmic parameters are chosen properly. We also demonstrate that certain mixtures converge to the optimal convex combination filter such that their steady-state performances can be better than the best constituent filter. We also perform the transient analysis of these updates in the mean and mean-square error senses.

In the second part, we investigate adaptive mixture methods that linearly combine outputs of m constituent filters running in parallel to model a desired signal. We use *Bregman divergences* and obtain certain multiplicative updates to train the linear combination weights under an affine constraint or without any constraints. We use unnormalized relative entropy and relative entropy to define two different Bregman divergences that produce an unnormalized exponentiated gradient update and a normalized exponentiated gradient update on the mixture weights, respectively. We then carry out the mean and the mean-square transient analysis of these adaptive algorithms when they are used to combine outputs of m constituent filters. We illustrate the accuracy of our results and demonstrate the effectiveness

of these updates for sparse mixture systems.

ÖZETÇE

Bu tez çalışmasında, bilinmeyen bir sistemi modellemek için paralel olarak çalışan birden fazla uyarlanır süzgecin çıktılarını birleştiren uyarlanır birleşim metotları incelenmektedir. Öncelikle iki farklı uyarlanır algoritmanın çıktılarını birleştiren üç farklı dışbükey birleşim metodu etüt edilmekte ve bunların kalıcı zaman ve geçici zaman performansları verilmektedir. Daha sonra Bregman ıraksaklıkları temelli olup birden fazla uyarlanır süzgecin çıktılarını birleştiren ilgin ve doğrusal birleşim metodları araştırılmakta ve bunların ortalama ve ortalama-karesel geçici zaman analizleri sunulmuktadır.

İlk kısımda, istenilen ve bilinmeyen bir sistemi modellemek için paralel olarak çalışan iki uyarlanır süzgecin çıktılarını uyarlanır biçimde birleştiren dışbükey kısıtlanmış birleşim metotları araştırılmaktadır. Birçok algoritma, bilinmeyen sistem doğrusal değilken ve rastgele yürüyüş modeline göre hareket ederken, kalıcı zamanda ortalama-karesel hatalarına göre kıyaslanmaktadır. Eğer belirli algoritma parametreleri uygun biçimde seçilirse, bu algoritmaların "evrensel" olacağı; yani kalıcı zamanda en iyi birleşen algoritmasının performansına ulaşacağı gösterilmektedir. Ayrıca bazı birleşimlerin optimal birleşim süzgecine yakınsayacağı ve bunların kalıcı zaman performanslarının en iyi birleşen algoritmasının performansından daha iyi olabileceği gösterilmektedir. Bu algoritmaların geçici zamanda ortalama ve ortalama-karesel analizleri de yapılmaktadır.

İkinci kısımda, istenilen işareti modellemek için paralel olarak çalışan m adet birleşen süzgecin çıktılarını doğrusal olarak birleştiren uyarlanır birleşim metodları araştırılmaktadır. Doğrusal birleşim ağırlıklarını ilgin kısıt altında veya kısıtsız olarak eğitmek için Bregman ıraklaklıkları kullanılmakta ve bazı çarpımsal algoritmalar elde edilmektedir. Birleşim ağırlıkları üzerine bir düzgelenmemiş üstellenmiş gradient algoritması ve bir düzgelenmiş üstellenmiş gradient algoritması üreten iki farklı Bregman ıraksaklığı tanımlamak için sırasıyla düzgelenmiş göreli entropi ve göreli entropi kullanılmaktadır. Daha sonra bu algoritmaların m birleşen algoritmayı birleştirdikleri durumdaki geçici zamanda ortalama ve ortalama-karesel analizleri yapılaktadır. Sonuçların doğruluğu ve bu algoritmaların etkinliği seyrek

birleşim sistemleri için gösterilmektedir.

ACKNOWLEDGMENTS

I first wish to express my sincere gratitude to my supervisor Dr. Süleyman Serdar Kozat for all of his guidance and encouragement throughout this process. I am very indebted to his patience and invaluable advices that inspired me and I am honored by his confidence and trust on my abilities. I express special thanks to the reading committee members, Dr. Alper Tunga Erdoğan, Dr. Sinem Çöleri Ergen, Dr. Özgür Barış Akan and Dr. Emre Mengi for their careful reading, valuable comments and suggestions on my thesis.

I would like to thank to Koç University and TÜBİTAK for providing me with financial support which made this study possible.

My sincere thanks to the members of the Competitive Signal Processing Laboratory and to all my friends for their friendship, help and support. Finally, I express my gratitude to my family for their lifetime support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

The problem of estimating or learning an unknown desired signal is heavily investigated in adaptive signal processing literature [1, 10, 15, 20]. 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 lack of information, mixture approaches are proposed that adaptively combine outputs of multiple constituent algorithms performing the same task in the adaptive signal processing under the adaptive mixture methods framework [1, 22, 29]. These parallel running algorithms can be seen as alternative hypotheses for modeling, which can be exploited for both performance improvement and robustness. In this thesis, we investigate convex, affine and linear combination methods and provide their steady-state and transient performances.

Adaptive mixture approaches are shown to improve the steady-state and transient performance over the constituent filters under certain scenarios [1, 9, 22, 30]. The steady-state analysis of convexly constrained, affinely constrained and unconstrained mixtures are carried out in [1, 9, 22], respectively. Specifically, the adaptive convex mixture of [1] is shown to be universal with respect to the constituent filters such that this algorithm achieves the excess mean-square error (EMSE) performance of the best constituent filter and, in certain cases, even outperforms both [1]. The transient analysis of this adaptive convex combination is studied in [25], where the time evolution of the mean and variance of the mixture weights is provided. Along these lines, an affinely constrained mixture of adaptive filters using a stochastic gradient update is introduced in [9]. The steady-state mean square error (MSE) of this affinely constrained mixture is shown to outperform the steady-state MSE of the best constituent filter in the mixture under certain conditions [9]. The transient analysis of this affinely constrained mixture for m constituent filters is carried out in [24]. The general linear mixture framework as well as the steady-state performances of different

mixture configurations are studied in [22].

In the first chapter, we study four convex combination methods to combine outputs of two adaptive filters to model a desired unknown system [1,20,27,30]. We first provide their MSE performances in the steady-state. We next perform the transient analysis of these convexly constrained updates in the mean and the mean square senses. In this framework, we have two adaptive filters that work in parallel in order to model an unknown system [1]. The outputs of these algorithms are then combined using another adaptive method in order to improve the overall performance [1]. The first adaptive algorithm [1] uses a stochastic gradient update on the convexly constrained mixture weights to minimize the final estimation error. The second algorithm minimizes an approximate final estimation error while penalizing the distance between the new and the old mixture weights [6, 20]. The third [30] and the fourth algorithms [27] employ specific performance-based updates on the combination weights.

In the second chapter, we investigate affine and linear combination methods based on *Bregman divergences* [10,15] that combine outputs of m constituent filters running in parallel to model an unknown system. This mixture framework consists of two stages [2–5,8,23]. In the first stage, there are several adaptive filters that work in parallel to model an unknown desired signal. In the second stage, we linearly combine the outputs of these adaptive filters to produce the final output. To train the linear mixture weights, we employ Bregman divergences and propose certain multiplicative updates under an affine constraint [9] or without any constraints [22]. We employ two different Bregman divergences based on unnormalized [15] and normalized relative entropy [21] producing the unnormalized exponentiated gradient update (EGU) and the exponentiated gradient update (EG) on the mixture weights, respectively. We then perform the mean and the mean-square transient analysis of these adaptive mixtures when they are used to combine outputs of m constituent filters.

1.1 Contributions

The contributions of the first part of this thesis are as follows:

- We present the steady-state and transient analysis of three convex combination methods that adaptively combines outputs of two adaptive filters.

- We show that if we use the EG algorithm [20] to update the mixing parameter, the combination filter performs, at least, as well as the best constituent filter in the steady-state.

- We analyze the steady-state behavior of [30] and show that with a proper selection of the forgetting factor, the combination filter performs as well as the best constituent filter in the steady-state.

- We demonstrate that if the mixture parameter in [27] is selected using a sufficiently large time window, then the combination filter can achieve the performance of the best constituent filter in the steady-state.

The contributions of the second part of this thesis are as follows:

- We use Bregman divergences to derive multiplicative updates on affinely constrained and unconstrained mixture weights that adaptively combine outputs of m constituent filters.

- We use the unnormalized relative entropy and the relative entropy to define two different Bregman divergences that produce the EGU algorithm and the EG algorithm to update the affinely constrained and unconstrained mixture weights.

- We perform the mean and the mean-square transient analysis of the affinely constrained and unconstrained mixtures using the EGU algorithm and the EG algorithm.

1.2 Content

Chapter 2 begins with a brief description of the convex mixture framework for the combination of two adaptive filters running in parallel with the error quantities and performance measures. In Section 2.2, we present four convex mixture methods in detail. In Section 2.3, we present the steady-state MSE analysis of these methods with the converged mixture weights in the steady-state. In Section 2.4, we provide a transient analysis of the corresponding algorithms. We illustrate the introduced results through simulations under the setup of [1] in Section 2.5. We demonstrate that our results accurately describe the behavior

of these algorithms both in the steady-state and during convergence in the studied setup. The chapter concludes with certain remarks.

In Chapter 3, we first present the general linear mixture framework that combines outputs of m adaptive filters. In Section 3.2, we study the affinely constrained and unconstrained mixture methods updated with the EGU algorithm and the EG algorithm. In Section 3.3, we first perform the transient analysis of the affinely constrained mixtures and then continue with the transient analysis of the unconstrained mixtures. Finally, in Section 3.4, we perform simulations to show the accuracy of our results and to compare performances of the different adaptive mixture methods. We conclude this chapter with remarks in Section 3.5.

Chapter 2

STEADY STATE AND TRANSIENT MSE ANALYSIS OF CONVEXLY CONSTRAINED MIXTURE METHODS

In this chapter, we first investigate and compare four well-known convexly constrained adaptive mixture methods to combine outputs of two adaptive filters [1, 20, 27, 30] with respect to their MSE in the steady-state. We then perform the transient analysis of these convexly constrained updates in the mean and the MSE senses. In this widely studied framework, we have two adaptive filters that work in parallel in order to model an unknown system [1]. The outputs of these algorithms are then combined using another adaptive method in order to improve the overall performance [1]. The first adaptive algorithm [1] uses a stochastic gradient update on the convexly constrained mixture parameter to minimize the final estimation error. The second algorithm is based on the exponentiated gradient (EG) algorithm [6,20]. The EG algorithm has extensive roots in sequential learning theory [13,31] and minimizes an approximate final estimation error while penalizing the distance between the new and the old mixture parameters. The third [30] and the fourth algorithms [27] use specific performance-based updates on the mixture parameters as further detailed in Section 2.2. Although we specifically concentrate on the combination of two filters for presentation clarity, our results can be readily extended to mixtures having more than two filters [2].

We first show that if we use the EG algorithm to update the mixing parameter, the resulting combination filter is universal with respect to the constituent filters such that the combination filter performs, at least, as well as the best constituent filter in the steady-state. Specifically, we show that the EMSE of the combination filter is as small as the best of the constituent filters and, in some cases, smaller than EMSEs of the component filters in the steady-state. We also show that the mixture parameter under the EG update converges to the optimal convex combination parameter that minimizes the EMSE. Note that the EG algorithm is shown to converge faster and has better tracking performance than the

stochastic gradient algorithms for sparse impulse responses in certain situations [6, 17, 20]. Hence, the EG algorithm can be preferred over the stochastic gradient based algorithms for mixtures having more than two filters and when the combination favors only a few of the constituent filters. We point out that although the MSE of the EG algorithm is studied using Euler discretization in [17] under certain assumptions for uncorrelated input regressors, our framework and the analysis are significantly different since we use the EG algorithm to combine outputs of adaptive filters, which are nonlinearly coupled, such that the assumptions of [17] do not hold. The third algorithm we investigate is based on a certain performance-based mixture of the constituent filters [30]. We analyze the steady-state behavior of [30] and show that with a proper selection of the forgetting factor, the combination filter is universal such that it performs as well as the best constituent filter in the steady-state. Although the algorithm of [30] is also shown to be universal in a strong deterministic sense [30], we show that the mixture parameter does not converge to the optimal convex combination parameter under our assumptions (which is also supported by our experiments). The fourth algorithm we investigate was studied in [27] and combines filters based on their performances within a time window. We demonstrate that if the mixture parameter in [27] is selected using a sufficiently large time window, the combination filter can achieve the performance of the best constituent filter in the steady-state. For all algorithms, we also perform the transient analysis in the mean and the MSE senses.

2.1 Problem Description

In this framework, we have two adaptive algorithms that run in parallel to model a desired signal $d(t)$. The desired signal $d(t)$ is given by $d(t) = \boldsymbol{w}_o^T(t)\boldsymbol{u}(t) + n(t)$, where $\boldsymbol{w}_o(t) \in \mathbb{R}^p$ is the desired system vector that varies according to a random walk model [28], i.e.,

$$\boldsymbol{w}_o(t+1) = \boldsymbol{w}_o(t) + \boldsymbol{q}(t),$$

where $\boldsymbol{q}(t)$ is a zero mean, i.i.d. random vector with covariance matrix $\boldsymbol{Q} = E[\boldsymbol{q}(t)\boldsymbol{q}^T(t)]$, $\boldsymbol{u}(t) \in \mathbb{R}^p$ is the input regressor with zero mean and correlation matrix $\boldsymbol{R} = E[\boldsymbol{u}(t)\boldsymbol{u}^T(t)]$ and the observation noise $n(t)$ is i.i.d. with zero mean and variance $E[n^2(t)] = \sigma_n^2$. The cross correlation vector between the desired signal and the input regressor is $\boldsymbol{p}(t) = E[d(t)\boldsymbol{u}(t)]$.

To model the desired signal $d(t)$, we have two parallel running constituent filters each producing estimates $\hat{d}_1(t) = \boldsymbol{w}_1^T(t)\boldsymbol{u}(t)$ and $\hat{d}_2(t) = \boldsymbol{w}_2^T(t)\boldsymbol{u}(t)$ using the weight vectors $\boldsymbol{w}_1(t)$, $\boldsymbol{w}_2(t)$ respectively. For each constituent filter, we define the estimation error, the *a priori* error and the *a posteriori* error as

$$\begin{aligned} e_i(t) &\triangleq d(t) - \hat{d}_i(t) = d(t) - \boldsymbol{w}_i^T(t)\boldsymbol{u}(t), \\ e_{a,i}(t) &\triangleq [\boldsymbol{w}_o(t) - \boldsymbol{w}_i(t)]^T\boldsymbol{u}(t), \\ e_{p,i}(t) &\triangleq [\boldsymbol{w}_o(t) - \boldsymbol{w}_i(t+1)]^T\boldsymbol{u}(t), \end{aligned}$$

respectively. For each filter, we also define MSE as $J_i(t) \triangleq E[e_i^2(t)]$ and excess MSE as $J_{\text{ex},i}(t) \triangleq E[e_{a,i}^2(t)]$, with limiting values $J_i \triangleq \lim_{t\to\infty} J_i(t)$, $J_{\text{ex},i} \triangleq \lim_{t\to\infty} J_{\text{ex},i}(t)$ (if the limits exist). We also define the cross correlation between the *a priori* errors as $J_{\text{ex},12}(t) \triangleq E[e_{a,1}(t)e_{a,2}(t)]$ with limiting value $J_{\text{ex},12} \triangleq \lim_{t\to\infty} J_{\text{ex},12}(t)$. We also define $\Delta J_i(t) = J_{\text{ex},i}(t) - J_{\text{ex},12}(t)$ for $i = 1, 2$ with the limiting values $\Delta J_i = J_{\text{ex},i} - J_{\text{ex},12}$ [1].

The outputs of the constituent filters are then combined using another adaptive layer to produce the final estimate of the desired signal as

$$\hat{d}(t) = \lambda(t)\hat{d}_1(t) + [1 - \lambda(t)]\hat{d}_2(t),$$

where $\lambda(t)$ is the mixing parameter constrained to be in $[0, 1]$. If $\boldsymbol{y}(t) \triangleq [\hat{d}_1(t)\ \hat{d}_2(t)]^T$ and $\boldsymbol{w}(t) \triangleq [\lambda(t)\ 1 - \lambda(t)]^T$, then we have

$$\hat{d}(t) = \boldsymbol{w}^T(t)\boldsymbol{y}(t).$$

The final estimation error is given as $e(t) = d(t) - \hat{d}(t)$. In this chapter, we investigate four methods to train the combination weight $\lambda(t)$. Assuming convergence, the optimal mean combination weights in terms of minimizing the MSE under convex constraint are given by [1]

$$\boldsymbol{w}_{\text{o,c}} \triangleq \begin{cases} \begin{bmatrix} 1 & 0 \end{bmatrix}^T & : J_{\text{ex},1} \le J_{\text{ex},12} \le J_{\text{ex},2} \\ \begin{bmatrix} 0 & 1 \end{bmatrix}^T & : J_{\text{ex},2} \le J_{\text{ex},12} \le J_{\text{ex},1} \\ \begin{bmatrix} \frac{\Delta J_2}{\Delta J_1 + \Delta J_2} & \frac{\Delta J_1}{\Delta J_1 + \Delta J_2} \end{bmatrix}^T & : J_{\text{ex},12} < J_{\text{ex},i}, i = 1, 2 \end{cases} \tag{2.1}$$

in the steady-state.

2.2 Four Convexly Constrained Mixture Methods

In this section, we present four methods to train the mixture parameter $\lambda(t)$. The first adaptive algorithm [1] uses a stochastic gradient update on the convexly constrained mixture parameter to minimize the final estimation error. The second algorithm is based on the exponentiated gradient (EG) algorithm [6, 20] and minimizes an approximate final estimation error while penalizing the distance between the new and the old mixture parameters. The third [30] and the fourth algorithms [27] use specific performance-based updates on the mixture parameters.

2.2.1 Algorithm 1

For the convexly constrained algorithm from [1], the mixture parameter is given by

$$\lambda_\alpha(t) = \frac{1}{1+\exp[-\alpha(t)]},$$

where $\alpha(t)$ is trained using a stochastic gradient update to minimize the final prediction error as

$$\begin{aligned}\alpha(t+1) &= \alpha(t) - \frac{\mu_\alpha}{2}\nabla_{\alpha(t)}e^2(t) \\ &= \alpha(t) + \mu_\alpha e(t)[\hat{d}_1(t) - \hat{d}_2(t)]\lambda_\alpha(t)[1-\lambda_\alpha(t)]. \end{aligned} \tag{2.2}$$

For (2.2), we have [1]

$$J_{\text{ex}} = \begin{cases} J_{\text{ex},1} & : J_{\text{ex},1} \le J_{\text{ex},12} < J_{\text{ex},2}, \\ J_{\text{ex},12} + \frac{\Delta J_1 \Delta J_2}{\Delta J_1 + \Delta J_2} & : J_{\text{ex},12} < J_{\text{ex},1} < J_{\text{ex},2}, \end{cases}$$

where J_{ex} is the EMSE of the combination filter and $J_{\text{ex},12} + \frac{\Delta J_1 \Delta J_2}{\Delta J_1 + \Delta J_2} < J_{\text{ex},1}$. Furthermore, if $\boldsymbol{w}_\alpha(t) \stackrel{\triangle}{=} [\lambda_\alpha(t)\ 1-\lambda_\alpha(t)]^T$, then we have [1]

$$\lim_{t\to\infty} E[\boldsymbol{w}_\alpha(t)] = \begin{cases} \begin{bmatrix} 1 & 0 \end{bmatrix}^T & : J_{\text{ex},1} \le J_{\text{ex},12} < J_{\text{ex},2}, \\ \begin{bmatrix} \frac{\Delta J_2}{\Delta J_1 + \Delta J_2} & \frac{\Delta J_1}{\Delta J_1 + \Delta J_2} \end{bmatrix}^T & : J_{\text{ex},12} < J_{\text{ex},1} < J_{\text{ex},2}. \end{cases}$$

Hence, in the steady-state, the mixture performs as well as the best component filter

and, in some cases, outperforms both. Moreover, the combination weight vector $\boldsymbol{w}_\alpha(t)$ converges to the optimal weight vector $\boldsymbol{w}_{o,c}$ under the convex constraint.

2.2.2 Algorithm 2

The second convexly constrained update is based on the exponentiated gradient (EG) algorithm [20]. The EG algorithm has extensive roots in competitive online learning theory and has been used in different signal processing problems such as in echo cancellation [6, 17]. Here, we use the EG algorithm to train the mixture weights, where the combination weight is updated as [6, 20]

$$\lambda_\rho(t+1) = \arg\min_{\lambda\in[0,1]} \left\{ \mathrm{d}(\boldsymbol{w}, \boldsymbol{w}_\rho(t)) + \frac{\mu_\rho}{2}\left[e^2(t) + \frac{\partial (d(t) - \boldsymbol{w}^T\boldsymbol{y}(t))^2}{\partial\lambda}\Big|_{\lambda=\lambda_\rho(t)} (\lambda - \lambda_\rho(t)) \right] \right\} \tag{2.3}$$

$$= \frac{\lambda_\rho(t)\exp[\mu_\rho e(t)\hat{d}_1(t)]}{\lambda_\rho(t)\exp[\mu_\rho e(t)\hat{d}_1(t)] + [1-\lambda_\rho(t)]\exp[\mu_\rho e(t)\hat{d}_2(t)]}, \tag{2.4}$$

where $\mathrm{d}(\boldsymbol{w}, \boldsymbol{w}_\rho(t)) = \lambda \ln\left(\frac{\lambda}{\lambda_\rho(t)}\right) + (1-\lambda)\ln\left(\frac{1-\lambda}{1-\lambda_\rho(t)}\right)$ is the Kullback-Leibler distance between the old and new weights, the second term on the right hand side of (2.3) is the first order Taylor's approximation of $(d(t) - \boldsymbol{w}^T\boldsymbol{y}(t))^2$ around $\lambda = \lambda_\rho(t)$, measuring the "fit" of the new weight to the data, $\boldsymbol{w} = [\lambda \;\; 1-\lambda]^T$, $\boldsymbol{w}_\rho(t) \stackrel{\triangle}{=} [\lambda_\rho(t) \;\; 1-\lambda_\rho(t)]^T$ and $e(t) = d(t) - \boldsymbol{w}_\rho^T(t)\boldsymbol{y}(t)$. We show in Appendix A that the update on $\lambda_\rho(t)$ in (2.4) can be written as

$$\lambda_\rho(t) = \frac{1}{1+\exp[-\rho(t)]}, \tag{2.5}$$

with

$$\begin{aligned}\rho(t+1) &= \rho(t) + \mu_\rho e(t)(\hat{d}_1(t) - \hat{d}_2(t)) \\ &= \rho(t) + \mu_\rho\{\lambda_\rho(t)e_{a,1}(t) + [1-\lambda_\rho(t)]e_{a,2}(t) + n(t)\}[e_{a,2}(t) - e_{a,1}(t)].\end{aligned} \tag{2.6}$$

We note that the update in (2.6) is similar to the update in (2.2) without the extra $[\lambda(t)(1-\lambda(t))]$ multiplier in (2.2). In [1], it is pointed out that the update in (2.2) may slow down when $\lambda(t)$ is too close to 0 or 1 due to $[\lambda(t)(1-\lambda(t))]$. As a possible remedy to this problem, $\lambda(t)$ is restricted to an interval excluding 0 and 1 [1]. Note that this problem is

not present in (2.6).

2.2.3 Algorithm 3

The third update uses a performance-based mixture of the component filters and has deep roots in computational learning theory [13, 31]. Here, the combination weights are selected as certain functions of the accumulated loss of each constituent filter as

$$\lambda_\epsilon(t) = \frac{\exp\{-\mu_\epsilon \sum_{i=1}^{t-1}[a^{(t-1-i)}e_1^2(i)]\}}{\exp\{-\mu_\epsilon \sum_{i=1}^{t-1}[a^{(t-1-i)}e_1^2(i)]\} + \exp\{-\mu_\epsilon \sum_{i=1}^{t-1}[a^{(t-1-i)}e_2^2(i)]\}}, \tag{2.7}$$

where $0 < a \leq 1$. As shown in Appendix A, the same update on $\lambda_\epsilon(t)$ can be written as

$$\lambda_\epsilon(t) = \frac{1}{1+\exp[-\epsilon(t)]}, \tag{2.8}$$

with

$$\epsilon(t+1) = a\epsilon(t) + \mu_\epsilon(e_{a,2}(t) - e_{a,1}(t))(e_1(t) + e_2(t)). \tag{2.9}$$

2.2.4 Algorithm 4

The fourth update we investigate is studied in [27]. Here, the combination weight is given by

$$\lambda_\gamma(t) = \frac{[\sum_{n=0}^{M-1} e_1^2(t-n)]^{-\frac{M}{2}}}{[\sum_{n=0}^{M-1} e_1^2(t-n)]^{-\frac{M}{2}} + [\sum_{n=0}^{M-1} e_2^2(t-n)]^{-\frac{M}{2}}}, \tag{2.10}$$

where M is the time window to evaluate the performance-based weighting. We show in Appendix A that the same update on $\lambda_\gamma(t)$ can be written as

$$\lambda_\gamma(t) = \frac{1}{1+\exp[-\gamma(t)]},$$

where

$$\gamma(t) \triangleq \frac{M}{2} \ln\left[\frac{\sum_{n=0}^{M-1} e_2^2(t-n)}{\sum_{n=0}^{M-1} e_1^2(t-n)}\right]. \tag{2.11}$$

2.3 Steady State Analysis of the Convexly Constrained Mixtures

In this section, we present steady-state analysis of the convexly constrained mixture methods. The *a priori* error of the combination filter is $e_a(t) = \lambda(t)e_{a,1}(t) + (1-\lambda(t))e_{a,2}(t)$. If $J_{\text{ex}}(t) \triangleq E[e_a^2(t)]$, then we get

$$J_{\text{ex}}(t) = E[\lambda^2(t)e_{a,1}^2(t) + (1-\lambda(t))^2 e_{a,2}^2(t) + 2\lambda(t)(1-\lambda(t))e_{a,1}(t)e_{a,2}(t)]$$

and $J_{\text{ex}} \triangleq \lim_{t\to\infty} J_{\text{ex}}(t)$. Without loss of generality, we assume that $J_{\text{ex},1} < J_{\text{ex},2}$ in the following. Hence, for each algorithm, we have two separate cases depending on the relative value of $J_{\text{ex},12}$, i.e., $J_{\text{ex},1} \leq J_{\text{ex},12} < J_{\text{ex},2}$ or $J_{\text{ex},12} < J_{\text{ex},1} < J_{\text{ex},2}$, to investigate the steady-state behavior.

2.3.1 Steady State Analysis of Algorithm 2

The derivations follow as in [1]. Here, we first obtain an expression for the adaptation parameter in the steady-state. If $\bar{\lambda}_\rho(t) \triangleq E[\lambda_\rho(t)]$, then, as $t \to \infty$, we get

$$E[\rho(t+1)] = E[\rho(t)] + \mu_\rho(1-\bar{\lambda}_\rho(t))\Delta J_2(t) - \mu_\rho\bar{\lambda}_\rho(t)\Delta J_1(t), \tag{2.12}$$

after some algebra, where we assume that $\lambda_\rho(t)$ and $e_{a,i}(t)$ are independent in the steady-state for $i = 1, 2$ [1]. Furthermore, under the assumption of zero variance for $\lambda_\rho(t)$ as $t \to \infty$ [1], we get

$$J_{\text{ex}} = \bar{\lambda}_\rho^2 J_{\text{ex},1} + (1-\bar{\lambda}_\rho)^2 J_{\text{ex},2} + 2\bar{\lambda}_\rho(1-\bar{\lambda}_\rho)J_{\text{ex},12}, \tag{2.13}$$

where

$$\bar{\lambda}_\rho \triangleq \lim_{t\to\infty} E[\lambda_\rho(t)].$$

Depending on variances and cross correlation of the *a priori* errors, we have two cases:

a) $J_{\text{ex},1} \leq J_{\text{ex},12} < J_{\text{ex},2}$:

Here, we have $\Delta J_1 \leq 0$ and $\Delta J_2 > 0$ so that the term

$$(1-\bar{\lambda}_\rho(t))\Delta J_2(t) - \bar{\lambda}_\rho(t)\Delta J_1(t)$$

is positive since $1 > \bar{\lambda}_\rho(t) > 0$ for all t. Then, we get $E[\rho(t)] \to \infty$ as $t \to \infty$. This implies that $\rho(t) \to \infty$ and

$$\rho(t) \to \infty,$$
$$\lambda_\rho(t) \to 1$$

almost surely as $t \to \infty$ so that

$$J_{\text{ex}} = J_{\text{ex},1}.$$

That is, in this case, the combination performs as well as the best component filter. In addition, since we have

$$\lim_{t\to\infty} E[\boldsymbol{w}_\rho(t)] = [1\ 0]^T,$$

we conclude that the combination vector $\boldsymbol{w}_\rho(t)$ converges to the optimal weight vector $\boldsymbol{w}_{\text{o,c}}$ under the convex constraint.

b) $J_{\text{ex},12} < J_{\text{ex},1} < J_{\text{ex},2}$:

We have $\Delta J_i > 0, i = 1, 2$. As $t \to \infty$, a stationary point of (2.12) may be characterized by

$$(1 - \bar{\lambda}_\rho(t))\Delta J_2(t) = \bar{\lambda}_\rho(t)\Delta J_1(t),$$

so that

$$\bar{\lambda}_\rho = \frac{\Delta J_2}{\Delta J_1 + \Delta J_2}.$$

If we substitute $\bar{\lambda}_\rho$ in (2.13), then we get

$$J_{\text{ex}} = J_{\text{ex},12} + \frac{\Delta J_1 \Delta J_2}{\Delta J_1 + \Delta J_2},$$

after some algebra. Using

$$0 < \frac{\Delta J_i}{\Delta J_1 + \Delta J_2} < 1$$

yields

$$J_{\text{ex}} < \min\{J_{\text{ex},1}, J_{\text{ex},2}\}.$$

Thus, the combination filter outperforms both of the constituent filters. In addition,

since we have

$$\lim_{t\to\infty} E[\boldsymbol{w}_\rho(t)] = \left[\begin{array}{cc} \frac{\Delta J_2}{\Delta J_1+\Delta J_2} & \frac{\Delta J_1}{\Delta J_1+\Delta J_2} \end{array} \right]^T,$$

the combination weight $\boldsymbol{w}_\rho(t)$ converges to the optimal weight vector $\boldsymbol{w}_{\mathrm{o,c}}$ under the convex constraint.

Hence, the combination filter is universal with respect to the constituent filters and its weight vector converges to its optimal value.

2.3.2 Steady State Analysis of Algorithm 3

To obtain an expression for the adaptation parameter in the steady-state, we use

$$\begin{aligned} E[\epsilon(t+1)] &= aE[\epsilon(t)] + \mu_\epsilon E[(e_{a,2}(t) - e_{a,1}(t))(e_1(t) + e_2(t))] \\ &= aE[\epsilon(t)] + \mu_\epsilon (J_{\mathrm{ex},2}(t) - J_{\mathrm{ex},1}(t)), \end{aligned} \tag{2.14}$$

where we assume that $e_{a,i}(t)$ and $n(t)$ are independent for $i = 1, 2$ [28]. Along with the configuration of EMSEs, we need to consider also $0 < a < 1$ and $a = 1$ cases separately.

a) $0 < a < 1$:

For convergence of (2.14), if

$$d(t) \stackrel{\triangle}{=} \sum_{i=0}^{t} a^{t-i}(J_{\mathrm{ex},2}(i) - J_{\mathrm{ex},1}(i)) - \sum_{i=0}^{t} a^{t-i}(J_{\mathrm{ex},2} - J_{\mathrm{ex},1}),$$

then we recognize that

$$d(t+1) = ad(t) + b(t+1) - bi$$

so

$$|d(t+1)| \leq a|d(t)| + |(J_{\mathrm{ex},2}(t) - J_{\mathrm{ex},1}(t)) - (J_{\mathrm{ex},2} - J_{\mathrm{ex},1})|$$

by the triangular inequality where $b(t) \stackrel{\triangle}{=} J_{\mathrm{ex},2}(t) - J_{\mathrm{ex},1}(t)$ and $b \stackrel{\triangle}{=} J_{\mathrm{ex},2} - J_{\mathrm{ex},1}$. We show in Appendix A that

$$d(t) \to 0$$

as $t \to \infty$ so that

$$\lim_{t\to\infty} E[\epsilon(t)] = \frac{\mu_\epsilon(J_{\text{ex},2} - J_{\text{ex},1})}{1-a}. \tag{2.15}$$

The final EMSE of the combination filter is

$$J_{\text{ex}} = \bar{\lambda}_\epsilon^2 J_{\text{ex},1} + (1-\bar{\lambda}_\epsilon)^2 J_{\text{ex},2} + 2(1-\bar{\lambda}_\epsilon)\bar{\lambda}_\epsilon J_{\text{ex},12},$$

under the assumption of zero variance for $\lambda_\epsilon(t)$ as $t \to \infty$ [1] for any given a where

$$\bar{\lambda}_\epsilon \stackrel{\triangle}{=} \lim_{t\to\infty} E[\lambda_\epsilon(t)].$$

Note that (2.15) does not depend on $J_{\text{ex},12}$. Depending on the variances and the cross-EMSE of the *a priori* errors, there are two sub-cases:

a.1) $J_{\text{ex},1} \leq J_{\text{ex},12} < J_{\text{ex},2}$:

Under this configuration, the optimal combination parameter λ in (2.1) is equal to 1 and the EMSE of the optimal combination filter is $J_{\text{ex},1}$. Hence, for the combination filter to achieve the performance of the best constituent filter, we need to have $\lambda_\epsilon = 1$, i.e., $E[\epsilon(t)] \to \infty$ as $t \to \infty$, which is true if and only if $a = 1$. For any a, the difference between the EMSEs of the combination filter and the best constituent filter is

$$\begin{aligned} f(\bar{\lambda}_\epsilon) &\stackrel{\triangle}{=} J_{\text{ex}} - J_{\text{ex},1} \\ &= (1-\bar{\lambda}_\epsilon)[(1+\bar{\lambda}_\epsilon)(J_{\text{ex},12} - J_{\text{ex},1}) + (1-\bar{\lambda}_\epsilon)(J_{\text{ex},2} - J_{\text{ex},12})] \geq 0 \end{aligned} \tag{2.16}$$

where the equality is reached if and only if $a = 1$ so that the update (2.9) does not achieve the performance of the best constituent filter if $a \neq 1$.

a.2) $J_{\text{ex},12} < J_{\text{ex},1} < J_{\text{ex},2}$:

Here, the difference between the EMSEs of the combination filter and the best constituent filter is, i.e., $f(\bar{\lambda}_\epsilon)$ in (2.16), a convex function of $\bar{\lambda}_\epsilon$ with roots $\frac{\Delta J_2 - \Delta J_1}{\Delta J_2 + \Delta J_1}$ and 1. Hence, for $\bar{\lambda}_\epsilon \in (\frac{\Delta J_2 - \Delta J_1}{\Delta J_2 + \Delta J_1}, 1)$, $f(\cdot)$ is negative, i.e., $J_{\text{ex}} < J_{\text{ex},1}$. We note that

$$\bar{\lambda}_\epsilon \in \left(\frac{\Delta J_2 - \Delta J_1}{\Delta J_2 + \Delta J_1}, 1\right)$$

if and only if

$$a \in \left(1 + \mu_\epsilon \frac{\Delta J_2 - \Delta J_1}{\ln\left(\frac{2\Delta J_1}{\Delta J_2 - \Delta J_1}\right)}, 1\right)$$

assuming that

$$\mu_\epsilon \frac{\Delta J_2 - \Delta J_1}{\ln\left(\frac{2\Delta J_1}{\Delta J_2 - \Delta J_1}\right)} < 0.$$

Then, the combination filter outperforms the constituent filters for any $a \in \left(1 + \mu_\epsilon \frac{\Delta J_2 - \Delta J_1}{\ln\left(\frac{2\Delta J_1}{\Delta J_2 - \Delta J_1}\right)}, 1\right)$.

b) $a = 1$:

We have

$$E[\epsilon(t+1)] = E[\epsilon(t)] + K(t),$$

where

$$K(t) \stackrel{\triangle}{=} \mu_\epsilon (J_{\text{ex},2}(t) - J_{\text{ex},1}(t))$$

converges to a positive constant since $J_{\text{ex},1} < J_{\text{ex},2}$ so that $E[\epsilon(t)] \to \infty$ as $t \to \infty$. This implies that

$$\epsilon(t) \to \infty,$$
$$\lambda_\epsilon(t) \to 1$$

almost surely as $t \to \infty$ so that

$$J_{\text{ex}} = J_{\text{ex},1}.$$

Thus, the combination filter performs as well as the best component filter. The final combination weight vector is

$$\lim_{t\to\infty} E[\boldsymbol{w}_\epsilon(t)] = [1\ 0]^T.$$

Hence, $a = 1$ is a necessary condition for the combination filter to achieve the performance of the best constituent filter. Note that when $a \neq 1$, the combination filter may outperform the constituent filters in specific configurations of EMSEs. However, if the cross EMSE is

$J_{\text{ex},12} > J_{\text{ex},1}$ and $a \neq 1$, then the combination performs worse than the best constituent filter. Hence, unlike [1], the algorithm of [30] achieves (but not outperforms) the best constituent filter when $a = 1$ and if $a \neq 1$, then the algorithm may outperform or perform worse than the best constituent filter depending on the configuration of EMSEs. Moreover, the weight vector convergence does not appear.

2.3.3 Steady State Analysis of Algorithm 4

To get the steady-state behavior, we use

$$E[\lambda_\gamma(t)] \approx \frac{1}{1 + E\left[\frac{\sum_{n=0}^{M-1} e_2^2(t-n)}{\sum_{n=0}^{M-1} e_1^2(t-n)}\right]^{-\frac{M}{2}}} \approx \frac{1}{1 + \left[\frac{\sum_{n=0}^{M-1} E[e_2^2(t-n)]}{\sum_{n=0}^{M-1} E[e_1^2(t-n)]}\right]^{-\frac{M}{2}}}. \tag{2.17}$$

We emphasize that although the approximations in (2.17) are strong especially for small M, we observe a close agreement with our simulations for relatively large M, e.g., $M > 30$. Since as $t \to \infty$, $E[e_i^2(t)] \to J_{\text{ex},i} + \sigma_n^2$ for $i = 1, 2$, we get

$$\bar{\lambda}_\gamma \stackrel{\triangle}{=} \lim_{t\to\infty} E[\lambda_\gamma(t)] = \frac{1}{1 + \left[\frac{J_{\text{ex},2}+\sigma_n^2}{J_{\text{ex},1}+\sigma_n^2}\right]^{-\frac{M}{2}}}, \tag{2.18}$$

and the final EMSE of the combination filter is $J_{\text{ex}} = \bar{\lambda}_\gamma^2 J_{\text{ex},1} + (1-\bar{\lambda}_\gamma)^2 J_{\text{ex},2} + 2(1-\bar{\lambda}_\gamma)\bar{\lambda}_\gamma J_{\text{ex},12}$ for any given M under the assumption of zero variance for $\lambda_\gamma(t)$ in the steady-state [1]. Depending on M, we have two cases:

a) $M \to \infty$: Since we have $(J_{\text{ex},2} + \sigma_n^2)/(J_{\text{ex},1} + \sigma_n^2) > 1$, we get $\lim_{t\to\infty} E[\lambda_\gamma(t)] = 1$. Hence,

$$J_{\text{ex}} = J_{\text{ex},1}. \tag{2.19}$$

Thus, the combination filter performs as well as the best constituent filter. The final combination weight vector is $\lim_{t\to\infty} E[\boldsymbol{w}_\gamma(t)] = [1 \;\; 0]^T$.

b) $M < \infty$: Depending on the *a priori* errors and the cross-EMSE between the component filters, there are two subcases:

b.1) $J_{\text{ex},1} \leq J_{\text{ex},12} < J_{\text{ex},2}$: In this case, the optimal combination parameter λ in (2.1) is 1 and the EMSE of the optimal combination filter is $J_{\text{ex},1}$. The combination

filter achieves the performance of the best constituent filter if $E[\gamma(t)] \to \infty$ as $t \to \infty$ if and only if $M \to \infty$. The difference between the EMSEs of the combination filter and the best constituent filter is

$$J_{\text{ex}} - J_{\text{ex},1} = (1 - \bar{\lambda}_\gamma)[(1 + \bar{\lambda}_\gamma)(J_{\text{ex},12} - J_{\text{ex},1}) + (1 - \bar{\lambda}_\gamma)(J_{\text{ex},2} - J_{\text{ex},12})] \geq 0,$$

where the equality is reached if and only if $M \to \infty$ so that the algorithm does not achieve the performance of the best constituent filter if $M < \infty$.

b.2) $J_{\text{ex},12} < J_{\text{ex},1} < J_{\text{ex},2}$: In this case, the difference between the EMSEs of the combination filter and the best constituent filter, i.e., $f(\bar{\lambda}_\gamma)$ in (2.16), is negative for $M \in (2\frac{\log(2\Delta J_1) - \log(\Delta J_2 - \Delta J_1)}{\log(J_{\text{ex},1}+\sigma_n^2) - \log(J_{\text{ex},2}+\sigma_n^2)}, \infty)$ so that the combination filter outperforms the constituent filters, i.e., $J_{\text{ex}} < J_{\text{ex},1}$.

Hence, $M \to \infty$ is a necessary condition for the combination filter to perform as well as the best constituent filter. The combination filter using the update rule (2.10) with $M < \infty$ may outperform the constituent filters in certain configurations of the EMSEs. However, if the cross EMSE is sufficiently large, then the combination filter performs worse than the best component filter when $M < \infty$. Hence, unlike [1], update (2.10) achieves (but not outperforms) the best constituent filter when $M \to \infty$ and if $M < \infty$, then the algorithm may outperform or perform worse than the best constituent filter depending on the configuration of EMSEs. Moreover, it does not offer the desirable weight vector convergence.

2.4 Transient Analysis of the Convexly Constrained Mixtures

In this section, we perform mean and mean-square convergence analysis of the studied algorithms. The derivations follow [11]. We use the following assumptions [11]:

- **A1)** $n(t)$ is independent of $\boldsymbol{u}(t)$.
- **A2)** $\rho(t), \epsilon(t), \gamma(t)$ vary slowly enough so that

$$E[e_{a,i}^k(t) e_{a,j}^l(t) h(t) | h(t)] = E[e_{a,i}^k(t) e_{a,j}^l(t)] h(t),$$

where $h(t) \in \{\rho(t), \epsilon(t), \gamma(t)\}$, $i, j = 1, 2$ and $k, l = 0, .., 4$, $k + l \leq 4$.

- **A3)** $e_{a,1}(t)$ and $e_{a,2}(t)$ are jointly Gaussian and zero mean, implying [11]

$$\begin{aligned}
&E[e_{a,i}^4(t)] = 3J_{\text{ex},i}^2(t), \quad i = 1, 2, \\
&E[e_{a,1}^k(t)e_{a,2}^l(t)] = 0, \quad k + l = 3, \\
&E[e_{a,1}^k(t)e_{a,2}^l(t)] = 3J_{\text{ex},1}(t)J_{\text{ex},12}(t), \quad k = 3, l = 1, \\
&E[e_{a,1}^k(t)e_{a,2}^l(t)] = 3J_{\text{ex},12}(t)J_{\text{ex},2}(t), \quad k = 1, l = 3, \\
&E[e_{a,1}^k(t)e_{a,2}^l(t)] = 2J_{\text{ex},12}^2(t) + J_{\text{ex},1}(t)J_{\text{ex},2}(t), \quad k = l = 2.
\end{aligned}$$

2.4.1 Transient Analysis of Algorithm 2

The update (2.6) can be written as

$$\begin{aligned}
\rho(t+1) = \rho(t) + \mu_\rho \big[&-\lambda_\rho(t)e_{a,1}^2(t) + (1 - \lambda_\rho(t))e_{a,2}^2(t) \\
&+ (2\lambda_\rho(t) - 1)e_{a,1}(t)e_{a,2}(t) + n(t)(e_{a,2}(t) - e_{a,1}(t))\big]. \qquad (2.20)
\end{aligned}$$

The first order Taylor's approximation of

$$\lambda_\rho(\rho(t)) \stackrel{\triangle}{=} 1/(1 + \exp(-\rho(t)))$$

around $\bar{\rho}(t) \stackrel{\triangle}{=} E[\rho(t)]$ is given by

$$\begin{aligned}
\lambda_\rho(\rho(t)) &\approx \lambda_\rho(\bar{\rho}(t)) + \frac{d\lambda_\rho}{d\rho(t)}(\bar{\rho}(t))(\rho(t) - \bar{\rho}(t)) \\
&= \bar{\lambda}_\rho(t) + \bar{\lambda}_\rho(t)(1 - \bar{\lambda}_\rho(t))(\rho(t) - \bar{\rho}(t)), \qquad (2.21)
\end{aligned}$$

where $\bar{\lambda}_\rho(t) \stackrel{\triangle}{=} \lambda_\rho(\bar{\rho}(t))$. Using (2.21) in (2.20) yields

$$\begin{aligned}
\rho(t+1) = \rho(t) + \mu_\rho \big[&-(\bar{\lambda}_\rho(t) + \bar{\lambda}_\rho(t)(1 - \bar{\lambda}_\rho(t))(\rho(t) - \bar{\rho}(t)))e_{a,1}^2(t) \\
&+ (1 - \bar{\lambda}_\rho(t) - \bar{\lambda}_\rho(t)(1 - \bar{\lambda}_\rho(t))(\rho(t) - \bar{\rho}(t)))e_{a,2}^2(t) \\
&+ (2\bar{\lambda}_\rho(t) + 2\bar{\lambda}_\rho(t)(1 - \bar{\lambda}_\rho(t))(\rho(t) - \bar{\rho}(t)) - 1)e_{a,1}(t)e_{a,2}(t) \\
&+ n(t)(e_{a,2}(t) - e_{a,1}(t))\big]. \qquad (2.22)
\end{aligned}$$

Taking the expectation of (2.22) and using A1, A2 yields

$$\bar{\rho}(t+1) = \bar{\rho}(t) + \mu_\rho \left[-\bar{\lambda}_\rho(t) J_{\text{ex},1}(t) + (1-\bar{\lambda}_\rho(t)) J_{\text{ex},2}(t) + (2\bar{\lambda}_\rho(t) - 1) J_{\text{ex},12}(t) \right]. \quad (2.23)$$

Moreover, by using (2.21) in

$$e_a(t) = \lambda_\rho(t) e_{a,1}(t) + (1-\lambda_\rho(t)) e_{a,2}(t),$$

we get

$$e_a(t) = (\bar{\lambda}_\rho(t) + \bar{\lambda}_\rho(t)(1-\bar{\lambda}_\rho(t))(\rho(t) - \bar{\rho}(t)))(e_{a,1}(t) - e_{a,2}(t)) + e_{a,2}(t), \quad (2.24)$$

which yields $E[e_a(t)] = 0$ using A1 and A2. We next find the EMSE of the combination filter by squaring (2.24) and taking the expectation, yielding

$$\begin{aligned} E[e_a^2(t)] = & \left[\bar{\lambda}_\rho^2(t) + \sigma_\rho^2(t) \bar{\lambda}_\rho^2(t) (1-\bar{\lambda}_\rho(t))^2 \right] \left[J_{\text{ex},1}(t) + J_{\text{ex},2}(t) - 2J_{\text{ex},12}(t) \right] \\ & + 2\bar{\lambda}_\rho(t)(J_{\text{ex},12}(t) - J_{\text{ex},2}(t)) + J_{\text{ex},2}(t), \end{aligned} \quad (2.25)$$

where $\sigma_\rho^2(t) \stackrel{\triangle}{=} E[(\rho(t) - \bar{\rho}(t))^2]$ with A1 and A2. To evaluate (2.25), we need have $\sigma_\rho^2(t)$. To obtain a recursion for $\sigma_\rho^2(t)$, we square (2.22), take the expected value and subtract the square of (2.23), yielding, using A1, A2 and A3, after straightforward algebra,

$$\sigma_\rho^2(t+1) = (1 + 2\mu_\rho G_1(t) + \mu_\rho^2 G_2(t)) \sigma_\rho^2(t) + {\mu_\rho}^2 F(t), \quad (2.26)$$

where, omitting t,

$$\begin{aligned} F = & 2(1-\bar{\lambda}_\rho)^2 J_{\text{ex},2}^2 + (2\bar{\lambda}_\rho - 1)^2 [J_{\text{ex},12}^2 + J_{\text{ex},1} J_{\text{ex},2}] + 2\bar{\lambda}_\rho^2 J_{\text{ex},1}^2 \\ & + 4(2\bar{\lambda}_\rho - 1)(1-\bar{\lambda}_\rho) J_{\text{ex},12} J_{\text{ex},2} - 4\bar{\lambda}_\rho (1-\bar{\lambda}_\rho) J_{\text{ex},12}^2 \\ & - 4\bar{\lambda}_\rho (2\bar{\lambda}_\rho - 1) J_{\text{ex},1} J_{\text{ex},12} + (J_{\text{ex},1} + J_{\text{ex},2} - 2J_{\text{ex},12}) \sigma_n^2, \end{aligned} \quad (2.27)$$

$$G_1 = -\bar{\lambda}_\rho (1-\bar{\lambda}_\rho)[J_{\text{ex},1} + J_{\text{ex},2} - 2J_{\text{ex},12}], \quad (2.28)$$

$$G_2 = 3\bar{\lambda}_\rho^2 (1-\bar{\lambda}_\rho)^2 \left[J_{\text{ex},1}^2 + 2(J_{\text{ex},1} J_{\text{ex},2} + 2J_{\text{ex},12}^2) - 4J_{\text{ex},1} J_{\text{ex},12} - 4J_{\text{ex},12} J_{\text{ex},2} + J_{\text{ex},2}^2 \right]. \quad (2.29)$$

Here, we analyze the bias/variance relation of Algorithm 2. From (2.23), when the step size is large, the combination filter could better track the constituent filters. However, a larger step size may cause $\sigma_\rho^2(t)$ to be large so that the EMSE of the combination filter (2.25) may become unstable during the initial iterations. Note that from (2.26) when $\bar{\lambda}_\rho \triangleq \lim_{t\to\infty} \bar{\lambda}_\rho(t) = 0$ or 1, $\sigma_\rho^2(t)$ is unbounded as $t \to \infty$ since $G_1 \triangleq \lim_{t\to\infty} G_1(t) = 0$, $G_2 \triangleq \lim_{t\to\infty} G_2(t) = 0$ and $F \triangleq \lim_{t\to\infty} F(t) > 0$. However, in our simulations, we observe that $\bar{\lambda}_\rho^2(t)(1-\bar{\lambda}_\rho(t))^2$ converges to 0 faster than $\sigma_\rho^2(t)$ goes to infinity so that the term

$$\lim_{t\to\infty} \sigma_\rho^2(t)\bar{\lambda}_\rho^2(t)(1-\bar{\lambda}_\rho(t))^2 = 0$$

in (2.25). Hence, the effect of the variance of the combination parameter on the EMSE of the combination filter diminishes in the steady-state when $\bar{\lambda}_\rho = 0$ or 1 so that the EMSE of the combination filter converges to the EMSE of the best constituent filter in the mean and the MSE senses. When $\bar{\lambda}_\rho = \frac{\Delta J_2}{\Delta J_1 + \Delta J_2}$, we observe from (2.26) that $\sigma_\rho^2(t)$ converges when $|1 + 2\mu_\rho G_1(t) + \mu_\rho^2 G_2(t)| < 1$ for all t, i.e., $-2 < 2\mu_\rho G_1(t) + \mu_\rho^2 G_2(t) < 0$ and under this condition

$$\sigma_\rho^2 \triangleq \lim_{t\to\infty} \sigma_\rho^2(t) = -\frac{\mu_\rho F}{2G_1 + \mu_\rho G_2}.$$

We observe from (2.27) that $F > 0$ and from (2.28), (2.29) that $2G_1 + \mu_\rho G_2 < 0$ when $-2 < 2\mu_\rho G_1(t) + \mu_\rho^2 G_2(t) < 0$ for all t so that $\sigma_\rho^2 > 0$ and the term $\sigma_\rho^2(t)\bar{\lambda}_\rho^2(t)(1-\bar{\lambda}_\rho(t))^2$ in (2.25) converges. Hence, from (2.25), there is a bias term

$$\sigma_\rho^2 \bar{\lambda}_\rho^2 (1-\bar{\lambda}_\rho)^2 (\Delta J_1 + \Delta J_2)$$

in the EMSE of the combination filter in the steady-state which introduces a bias/variance trade-off as in the stochastic gradient algorithms [28], e.g., the trade-off between the bias and the step size of LMS algorithm. Since all the terms in (2.27), (2.28) and (2.29) can be calculated (recursively), this concludes the transient analysis of Algorithm 2.

2.4.2 Transient Analysis of Algorithm 3

The update rule for $\epsilon(t)$ can be written as

$$\epsilon(t+1) = a\epsilon(t) + \mu_\epsilon \left[e_{a,2}^2(t) - e_{a,1}^2(t) + 2n(t)(e_{a,2}(t) - e_{a,1}(t)) \right], \tag{2.30}$$

yielding

$$\bar{\epsilon}(t+1) = a\bar{\epsilon}(t) + \mu_\epsilon \left[J_{\text{ex},2}(t) - J_{\text{ex},1}(t)\right], \tag{2.31}$$

with A1. We next use the first order Taylor's approximation of $\lambda_\epsilon(t)$ around the expected value $\bar{\epsilon}(t) \stackrel{\triangle}{=} E[\epsilon(t)]$ as $\lambda_\epsilon(t) \approx \bar{\lambda}_\epsilon(t) + \bar{\lambda}_\epsilon(t)(1-\bar{\lambda}_\epsilon(t))(\epsilon(t) - \bar{\epsilon}(t))$, where $\bar{\lambda}_\epsilon(t) \stackrel{\triangle}{=} \lambda(\bar{\epsilon}(t))$. Applying this to $e_a(t) = \lambda_\epsilon(t)e_{a,1}(t) + (1-\lambda_\epsilon(t))e_{a,2}(t)$ yields

$$e_a(t) = [\bar{\lambda}_\epsilon(t) + \bar{\lambda}_\epsilon(t)(1-\bar{\lambda}_\epsilon(t)(\epsilon(t) - \bar{\epsilon}(t)))][e_{a,1}(t) - e_{a,2}(t)] + e_{a,2}(t), \tag{2.32}$$

and $E[e_a(t)] = 0$ with A1 and A2. We obtain EMSE of the combination filter by squaring (2.32) and taking the expectation as

$$\begin{aligned} E[e_a^2(t)] = &\left[\bar{\lambda}_\epsilon^2(t) + \sigma_\epsilon^2(t)\bar{\lambda}_\epsilon^2(t)(1-\bar{\lambda}_\epsilon(t))^2\right]\left[J_{\text{ex},1}(t) + J_{\text{ex},2}(t) - 2J_{\text{ex},12}(t)\right] \\ &+ 2\bar{\lambda}_\epsilon(t)(J_{\text{ex},12}(t) - J_{\text{ex},2}(t)) + J_{\text{ex},2}(t), \end{aligned} \tag{2.33}$$

where $\sigma_\epsilon^2(t) \stackrel{\triangle}{=} E[(\epsilon(t) - \bar{\epsilon}(t))^2]$ is the variance of $\epsilon(t)$ using A1 and A2. To obtain a recursion for $\sigma_\epsilon^2(t)$, we square (2.30), take expectation and then subtract the square of (2.31). This yields, using A1, A2 and A3,

$$\sigma_\epsilon^2(t+1) = a^2\sigma_\epsilon^2(t) + 2{\mu_\epsilon}^2\left[J_{\text{ex},1}^2(t) + J_{\text{ex},2}^2(t) - 2J_{\text{ex},12}^2(t) + 2\sigma_n^2(J_{\text{ex},1}(t) + J_{\text{ex},2}(t) - 2J_{\text{ex},12}(t))\right]. \tag{2.34}$$

Here, we analyze the bias/variance relation of Algorithm 3. From (2.31), the combination filter could better track the constituent filters when the step size is large. However, a larger step size may cause $\sigma_\epsilon^2(t)$ to be large so that the EMSE of the combination filter (2.33) may become unstable during the initial iterations. When $0 < a < 1$, we have $\bar{\epsilon} \stackrel{\triangle}{=} \lim_{t\to\infty} \bar{\epsilon}(t) = \dfrac{\mu_\epsilon(J_{\text{ex},2} - J_{\text{ex},1})}{1-a}$ and $\bar{\lambda}_\epsilon \stackrel{\triangle}{=} \lim_{t\to\infty} \bar{\lambda}_\epsilon(t)$. From (2.34), $\sigma_\epsilon^2(t)$ converges and

$$\sigma_\epsilon^2 \stackrel{\triangle}{=} \lim_{t\to\infty} \sigma_\epsilon^2(t) = \frac{2{\mu_\epsilon}^2\left[J_{\text{ex},1}^2 + J_{\text{ex},2}^2 - 2J_{\text{ex},12}^2 + 2\sigma_n^2(J_{\text{ex},1} + J_{\text{ex},2} - 2J_{\text{ex},12})\right]}{1-a^2}.$$

Hence, the term $\sigma_\epsilon^2(t)\bar{\lambda}_\epsilon^2(t)(1-\bar{\lambda}_\epsilon(t))^2$ in (2.33) converges. Note that from (2.33) this term introduces a bias in the EMSE of the combination filter in the steady-state. When $a = 1$, it follows from (2.31) that $\bar{\lambda}_\epsilon = 0$ or 1. From (2.34), $\sigma_\epsilon^2(t)$ is unbounded as $t \to \infty$. However,

in our simulations, we observe that $\bar{\lambda}_\epsilon^2(t)(1-\bar{\lambda}_\epsilon(t))^2$ converges to 0 faster than $\sigma_\epsilon^2(t)$ goes to infinity so that the term

$$\lim_{t\to\infty} \sigma_\epsilon^2(t)\bar{\lambda}_\epsilon^2(t)(1-\bar{\lambda}_\epsilon(t))^2 = 0$$

in (2.33). Hence, the effect of the variance of the combination parameter on the EMSE of the combination filter diminishes in the steady-state when $a = 1$ so that the EMSE of the combination filter converges to the EMSE of the best constituent filter in the mean and the MSE senses. This concludes the transient analysis of Algorithm 3.

2.4.3 Transient Analysis of Algorithm 4

Taking expectation of (2.11) yields

$$\bar{\gamma}(t) = \frac{M}{2} E\left[\ln\left(\frac{\sum_{n=0}^{M-1} e_2^2(t-n)}{\sum_{n=0}^{M-1} e_1^2(t-n)}\right)\right] \approx \frac{M}{2}\ln\left(\frac{\sum_{n=0}^{M-1} J_{\text{ex},2}(t-n)}{\sum_{n=0}^{M-1} J_{\text{ex},1}^2(t-n)}\right).$$

If we use the first order Taylor's approximation of $\lambda_\gamma(t)$ around the expected value $\bar{\gamma}(t) \stackrel{\triangle}{=} E[\gamma(t)]$, then we get

$$\lambda_\gamma(t) \approx \bar{\lambda}_\gamma(t) + \bar{\lambda}_\gamma(t)(1-\bar{\lambda}_\gamma(t))(\gamma(t)-\bar{\gamma}(t)), \tag{2.35}$$

where $\bar{\lambda}_\gamma(t) \stackrel{\triangle}{=} \lambda(\bar{\gamma}(t))$. Using (2.35) in $e_a(t)$ yields

$$e_a(t) = \left[\bar{\lambda}_\gamma(t) + \bar{\lambda}_\gamma(t)(1-\bar{\lambda}_\gamma(t)(\gamma(t)-\bar{\gamma}(t)))\right][e_{a,1}(t) - e_{a,2}(t)] + e_{a,2}(t) \tag{2.36}$$

and $E[e_a(t)] = 0$ under A1 and A2. To get the EMSE of the combination filter, we first use the first order Taylor's approximation of $\lambda_\gamma^2(t)$ around the expected value $\bar{\gamma}(t) \stackrel{\triangle}{=} E[\gamma(t)]$ to get

$$\lambda_\gamma^2(t) \stackrel{\triangle}{=} \bar{\lambda}_\gamma^2(t) + 2\bar{\lambda}_\gamma^2(t)(1-\bar{\lambda}_\gamma(t))(\gamma(t)-\bar{\gamma}(t)). \tag{2.37}$$

Using (2.35) and (2.37) in $e_a^2(t)$ and taking expectation yields

$$E[e_a^2(t)] = \bar{\lambda}_\gamma^2(t)J_{\text{ex},1}(t) + (1-\bar{\lambda}_\gamma^2(t))^2 J_{\text{ex},2}(t) + 2\bar{\lambda}_\gamma^2(t)(1-\bar{\lambda}_\gamma^2(t))J_{\text{ex},12}(t). \tag{2.38}$$

This concludes the transient analysis of Algorithm 4.

2.5 Simulations

In this section, we present performance of the combination algorithms through simulations using the setup of [1]. Here, we have two LMS filters with the same input regressor and different step sizes running in parallel as the constituent filters with updates $\boldsymbol{w}_i(t+1) = \boldsymbol{w}_i(t) + \mu_i e_i(t)\boldsymbol{u}(t)$, for $i = 1, 2$, where $\mu_1 = 0.1$ and $\mu_2 = 0.001$. The input regressor $\boldsymbol{u}(t) \in \mathbb{R}^7$ is zero mean and i.i.d. Gaussian with variance selected to yield $\mathrm{Tr}(\boldsymbol{R})$=1, where $\mathrm{Tr}(\cdot)$ is the trace. The underlying signal is generated as $d(t) = \boldsymbol{w}_o^T(t)\boldsymbol{u}(t) + n(t)$, where $n(t)$ is the additive i.i.d Gaussian noise with variance $\sigma_n^2 = 0.01$ and $\boldsymbol{w}_o(t+1) = \boldsymbol{w}_o(t) + \boldsymbol{q}(t)$. The initial value of $\boldsymbol{w}_o(t)$ is selected as [1]

$$\boldsymbol{w}_o(0) = [0.90, -0.53, 0.21, -0.028, 0.78, 0.52, -0.08]^T.$$

Theoretical EMSEs of the combination filters and the cross-EMSE between them are given by

$$J_{\mathrm{ex},i} = \frac{\mu_i \sigma_n^2 \mathrm{Tr}(\boldsymbol{R}) + \mu_i^{-1}\mathrm{Tr}(\boldsymbol{Q})}{2 - \mu_i \mathrm{Tr}(\boldsymbol{R})},$$
$$J_{\mathrm{ex},12} = \frac{\mu_{12} \sigma_n^2 \mathrm{Tr}(\boldsymbol{R}) + 2\frac{\mathrm{Tr}(\boldsymbol{Q})}{\mu_1+\mu_2}}{2 - \mu_{12} \mathrm{Tr}(\boldsymbol{R})}$$

under the separation assumption [1], where $\mu_{12} = \frac{2\mu_1\mu_2}{\mu_1+\mu_2}$ and theoretical $J_{\mathrm{ex},i}$ attains the minimum at

$$\mu_{\mathrm{opt}} = \sqrt{\frac{\mathrm{Tr}(\boldsymbol{Q})}{\sigma_n^2 \mathrm{Tr}(\boldsymbol{R})} + \frac{[\mathrm{Tr}(\boldsymbol{Q})]^2}{4\sigma_n^4}} - \frac{\mathrm{Tr}(\boldsymbol{Q})}{2\sigma_n^2}.$$

We measure the performance using the same figure of merit as in [1]. The normalized square deviation (NSD) of the component filters and the combination filters are defined as

$$\mathrm{NSD}_i \triangleq J_{\mathrm{ex},i}/J_{\mathrm{ex,opt}},$$
$$\mathrm{NSD}_{\mathrm{alg2}} \triangleq J_{\mathrm{ex,alg2}}/J_{\mathrm{ex,opt}},$$
$$\mathrm{NSD}_{\mathrm{alg3}} \triangleq J_{\mathrm{ex,alg3}}/J_{\mathrm{ex,opt}},$$
$$\mathrm{NSD}_{\mathrm{alg4}} \triangleq J_{\mathrm{ex,alg4}}/J_{\mathrm{ex,opt}},$$

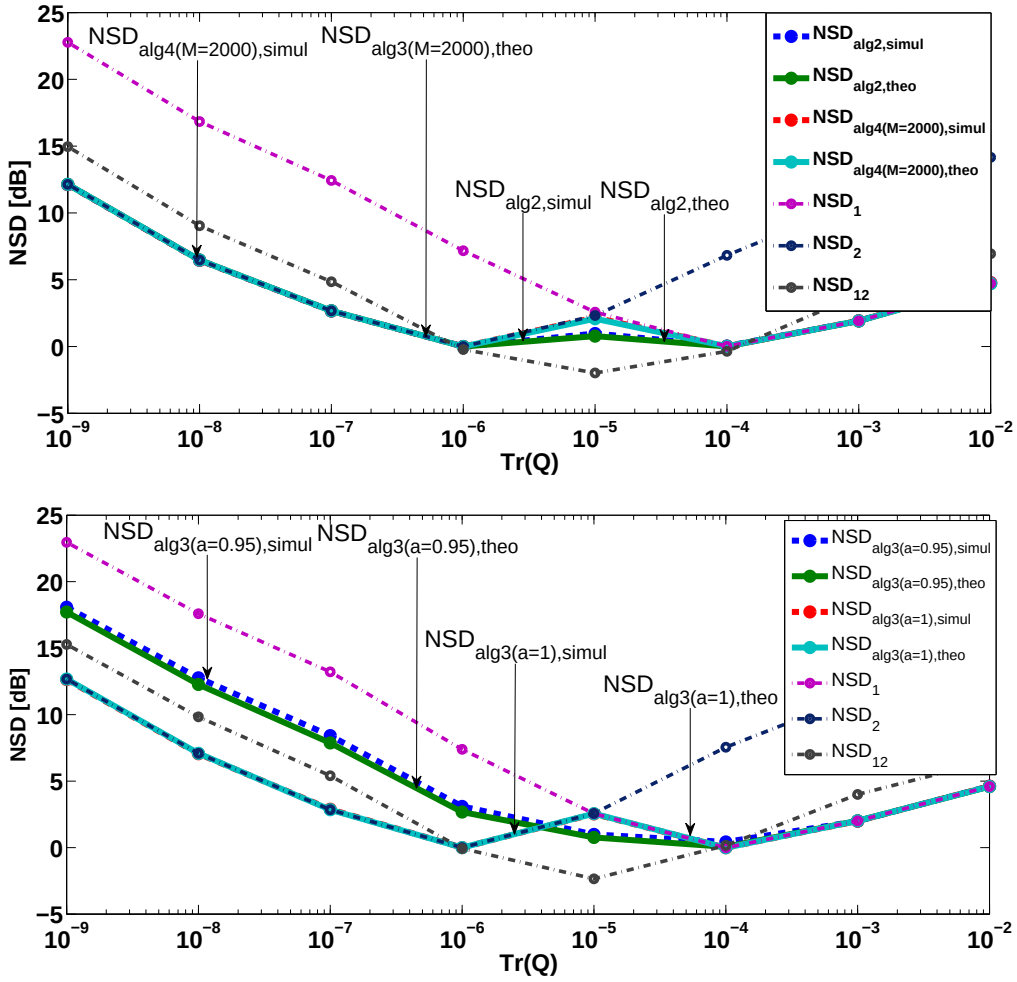


Figure 2.1: Theoretical and simulated NSDs as a function of $\mathrm{Tr}(\boldsymbol{Q})$. (a) 2nd and 3rd combination filters, $\mu_\rho = 30$, $M = 2000$. (b) 4rd combination filter, $\mu_\epsilon = 30$, $a = 1$, $a = 0.95$.

where $J_{\mathrm{ex,alg}i}$ is the EMSE of the ith combination filter and $J_{\mathrm{ex,opt}}$ is the EMSE calculated using μ_{opt}.

In Fig. 2.1, we plot the NSDs for all algorithms as a function of $\mathrm{Tr}(\boldsymbol{Q})$, $\boldsymbol{Q} = E[\boldsymbol{q}(t)\boldsymbol{q}^T(t)]$. For these simulations, the step size in (2.6) is set to $\mu_\rho = 30$ and the step size in (2.9) is set to $\mu_\epsilon = 30$ to guarantee convergence. To test our theoretical analysis on the forgetting factor in (2.9), we simulate (2.9) using $a = 1$ and $a = 0.95$. We test the update in (2.10) using a time window $M = 2000$. The simulations are done over 6x10^5 samples, averaged over 20 independent trials. The final EMSEs are calculated by averaging the last 1000 samples of each iteration.

We observe in Fig. 2.1a that the combination filter using the EG update (2.6) is universal with respect to the combination filters and even performs better than both when $J_{\text{ex},12} < \min\{J_{\text{ex},1}, J_{\text{ex},2}\}$ (as shown in Section 2.2.2). The update (2.10) with $M = 2000$ achieves the performance of the best constituent filter since $M = 2000$ is sufficiently large to yield (2.19). Similarly, the update (2.9) is also universal when $a = 1$ such that it achieves the performance of the best constituent filter for all $\text{Tr}(\boldsymbol{Q})$ in Fig. 2.1b. For the update (2.9) with $a = 0.95$, we observe that for certain $\text{Tr}(\boldsymbol{Q})$, the update performs better than both constituent filters. However, since $a \neq 1$, the update (2.9) performs worse than the best constituent filter as predicted in (2.16) and Section III.C.a.2 for certain $\text{Tr}(\boldsymbol{Q})$. For all algorithms, we observe that our steady-state analysis accurately describes the simulations.

For the simulations related to the transient analysis, the underlying signal is generated from a stationary model as $d(t) = \boldsymbol{w}_o^T \boldsymbol{u}(t) + n(t)$ [1], where $n(t)$ is the additive i.i.d noise with variance $\sigma_n^2 = 0.01$ and $\boldsymbol{w}_o = [0.24, -0.45, -0.35, 0.04, -0.17, 0.74, 0.14]^T$. Moreover, to test the switching performance, we abruptly change $\boldsymbol{w}_o$ to
$\boldsymbol{w}_o = [0.34, 0.45, -0.41, 0.46, 0.14, -0.44, -0.24]^T$ in the middle of the simulations [1]. Here, the input regressor $\boldsymbol{u}(t) \in \mathbb{R}^7$ is zero mean i.i.d. Gaussian, where the variance of each entry is set to 1. As the constituent filters, we have two LMS filters with the same input regressor and different step sizes running in parallel with updates

$$\boldsymbol{w}_i(t+1) = \boldsymbol{w}_i(t) + \mu_i e_i(t) \boldsymbol{u}(t),$$

for $i = 1, 2$, $\mu_1 = 0.15$, $\mu_2 = 0.002$. For the combination algorithms, we set $\mu_\rho = 1$ for Algorithm 2, $\mu_\epsilon = 1$ and $a = 0.98$ for Algorithm 3, $M = 200$ for the Algorithm 4. Results are averaged over 1000 independent trials. In Fig. 2.2a, we plot the MSE curve for Algorithm 2, labeled as "$\text{Alg.2}_{\text{simul}}$", the theoretical derived MSE curve using (2.25), labelled as "$\text{Alg.2}_{\text{theory}}$". In Fig. 2.2a, we also plot the theoretical derived MSE curve, where we set $\sigma_\rho(t) = 0$, labeled as "$\text{Alg.2}_{\text{theory}(\sigma_\rho=0)}$", as suggested in [11]. We observe that our analysis closely describes the transient behavior of Algorithm 2 in these simulations. We repeat the same experiment for Algorithm 3 and display the results in Fig. 2.2b. We use the same labeling as in Fig. 2.2a, however, use (2.33) to calculate the theoretical curves. We point out that since $a = 0.98$, as predicted from the steady-state analysis, the mixture does not converge to the best constituent filter as seen in Fig. 2.2b (unlike Algorithm 2

μ_α	μ_ρ	Alg. 3	Alg. 4	n (iteration index)
$\mu_\alpha = 0.5$	$\mu_\rho = 0.5$	a=0.999, $\mu_\epsilon = 0.5$	M=2000	$n_1 = 1864$, $n_2 = 1092$, $n_3 = 3055$, $n_4 = 2595$
$\mu_\alpha = 5$	$\mu_\rho = 5$	a=0.999, $\mu_\epsilon = 5$	M=1700	$n_1 = 1063$, $n_2 = 1009$, $n_3 = 3181$, $n_4 = 2336$
$\mu_\alpha = 100$	$\mu_\rho = 100$	a=0.999, $\mu_\epsilon = 100$	M=1400	$n_1 = 1063$,$n_2 = 1011$, $n_3 = 3193$, $n_4 = 2088$

Table 2.1: Performance of Algorithm 3 and Algorithm 4 for different parameters

in Fig. 2.2a). The same simulations are performed for Algorithm 4 as shown in Fig. 2.2c, however, we used (2.38) to calculate the theoretical curve. We again observe that our transient analysis closely describes the behavior of Algorithm 3 and 4. We observe that $M = 200$ is sufficiently large for these simulations that the mixture converges to the best constituent filter.

Here, we investigate the trade-off between the transient and steady-state behaviors for the combination algorithms as follows. In this setup, the desired signal is generated as $d(t) = \boldsymbol{w}_o^T \boldsymbol{u}(t) + n(t)$, where $n(t)$ is the additive i.i.d noise with variance $\sigma_n^2 = 0.01$, $\boldsymbol{w}_o = [0.25, -0.47, -0.37, 0.04, -0.18, 0.78, 0.14]^T$ and the input regressor $\boldsymbol{u}(t) \in \mathbb{R}^7$ is zero mean i.i.d. Gaussian, where the variance of each entry is set to 1. As the input filters, there are two LMS filters running in parallel to model $d(t)$ with the same input regressor and the step sizes $\mu_1 = 0.15$, $\mu_2 = 0.002$, respectively. We first fix the step size of Algorithm 1, i.e., $\mu_\alpha = 0.5$, and generate the theoretical MSE(n) curve versus iteration index n. Then, we determine the value of n where MSE(n) is 3 dB above the minimum MSE and label it n_1. We adjust the step size of Algorithm 2 μ_ρ, the step size μ_ϵ and the forgetting factor a of Algorithm 3 and the time window M of Algorithm 4 such that the final MSE of each algorithm is equal to the final MSE of Algorithm 1. Then, the theoretical MSE(n) curve versus iteration index n for each algorithm is generated and the values of n where MSE(n) is 3 dB above the minimum MSE are determined and labeled by n_2, n_3, n_4, respectively. The performance of the combination algorithm with the smallest n_i is the best for this example. We repeat this process for different selections of μ_α including $\mu_\alpha = 5$ and $\mu_\alpha = 100$ and summarize the results in Table 2.1. We observe that in these simulations Algorithm 2 provide a better converge trade-off.

2.6 Conclusion

We investigated and compared four convexly constrained adaptive mixture methods to adaptively combine outputs of constituent filters that work in parallel on system modeling. We derived the corresponding MSEs and the converged mixture weights in the steady-state under nonstationary random walk model. We also performed the transient analysis in the mean and MSE sense for all algorithms. We observe that these convex mixture methods are universal such that they achieve the performance of the best constituent filter in the steady-state. Our main contributions in this chapter can be listed as follows:

1. We show that the algorithm from [20] is universal and its combined weight vector converges to the optimal convex mixture;

2. We demonstrate that the algorithm from [30] is only universal if the memory constant is unitary (no decay is allowed if universality is required), but the weight vector does not convergence to the optimal convex mixture;

3. We show that the algorithm from [27] is always universal (but not better than the best filter) only for very long windows, however, does not offer the desirable weight vector convergence.

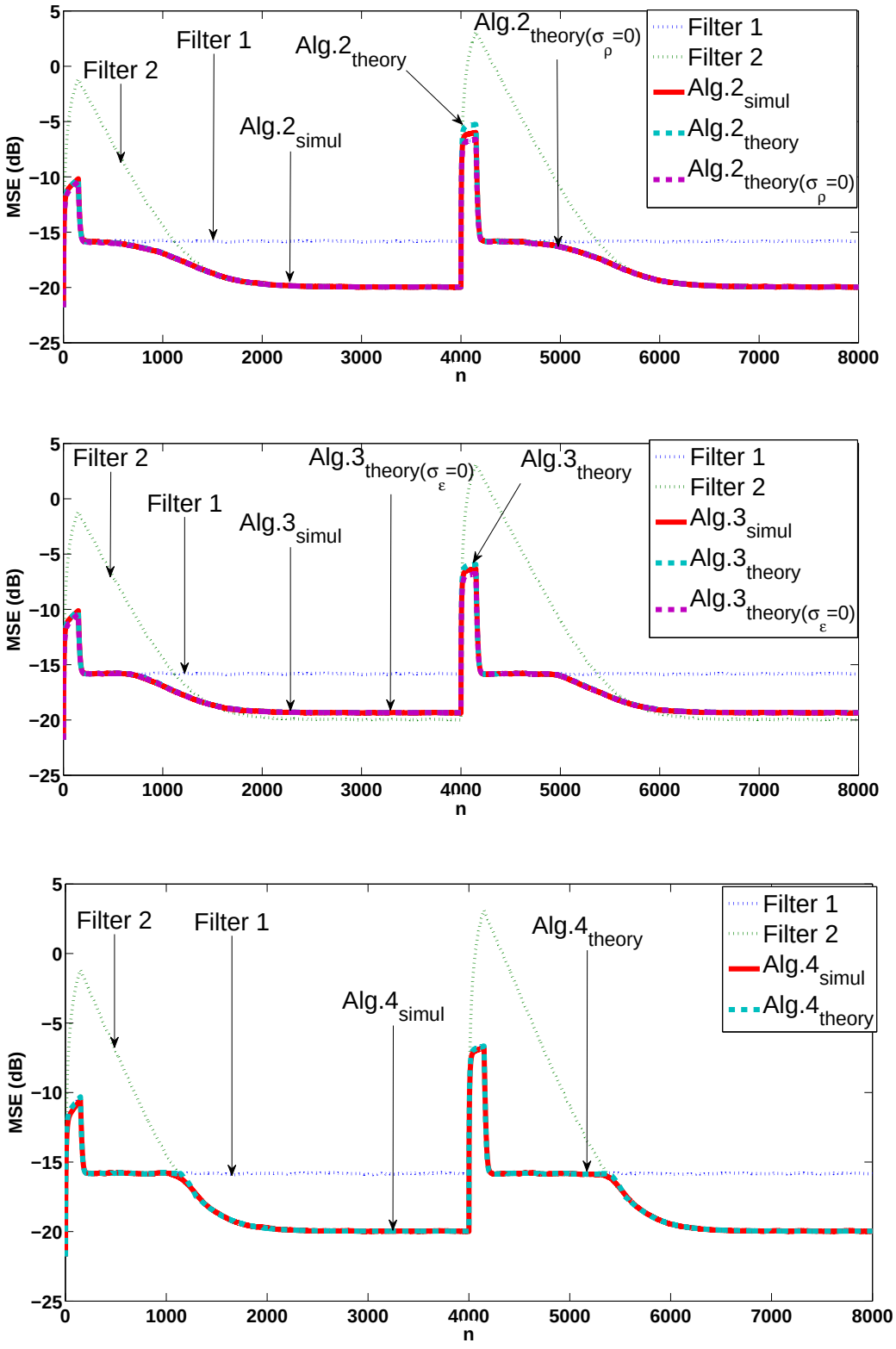


Figure 2.2: MSE curves for all algorithms. Labels are described in the text. (a) Algorithm 2, $\mu_\rho = 1$. (b) Algorithm 3, $\mu_\epsilon = 1$ and $a = 0.98$. (c) Algorithm 4, $M = 200$.

Chapter 3

ADAPTIVE MIXTURE METHODS BASED ON BREGMAN DIVERGENCES

In this chapter, we study adaptive mixture methods based on *Bregman divergences* [10, 15] that combine outputs of m constituent filters running in parallel on the same task. The overall system has two stages [2–5,8,23]. The first stage contains adaptive filters running in parallel to model a desired signal. The outputs of these adaptive filters are then linearly combined to produce the final output of the overall system in the second stage. We use Bregman divergences and obtain certain multiplicative updates [21], [15], [16] to train these linear combination weights under an affine constraint [9] or without any constraints [22]. We use unnormalized [15] and normalized relative entropy [21] to define two different Bregman divergences that produce the unnormalized exponentiated gradient update (EGU) and the exponentiated gradient update (EG) on the mixture weights [21], respectively. We then perform the mean and the mean-square transient analysis of these adaptive mixtures when they are used to combine outputs of m constituent filters. We emphasize that to the best of our knowledge, this is the first mean and mean-square transient analysis of the EGU algorithm and the EG algorithm in the mixture framework (which naturally covers the classical framework also [7,28]). We illustrate the accuracy of our results through simulations in different configurations and demonstrate advantages of the introduced algorithms for sparse mixture systems.

We use Bregman divergences to derive multiplicative updates on the mixture weights. Specifically, we use the unnormalized relative entropy and the relative entropy as distance measures and obtain the EGU algorithm and the EG algorithm to update the combination weights under an affine constraint or without any constraints. We then carry out the mean and the mean-square transient analysis of these adaptive mixtures when they are used to combine m constituent filters. We point out that the EG algorithm is widely used in sequential learning theory [31] and minimizes an approximate final estimation error

while penalizing the distance between the new and the old filter weights. In network and acoustic echo cancellation applications, the EG algorithm is shown to converge faster than the LMS algorithm [26,28] when the system impulse response is sparse [7]. Similarly, in our simulations, we observe that using the EG algorithm to train the mixture weights yields increased convergence speed compared to using the LMS algorithm to train the mixture weights [9, 22] when the combination favors only a few of the constituent filters in the steady state, i.e., when the final steady-state combination vector is sparse. We also observe that the EGU algorithm and the LMS algorithm show similar performance when they are used to train the mixture weights even if the final steady-state mixture is sparse. In this sense, we emphasize that we do not force the system to be sparse in order to make sure that the EG algorithm performs better than the LMS algorithm. However, if the final steady-state vector is sparse, than the EG could increase the convergence speed.

To summarize, the main contributions of this chapter are as follows:

- We use Bregman divergences to derive multiplicative updates on affinely constrained and unconstrained mixture weights adaptively combining outputs of m constituent filters.

- We use the unnormalized relative entropy and the relative entropy to define two different Bregman divergences that produce the EGU algorithm and the EG algorithm to update the affinely constrained and unconstrained mixture weights.

- We perform the mean and the mean-square transient analysis of the affinely constrained and unconstrained mixtures using the EGU algorithm and the EG algorithm.

3.1 Problem Description

3.1.1 Notation

In this chapter, all vectors are column vectors and represented by boldface lowercase letters. Matrices are represented by boldface capital letters. For presentation purposes, we work only with real data. Given a vector $\boldsymbol{w}$, $w^{(i)}$ denotes the ith individual entry of $\boldsymbol{w}$, $\boldsymbol{w}^T$ is the transpose of $\boldsymbol{w}$, $\|\boldsymbol{w}\|_1 \stackrel{\triangle}{=} \sum_i |w^{(i)}|$ is the l_1 norm; $\|\boldsymbol{w}\| \stackrel{\triangle}{=} \sqrt{\boldsymbol{w}^T\boldsymbol{w}}$ is the l_2 norm. For a matrix $\boldsymbol{W}$, $\boldsymbol{t}_{r,n}(\boldsymbol{W})$ is the trace. For a vector $\boldsymbol{w}$, $\mathrm{diag}(\boldsymbol{w})$ represents a diagonal matrix

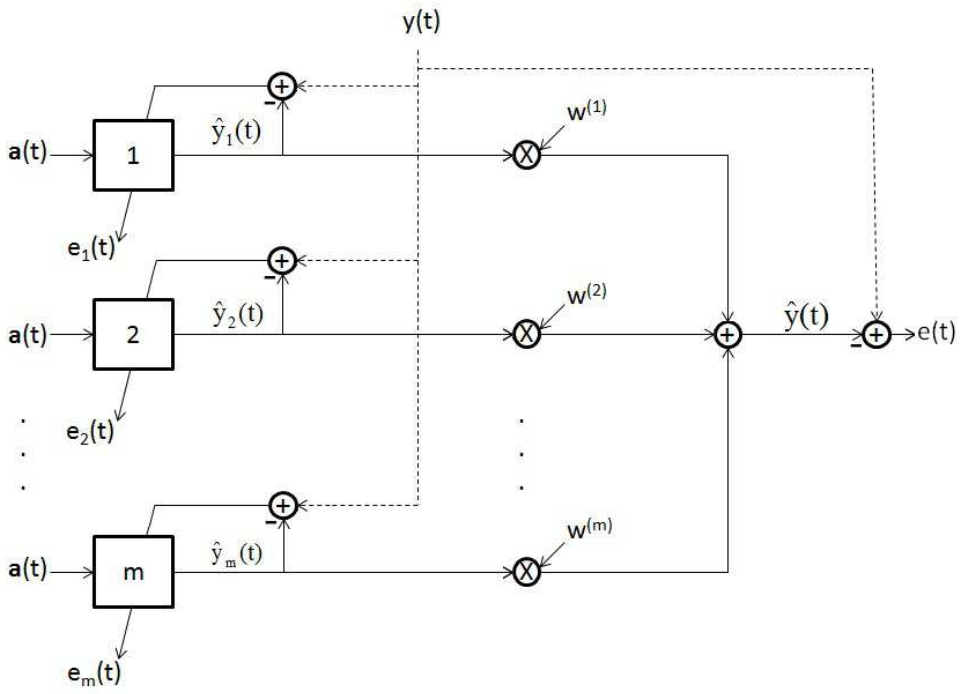


Figure 3.1: A linear mixture of outputs of m adaptive filters.

formed using the entries of $\boldsymbol{w}$. For a matrix $\boldsymbol{W}$, diag($\boldsymbol{W}$) represents a column vector that contains the diagonal entries of $\boldsymbol{W}$. For two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, we define the concatenation $[\boldsymbol{v}_1; \boldsymbol{v}_2] \stackrel{\triangle}{=} [\boldsymbol{v}_1^T \ \boldsymbol{v}_2^T]^T$. For a random variable v, $\bar{v}$ is the expected value. For a random vector $\boldsymbol{v}$ (or a random matrix $\boldsymbol{V}$), $\bar{\boldsymbol{v}}$ (or $\bar{\boldsymbol{V}}$) represents the expected value of each entry. Vectors (or matrices) $\mathbf{1}$ and $\mathbf{0}$, with an abuse of notation, denote vectors (or matrices) of all ones or zeros, respectively, where the size of the vector (or the matrix) is understood from the context.

3.1.2 System Description

The framework that we study has two stages. In the first stage, we have m adaptive filters producing outputs $\hat{y}_i(t)$, $i = 1, \ldots, m$, running in parallel to model a desired signal $y(t)$ as seen in Fig. 1. Here, $\boldsymbol{a}(t)$ is generated from a zero mean stochastic process and $y(t)$ is generated from a zero-mean stationary stochastic process. The second stage is the mixture stage, where the outputs of the first stage filters are combined to improve the steady-state and/or the transient performance over the constituent filters. We linearly combine the

outputs of the first stage filters to produce the final output as $\hat{y}(t) = \boldsymbol{w}^T(t)\boldsymbol{x}(t)$, where $\boldsymbol{x}(t) \stackrel{\triangle}{=} [\hat{y}_1(t), \ldots, \hat{y}_m(t)]^T$ and train the mixture weights using multiplicative updates (or exponentiated gradient updates) [15]. We point out that in order to satisfy the constraints and derive the multiplicative updates [21], [12], we use reparametrization of the mixture weights as $\boldsymbol{w}(t) = \boldsymbol{f}(\boldsymbol{\lambda}(t))$ and perform the update on $\boldsymbol{\lambda}(t)$ as

$$\boldsymbol{\lambda}(t+1) = \arg\min_{\boldsymbol{\lambda}} \Big\{ d(\boldsymbol{\lambda}, \boldsymbol{\lambda}(t)) + \mu\, l\big(y(t), \boldsymbol{f}^T(\boldsymbol{\lambda})\boldsymbol{x}(t)\big) \Big\}, \tag{3.1}$$

where μ is the learning rate of the update, $d(\cdot,\cdot)$ is an appropriate distance measure and $l(\cdot,\cdot)$ is the instantaneous loss. We emphasize that in (3.1), the updated vector $\boldsymbol{\lambda}$ is forced to be close to the present vector $\boldsymbol{\lambda}(t)$ by $d(\boldsymbol{\lambda}(t+1), \boldsymbol{\lambda}(t))$, while trying to accurately model the current data by $l\big(y(t), \boldsymbol{f}^T(\boldsymbol{\lambda})\boldsymbol{x}(t)\big)$. However, instead of directly minimizing (3.1), a linearized version of (3.1)

$$\begin{aligned} \boldsymbol{\lambda}(t+1) = \arg\min_{\boldsymbol{\lambda}} \Big\{ & d(\boldsymbol{\lambda}, \boldsymbol{\lambda}(t)) + l\left(y(t), \boldsymbol{f}^T(\boldsymbol{\lambda}(t))\boldsymbol{x}(t)\right) \\ & + \mu \nabla_{\boldsymbol{\lambda}} l\left(y(t), \boldsymbol{f}^T(\boldsymbol{\lambda})\boldsymbol{x}(t)\right)^T \Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}(t)} (\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)) \Big\} \end{aligned} \tag{3.2}$$

is minimized to get the desired update. As an example, if we use the l_2-norm as the distance measure, i.e., $d(\boldsymbol{\lambda}, \boldsymbol{\lambda}(t)) = \|\boldsymbol{\lambda} - \boldsymbol{\lambda}(t)\|^2$, and the square error as the instantaneous loss, i.e., $l\big(y(t), \boldsymbol{f}^T(\boldsymbol{\lambda})\boldsymbol{x}(t)\big) = [y(t) - \boldsymbol{f}^T(\boldsymbol{\lambda})\boldsymbol{x}(t)]^2$ with $\boldsymbol{f}(\boldsymbol{\lambda}) = \boldsymbol{\lambda}$, then we get the stochastic gradient update on $\boldsymbol{w}(t)$, i.e.,

$$\boldsymbol{w}(t+1) = \boldsymbol{w}(t) + \mu e(t)\boldsymbol{x}(t),$$

in (3.2).

In the next section, we use the unnormalized relative entropy

$$d_1(\boldsymbol{\lambda}, \boldsymbol{\lambda}(t)) = \left\{ \sum_{i=1}^{m} \left[\lambda^{(i)} \ln \left(\frac{\lambda^{(i)}}{\lambda^{(i)}(t)} \right) + \lambda^{(i)}(t) - \lambda^{(i)} \right] \right\} \tag{3.3}$$

for positively constrained $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}(t)$, $\boldsymbol{\lambda} \in \mathbb{R}_+^m$, $\boldsymbol{\lambda}(t) \in \mathbb{R}_+^m$, and the relative entropy

$$d_2(\boldsymbol{\lambda}, \boldsymbol{\lambda}(t)) = \left\{ \sum_{i=1}^{m} \left[\lambda^{(i)} \ln \left(\frac{\lambda^{(i)}}{\lambda^{(i)}(t)} \right) \right] \right\}, \tag{3.4}$$

where $\boldsymbol{\lambda}$ is constrained to be in an extended simplex such that $\lambda^{(i)} \geq 0$, $\sum_{k=1}^{m} \lambda^{(i)} = u$ for some $u \geq 1$ as the distance measures, with appropriately selected $\boldsymbol{f}(\cdot)$ to derive updates on mixture weights under different constraints. We first investigate affinely constrained mixture of m adaptive filters, and then continue with the unconstrained mixture using (3.3) and (3.4) as the distance measures.

3.2 Adaptive Mixture Algorithms

In this section, we investigate affinely constrained and unconstrained mixtures updated with the EGU algorithm and the EG algorithm.

3.2.1 Affinely Constrained Mixture

When an affine constraint is imposed on the mixture such that $\boldsymbol{w}^T(t)\mathbf{1} = 1$, we get

$$\begin{aligned}
&\hat{y}(t) = \boldsymbol{w}(t)^T \boldsymbol{x}(t), \\
&e(t) = y(t) - \hat{y}(t), \\
&w^{(i)}(t) = \lambda^{(i)}(t), \;\; i = 1, \ldots, m-1, \\
&w^{(m)}(t) = 1 - \sum_{i=1}^{m-1} \lambda^{(i)}(t),
\end{aligned}$$

where the $m-1$ dimensional vector $\boldsymbol{\lambda}(t) \stackrel{\triangle}{=} [\lambda^{(1)}(t), \ldots, \lambda^{(m-1)}(t)]^T$ is the unconstrained weight vector, i.e., $\boldsymbol{\lambda}(t) \in \mathbb{R}^{m-1}$. Using $\boldsymbol{\lambda}(t)$ as the unconstrained weight vector, the error can be written as $e(t) = \big[y(t) - \hat{y}_m(t)\big] - \boldsymbol{\lambda}^T(t)\boldsymbol{\delta}(t)$, where $\boldsymbol{\delta}(t) \stackrel{\triangle}{=} [\hat{y}_1(t) - \hat{y}_m(t), \ldots, \hat{y}_{m-1}(t) - \hat{y}_m(t)]^T$. To be able to derive a multiplicative update on $\boldsymbol{\lambda}(t)$, we use

$$\boldsymbol{\lambda}(t) = \boldsymbol{\lambda}_1(t) - \boldsymbol{\lambda}_2(t),$$

where $\boldsymbol{\lambda}_1(t)$ and $\boldsymbol{\lambda}_2(t)$ are constrained to be nonnegative, i.e., $\boldsymbol{\lambda}_i(t) \in \mathbb{R}_+^{m-1}$, $i = 1, 2$. After we collect nonnegative weights in $\boldsymbol{\lambda}_a(t) = [\boldsymbol{\lambda}_1(t); \boldsymbol{\lambda}_2(t)]$, we define a function of loss $e(t)$ as

$$l_{\mathrm{a}}\left(\boldsymbol{\lambda}_a(t)\right) \stackrel{\triangle}{=} e^2(t)$$

and update positively constrained $\boldsymbol{\lambda}_a(t)$ as follows.

Unnormalized Relative Entropy

Using the unconstrained relative entropy as the distance measure, we get

$$\boldsymbol{\lambda}_a(t+1) = \arg\min_{\boldsymbol{\lambda}} \left\{ \sum_{i=1}^{2(m-1)} \left[\lambda^{(i)} \ln \left(\frac{\lambda^{(i)}}{\lambda_a^{(i)}(t)} \right) + \lambda_a^{(i)}(t) - \lambda^{(i)} \right] + \mu \left[l_{\mathrm{a}} \left(\boldsymbol{\lambda}_a(t) \right) + \nabla_{\boldsymbol{\lambda}} l_{\mathrm{a}} \left(\boldsymbol{\lambda} \right)^T \big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}_a(t)} \left(\boldsymbol{\lambda} - \boldsymbol{\lambda}_a(t) \right) \right] \right\}.$$

After some algebra this yields

$$\lambda_a^{(i)}(t+1) = \lambda_a^{(i)}(t) \exp\left\{ \mu e(t) (\hat{y}_i(t) - \hat{y}_m(t)) \right\}, i = 1, \ldots, m-1,$$
$$\lambda_a^{(i)}(t+1) = \lambda_a^{(i)}(t) \exp\left\{ -\mu e(t) (\hat{y}_{i-m+1}(t) - \hat{y}_m(t)) \right\}, i = m, \ldots, 2(m-1),$$

providing the multiplicative updates on $\boldsymbol{\lambda}_1(t)$ and $\boldsymbol{\lambda}_2(t)$.

Relative Entropy

Using the relative entropy as the distance measure, we get

$$\boldsymbol{\lambda}_a(t+1) = \arg\min_{\boldsymbol{\lambda}} \left\{ \sum_{i=1}^{2(m-1)} \left[\lambda^{(i)} \ln \left(\frac{\lambda^{(i)}}{\lambda_a^{(i)}(t)} \right) + \gamma(u - \mathbf{1}^T \boldsymbol{\lambda}) \right] + \mu \left[l_{\mathrm{a}} \left(\boldsymbol{\lambda}_a(t) \right) + \nabla_{\boldsymbol{\lambda}} l_{\mathrm{a}} \left(\boldsymbol{\lambda} \right)^T \big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}_a(t)} \left(\boldsymbol{\lambda} - \boldsymbol{\lambda}_a(t) \right) \right] \right\},$$

where γ is the Lagrange multiplier. This yields

$$\lambda_a^{(i)}(t+1) = u \frac{\lambda_a^{(i)}(t) \exp\left\{ \mu e(t) (\hat{y}_i(t) - \hat{y}_m(t)) \right\}}{\sum_{k=1}^{m-1} \left[\lambda_a^{(k)}(t) \exp\left\{ \mu e(t) (\hat{y}_k(t) - \hat{y}_m(t)) \right\} + \lambda_a^{(k+m-1)}(t) \exp\left\{ -\mu e(t) (\hat{y}_k(t) - \hat{y}_m(t)) \right\} \right]},$$
$$i = 1, \ldots, m-1,$$
$$\lambda_a^{(i)}(t+1) = u \frac{\lambda_a^{(i)}(t) \exp\left\{ -\mu e(t) (\hat{y}_{i-m+1}(t) - \hat{y}_m(t)) \right\}}{\sum_{k=1}^{m-1} \left[\lambda_a^{(k)}(t) \exp\left\{ \mu e(t) (\hat{y}_k(t) - \hat{y}_m(t)) \right\} + \lambda_a^{(k+m-1)}(t) \exp\left\{ -\mu e(t) (\hat{y}_k(t) - \hat{y}_m(t)) \right\} \right]},$$
$$i = m, \ldots, 2(m-1),$$

providing the multiplicative updates on $\boldsymbol{\lambda}_a(t)$.

3.2.2 Unconstrained Mixture

Without any constraints on the combination weights, the mixture stage can be written as

$$\hat{y}(t) = \boldsymbol{w}^T(t)\boldsymbol{x}(t),$$
$$e(t) = y(t) - \hat{y}(t),$$

where $\boldsymbol{w}(t) \in \mathbb{R}^m$. To be able to derive a multiplicative update, we use a change of variables,

$$\boldsymbol{w}(t) = \boldsymbol{w}_1(t) - \boldsymbol{w}_2(t),$$

where $\boldsymbol{w}_1(t)$ and $\boldsymbol{w}_2(t)$ are constrained to be nonnegative, i.e., $\boldsymbol{w}_i(t) \in \mathbb{R}_+^m$, $i = 1, 2$. We then collect the nonnegative weights $\boldsymbol{w}_a(t) = [\boldsymbol{w}_1(t); \boldsymbol{w}_2(t)]$ and define a function of the loss $e(t)$ as

$$l_{\mathrm{u}}\left(\boldsymbol{w}_a(t)\right) \stackrel{\triangle}{=} e^2(t).$$

Unnormalized Relative Entropy

Defining cost function similar to (4) and minimizing it with respect to $\boldsymbol{w}$ yields

$$w_a^{(i)}(t+1) = w_a^{(i)}(t) \exp\left\{\mu e(t)\hat{y}_i(t)\right\}, i = 1, \ldots, m,$$
$$w_a^{(i)}(t+1) = w_a^{(i)}(t) \exp\left\{-\mu e(t)\hat{y}_{i-m}(t)\right\}, i = m+1, \ldots, 2m,$$

providing the multiplicative update on $\boldsymbol{w}_a(t)$.

Relative Entropy

Using the relative entropy under the simplex constraint on $\boldsymbol{w}$, we get the updates

$$w_a^{(i)}(t+1) = u\frac{w_a^{(i)}(t)\exp\left\{\mu e(t)\hat{y}_i(t)\right\}}{\sum_{k=1}^{m}\left[w_a^{(k)}(t)\exp\left\{\mu e(t)\hat{y}_k(t)\right\} + w_a^{(k+m)}(t)\exp\left\{-\mu e(t)\hat{y}_k(t)\right\}\right]},$$

$i = 1, \ldots, m,$

$$w_a^{(i)}(t+1) = u\frac{w_a^{(i)}(t)\exp\left\{-\mu e(t)\hat{y}_{i-m}(t)\right\}}{\sum_{k=1}^{m}\left[w_a^{(k)}(t)\exp\left\{\mu e(t)\hat{y}_k(t)\right\} + w_a^{(k+m)}(t)\exp\left\{-\mu e(t)\hat{y}_k(t)\right\}\right]},$$

$i = m+1 \ldots, 2m.$

In the next section, we study the transient analysis of these four adaptive mixture algorithms.

3.3 Transient Analysis

In this section, we study the mean and the mean-square transient analysis of the adaptive mixture methods. We start with the affinely constrained combination.

3.3.1 Affinely Constrained Mixture

We first perform the transient analysis of the mixture weights updated with the EGU algorithm. Then, we continue with the transient analysis of the mixture weights updated with the EG algorithm.

Unconstrained Relative Entropy

For the affinely constrained mixture updated with the EGU algorithm, using Taylor Series, we have the multiplicative update as

$$\begin{aligned}\lambda_1^{(i)}(t+1) &= \lambda_1^{(i)}(t)\exp\left\{\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\right\},\\ &= \lambda_1^{(i)}(t)\sum_{k=0}^{\infty}\frac{\big(\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\big)^k}{k!},\end{aligned} \tag{3.5}$$

$$\begin{aligned}\lambda_2^{(i)}(t+1) &= \lambda_2^{(i)}(t)\exp\left\{-\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\right\},\\ &= \lambda_2^{(i)}(t)\sum_{k=0}^{\infty}\frac{\big(-\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\big)^k}{k!},\end{aligned} \tag{3.6}$$

for $i = 1, \ldots, m-1$. If $e(t)$ and $\hat{y}_i(t) - \hat{y}_m(t)$ for each $i = 1, \ldots, m-1$ are bounded, then we can write (3.5) and (3.6) as

$$\lambda_1^{(i)}(t+1) \approx \lambda_1^{(i)}(t)\big(1+\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t)) + O(\mu^2)\big), \tag{3.7}$$

$$\lambda_2^{(i)}(t+1) \approx \lambda_2^{(i)}(t)\big(1-\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t)) + O(\mu^2)\big), \tag{3.8}$$

for $i = 1, \ldots, m-1$. Since μ is usually relatively small [15], we approximate (3.7) and (3.8) as

$$\lambda_1^{(i)}(t+1) \approx \lambda_1^{(i)}(t)\big(1+\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\big), \tag{3.9}$$

$$\lambda_2^{(i)}(t+1) \approx \lambda_2^{(i)}(t)\big(1-\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\big). \tag{3.10}$$

In our simulations, we illustrate the accuracy of the approximations (3.9) and (3.10) under the mixture framework. Using (3.9) and (3.10), we can obtain updates on $\boldsymbol{\lambda}_1(t)$ and $\boldsymbol{\lambda}_2(t)$ as

$$\boldsymbol{\lambda}_1(t+1) = \big(I + \mu e(t)\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\big)\boldsymbol{\lambda}_1(t), \tag{3.11}$$

$$\boldsymbol{\lambda}_2(t+1) = \big(I - \mu e(t)\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\big)\boldsymbol{\lambda}_2(t). \tag{3.12}$$

Collecting the weights in $\boldsymbol{\lambda}_a(t) = [\boldsymbol{\lambda}_1(t); \boldsymbol{\lambda}_2(t)]$, using the updates (3.11) and (3.12), we can write update on $\boldsymbol{\lambda}_a(t)$ as

$$\boldsymbol{\lambda}_a(t+1) = \big(I + \mu e(t)\mathrm{diag}\big(\boldsymbol{u}(t)\big)\big)\boldsymbol{\lambda}_a(t), \tag{3.13}$$

where $\boldsymbol{u}(t)$ is defined as $\boldsymbol{u}(t) \stackrel{\triangle}{=} [\boldsymbol{\delta}(t); -\boldsymbol{\delta}(t)]$.

For the desired signal $y(t)$, we can write $y(t) - \hat{y}_m(t) = \boldsymbol{\lambda}_0^T(t)\boldsymbol{\delta}(t) + e_0(t)$, where $\boldsymbol{\lambda}_0(t)$ is the optimum MSE solution at time t such that $\boldsymbol{\lambda}_0(t) \stackrel{\triangle}{=} \boldsymbol{R}^{-1}(t)\boldsymbol{p}(t)$, $\boldsymbol{R}(t) \stackrel{\triangle}{=} E\big[\boldsymbol{\delta}(t)\boldsymbol{\delta}^T(t)\big]$, $\boldsymbol{p}(t) \stackrel{\triangle}{=} E\big\{\boldsymbol{\delta}(t)\big[y(t) - \hat{y}_m(t)\big]\big\}$ and $e_0(t)$ is zero-mean and uncorrelated with $\boldsymbol{\delta}(t)$. We next show that the mixture weights converge to the optimum solution in the steady-state such that $\lim_{t\to\infty} E\big[\boldsymbol{\lambda}(t)\big] = \lim_{t\to\infty} \boldsymbol{\lambda}_0(t)$ for properly selected μ.

Subtracting (3.12) from (3.11), we obtain

$$\begin{aligned}\boldsymbol{\lambda}(t+1) &= \boldsymbol{\lambda}(t) + \mu e(t)\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\big(\boldsymbol{\lambda}_1(t) + \boldsymbol{\lambda}_2(t)\big),\\ &= \boldsymbol{\lambda}(t) - \mu e(t)\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}(t) + 2\mu e(t)\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}_1(t).\end{aligned} \tag{3.14}$$

Defining $\boldsymbol{\varepsilon}(t) \stackrel{\triangle}{=} \boldsymbol{\lambda}_0(t) - \boldsymbol{\lambda}(t)$ and using $e(t) = \boldsymbol{\delta}^T(t)\boldsymbol{\varepsilon}(t) + e_0(t)$ in (3.14) yield

$$\begin{aligned}\boldsymbol{\lambda}(t+1) &= \boldsymbol{\lambda}(t) - \mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}(t)\boldsymbol{\delta}^T(t)\boldsymbol{\varepsilon}(t) - \mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}(t)e_0(t)\\ &+ 2\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}_1(t)\boldsymbol{\delta}^T(t)\boldsymbol{\varepsilon}(t) + 2\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}_1(t)e_0(t).\end{aligned} \tag{3.15}$$

In (3.15), subtracting both sides from $\boldsymbol{\lambda}_0(t+1)$, we have

$$\begin{aligned}\boldsymbol{\varepsilon}(t+1) &= \boldsymbol{\varepsilon}(t) + \mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}(t)\boldsymbol{\delta}^T(t)\boldsymbol{\varepsilon}(t) + \mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}(t)e_0(t)\\ &- 2\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}_1(t)\boldsymbol{\delta}^T(t)\boldsymbol{\varepsilon}(t) - 2\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}_1(t)e_0(t)\\ &+ \big[\boldsymbol{\lambda}_0(t+1) - \boldsymbol{\lambda}_0(t)\big].\end{aligned} \tag{3.16}$$

Taking expectation of both sides of (3.16) and using

$$\begin{aligned}E\big[\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}(t)e_0(t)\big] &= E\big[\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}(t)\big]E[e_0(t)] = 0,\\ E\big[2\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}_1(t)e_0(t)\big] &= E\big[2\mu\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\boldsymbol{\lambda}_1(t)\big]E[e_0(t)] = 0,\end{aligned}$$

and assuming that the correlation between $\boldsymbol{\varepsilon}(t)$ and $\boldsymbol{\lambda}_1(t)$, $\boldsymbol{\lambda}_2(t)$ is small enough to be safely omitted [24] yields

$$E\big[\boldsymbol{\varepsilon}(t+1)\big] = E\big[I - \mu\mathrm{diag}\big(\boldsymbol{\lambda}_1(t) + \boldsymbol{\lambda}_2(t)\big)\boldsymbol{\delta}(t)\boldsymbol{\delta}^T(t)\big]E\big[\boldsymbol{\varepsilon}(t)\big] + E\big[\boldsymbol{\lambda}_0(t+1) - \boldsymbol{\lambda}_0(t)\big]. \tag{3.17}$$

Assuming convergence of $\boldsymbol{R}(t)$ and $\boldsymbol{p}(t)$ (which is true for a wide range of adaptive methods in the first stage [25], [18, 28]), we obtain $\lim_{t\to\infty} E\big[\boldsymbol{\lambda}_0(t+1) - \boldsymbol{\lambda}_0(t)\big] = 0$. If μ is chosen such that the eigenvalues of $E\big[I - \mu\mathrm{diag}\big(\boldsymbol{\lambda}_1(t) + \boldsymbol{\lambda}_2(t)\big)\boldsymbol{\delta}(t)\boldsymbol{\delta}^T(t)\big]$ have strictly less than unit magnitude for every t, then
$\lim_{t\to\infty} E\big[\boldsymbol{\lambda}(t)\big] = \lim_{t\to\infty} \boldsymbol{\lambda}_0(t)$.

For the transient analysis of the MSE, we have

$$\begin{aligned}
E[e^2(t)] &= E\left\{\big[y(t) - \hat{y}_m(t)\big]^2\right\} - 2\bar{\boldsymbol{\lambda}}_a^T(t)E\left\{\big[y(t) - \hat{y}_m(t)\big][\boldsymbol{\delta}(t); -\boldsymbol{\delta}(t)]\right\} \\
&\quad + E\left\{\boldsymbol{\lambda}_a^T(t)[\boldsymbol{\delta}(t); -\boldsymbol{\delta}(t)][\boldsymbol{\delta}(t); -\boldsymbol{\delta}(t)]^T\boldsymbol{\lambda}_a(t)\right\}, \\
&= E\left\{\big[y(t) - \hat{y}_m(t)\big]^2\right\} - 2\bar{\boldsymbol{\lambda}}_a^T(t)E\left\{\big[y(t) - \hat{y}_m(t)\big]\boldsymbol{u}(t)\right\} \\
&\quad + \boldsymbol{t}_{r,n}\Big(E\left[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\right]E\left\{\boldsymbol{u}(t)\boldsymbol{u}(t)^T\right\}\Big), \\
&= E\left\{\big[y(t) - \hat{y}_m(t)\big]^2\right\} - 2\bar{\boldsymbol{\lambda}}_a^T(t)\boldsymbol{\gamma}(t) + \boldsymbol{t}_{r,n}\Big(E\left[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\right]\boldsymbol{\Gamma}(t)\Big),
\end{aligned} \tag{3.18}$$

where we define $\boldsymbol{\gamma}(t) \triangleq E\left\{\boldsymbol{u}(t)\big[y(t) - \hat{y}_m(t)\big]\right\}$ and $\boldsymbol{\Gamma}(t) \triangleq E\big[\boldsymbol{u}(t)\boldsymbol{u}^T(t)\big]$.

For the recursion of $\bar{\boldsymbol{\lambda}}_a(t) = E[\boldsymbol{\lambda}_a(t)]$, using (3.13), we get

$$\bar{\boldsymbol{\lambda}}_a(t+1) = \bar{\boldsymbol{\lambda}}_a(t) + \mu\mathrm{diag}\big(\boldsymbol{\gamma}(t)\big)\bar{\boldsymbol{\lambda}}_a(t) - \mu\mathrm{diag}\big(E[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)]\boldsymbol{\Gamma}(t)\big). \tag{3.19}$$

Using (3.13) and $e(t) = \big[y(t) - \hat{y}_m(t)\big] - \boldsymbol{\lambda}^T(t)\boldsymbol{\delta}(t)$, assuming $\boldsymbol{\lambda}_a(t)$ is Gaussian and assuming $\lambda_a^{(i)}(t)$ and $\lambda_a^{(j)}(t)$ are uncorrelated when $i \neq j$ (as in Chapter 9.4.2) [28], [24], defining the diagonal matrix
$\boldsymbol{D}(t) = E[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)] - \bar{\boldsymbol{\lambda}}_a(t)\bar{\boldsymbol{\lambda}}_a^T(t)$ and since μ is small, ignoring the terms that are pro-

portional to μ^2, we get a recursion for $E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]$ as

$$
\begin{aligned}
&E\big[\boldsymbol{\lambda}_a(t+1)\boldsymbol{\lambda}_a^T(t+1)\big] = E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big] + \mu\mathrm{diag}\big(\boldsymbol{\gamma}(t)\big)E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big] \\
&- \mu\mathrm{diag}\big(\boldsymbol{\Gamma}(t)\bar{\boldsymbol{\lambda}}_a(t)\big)E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big] - \mu E\big[\mathrm{diag}^2(\boldsymbol{u}(t))\big]\boldsymbol{D}(t)\mathbf{1}\bar{\boldsymbol{\lambda}}_a^T(t) \\
&- \mu\mathrm{diag}\big(\bar{\boldsymbol{\lambda}}_a(t)\big)\boldsymbol{\Gamma}(t)\boldsymbol{D}(t) + \mu E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]\mathrm{diag}\big(\boldsymbol{\gamma}(t)\big) \\
&- \mu E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]\mathrm{diag}\big(\boldsymbol{\Gamma}(t)\bar{\boldsymbol{\lambda}}_a(t)\big) - \mu\bar{\boldsymbol{\lambda}}_a(t)\mathbf{1}^T\boldsymbol{D}(t)E\big[\mathrm{diag}^2(\boldsymbol{u}(t))\big] \\
&- \mu\boldsymbol{D}(t)\boldsymbol{\Gamma}(t)\mathrm{diag}\big(\bar{\boldsymbol{\lambda}}_a(t)\big). \qquad (3.20)
\end{aligned}
$$

Defining $\boldsymbol{q}_a(t) \stackrel{\triangle}{=} \bar{\boldsymbol{\lambda}}_a(t)$ and $\boldsymbol{Q}_a(t) \stackrel{\triangle}{=} E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]$, we express (3.19) and (3.20) as a coupled recursions in Table 3.1.

Table 3.1: Time evolution of the mean and the variance of the affinely constrained mixture weights updated with the EGU algorithm

$\boldsymbol{q}_a(t+1) = \boldsymbol{q}_a(t) + \mu\mathrm{diag}\big(\boldsymbol{\gamma}(t)\big)\boldsymbol{q}_a(t) - \mu\mathrm{diag}\big(\boldsymbol{Q}_a(t)\boldsymbol{\Gamma}(t)\big),$
$\boldsymbol{Q}_a(t+1) = \Big(I + \mu\mathrm{diag}\big(\boldsymbol{\gamma}(t)\big) - \mu\mathrm{diag}\big(\boldsymbol{\Gamma}(t)\boldsymbol{q}_a(t)\big)\Big)\boldsymbol{Q}_a(t) - \mu E\big[\mathrm{diag}^2(\boldsymbol{u}(t))\big]\Big(\boldsymbol{Q}_a(t) - \boldsymbol{q}_a(t)\boldsymbol{q}_a^T(t)\Big)\mathbf{1}\boldsymbol{q}_a^T(t)$
$-\mu\mathrm{diag}\big(\boldsymbol{q}_a(t)\big)\boldsymbol{\Gamma}(t)\Big(\boldsymbol{Q}_a(t) - \boldsymbol{q}_a(t)\boldsymbol{q}_a^T(t)\Big) + \boldsymbol{Q}_a(t)\Big(\mu\mathrm{diag}\big(\boldsymbol{\gamma}(t)\big) - \mu\mathrm{diag}\big(\boldsymbol{\Gamma}(t)\boldsymbol{q}_a(t)\big)\Big)$
$-\mu\boldsymbol{q}_a(t)\mathbf{1}^T\Big(\boldsymbol{Q}_a(t) - \boldsymbol{q}_a(t)\boldsymbol{q}_a^T(t)\Big)E\big[\mathrm{diag}^2(\boldsymbol{u}(t))\big] - \mu\Big(\boldsymbol{Q}_a(t) - \boldsymbol{q}_a(t)\boldsymbol{q}_a^T(t)\Big)\boldsymbol{\Gamma}(t)\mathrm{diag}\big(\boldsymbol{q}_a(t)\big).$

In Table 3.1, we provide the mean and the variance recursions for $\boldsymbol{Q}_a(t)$ and $\boldsymbol{q}_a(t)$. To implement these recursions, one needs to only provide $\boldsymbol{\Gamma}(t)$ and $\boldsymbol{\gamma}(t)$. Note that $\boldsymbol{\Gamma}(t)$ and $\boldsymbol{\gamma}(t)$ are derived for a wide range of adaptive filters [25], [28]. If we use the mean and the variance recursions in (3.18), then we obtain the time evolution of the final MSE. This completes the transient analysis of the affinely constrained mixture weights updated with the EGU algorithm.

Relative Entropy

For the affinely constrained combination updated with the EG algorithm, we have the multiplicative updates as

$$
\begin{aligned}
\lambda_1^{(i)}(t+1) &= u\frac{\lambda_1^{(i)}(t)\exp\{\mu e(t)(\hat{y}_i(t) - \hat{y}_m(t))\}}{\sum_{k=1}^{m-1}\Big[\lambda_1^{(k)}(t)\exp\{\mu e(t)(\hat{y}_k(t) - \hat{y}_m(t))\} + \lambda_2^{(k)}(t)\exp\{-\mu e(t)(\hat{y}_k(t) - \hat{y}_m(t))\}\Big]}, \\
\lambda_2^{(i)}(t+1) &= u\frac{\lambda_2^{(i)}(t)\exp\{-\mu e(t)(\hat{y}_i(t) - \hat{y}_m(t))\}}{\sum_{k=1}^{m-1}\Big[\lambda_1^{(k)}(t)\exp\{\mu e(t)(\hat{y}_k(t) - \hat{y}_m(t))\} + \lambda_2^{(k)}(t)\exp\{-\mu e(t)(\hat{y}_k(t) - \hat{y}_m(t))\}\Big]},
\end{aligned}
$$

for $i = 1, \ldots, m-1$. Using the same approximations as in (3.7), (3.8), (3.9) and (3.10), we obtain

$$\lambda_1^{(i)}(t+1) \approx u \frac{\lambda_1^{(i)}(t)\big(1+\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\big)}{\sum_{k=1}^{m-1}\Big[\lambda_1^{(k)}(t)\big(1+\mu e(t)(\hat{y}_k(t)-\hat{y}_m(t))\big)+\lambda_2^{(k)}(t)\big(1-\mu e(t)(\hat{y}_k(t)-\hat{y}_m(t))\big)\Big]}, \tag{3.21}$$

$$\lambda_2^{(i)}(t+1) \approx u \frac{\lambda_2^{(i)}(t)\big(1-\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\big)}{\sum_{k=1}^{m-1}\Big[\lambda_1^{(k)}(t)\big(1+\mu e(t)(\hat{y}_k(t)-\hat{y}_m(t))\big)+\lambda_2^{(k)}(t)\big(1-\mu e(t)(\hat{y}_k(t)-\hat{y}_m(t))\big)\Big]}. \tag{3.22}$$

In our simulations, we illustrate the accuracy of the approximations (3.21) and (3.22) under the mixture framework. Using (3.21) and (3.22), we obtain updates on $\boldsymbol{\lambda}_1(t)$ and $\boldsymbol{\lambda}_2(t)$ as

$$\boldsymbol{\lambda}_1(t+1) = u \frac{\big(I+\mu e(t)\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\big)\boldsymbol{\lambda}_1(t)}{\big[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\big]\boldsymbol{\lambda}_a(t)}, \tag{3.23}$$

$$\boldsymbol{\lambda}_2(t+1) = u \frac{\big(I-\mu e(t)\mathrm{diag}\big(\boldsymbol{\delta}(t)\big)\big)\boldsymbol{\lambda}_2(t)}{\big[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\big]\boldsymbol{\lambda}_a(t)}. \tag{3.24}$$

Using updates (3.23) and (3.24), we can write update on $\boldsymbol{\lambda}_a(t)$

$$\boldsymbol{\lambda}_a(t+1) = u \frac{\big[I+\mu e(t)\mathrm{diag}\big(\boldsymbol{u}(t)\big)\big]\boldsymbol{\lambda}_a(t)}{\big[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\big]\boldsymbol{\lambda}_a(t)}. \tag{3.25}$$

For the recursion of $\bar{\boldsymbol{\lambda}}_a(t)$, using (3.25), we get

$$\begin{aligned} E\big[\boldsymbol{\lambda}_a(t+1)\big] &= E\left\{u\frac{\big[I+\mu e(t)\mathrm{diag}\big(\boldsymbol{u}(t)\big)\big]\boldsymbol{\lambda}_a(t)}{\big[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\big]\boldsymbol{\lambda}_a(t)}\right\}, \\ &\approx u\frac{E\big\{\big[I+\mu e(t)\mathrm{diag}\big(\boldsymbol{u}(t)\big)\big]\boldsymbol{\lambda}_a(t)\big\}}{E\big\{\big[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\big]\boldsymbol{\lambda}_a(t)\big\}}, \qquad (3.26) \\ &= u\frac{E\big[\boldsymbol{\lambda}_a(t)\big]+\mu\mathrm{diag}\big(\boldsymbol{\gamma}(t)\big)E\big[\boldsymbol{\lambda}_a(t)\big]-\mu\mathrm{diag}\big(E[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)]\boldsymbol{\Gamma}(t)\big)}{\big[\mathbf{1}^T+\mu\boldsymbol{\gamma}^T(t)\big]E\big[\boldsymbol{\lambda}_a(t)\big]-\mu\boldsymbol{t}_{r,n}\big(E[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)]\boldsymbol{\Gamma}(t)\big)}, \qquad (3.27) \end{aligned}$$

where in (3.26) we approximate expectation of the quotient with the quotient of the expectations. In our simulations, we also illustrate the accuracy of this approximation in the mixture framework. From (3.25), using the same approximation in (3.27), assuming $\boldsymbol{\lambda}_a(t)$ is Gaussian, assuming $\lambda_a^{(i)}(t)$ and $\lambda_a^{(j)}(t)$ are uncorrelated when $i \neq j$, and since μ is small, ignoring the terms that are proportional to μ^2, we get a recursion for $E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]$ as

$$E\big[\boldsymbol{\lambda}_a(t+1)\boldsymbol{\lambda}_a^T(t+1)\big] = u^2\frac{\boldsymbol{A}(t)}{b(t)}, \tag{3.28}$$

where $\boldsymbol{A}(t)$ is equal to the right hand side of (3.20) and

$$\begin{aligned} b(t) &= \mathbf{1}^T E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]\mathbf{1} + \mu \boldsymbol{p}^T(t) E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]\mathbf{1} \\ &- \mu \bar{\boldsymbol{\lambda}}_a^T(t)\boldsymbol{R}(t)E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]\mathbf{1} - \mu \mathbf{1}^T \boldsymbol{D}(t)\boldsymbol{R}(t)\bar{\boldsymbol{\lambda}}_a(t) \\ &- \mu \mathbf{1}^T \boldsymbol{D}(t) E\big[\mathrm{diag}^2(\boldsymbol{u}(t))\big]\mathbf{1}^T \bar{\boldsymbol{\lambda}}_a(t)\mathbf{1} \\ &+ \mu \mathbf{1}^T E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]\boldsymbol{p}(t) - \mu \mathbf{1}^T E\big[\boldsymbol{\lambda}_a(t)\boldsymbol{\lambda}_a^T(t)\big]\boldsymbol{R}(t)\bar{\boldsymbol{\lambda}}_a(t) \\ &- \mu \bar{\boldsymbol{\lambda}}_a^T(t)\boldsymbol{R}(t)\boldsymbol{D}(t)\mathbf{1} - \mu \mathbf{1}^T \bar{\boldsymbol{\lambda}}_a^T(t)\mathbf{1} E\big[\mathrm{diag}^2(\boldsymbol{u}(t))\big]\boldsymbol{D}(t). \end{aligned} \tag{3.29}$$

If we use the mean (3.27) and the variance (3.28), (3.29) recursions in (3.18), then we obtain the time evolution of the final MSE. This completes the transient analysis of the affinely constrained mixture weights updated with the EG algorithm.

3.3.2 Unconstrained Mixture

We use the unconstrained relative entropy and the relative entropy as distance measures to update unconstrained mixture weights. We first perform transient analysis of the mixture weights updated using the EGU algorithm. Then, we continue with the transient analysis of the mixture weights updated using the EG algorithm. Note that since the unconstrained case is close to the affinely constrained case, we only provide the necessary modifications to get the mean and the variance recursions for the transient analysis.

Unconstrained Relative Entropy

For the unconstrained combination updated with EGU, we have the multiplicative updates as

$$\begin{aligned} w_1^{(i)}(t+1) &= w_1^{(i)}(t)\exp\left\{\mu e(t)\hat{y}_i(t)\right\}, \\ w_2^{(i)}(t+1) &= w_2^{(i)}(t)\exp\left\{-\mu e(t)\hat{y}_i(t)\right\}, \end{aligned}$$

for $i = 1, \ldots, m$. Using the same approximations as in (3.7), (3.8), (3.9) and (3.10), we can obtain updates on $\boldsymbol{w}_1(t)$ and $\boldsymbol{w}_2(t)$ as

$$\boldsymbol{w}_1(t+1) \approx \big(I + \mu e(t)\mathrm{diag}\big(\boldsymbol{x}(t)\big)\big)\boldsymbol{w}_1(t), \tag{3.30}$$

$$\boldsymbol{w}_2(t+1) \approx \big(I - \mu e(t)\mathrm{diag}\big(\boldsymbol{x}(t)\big)\big)\boldsymbol{w}_2(t). \tag{3.31}$$

Collecting the weights in $\boldsymbol{w}_a(t) = [\boldsymbol{w}_1(t); \boldsymbol{w}_2(t)]$, using the updates (3.30) and (3.31), we can write update on $\boldsymbol{w}_a(t)$ as

$$\boldsymbol{w}_a(t+1) = \big(I + \mu e(t)\mathrm{diag}\big(\boldsymbol{u}(t)\big)\big)\boldsymbol{w}_a(t), \tag{3.32}$$

where $\boldsymbol{u}(t)$ is defined as $\boldsymbol{u}(t) \stackrel{\triangle}{=} [\boldsymbol{x}(t); -\boldsymbol{x}(t)]$.

For the desired signal $y(t)$, we can write $y(t) = \boldsymbol{w}_0^T(t)\boldsymbol{x}(t) + e_0(t)$, where $\boldsymbol{w}_0(t)$ is the optimum MSE solution at time t such that $\boldsymbol{w}_0(t) \stackrel{\triangle}{=} \boldsymbol{R}^{-1}(t)\boldsymbol{p}(t)$, $\boldsymbol{R}(t) \stackrel{\triangle}{=} E\big[\boldsymbol{x}(t)\boldsymbol{x}^T(t)\big]$, $\boldsymbol{p}(t) \stackrel{\triangle}{=} E\left\{\boldsymbol{x}(t)y(t)\right\}$ and $e_0(t)$ is zero-mean disturbance uncorrelated to $\boldsymbol{x}(t)$. To show that the mixture weights converge to the optimum solution in the steady-state such that $\lim_{t\to\infty} E\big[\boldsymbol{w}(t)\big] = \lim_{t\to\infty} \boldsymbol{w}_0(t)$, we follow similar lines as in the Section 4.1.1. We modify (3.14), (3.15), (3.16) and (3.17) such that $\boldsymbol{\lambda}$ will be replaced by $\boldsymbol{w}$, $\boldsymbol{\delta}(t)$ will be replaced by $\boldsymbol{x}(t)$ and $\boldsymbol{\varepsilon}(t) = \boldsymbol{w}_0(t) - \boldsymbol{w}(t)$. After these replacements, we obtain

$$\begin{aligned} E\big[\boldsymbol{\varepsilon}(t+1)\big] &= E\big[I - \mu\mathrm{diag}\big(\boldsymbol{w}_1(t) + \boldsymbol{w}_2(t)\big)\boldsymbol{x}(t)\boldsymbol{x}^T(t)\big]E\big[\boldsymbol{\varepsilon}(t)\big] \\ &+ E\big[\boldsymbol{w}_0(t+1) - \boldsymbol{w}_0(t)\big]. \end{aligned} \tag{3.33}$$

Since, we have $\lim_{t\to\infty} E\big[\boldsymbol{w}_0(t+1) - \boldsymbol{w}_0(t)\big] = 0$ for most adaptive filters in the first stage [28] and if μ is chosen so that all the eigenvalues of $E\big[I - \mu\mathrm{diag}\big(\boldsymbol{w}_1(t) + \boldsymbol{w}_2(t)\big)\boldsymbol{x}(t)\boldsymbol{x}^T(t)\big]$ have strictly less than unit magnitude for every t, then $\lim_{t\to\infty} E\big[\boldsymbol{w}(t)\big] = \lim_{t\to\infty} \boldsymbol{w}_0(t)$.

For the transient analysis of MSE, defining $\boldsymbol{\gamma}(t) \stackrel{\triangle}{=} E\left\{\boldsymbol{u}(t)y(t)\right\}$ and $\boldsymbol{\Gamma}(t) \stackrel{\triangle}{=} E\big[\boldsymbol{u}(t)\boldsymbol{u}^T(t)\big]$, (3.18) is modified as

$$E[e^2(t)] = E\left\{y^2(t)\right\} - 2\bar{\boldsymbol{w}}_a^T(t)\boldsymbol{\gamma}(t) + \boldsymbol{t}_{r,n}\Big(E\left[\boldsymbol{w}_a(t)\boldsymbol{w}_a^T(t)\right]\boldsymbol{\Gamma}(t)\Big). \tag{3.34}$$

Accordingly, we modify the mean recursion (3.19) and the variance recursion (3.20) such

that instead of $\boldsymbol{\lambda}_a(t)$ we use $\boldsymbol{w}_a(t)$. We also modify the Table 3.1 using $\boldsymbol{q}_a(t) \stackrel{\triangle}{=} \bar{\boldsymbol{w}}_a(t)$ and $\boldsymbol{Q}_a(t) \stackrel{\triangle}{=} E\big[\boldsymbol{w}_a(t)\boldsymbol{w}_a^T(t)\big]$. If we use this modified mean and variance recursions in (3.34), then we obtain the time evolution of the final MSE. This completes the transient analysis of the unconstrained mixture weights updated with the EGU algorithm.

Relative Entropy

For the unconstrained combination updated with the EG algorithm, we have the multiplicative updates as

$$w_a^{(i)}(t+1) = u\frac{w_a^{(i)}(t)\exp\{\mu e(t)\hat{y}_i(t)\}}{\sum_{k=1}^{m}\Big[w_a^{(k)}(t)\exp\{\mu e(t)\hat{y}_k(t)\} + w_a^{(k+m)}(t)\exp\{-\mu e(t)\hat{y}_k(t)\}\Big]},$$

$$i = 1, \ldots, m,$$

$$w_a^{(i)}(t+1) = u\frac{w_a^{(i)}(t)\exp\{-\mu e(t)\hat{y}_i(t)\}}{\sum_{k=1}^{m}\Big[w_a^{(k)}(t)\exp\{\mu e(t)\hat{y}_k(t)\} + w_a^{(k+m)}(t)\exp\{-\mu e(t)\hat{y}_k(t)\}\Big]},$$

$$i = m+1 \ldots, 2m.$$

Following similar lines, we modify (3.23), (3.24), (3.25), (3.27), (3.28) and (3.29) such that we replace $\boldsymbol{\delta}(t)$ with $\boldsymbol{x}(t)$, $\boldsymbol{\lambda}$ with $\boldsymbol{w}$ and $\boldsymbol{u}(t) = \big[\boldsymbol{x}(t); -\boldsymbol{x}(t)\big]$. Finally, we use the modified mean and variance recursions in (3.34) and obtain the time evolution of the final MSE. This completes the transient analysis of the unconstrained mixture weights updated with the EG algorithm.

3.4 Simulations

In this section, we illustrate the accuracy of our results and compare performances of different adaptive mixture methods through simulations. In our simulations, we observe that using the EG algorithm to train the mixture weights yields better performance compared to using the LMS algorithm or the EGU algorithm to train the mixture weights for combinations having more than two filters and when the combination favors only a few of the constituent filters. The LMS algorithm and the EGU algorithm perform similarly in our simulations when they are used to train the mixture weights. We also observe in our simula-

tions that the mixture weights under the EG update converge to the optimum combination vector faster than the mixture weights under the LMS algorithm.

To compare performances of the EG and LMS algorithms and illustrate the accuracy of our results in (3.27), (3.28) and (3.29) under different algorithmic parameters, the desired signal as well as the system parameters are selected as follows. First, a seventh-order linear filter,
$\boldsymbol{w}_o = [0.25, -0.47, -0.37, 0.045, -0.18, 0.78, 0.147]^T$, is chosen as in [24]. The underlying signal is generated using the data model $y(t) = \tau\, \boldsymbol{w}_o^T \boldsymbol{a}(t) + n(t)$, where $\boldsymbol{a}(t)$ is an i.i.d. Gaussian vector process with zero mean and unit variance entries, i.e., $E[\boldsymbol{a}(t)\boldsymbol{a}^T(t)] = \boldsymbol{I}$, $n(t)$ is an i.i.d. Gaussian noise process with zero mean and variance $E[n^2(t)] = 0.3$, and τ is a positive scalar to control SNR. Hence, the SNR of the desired signal is given by $\text{SNR} \triangleq 10\log(\frac{E[\tau^2(\boldsymbol{w}_o^T \boldsymbol{u}(t))^2]}{0.01}) = 10\log(\frac{\tau^2 \|\boldsymbol{w}_o\|^2}{0.01})$.

For the first experiment, we have SNR = 10dB. To model the unknown system we use four linear filters using the RLS algorithm, LMS algorithm, Sign-error LMS algorithm and Sign-sign LMS algorithm. We emphasize that depending on the underlying signal and/or application, one of these algorithms is preferable to the others, however, such a selection is only possible in hindsight. Hence, an adaptive combination could resolve such uncertainty [22]. In this experiment, there is a sudden change in the desired signal such that the target $\boldsymbol{w}_0$ changes in the middle of the simulations as seen in Fig. 3.2. In the start of the simulations, the desired signal is generated from a seventh-order linear filter $\boldsymbol{w}_0 = [0.25, -0.47, -0.37, 0.045, -0.18, 0.78, 0.147]^T$ [24], which is then replaced by $\boldsymbol{w}_0 = [0.62, 0.81, -0.74, 0.82, 0.26, -0.80, -0.44]^T$ at the 4000th sample. The constituent RLS algorithm is initialized after $\boldsymbol{w}_0$ is updated. The learning rates of these constituent filters are set to $\mu_{\text{LMS}} = 0.12$, $\mu_{\text{Sign-errorLMS}} = 0.11$ and $\mu_{\text{Sign-signLMS}} = 0.1$. The parameters for the RLS algorithm are set to $\lambda = 1$ and $\epsilon = 20$. Therefore, in the steady-state, we obtain the optimum combination vector approximately as $\boldsymbol{\lambda}_o = [1, 0, 0, 0]^T$, i.e., the final combination vector is sparse. In the second stage, we train the combination weights with the EG and LMS algorithms and compare performances of these algorithms. The EG algorithm has two parameters to adjust while the LMS algorithm has only one parameter to adjust. For the second stage, the learning rates for the EG and LMS algorithms are selected as $\mu_{\text{EG}} = 0.001$ and $\mu_{\text{LMS}} = 0.01$ such that the EMSEs of both mixtures converge to the same final EMSE

to provide a fair comparison. However, there exist a wide range of values for the step sizes so that the algorithms converge to very similar EMSEs. We select $u = 50$ for the EG algorithm. In Fig. 3.2a, we plot the weight of the RLS filter, i.e. $E[\boldsymbol{\lambda}^{(1)}(t)]$, updated with the EG and LMS algorithms. In Fig. 3.2b, we plot the EMSE curves for the adaptive mixture updated with the EG algorithm, the adaptive mixture updated with the LMS algorithm, the RLS filter with $\lambda = 1$ and $\epsilon = 20$, the Sign-error LMS filter with $\mu_{\text{Sign-errorLMS}} = 0.11$ and the LMS filter with $\mu_{\text{LMS}} = 0.12$. From Fig. 3.2a and Fig. 3.2b, we see that the EG algorithm performs better than the LMS algorithm such that the combination weight under the update of the EG algorithm converges to 1 faster than the combination weight under the update of the LMS algorithm. We also observe from these simulations that even after the sudden change in the statistics, the EG algorithm quickly recovers and performs better than the LMS algorithm. Furthermore the EMSE of the adaptive mixture updated with the EG algorithm converges faster than the EMSE of the adaptive mixture updated with the LMS algorithm. In Fig. 3.2c, we plot the theoretical values for $\bar{\lambda}_a^{(1)}(t)$ and $\bar{\lambda}_a^{(4)}(t)$ along with simulations. Note that in Fig. 3.2c we observe that $\bar{\lambda}^{(1)}(t) = \bar{\lambda}_a^{(1)}(t) - \bar{\lambda}_a^{(4)}(t)$ converges to 1 as predicted in our derivations. In Fig. 3.2d, we plot the theoretical values of $E\big[\lambda_a^{(1)}(t)^2\big]$ and $E\big[\lambda_a^{(2)}(t)\lambda_a^{(4)}(t)\big]$ along with simulations. As we observe from Fig. 3.2c and Fig. 3.2d, there is a close agreement between our results and simulations in these experiments. We observe similar results for the other cross terms.

We next model the unknown system using ten linear filters with the LMS update as the constituent filters. For this experiment, we have SNR = -10dB. The learning rates of two constituent filters are set to $\mu_1 = 0.002$ and $\mu_6 = 0.002$ while the learning rates for the rest of the constituent filters are selected randomly in $[0.1, 0.11]$. Therefore, in the steady-state, we obtain the optimum combination vector approximately as $\boldsymbol{\lambda}_o = [0.5, 0, 0, 0, 0, 0.5, 0, 0, 0, 0]^T$, i.e., the final combination vector is sparse. In the second stage, we train the combination weights with the EG and LMS algorithms and compare performances of these algorithms. For the second stage, the learning rates for the EG and LMS algorithms are selected as $\mu_{\text{EG}} = 0.0005$ and $\mu_{\text{LMS}} = 0.005$ such that the EMSEs of both mixtures converge to the same final EMSE to provide a fair comparison. However, there exist a wide range of values for the step sizes so that the algorithms converge to very similar EMSEs. We select $u = 500$ for the EG algorithm. In Fig. 3.3a, we plot the weight of the first constituent filter with

$\mu_1 = 0.002$, i.e. $E[\boldsymbol{\lambda}^{(1)}(t)]$, updated with the EG and LMS algorithms. In Fig. 3.3b, we plot the EMSE curves for the adaptive mixture updated with the EG algorithm, the adaptive mixture updated with the LMS algorithm, the first constituent filter with $\mu_1 = 0.002$ and the second constituent filter with $\mu_2 \in [0.1, 0.11]$. From Fig. 3.3a and Fig. 3.3b, we see that the EG algorithm performs better than the LMS algorithm such that the combination weight under the update of the EG algorithm converges to 0.5 faster than the combination weight under the update of the LMS algorithm. Furthermore the EMSE of the adaptive mixture updated with the EG algorithm converges faster than the EMSE of the adaptive mixture updated with the LMS algorithm. In Fig. 3.3c, to test the accuracy of (3.27), we plot the theoretical values for $\bar{\lambda}_a^{(1)}(t)$ and $\bar{\lambda}_a^{(10)}(t)$ along with simulations. Note in Fig. 3.3c we observe that $\bar{\lambda}^{(1)}(t) = \bar{\lambda}_a^{(1)}(t) - \bar{\lambda}_a^{(10)}(t)$ converges to 0.5 as predicted in our derivations. In Fig. 3.3d, to test the accuracy of (3.28) and (3.29), as an example, we plot the theoretical values of $E\big[\lambda_a^{(1)}(t)^2\big]$ and $E\big[\lambda_a^{(1)}(t)\lambda_a^{(3)}(t)\big]$ along with simulations. As we observe from Fig. 3.3c and Fig. 3.3d, there is a close agreement between our results and simulations in these experiments. We observe similar results for the other cross terms.

We next simulate the unconstrained mixtures updated with the EGU and EG algorithms. Here, we have two linear filters and both using the LMS update to train their weight vectors as the constituent filters. The learning rates for two constituent filters are set to $\mu_1 = 0.002$ and $\mu_2 = 0.1$ respectively. Therefore, in the steady-state, we obtain the optimum vector approximately as $\boldsymbol{w}_o = [1, 0]$. We have SNR = 1 for these simulations. The unconstrained mixture weights are first updated with the EGU algorithm. For the second stage, the learning rate for the EGU algorithm is selected as $\mu_{\text{EGU}} = 0.01$. The theoretical curves in the figures are produced using $\boldsymbol{\Gamma}(t)$ and $\boldsymbol{\gamma}(t)$ that are calculated from the simulations, since our goal is to illustrate the validity of derived equations. In Fig. 3.4a, we plot the theoretical values of $\bar{\boldsymbol{w}}_a^{(1)}(t)$, $\bar{\boldsymbol{w}}_a^{(2)}(t)$, $\bar{\boldsymbol{w}}_a^{(3)}(t)$ and $\bar{\boldsymbol{w}}_a^{(4)}(t)$ along with simulations. In Fig. 3.4b, as an example, we plot the theoretical values of $E\big[\boldsymbol{w}_a^{(1)}(t)^2\big]$, $E\big[\boldsymbol{w}_a^{(1)}(t)\boldsymbol{w}_a^{(2)}(t)\big]$, $E\big[\boldsymbol{w}_a^{(2)}(t)\boldsymbol{w}_a^{(3)}(t)\big]$ and $E\big[\boldsymbol{w}_a^{(3)}(t)\boldsymbol{w}_a^{(4)}(t)\big]$ along with simulations. We continue to update the mixture weights with the EG algorithm. For the second stage, the learning rate for the EG algorithm is selected as $\mu_{\text{EG}} = 0.01$. We select $u = 3$ for the EG algorithm. In Fig. 3.4c, we plot the theoretical values of $\bar{\boldsymbol{w}}_a^{(1)}(t)$, $\bar{\boldsymbol{w}}_a^{(2)}(t)$, $\bar{\boldsymbol{w}}_a^{(3)}(t)$ and $\bar{\boldsymbol{w}}_a^{(4)}(t)$ along with simulations. In Fig. 3.4d, as an example, we plot the theoretical values of $E\big[\boldsymbol{w}_a^{(2)}(t)^2\big]$,

$E\big[\boldsymbol{w}_a^{(1)}(t)\boldsymbol{w}_a^{(2)}(t)\big]$, $E\big[\boldsymbol{w}_a^{(2)}(t)\boldsymbol{w}_a^{(3)}(t)\big]$ and $E\big[\boldsymbol{w}_a^{(2)}(t)\boldsymbol{w}_a^{(4)}(t)\big]$ along with simulations. We observe a close agreement between our results and simulations.

To test the accuracy of the assumptions in (3.9) and (3.10), we plot in Fig. 3.5a, the difference

$$\frac{\| \exp\{\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\} - \{1+\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\} \|^2}{\sqrt{\| \exp\{\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\} \|^2 \| \{1+\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\} \|^2}}$$

for $i = 1$ with the same algorithmic parameters as in Fig. 3.3 and Fig. 3.4. To test the accuracy of the separation assumption in (3.27), we plot in Fig. 3.5b, the first parameter of the difference

$$\frac{\left\| E\left\{ u\frac{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}(t)\right)\right]\boldsymbol{\lambda}_a(t)}{\left[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\right]\boldsymbol{\lambda}_a(t)}\right\} - u\frac{E\left\{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}(t)\right)\right]\boldsymbol{\lambda}_a(t)\right\}}{E\left\{\left[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\right]\boldsymbol{\lambda}_a(t)\right\}}\right\|^2}{\sqrt{\left\| E\left\{ u\frac{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}(t)\right)\right]\boldsymbol{\lambda}_a(t)}{\left[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\right]\boldsymbol{\lambda}_a(t)}\right\}\right\|^2 \left\| u\frac{E\left\{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}(t)\right)\right]\boldsymbol{\lambda}_a(t)\right\}}{E\left\{\left[\mathbf{1}^T+\mu e(t)\boldsymbol{u}^T(t)\right]\boldsymbol{\lambda}_a(t)\right\}}\right\|^2}}$$

with the same algorithmic parameters as in Fig. 3.3 and Fig. 3.4. We observe that assumptions are fairly accurate for these algorithms in our simulations.

To illustrate the assumption that $\boldsymbol{\lambda}_a(t)$ have Gaussian distribution, we calculate the kurtosis of the "empirical" distribution of $\boldsymbol{\lambda}_a(t)$ under the setup of the chapter. Note that although not rigorous, the kurtosis is often used to measure the closeness of an empirical distribution to a Gaussian distribution [14,19]. For this experiment, we collect 2000 samples of $\boldsymbol{\lambda}_a(t)$ under the same algorithmic framework as in Fig. 3.4 and report the kurtosis values for randomly chosen t's. The corresponding kurtosis values are provided as a table in Fig. 3.6. As we observe from Table 3.6, the kurtosis values are close to 3 supporting the assumption that $\boldsymbol{\lambda}_a(t)$ follows Gaussian distribution.

To illustrate the assumption of $\lambda_a^{(i)}(t)$ and $\lambda_a^{(j)}(t)$ are uncorrelated for $j \neq i$, we perform 1000 iterations and plot the ensemble averaged curves that correspond to the difference $\frac{\|E[\lambda_a^{(i)}(t)\lambda_a^{(j)}(t)]-E[\lambda_a^{(i)}(t)]E[\lambda_a^{(j)}(t)]\|}{\sqrt{\|E[\lambda_a^{(i)}(t)\lambda_a^{(j)}(t)]\|^2\|E[\lambda_a^{(i)}(t)]E[\lambda_a^{(j)}(t)]\|^2}}$ for different randomly chosen i and j parameters with the same algorithmic parameters as in Fig. 3.3 both for the EG and EGU algorithms. In Fig. 3.7, we plot this difference for $\lambda_1^{(1)}(t)-\lambda_2^{(1)}(t)$ and $\lambda_1^{(2)}(t)-\lambda_2^{(2)}(t)$ pairs. We also plot the difference for $\lambda_1^{(3)}(t)-\lambda_1^{(8)}(t)$ and $\lambda_1^{(4)}(t)-\lambda_2^{(6)}(t)$ pairs. As we observe from the plots that it is reasonable to use this assumption to approximate the expectation of the product as the product of the expectations.

In the last simulations, we compare performances of the EGU, EG and LMS algorithms updating the affinely mixture weights under different algorithmic parameters. Algorithmic parameters and constituent filters are selected as in Fig. 3.3 under SNR = -5 and 5. For the second stage, under SNR = -5, learning rates for the EG, EGU and LMS algorithms are selected as $\mu_{\text{EG}} = 0.0005$, $\mu_{\text{EGU}} = 0.005$ and $\mu_{\text{LMS}} = 0.005$ such that the EMSEs converge to the same final EMSE to provide a fair comparison. However, there exist a wide range of values for the step sizes so that the algorithms converge to very similar EMSEs. We choose u = 500 for the EG algorithm. In Fig. 3.8a, we plot the EMSE curves for the adaptive mixture updated with the EG algorithm, the adaptive mixture updated with the EGU algorithm, the adaptive mixture updated with the LMS algorithm, first constituent filter with $\mu_1 = 0.002$ and second constituent filter with $\mu_2 \in [0.1, 0.11]$ under SNR = -5. Under SNR = 5, learning rates for the EG, EGU and LMS algorithms are selected as $\mu_{\text{EG}} = 0.002$, $\mu_{\text{EGU}} = 0.005$ and $\mu_{\text{LMS}} = 0.005$. We choose u = 100 for the EG algorithm. In Fig. 3.8b, we plot same EMSE curves as in Fig. 3.8a. We observe that the EG algorithm performs better than the EGU and LMS algorithms such that EMSE of the adaptive mixture updated with the EG algorithm converges faster than the EMSE of adaptive mixtures updated with the EGU and LMS algorithms. We also observe that the EGU and LMS algorithms show similar performances when they are used to train the mixture weights.

3.5 Conclusion

In this chapter, we investigated adaptive mixture methods based on Bregman divergences combining outputs of m adaptive filters to model a desired signal. We used the unnormalized relative entropy and relative entropy as distance measures that produce the exponentiated gradient update with unnormalized weights (EGU) and the exponentiated gradient update with positive and negative weights (EG) to train the mixture weights under the affine constraints or without any constraints. We provided the transient analysis of these methods updated with the EGU and EG algorithms. In our simulations, we compared performances of the EG, EGU and LMS algorithms and observe that the EG algorithm performs better than the EGU and LMS algorithms when the combination vector in steady-state is sparse. We observe that the EGU and LMS algorithms show similar performance when they are used to train the mixture weights. We also observe a close agreement between the simulations

and our theoretical results.

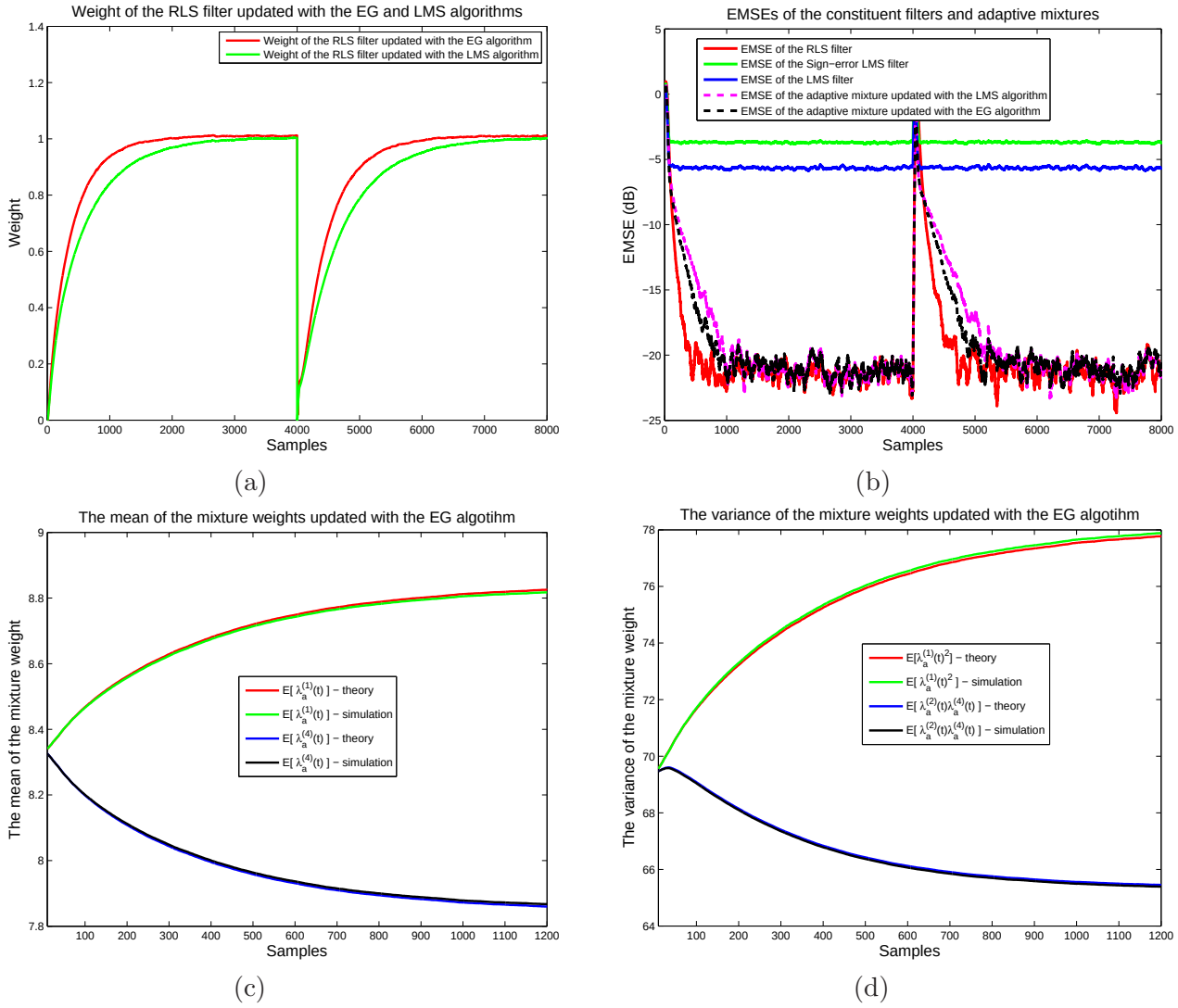


Figure 3.2: Using RLS, LMS, Sign-error LMS, Sign-sign LMS filters as constituent filters, where learning rates are $\mu_{\text{LMS}} = 0.12$, $\mu_{\text{Sign-errorLMS}} = 0.11$ and $\mu_{\text{Sign-signLMS}} = 0.1$. For the RLS filter, $\lambda = 1$ and $\epsilon = 20$. SNR = 10dB. For the mixture stage, the EG algorithm has $\mu_{\text{EG}} = 0.001$ and the LMS algorithm has $\mu_{\text{LMS}} = 0.01$. For the EG algorithm, $u = 50$. (a) The weight of the RLS filter in the mixture, i.e., $E[\boldsymbol{\lambda}^{(1)}(t)]$. (b) The EMSE curves for adaptive mixture updated with the EG algorithm, the adaptive mixture updated with the LMS algorithm, the RLS filter, the Sign-error LMS filter and the LMS filter. (c) Theoretical values $\bar{\lambda}_a^{(1)}(t)$ and $\bar{\lambda}_a^{(4)}(t)$ and simulations. (d) Theoretical values $E\big[\lambda_a^{(1)}(t)^2\big]$ and $E\big[\lambda_a^{(2)}(t)\lambda_a^{(4)}(t)\big]$ and simulations.

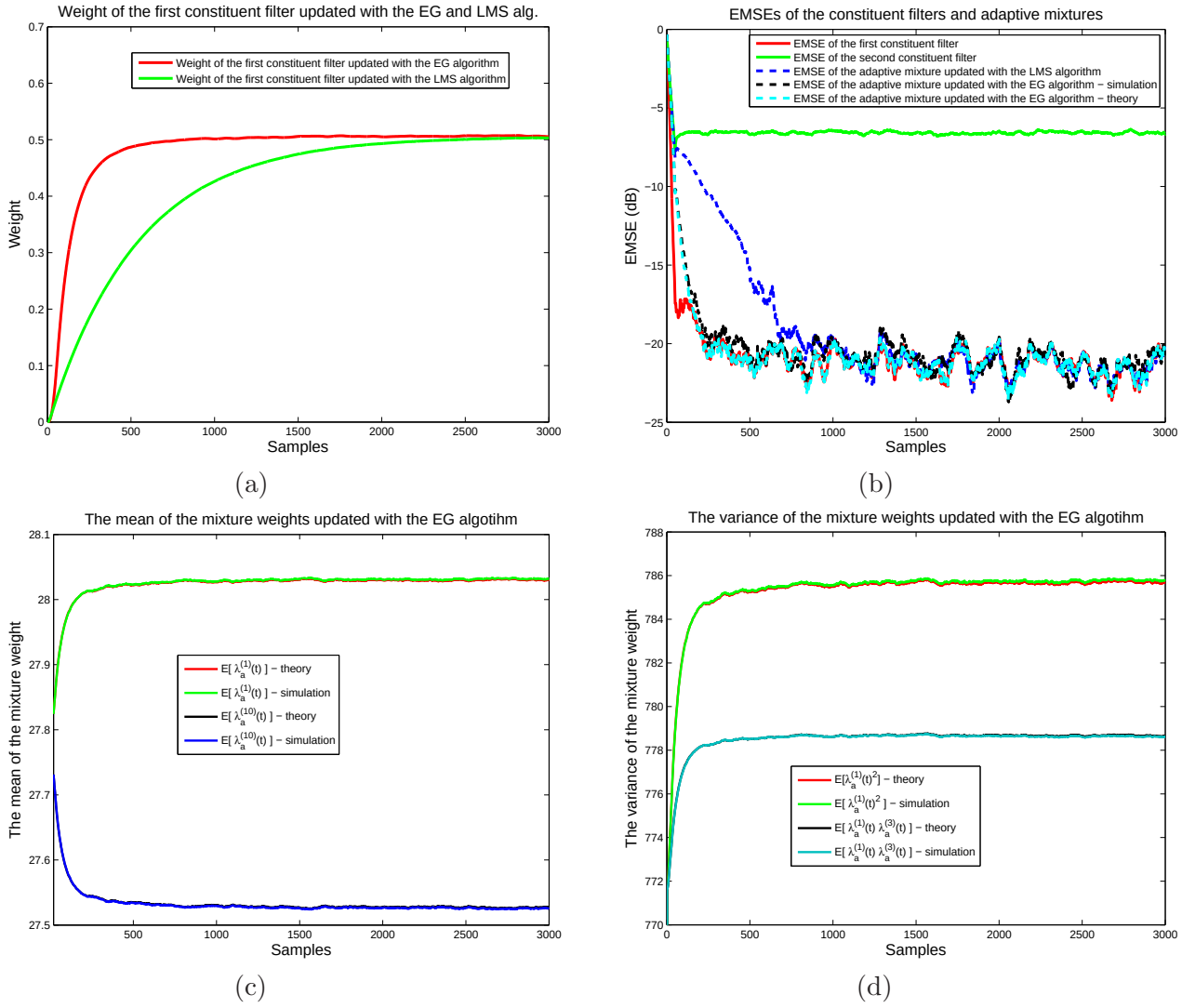


Figure 3.3: Using 10 LMS filters as constituent filters, where learning rates for 2 constituent filters are $\mu = 0.002$ and for the rest are $\mu \in [0.1, 0.11]$. SNR = -10dB. For the mixture stage, the EG algorithm has $\mu_{\text{EG}} = 0.0005$ and the LMS algorithm has $\mu_{\text{LMS}} = 0.005$. For the EG algorithm, $u = 500$. (a) The weight of the first constituent filter in the mixture, i.e., $E[\boldsymbol{\lambda}^{(1)}(t)]$. (b) The EMSE curves for adaptive mixture updated with the EG algorithm, the adaptive mixture updated with the LMS algorithm, the first constituent filter and the second constituent filter. (c) Theoretical values $\bar{\lambda}_a^{(1)}(t)$ and $\bar{\lambda}_a^{(10)}(t)$ and simulations. (d) Theoretical values $E\left[\lambda_a^{(1)}(t)^2\right]$ and $E\left[\lambda_a^{(1)}(t)\lambda_a^{(3)}(t)\right]$ and simulations.

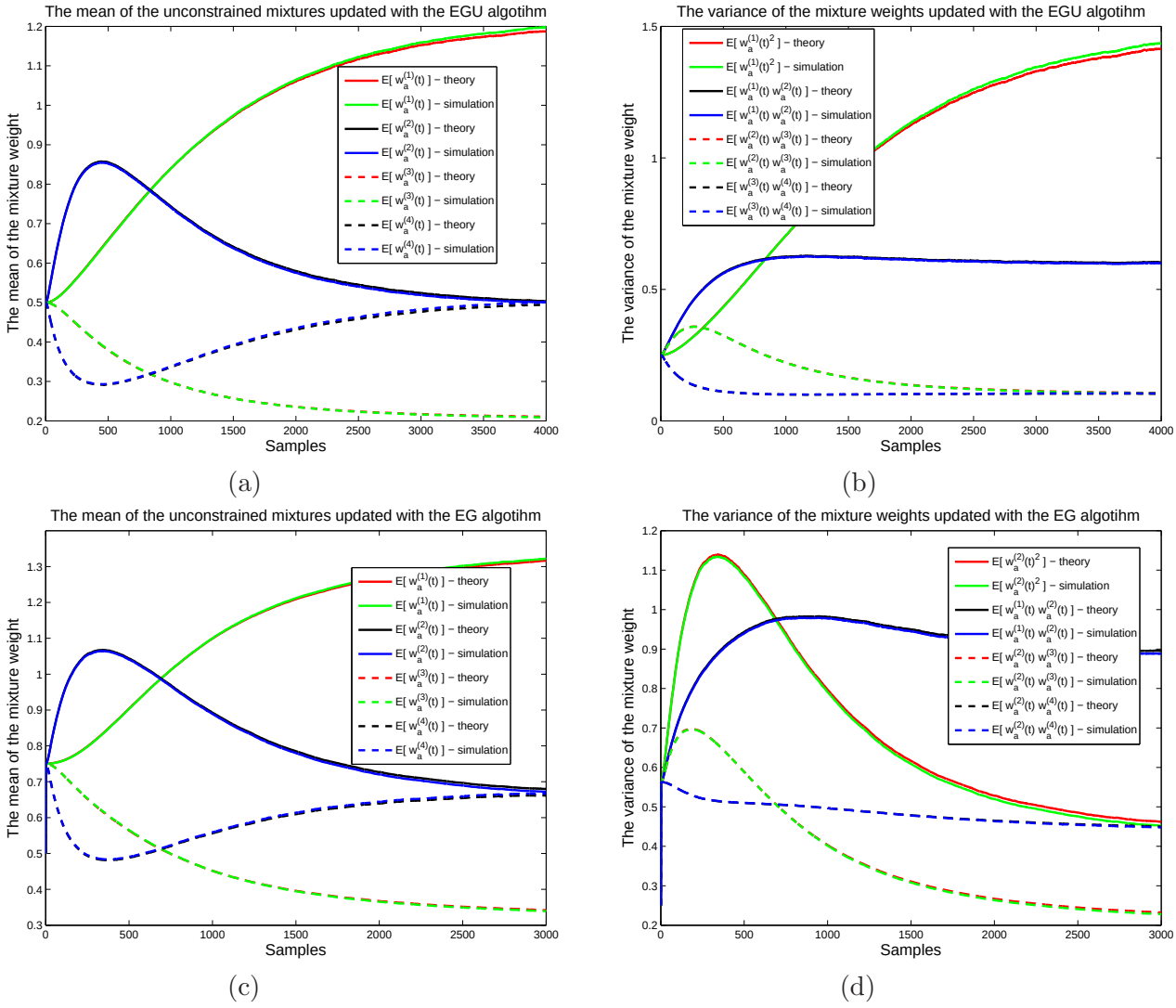
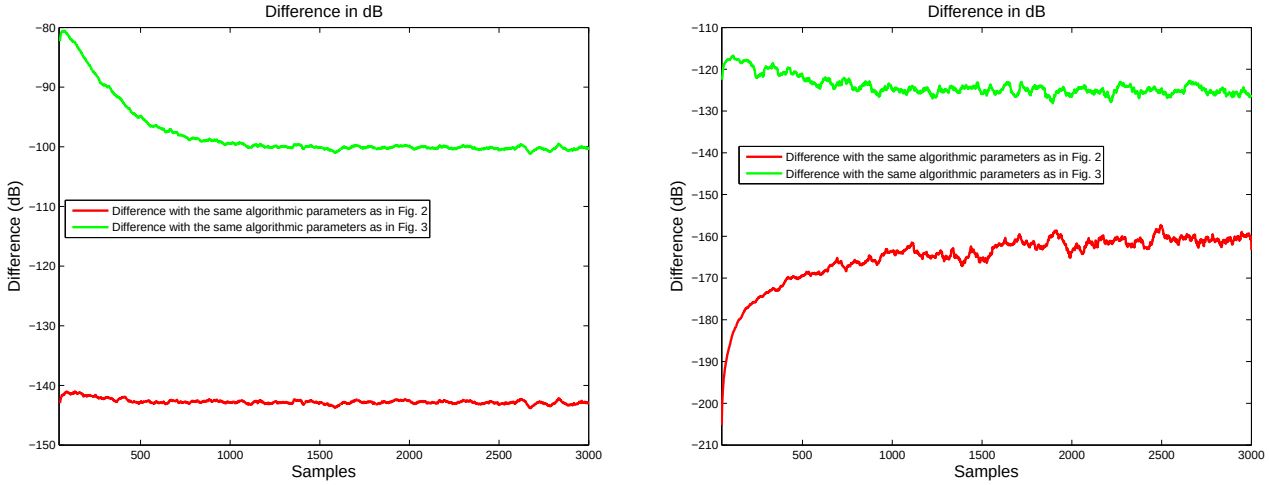


Figure 3.4: Two LMS filters as constituent filters with learning rates $\mu_1 = 0.002$ and $\mu_2 = 0.1$, respectively. SNR = 1dB. For the second stage, the EGU algorithm has $\mu_{\text{EGU}} = 0.01$ and the EG algorithm has $\mu_{\text{EG}} = 0.01$. For the EG algorithm, $u = 3$. (a) Theoretical values for the mixture weights updated with the EGU algorithm and simulations. (b) Theoretical values $E\big[\boldsymbol{w}_a^{(1)}(t)^2\big]$, $E\big[\boldsymbol{w}_a^{(1)}(t)\boldsymbol{w}_a^{(2)}(t)\big]$, $E\big[\boldsymbol{w}_a^{(2)}(t)\boldsymbol{w}_a^{(3)}(t)\big]$ and $E\big[\boldsymbol{w}_a^{(3)}(t)\boldsymbol{w}_a^{(4)}(t)\big]$ and simulations. (c) Theoretical mixture weights updated with the EG algorithm and simulations. (d) Theoretical values $E\big[\boldsymbol{w}_a^{(2)}(t)^2\big]$, $E\big[\boldsymbol{w}_a^{(1)}(t)\boldsymbol{w}_a^{(2)}(t)\big]$, $E\big[\boldsymbol{w}_a^{(2)}(t)\boldsymbol{w}_a^{(3)}(t)\big]$ and $E\big[\boldsymbol{w}_a^{(2)}(t)\boldsymbol{w}_a^{(4)}(t)\big]$ and simulations.

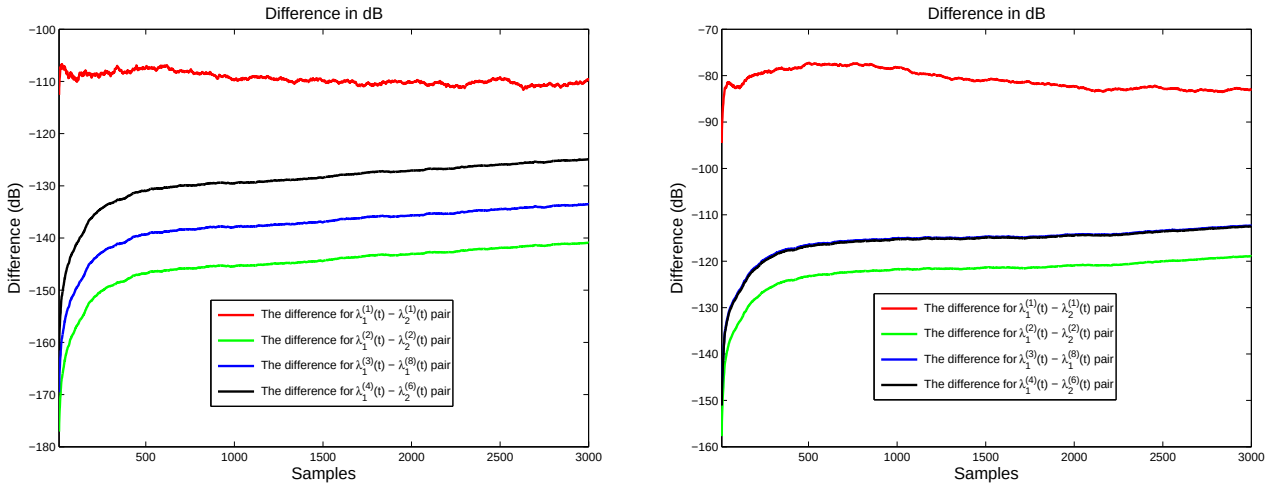


(a) (b)

Figure 3.5: (a) The difference $\frac{\|\exp\{\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\}-\{1+\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\}\|^2}{\sqrt{\|\exp\{\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\}\|^2\|\{1+\mu e(t)(\hat{y}_i(t)-\hat{y}_m(t))\}\|^2}}$ for $i = 1$ with the same algorithmic parameters as in Fig. 3.3 and Fig. 3.4. (b) The first parameter of the difference $\frac{\left\|E\left\{u\frac{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}_{(t)}\right)\right]\boldsymbol{\lambda}_{a(t)}}{\left[\mathbf{1}^T{}_{+\mu e(t)\boldsymbol{u}^T(t)}\right]\boldsymbol{\lambda}_{a(t)}}\right\}-u\frac{E\left\{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}_{(t)}\right)\right]\boldsymbol{\lambda}_{a(t)}\right\}}{E\left\{\left[\mathbf{1}^T{}_{+\mu e(t)\boldsymbol{u}^T(t)}\right]\boldsymbol{\lambda}_{a(t)}\right\}}\right\|^2}{\sqrt{\left\|E\left\{u\frac{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}_{(t)}\right)\right]\boldsymbol{\lambda}_{a(t)}}{\left[\mathbf{1}^T{}_{+\mu e(t)\boldsymbol{u}^T(t)}\right]\boldsymbol{\lambda}_{a(t)}}\right\}\right\|^2\left\|u\frac{E\left\{\left[I+\mu e(t)\mathrm{diag}\left(\boldsymbol{u}_{(t)}\right)\right]\boldsymbol{\lambda}_{a(t)}\right\}}{E\left\{\left[\mathbf{1}^T{}_{+\mu e(t)\boldsymbol{u}^T(t)}\right]\boldsymbol{\lambda}_{a(t)}\right\}}\right\|^2}}$ with the same algorithmic parameters as in Fig. 3.3 and Fig. 3.4.

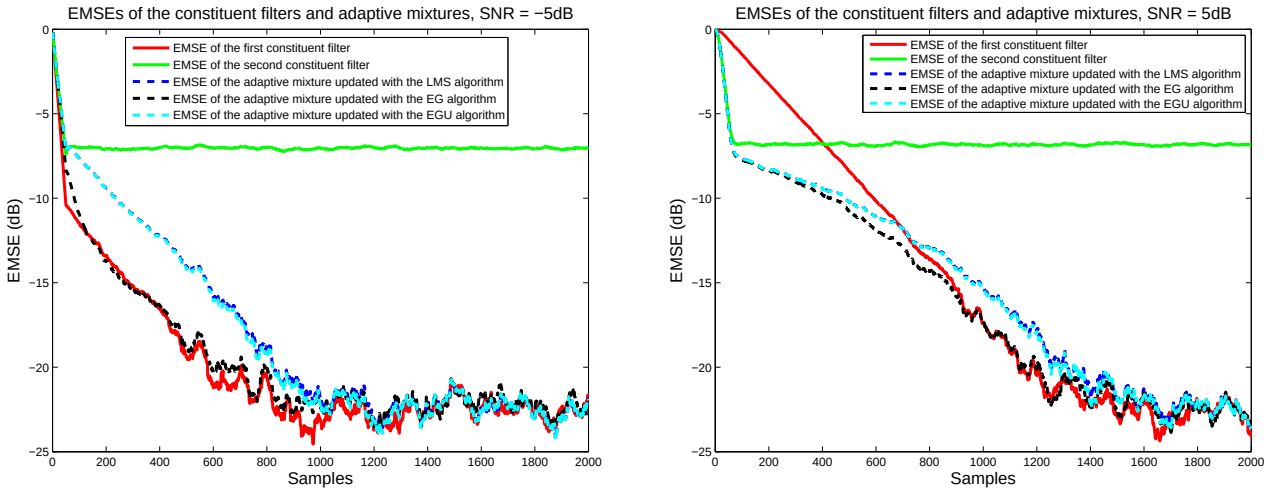
	t = 150	t = 320	t = 560	t = 920	t = 1210	t = 1440	t = 1760
$\boldsymbol{\lambda}_a^{(1)}(t)$	2,94	3,25	3,20	2,93	2,90	3,00	2,97
$\boldsymbol{\lambda}_a^{(2)}(t)$	2,71	3,05	3,43	3,02	2,96	3,13	2,88

Figure 3.6: Empirical kurtosis values. Experimental setup is from Fig. 3.4.



(a) (b)

Figure 3.7: (a) The difference $\frac{\|E[\lambda_a^{(i)}(t)\lambda_a^{(j)}(t)] - E[\lambda_a^{(i)}(t)]E[\lambda_a^{(j)}(t)]\|}{\sqrt{\|E[\lambda_a^{(i)}(t)\lambda_a^{(j)}(t)]\|^2 \|E[\lambda_a^{(i)}(t)]E[\lambda_a^{(j)}(t)]\|^2}}$ for different randomly chosen i and j parameters with the same algorithmic parameters as in Fig. 3.8(a) for the EG algorithm. (b) The difference $\frac{\|E[\lambda_a^{(i)}(t)\lambda_a^{(j)}(t)] - E[\lambda_a^{(i)}(t)]E[\lambda_a^{(j)}(t)]\|}{\sqrt{\|E[\lambda_a^{(i)}(t)\lambda_a^{(j)}(t)]\|^2 \|E[\lambda_a^{(i)}(t)]E[\lambda_a^{(j)}(t)]\|^2}}$ for different randomly chosen i and j parameters with the same algorithmic parameters as in Fig. 3.8(a) for the EGU algorithm.



(a) (b)

Figure 3.8: Algorithmic parameters and constituent filters are selected as in Fig. 3.3 under SNR = -5dB. For the second stage, the EG algorithm has $\mu_{\mathrm{EG}} = 0.0005$, the EGU algorithm has $\mu_{\mathrm{EGU}} = 0.005$ and the LMS algorithm has $\mu_{\mathrm{LMS}} = 0.005$. For the EG algorithm, u = 500. (a) the EMSE curves for the adaptive mixture updated with the EG algorithm, the adaptive mixture updated with the EGU algorithm, the adaptive mixture updated with the LMS algorithm (approximately same as the EGU algorithm), the first constituent filter and the second constituent filter. Next, SNR = 5dB. For the second stage, the EG algorithm has $\mu_{\mathrm{EG}} = 0.002$, the EGU algorithm has $\mu_{\mathrm{EGU}} = 0.005$ and the LMS algorithm has $\mu_{\mathrm{LMS}} = 0.005$. For the EG algorithm, u = 100. (b) the EMSE curves for the adaptive mixture updated with the EG algorithm, the adaptive mixture updated with the EGU algorithm, the adaptive mixture updated with the LMS algorithm (approximately same as the EGU algorithm), the first constituent filter and the second constituent filter.

Chapter 4

CONCLUSIONS

In this thesis, we study convex, affine and linear combination methods that adaptively combine outputs of several adaptive filters working in parallel on the same task. Chapter 2 deals with four different convex mixture methods and presents their steady-state and transient MSE performances. Chapter 3 is dedicated to affine and linear mixture methods based on Bregman divergences and provides their mean and mean-square transient analyses.

In the first chapter, we investigate four convex combination methods to adaptively combine outputs of two adaptive filters running in parallel to model an unknown system. We first perform the steady-state MSE analysis and provide the corresponding MSEs and the mixture weights of the combination algorithms in the steady-state under nonstationary random walk model. We next present the mean and the mean-square transient analysis for the studied algorithms. We observe that these convexly constrained combination methods are universal such that they achieve the performance of the best constituent filter in the steady-state. We observe that the EG update (2.6) under the mixture of experts framework can also outperform the best constituent filter under certain configuration of the EMSEs of the constituent filters (similar to the algorithm from [1]). We also demonstrate that the MSE in the steady-state of the algorithms from [30] and [27] heavily depends on the corresponding algorithmic parameters, i.e., the forgetting factor in [30] and the window length in [27]. We observe that our derivations accurately describe the behavior of all algorithms under the setup of [1].

In the second chapter, we analyze affine and linear mixture methods based on Bregman divergences combining outputs of several parallel running adaptive filters to model an unknown desired system. We use the unnormalized relative entropy and relative entropy and propose the exponentiated gradient update with unnormalized weights (EGU) and the exponentiated gradient update with positive and negative weights (EG) to update the convex weights under the affine constraint or without any constraints. We present the mean and

mean-square transient analyses of the studied algorithms. In our simulations, we observe that our derivations accurately describe the behavior of the EGU and EG algorithms.

Chapter 5

APPENDIX A

1) For the update (2.4), we have

$$\begin{aligned}\lambda_\rho(t+1) &= \frac{\lambda_\rho(t)\exp[\mu_\rho e(t)\hat{d}_1(t)]}{\lambda_\rho(t)\exp[\mu_\rho e(t)\hat{d}_1(t)] + (1-\lambda_\rho(t))\exp[\mu_\rho e(t)\hat{d}_2(t)]} \\ &= \frac{1}{1+\frac{1-\lambda_\rho(t)}{\lambda_\rho(t)}\exp[-\mu_\rho e(t)[\hat{d}_1(t)-\hat{d}_2(t)]]} \\ &= \frac{1}{1+\exp[-\rho(t)]\exp[-\mu_\rho e(t)[\hat{d}_1(t)-\hat{d}_2(t)]]} = \frac{1}{1+\exp[-\rho(t+1)]}\end{aligned}$$

with $\exp[-\rho(t)] = \frac{1-\lambda_\rho(t)}{\lambda_\rho(t)}$ and $\rho(t+1) \stackrel{\triangle}{=} \rho(t) + \mu_\rho e(t)[\hat{d}_1(t) - \hat{d}_2(t)]]$. □

2) For the update (2.7), we have

$$\begin{aligned}\lambda_\epsilon(t+1) &= \frac{\exp[-\mu_\epsilon \sum_{i=1}^{t} a^{(t-i)} e_1^2(i)]}{\exp[-\mu_\epsilon \sum_{i=1}^{t} a^{(t-i)} e_1^2(i)] + \exp[-\mu_\epsilon \sum_{i=1}^{t} a^{(t-i)} e_2^2(i)]} \\ &= \frac{1}{1+\exp[-\mu_\epsilon \sum_{i=1}^{t} a^{(t-i)}(e_2^2(i) - e_1^2(i))]} \\ &= \frac{1}{1+\exp[-a\epsilon(t) - \mu_\epsilon(e_2^2(t) - e_1^2(t))]} = \frac{1}{1+\exp[-\epsilon(t+1)]}\end{aligned}$$

with $\exp[-\epsilon(t)] = \frac{1-\lambda_\epsilon(t)}{\lambda_\epsilon(t)}$ and $\epsilon(t+1) \stackrel{\triangle}{=} a\epsilon(t) + \mu_\epsilon(e_2^2(t) - e_1^2(t))$. □

3) Let $s(t) = \sum_{i=0}^{t} a^{t-i} b(i)$ where $0 < a < 1$ and $b(t) \to b$ as $t \to \infty$. If $d(t) \stackrel{\triangle}{=} s(t) - c(t)$ where $c(t) \stackrel{\triangle}{=} b \sum_{i=0}^{t} a^i$, then we get $d(t+1) = ad(t) + b(t+1) - b$. Hence, we have

$$|d(t+1)| \leq |a||d(t)| + |b(t+1) - b|$$

by the triangular inequality. If $l \stackrel{\triangle}{=} \limsup_t d(t)$, then $|l| \leq |a||l|$ where we use $\limsup_t |b(t+1) - b| = 0$. Since $a < 1$, we get $l = 0$. Moreover, if $k \stackrel{\triangle}{=} \limsup_t(-d(t))$, then we have $|k| \leq |a||k|$ by the same reasoning. This yields $k = 0$. However, $\limsup_t(-d(t)) = -\liminf_t d(t)$ implies $\limsup_t d(t) = \liminf_t d(t) = 0$. Since a sequence is convergent if and only if limit superior and limit inferior of the sequence are equal, $s(t)$ is convergent. Furthermore, we

can write $s(t) = as(t-1) + b(t)$. By the uniqueness of the limit, we have

$$\lim_{t\to\infty} s(t) = \frac{b}{1-a}.$$

If we let $b(t) = J_{\text{ex},2}(t) - J_{\text{ex},1}(t)$ and $s(t) = E[\epsilon(t)]$, then we get $b = J_{\text{ex},2} - J_{\text{ex},1}$. By using the above result, we conclude that

$$\lim_{t\to\infty} E[\epsilon(t)] = \frac{\mu_\epsilon(J_{\text{ex},2} - J_{\text{ex},1})}{1-a}.$$

□

4) For the update (2.10), we have

$$\begin{aligned}
\lambda_\gamma(t) &= \frac{[\sum_{n=0}^{M-1} e_1^2(t-n)]^{-\frac{M}{2}}}{[\sum_{n=0}^{M-1} e_1^2(t-n)]^{-\frac{M}{2}} + [\sum_{n=0}^{M-1} e_2^2(t-n)]^{-\frac{M}{2}}} \\
&= \frac{1}{1 + \left[\frac{\sum_{n=0}^{M-1} e_2^2(t-n)}{\sum_{n=0}^{M-1} e_2^2(t-n)}\right]^{-\frac{M}{2}}} \\
&= \frac{1}{1+\exp[-\gamma(t)]},
\end{aligned}$$

where $\gamma(t) \triangleq \frac{M}{2} \ln\left[\frac{\sum_{n=0}^{M-1} e_2^2(t-n)}{\sum_{n=0}^{M-1} e_1^2(t-n)}\right]$. □

BIBLIOGRAPHY

[1] J. Arenas-Garcia, A. R. Figueiras-Vidal, and A. H. Sayed. Mean-square performance of a convex combination of two adaptive filters. *IEEE Transactions on Signal Processing*, 54:1078–1090, 2006.

[2] J. Arenas-Garcia, V. Gomez-Verdejo, and A. R. Figueiras-Vidal. New algorithms for improved adaptive convex combination of LMS transversal filters. *IEEE Transactions on Instrumentation and Measurement*, 54:2239–2249, December 2005.

[3] J. Arenas-Garcia, V. Gomez-Verdejo, M. Martinez-Ramon, and A. R. Figueiras-Vidal. Separate-variable adaptive combination of LMS adaptive filters for plant identification. In *Proc. of the 13th IEEE Int. Workshop Neural Networks Signal Processing*, pages 239–248, 2003.

[4] J. Arenas-Garcia, M. Martinez-Ramon, V. Gomez-Verdejo, and A. R. Figueiras-Vidal. Multiple plant identifier via adaptive LMS convex combination. In *Proc. of the IEEE Int. Symp. Intel. Signal Processing*, pages 137–142, 2003.

[5] J. Arenas-Garcia, M. Martinez-Ramon, A. Navia-Vazquez, and A. R. Figueiras-Vidal. Plant identification via adaptive combination of transversal filters. *Signal Processing*, 86:2430–2438, 2006.

[6] J. Benesty and Y. Huang. The LMS, PNLMS, and exponentiated gradient algorithms. In *Proc. EUSIPCO*, volume 1, 2004.

[7] Jacob Benesty and Yiteng (Arden) Huang. The LMS, PNLMS, and Exponentiated Gradient algorithms. *Proc. Eur. Signal Process. Conf. (EUSIPCO)*, pages 721–724, 2004.

[8] J. C. M. Bermudez, N. J. Bershad, and J. Y. Tourneret. Stochastic analysis of an error

power ratio scheme applied to the affine combination of two lms adaptive filters. *Signal Processing*, 91:2615–2622, 2011.

[9] N. J. Bershad, J. C. M. Bermudez, and J. Tourneret. An affine combination of two lms adaptive filters: Transient mean-square analysis. *IEEE Trans. on Signal Proc.*, 56(5):1853–1864, 2008.

[10] C. Boukis, D.P. Mandic, and A. G. Constantinides. A class of stochastic gradient algorithms with exponentiated error cost functions. *Digital Signal Processing*, 19:201–212, 2009.

[11] R. Candido, M. T. M. Silva, R. Candido, and V. H. Nascimento. Transient and steady-state analysis of the affine combination of two adaptive filters. *IEEE Transactions on Signal Processing*, 58(8):4064–4078, 2010.

[12] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

[13] N. Cesa-Bianchi and G. Lugosi. On prediction of individual sequences relative to a set of experts. *IEEE International Symposium on Information Theor*, pages 16–21, Massachusetts, 1998.

[14] R. A. Fisher. The moments of the distribution for normal samples of measures of departure from normality. *Proc. R. Soc. A*, 130:16–28, 1930.

[15] David P. Helmbold, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. A comparison of new and old algorithms for a mixture estimation problem. *Machine Learning*, 27:97–119, 1997.

[16] David P. Helmbold, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4):325–347, 1998.

[17] S. I. Hill and R. C. Williamson. Convergence of exponentiated gradient algorithms. *IEEE Transactions on Signal Processing*, 49:1208–1215, 2001.

[18] B. Jelfs, D. P. Mandic, and S. C. Douglas. An adaptive approach for the identification of improper complex signals. *Signal Processing*, 92:335–344, 2012.

[19] D. N. Joanes and C. A. Gill. Comparing measures of sample skewness and kurtosis. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47:183–189, 1998.

[20] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Journal of Information and Computation*, 132:1–64, 1997.

[21] J. Kivinen and M.K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Inform. Comput.*, 132:1–64, Jan. 1997.

[22] S. S. Kozat, A. T. Erdogan, A. C. Singer, and A. H. Sayed. Steady state MSE performance analysis of mixture approaches to adaptive filtering. 58(8):4050–4063, August 2010.

[23] S. S. Kozat and A. C. Singer. Multi-stage adaptive signal processing algorithms. In *Proceedings of SAM Signal Proc. Workshop*, pages 380–384, 2000.

[24] Suleyman S. Kozat, Alper T. Erdogan, Andrew C. Singer, and Ali H. Sayed. Transient analysis of adaptive affine combinations. *IEEE Transactions on Signal Processing*, 59(12):6227 – 6232, 2011.

[25] V. H. Nascimento, M. T. M. Silva, and J. Arenas-Garcia. A transient analysis for the convex combination of adaptive filters. *IEEE Transactions on Signal Processing*, 58(8):4064–4078, 2009.

[26] P. A. Naylor, J. Cui, and M. Brookes. Adaptive algorithms for sparse echo cancellation. *Signal Processing*, 86:1182–1192, 2006.

[27] M. Niedzwiecki. Identification of nonstationary stochastic systems using parallel estimation schemes. *IEEE Trans. Autom. Control*, 35(3):329–334, 1990.

[28] A. H. Sayed. *Fundamentals of Adaptive Filtering*. John Wiley and Sons, 2003.

[29] A. C. Singer and M. Feder. Universal linear prediction by model order weighting. *IEEE Transactions on Signal Processing*, 47(10):2685–2699, 1999.

[30] A. C. Singer and M. Feder. Universal linear prediction by model order weighting. *IEEE Transactions on Signal Processing*, 47(10):2685–2699, 1999.

[31] V. Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56:153–173, 1998.