Variants of non-negative least-mean-square algorithm and convergence analysis

Jie Chen, Cédric Richard, José C. M. Bermudez, Paul Honeine

Variants of non-negative least-mean-square algorithm and convergence analysis

Jie Chen, *Student Member, IEEE*, Cédric Richard, *Senior Member, IEEE*,
Jose Carlos M. Bermudez, *Senior Member, IEEE*, Paul Honeine, *Member, IEEE*

***Abstract*—Due to the inherent physical characteristics of systems under investigation, non-negativity is one of the most interesting constraints that can usually be imposed on the parameters to estimate. The Non-Negative Least-Mean-Square algorithm (NNLMS) was proposed to adaptively find solutions of a typical Wiener filtering problem but with the side constraint that the resulting weights need to be non-negative. It has been shown to have good convergence properties. Nevertheless, certain practical applications may benefit from the use of modified versions of this algorithm. In this paper, we derive three variants of NNLMS. Each variant aims at improving the NNLMS performance regarding one of the following aspects: sensitivity of input power, unbalance of convergence rates for different weights and computational cost. We study the stochastic behavior of the adaptive weights for these three new algorithms for non-stationary environments. This study leads to analytical models to predict the first and second order moment behaviors of the weights for Gaussian inputs. Simulation results are presented to illustrate the performance of the new algorithms and the accuracy of the derived models.**

I. Introduction

Optimization of a cost function given a set of constraints is a common objective in signal processing estimation problems. The constraints are usually imposed by system specifications which provide *a priori* information on the feasible set of solutions. The solution of estimation problems under constraints poses special problems for online applications. Common real-time signal processing restrictions on computational complexity and memory requirements tend to rule out several good solutions to the constrained optimization problem.

Non-negativity is one of the most commonly stated constraints. It is often imposed on the parameters to estimate in order to avoid physically absurd and uninterpretable results. Non-negativity constraints have been used for image deblurring [1], deconvolution of system impulse response estimation [2] and audio processing [3]. Another similar problem is the non-negative matrix factorization (NMF), which is now a popular dimension reduction technique used in many applications [4], [5], [6]. This problem is closely related to blind deconvolution, and has found direct application in neuroscience [7] and in hyperspectral imaging [8]. Separation of non-negative mixture of non-negative sources has also been considered in [9], [10].

Over the last fifteen years, a variety of methods have been developed to tackle non-negative least-square (NNLS) problems. Active set techniques for NNLS use the fact that if the set of variables which activate constraints is known, then the solution of the constrained least-square problem can be obtained by solving an unconstrained one that includes only inactive variables. The active set algorithm of Lawson and Hanson [11] is a batch resolution technique for NNLS problems. It has become a standard among the most frequently used methods. In [12], Bro and De Jong introduced a modification of the latter, called Fast NNLS, which takes advantage of the special characteristics of iterative algorithms involving repeated use of non-negativity constraints. Projected gradient algorithms [13], [14], [15], [16] form another class, which is based on successive projections onto the feasible region. In [17], Lin used this kind of algorithm for NMF problems. Low memory requirements and simplicity make algorithms in this class attractive for large scale problems. Nevertheless, they are characterized by slow convergence rate if not combined with appropriate step size selection. The class of multiplicative algorithms is very popular for dealing with NMF problems [5], [18]. Particularly efficient updates were derived in this way for a large number of problems involving non-negativity constraints [19]. However, these algorithms require batch processing, which is not suitable for online system identification problems. In [20], the problem of online system identification under non-negativity constraints on the parameters to estimate was investigated. An LMS-type adaptive algorithm, called Non-Negative Least-Mean-Square (NNLMS) was proposed to solve the Wiener problem under the constraint that the resulting weights need to be non-negative. It was based on the stochastic gradient descent approach combined with a fixed point iteration which converges to a solution satisfying the Karush-Kuhn-Tucker conditions. The stochastic behavior of this algorithm was also analyzed in [20].

In this paper, we extend the work of [20] and derive useful variants of the NNLMS algorithm. Each of these variants is derived to improve the NNLMS properties in some sense. A normalized algorithm is proposed to reduce the NNLMS performance sensitivity to the input power value. An exponential algorithm is proposed to improve the balance of weight convergence rates. Compared to NNLMS, the new algorithm leads to faster convergence of the weights in the active set (weights for which the inequality constraint is satisfied with

This work has been partly supported by CNPq grant No. 305377/2009-4.

J. Chen and P. Honeine are with the Institute Charles Delaunay, CNRS, University of Technology of Troyes, 10010 Troyes cedex, France (e-mail: chenjieg@sina.com; paul.honeine@utt.fr)

C. Richard is with côte d'Azur Observatory, CNRS, University of Nice Sophia-Antipolis, Parc Valrose, 06108 Nice cedex 2, France (e-mail: cedric.richard@unice.fr)

J.-C. M. Bermudez is with the Department of Electrical Engineering, Federal University of Santa Catarina 88040-900, Florian?polis, SC, Brazil (e-mail: j.bermudez@ieee.org).

the equal sign). Finally, a sign-based algorithm is proposed to reduce implementation cost in critical real-time applications.

The rest of this paper is organized as follows. Section II reviews the system identification problem under non-negative constraints and the NNLMS algorithm. Section III motivates and introduces the NNLMS variants. In Section IV and Section V, the transient behavior of each of these variants is analyzed. Analytical models are derived for the mean weight and for the mean-square error behavior. The accuracy of these models is illustrated through simulations. A final example compares the performance of the proposed algorithms with those of NLMS and Projected NLMS in solving an unconstrained non-negative parameter estimation problem.

II. Review of Non-Negative Least-Mean-Square Algorithm

Consider the estimation problem depicted in Fig. 1. The unknown system is characterized by real-valued observations

$$y(n) = \boldsymbol{\alpha}^{*\top} \boldsymbol{x}(n) + z(n), \tag{1}$$

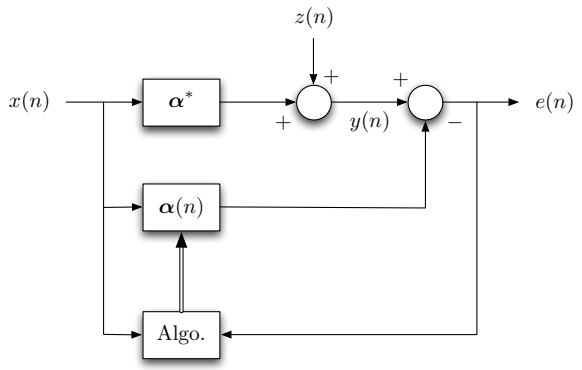


Fig. 1. Adaptive system under study

where $\boldsymbol{\alpha}^* = [\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*]^\top$ is the vector of the model parameters and $\boldsymbol{x}(n) = [x(n), x(n-1), \ldots, x(n-N+1)]^\top$ is the input data vector. The input signal $x(n)$ and the additive noise $z(n)$ are assumed stationary and zero-mean.

In certain applications, inherent physical characteristics of systems under investigation impose a non-negativity constraint on the estimate $\boldsymbol{\alpha}$ of the system parameters. Therefore, the problem of identifying the optimum non-negative model can be formalized as follows

$$\begin{aligned} \boldsymbol{\alpha}^{\mathrm{o}} = \ & \arg\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) \\ & \text{subject to } \alpha_i^{\mathrm{o}} \geq 0, \quad \forall i, \end{aligned} \tag{2}$$

where $J(\boldsymbol{\alpha})$ is a continuously differentiable and strictly convex cost function in $\mathbb{R}^N$, and $\boldsymbol{\alpha}^{\mathrm{o}}$ is the solution to the constrained optimization problem.

A. A fixed-point iteration scheme

To solve the problem (2), let us consider its Lagrangian function $Q(\boldsymbol{\alpha}, \boldsymbol{\lambda})$ given by [21]

$$Q(\boldsymbol{\alpha}, \boldsymbol{\lambda}) = J(\boldsymbol{\alpha}) - \boldsymbol{\lambda}^\top \boldsymbol{\alpha},$$

where $\boldsymbol{\lambda}$ is the vector of non-negative Lagrange multipliers. The Karush-Kuhn-Tucker conditions must necessarily be satisfied at the optimum defined by $\boldsymbol{\alpha}^{\mathrm{o}}$, $\boldsymbol{\lambda}^{\mathrm{o}}$, namely,

$$\nabla_\alpha Q(\boldsymbol{\alpha}^{\mathrm{o}}, \boldsymbol{\lambda}^{\mathrm{o}}) = 0 \tag{3}$$

$$\alpha_i^{\mathrm{o}} \, [\boldsymbol{\lambda}^{\mathrm{o}}]_i = 0, \quad i = 1, \ldots, N \tag{4}$$

where ∇_α stands for the gradient operator with respect to $\boldsymbol{\alpha}$. Using $\nabla_\alpha Q(\boldsymbol{\alpha}, \boldsymbol{\lambda}) = \nabla_\alpha J(\boldsymbol{\alpha}) - \boldsymbol{\lambda}$, these equations can be combined into the following expression

$$\alpha_i^{\mathrm{o}} \, [-\nabla_\alpha J(\boldsymbol{\alpha}^{\mathrm{o}})]_i = 0, \quad i = 1, \ldots, N \tag{5}$$

where the extra minus sign is just used to make a gradient descent of $J(\boldsymbol{\alpha})$ apparent. To solve equation (5) iteratively, two important points have to be noticed. First, $\boldsymbol{D}(-\nabla_\alpha J(\boldsymbol{\alpha}))$ is also a descent direction of $J(\boldsymbol{\alpha})$ if $\boldsymbol{D}$ is a symmetric positive definite matrix. Second, equations of the form $\varphi(u) = 0$ can be solved with a fixed-point iteration algorithm by considering the problem $u = u + \varphi(u)$ under some conditions on function φ. Implementing this strategy with equation (5) leads to the component-wise gradient descent algorithm

$$\alpha_i(n+1) = \alpha_i(n) + \eta_i(n) f_i(\boldsymbol{\alpha}(n)) \, \alpha_i(n) [-\nabla_\alpha J(\boldsymbol{\alpha}(n))]_i \tag{6}$$

where $\eta_i(n)$ a positive step size required to get a contraction scheme and to control the convergence rate, and $f_i(\boldsymbol{\alpha}(n)) > 0$ is the i-th entry of a diagonal matrix $\boldsymbol{D}(n)$. Function $f_i(\boldsymbol{\alpha}(n))$ in (6) is an arbitrary positive function of $\boldsymbol{\alpha}(n)$. Some criteria $J(\boldsymbol{\alpha})$ are defined only for parameter vectors $\boldsymbol{\alpha}$ with positive entries, e.g., the Itakura-Saito distance and the Kullback-Leibler divergence. If necessary, this condition can be managed by an appropriate choice of the step size parameter. Let us assume that $\alpha_i(n) \geq 0$. Non-negativity of $\alpha_i(n+1)$ in (6) is guaranteed if

$$1 + \eta_i(n) f_i(\boldsymbol{\alpha}(n)) [-\nabla_\alpha J(\boldsymbol{\alpha}(n)]_i \geq 0. \tag{7}$$

If $[\nabla_\alpha J(\boldsymbol{\alpha}(n))]_i \leq 0$, condition (7) is clearly satisfied and non-negativity does not impose any restriction on the step size. Conversely, if $[\nabla_\alpha J(\boldsymbol{\alpha}(n))]_i > 0$, non-negativity of $\alpha_i(n+1)$ holds if

$$0 \leq \eta_i(n) \leq \frac{1}{f_i(\boldsymbol{\alpha}(n)) \, [\nabla_\alpha J(\boldsymbol{\alpha}(n))]_i}. \tag{8}$$

Using a single step size $\eta(n)$ in $(0, \eta_{\max}(n)]$ for all entries of $\boldsymbol{\alpha}$ so that

$$\eta_{\max}(n) = \min_i \frac{1}{f_i(\boldsymbol{\alpha}(n)) \, [\nabla_\alpha J(\boldsymbol{\alpha}(n))]_i}, \quad i = 1, \ldots, N \tag{9}$$

the update equation (6) can be written in vector form as

$$\boldsymbol{\alpha}(n+1) = \boldsymbol{\alpha}(n) + \eta(n) \, \boldsymbol{d}(n) \tag{10}$$

where the ith entry of the weight adjustment direction $\boldsymbol{d}(n)$ defined as follows

$$[\boldsymbol{d}(n)]_i = f_i(\boldsymbol{\alpha}(n)) \, \alpha_i(n) [-\nabla_\alpha J(\boldsymbol{\alpha}(n))]_i \tag{11}$$

is a descent direction of $J(\boldsymbol{\alpha})$ because $f_i[\boldsymbol{\alpha}(n)] \, \alpha_i(n) \geq 0$. It should be noted that condition (9) on the step size $\eta(n)$ guarantees the non-negativity of $\boldsymbol{\alpha}(n)$ for all n, but does not ensure algorithm stability.

B. The Non-Negative Least-Mean-Square algorithm

Let us now consider the mean-square error criterion $J_{\rm mse}(\boldsymbol{\alpha}) = E\{[y(n) - \boldsymbol{\alpha}^\top \boldsymbol{x}(n)]^2\}$ to be minimized with respect to $\boldsymbol{\alpha}$ so that

$$\begin{aligned} \boldsymbol{\alpha}^o = \; & \arg\min_{\boldsymbol{\alpha}} E\{[y(n) - \boldsymbol{\alpha}^\top \boldsymbol{x}(n)]^2\} \\ & \text{subject to } \alpha_i^o \geq 0, \quad \forall i \end{aligned} \tag{12}$$

where the non-negativity constraint applies only to the optimum solution because $J_{\rm mse}(\boldsymbol{\alpha})$ is defined for $\boldsymbol{\alpha}$ with entries α_i that can be positive or negative. The gradient of $J_{\rm mse}(\boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$ is $\nabla_\alpha J(\boldsymbol{\alpha}) = 2\,(\boldsymbol{R}_x\boldsymbol{\alpha} - \boldsymbol{r}_{xy})$, where $\boldsymbol{R}_x$ is the autocorrelation matrix of $\boldsymbol{x}(n)$ and $\boldsymbol{r}_{xy}$ is the correlation vector between $\boldsymbol{x}(n)$ and $y(n)$. Following a stochastic gradient approach, the second-order moments $\boldsymbol{R}_x$ and $\boldsymbol{r}_{xy}$ are replaced in (11) by the instantaneous estimates $\boldsymbol{x}(n)\,\boldsymbol{x}^\top(n)$ and $y(n)\,\boldsymbol{x}(n)$, respectively. Then, choosing $f_i(\boldsymbol{\alpha}(n)) = 1/2$ for all i in (6), a fixed positive step-size η, defining $\boldsymbol{D}_x(n) = \mathrm{diag}\{\boldsymbol{x}(n)\}$ and $\boldsymbol{D}_\alpha(n) = \mathrm{diag}\{\boldsymbol{\alpha}(n)\}$, and noting that $\boldsymbol{D}_\alpha(n)\,\boldsymbol{x}(n) = \boldsymbol{D}_x(n)\,\boldsymbol{\alpha}(n)$ leads to the stochastic update given by

$$\boldsymbol{\alpha}(n+1) = \boldsymbol{\alpha}(n) + \eta\, e(n)\, \boldsymbol{D}_x(n)\, \boldsymbol{\alpha}(n) \tag{13}$$

where $e(n) = y(n) - \boldsymbol{\alpha}^\top(n)\,\boldsymbol{x}(n)$. A detailed study of this algorithm, named Non-Negative LMS (NNLMS) can be found in [20].

III. Variants of the Non-Negative Least-Mean-Square Algorithm

A. Normalized NNLMS

A direct extension of the original algorithm is the Normalized NNLMS. Conditioned on $\boldsymbol{\alpha}(n)$, the product $e(n)\boldsymbol{D}_x(n)$ in (13) has dimension of signal power. Thus, η is inversely proportional to signal power. Hence, setting a constant value for η leads to different weight updates for different signal power levels. This is the same sensitivity to signal power verified in the LMS algorithm. A popular way to address this limitation is to normalize the weight update by the input vector squared ℓ_2-norm which yields the Normalized NNLMS update equation

$$\boldsymbol{\alpha}_{\rm N}(n+1) = \boldsymbol{\alpha}_{\rm N}(n) + \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)}\, e(n)\, \boldsymbol{D}_x(n)\, \boldsymbol{\alpha}_{\rm N}(n) \tag{14}$$

Like in Normalized LMS (NLMS) algorithm, adding a small positive regularization parameter ϵ to the denominator $\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)$ may be necessary to avoid numerical difficulties when $\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)$ becomes very small. The resulting ϵ-Normalized NNLMS will then be

$$\boldsymbol{\alpha}_{\rm N}(n+1) = \boldsymbol{\alpha}_{\rm N}(n) + \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n) + \epsilon}\, e(n)\, \boldsymbol{D}_x(n)\, \boldsymbol{\alpha}_{\rm N}(n) \tag{15}$$

where we maintained the notation $\boldsymbol{\alpha}_{\rm N}(n)$ because (14) is a particular case of (15) for $\epsilon = 0$. From now on, we refer to (15) simply as the Normalized NNLMS algorithm.

B. Exponential NNLMS

Each component $\alpha_i(n)$ in the update term of (13) can be viewed as a distinct variable step size adjustment along the ith axis[1]. Hence, each component of $\boldsymbol{\alpha}(n)$ will have a different convergence rate in general. Specifically in the case of weights in the active set (those that tend to zero in steady-state), the convergence rate will progressively reduce in time, becoming very small near steady-state. To alleviate this convergence rate unbalance, we introduce the Exponential NNLMS algorithm.

To achieve a faster convergence for the adaptive coefficients as they get close to zero we propose the use of $f_i(\boldsymbol{\alpha}_E(n)) = \alpha_{{\rm E}_i}^{\gamma-1}(n)$ in (11), with parameter γ chosen in order to attract small values of $\alpha_{{\rm E}_i}(n)$ towards zero. This leads to the ith weight update equation

$$\alpha_{{\rm E}_i}(n+1) = \alpha_{{\rm E}_i}(n) + \eta\, e(n)\, x_i(n)\, \alpha_{{\rm E}_i}^{\gamma}(n). \tag{16}$$

For $0 < \gamma < 1$, the ith weight update in (16) becomes larger than that in (13) when $|\alpha_{{\rm E}_i}(n)| < 1$, thus accelerating convergence towards a null steady-state coefficient value.

The condition for $\alpha_{{\rm E}_i}(n+1) \geq 0$ given $\alpha_{{\rm E}_i}(n) \geq 0$ can be easily determined from (16) as

$$\eta \leq \frac{1}{e(n)\, \alpha_{{\rm E}_i}^{\gamma-1}(n)}, \quad \forall\, i, n. \tag{17}$$

This condition, however, is not useful for design purposes, since it requires *a priori* knowledge of the algorithm behavior. We then propose a modified version of the update equation (16) that allows for instantaneous negative values of $\alpha_{{\rm E}_i}(n)$. The problem with real and negative instantaneous values of $\alpha_{{\rm E}_i}(n)$ is that it may lead to a complex value for $\alpha_{{\rm E}_i}^{\gamma}$ for $0 < \gamma < 1$. To obtain always real values for $\alpha_{{\rm E}_i}^{\gamma}$ we propose to use $\gamma = p/q$ with p and q odd integers and $0 < p < q$. The oddness of p and q guarantees that $\mathrm{sgn}(\alpha_{{\rm E}_i}^{\gamma}(n)) = \mathrm{sgn}(\alpha_{{\rm E}_i}(n))$. Then, the real solution for $\alpha_{{\rm E}_i}^{\gamma}$ can be obtained by calculating $\mathrm{sgn}\{\alpha_{{\rm E}_i}\}\,|\alpha_{{\rm E}_i}|^{\gamma}$. This leads to the following weight update equation for the Exponential NNLMS algorithm in vector form:

$$\boldsymbol{\alpha}_{\rm E}(n+1) = \boldsymbol{\alpha}_{\rm E}(n) + \eta\, e(n)\, \boldsymbol{D}_x(n)\, \boldsymbol{\alpha}_{\rm E}^{(\gamma)}(n) \tag{18a}$$

with the ith component of $\boldsymbol{\alpha}_{\rm E}^{(\gamma)}(n)$ defined as

$$[\boldsymbol{\alpha}_{\rm E}^{(\gamma)}(n)]_i = \mathrm{sign}\{\alpha_{{\rm E}_i}(n)\}\,|\alpha_{{\rm E}_i}(n)|^{\gamma}. \tag{18b}$$

As in the gamma correction used in image processing, an exponent in the range $0 < \gamma < 1$ reduces the dynamic range of each $\alpha_{{\rm E}_i}(n)$. Large values of $\alpha_{{\rm E}_i}(n)$ will be compressed towards 1 and small values of $\alpha_{{\rm E}_i}(n)$ will be increased to prevent from stalling convergence. When $\gamma = 1$, the update equation degenerates into the NNLMS algorithm (13). Using $\gamma > 1$ is generally not recommended, as it tends to spread the vector component values.

[1]Note that Eq. (13) can be written as $\boldsymbol{\alpha}(n+1) = \boldsymbol{\alpha}(n) + \eta\boldsymbol{D}_\alpha(n)\boldsymbol{x}(n)e(n)$. Hence, $\boldsymbol{D}_\alpha(n)$ multiplies the ith gradient component $e(n)x(n-i+1)$ of the classical LMS algorithm by $\alpha_i(n)$.

C. Sign-Sign NNLMS

Like Sign-Sign LMS, which has been included in the CCITT standard for adaptive differential pulse code modulation [22], the motivation for introducing a Sign-Sign NNLMS algorithm is its computational simplicity and its robustness against disturbances [23]. Replacing the input regressor vector and the estimation error in the update term with their signs reduces computation time and dynamic range requirements by replacing multiplications with shifts in real-time implementations. The Sign-Sign NNLMS algorithm is given by

$$\boldsymbol{\alpha}_{\mathrm{S}}(n+1) = \boldsymbol{\alpha}_{\mathrm{S}}(n) + \eta\, \mathrm{sgn}\{e(n)\}\, \mathrm{sgn}\{\boldsymbol{D}_x(n)\}\, \boldsymbol{\alpha}_{\mathrm{S}}(n) \quad (19)$$

After the two signs are evaluated, the ith component update is given by

$$\alpha_{\mathrm{S}_i}(n+1) = \alpha_{\mathrm{S}_i}(n) \pm \eta\, \alpha_{\mathrm{S}_i}(n) \quad (20)$$

where the sign before η is determined by $\mathrm{sgn}\{e(n)x_i(n)\}$. The step-size η is usually selected as a power of 2^{-1}, say $\eta = 2^{-m}$ for some integer $m > 0$. In this case, Equation (20) can be efficiently implemented using shift-add operations. Moreover, the non-negativity constraint will be always satisfied if $\boldsymbol{\alpha}_{\mathrm{S}}$ is initialized with a positive vector and $0 < \eta < 1$. Table I compares the computation complexities of NNLMS and its three variants described above. The rightmost column describes the anticipated property of each algorithm, to be verified in the following.

IV. Mean Weight Behavior

Convergence in the mean sense of the NNLMS algorithm (13) has been studied in [20] for a stationary environment. We now study the stochastic behavior of the NNLMS variants introduced in Section III for fixed step sizes and for a time variant unconstrained solution given by

$$\boldsymbol{\alpha}^*(n) = \boldsymbol{\alpha}^*_{\mathrm{o}}(n) + \boldsymbol{\xi}(n) \quad (21)$$

where $\boldsymbol{\alpha}^*_{\mathrm{o}}(n)$ is a deterministic time-variant mean and $\boldsymbol{\xi}(n)$ is zero-mean with covariance matrix $\boldsymbol{\Xi} = \sigma^2_\xi \boldsymbol{I}$ and independent of any other signal. This simple model provides some information on how the performances of the proposed algorithms are affected by a time variant optimal solution which consists of a deterministic trajectory and a random perturbation. The analysis using more elaborate non-stationarity models such as the random walk model or the autoregressive model [24] leads to mathematically intractable situations. This is due to the extra multiplication of the weight update term by a function of $\boldsymbol{\alpha}(n)$ in (13), (15), (18) and (19), as compared to the LMS algorithm. For the random walk model, the recursive equation for the covariance matrix of the adaptive weight vector becomes a function of the optimal weight covariance matrix, which becomes unbounded as time progresses [24]. For the autoregressive model, a nonlinear term given by the product of the weight error vector and the optimal weight update makes it impossible to determine a recursive adaptive weight vector covariance matrix equation in the state-space form [25]. The model (21) leads to a tractable analysis and still permits inferences about the behavior of the algorithms in randomly time variant environments by varying the power σ^2_ξ of $\boldsymbol{\xi}(n)$. Inferences on the ability of the algorithm to track mean weight variations are also possible but require a different model run for each type of mean time variation of $\boldsymbol{\alpha}^*_{\mathrm{o}}(n)$ to be investigated.

To conserve space and to simplify notation without ambiguity, from now on we use the generic notations $\boldsymbol{\alpha}(n)$ and $\alpha_i(n)$ whenever the given expression is valid for all the algorithms under study. Notations $\boldsymbol{\alpha}_{\mathrm{N}}$, $\boldsymbol{\alpha}_{\mathrm{E}}$ and $\boldsymbol{\alpha}_{\mathrm{S}}$ will be used only for expressions which are specific to the corresponding algorithm. The same notational observation applies to any vector or matrix when referring to a specific algorithm.

For the analyses that follow, we shall define the weight error vector with respect to the unconstrained solution $\boldsymbol{\alpha}^*(n)$ as

$$\tilde{\boldsymbol{v}}(n) = \boldsymbol{\alpha}(n) - \boldsymbol{\alpha}^*(n). \quad (22)$$

and the weight error vector with respect to the mean unconstrained solution $\boldsymbol{\alpha}^*_{\mathrm{o}}(n)$ as

$$\boldsymbol{v}(n) = \boldsymbol{\alpha}(n) - \boldsymbol{\alpha}^*_{\mathrm{o}}(n). \quad (23)$$

The two vectors are related by $\tilde{\boldsymbol{v}}(n) = \boldsymbol{v}(n) - \boldsymbol{\xi}(n)$.

A. Statistical assumptions

The following analysis is performed for $x(n)$ and $z(n)$ zero-mean stationary Gaussian and for $z(n)$ white and statistically independent of any other signal. We assume in the subsequent mean weight behavior analysis that the input and weight vectors are statistically independent, according to the Independence Assumption [24]. This assumption is typical in the study of adaptive algorithms. It is sometimes used for simplification and frequently required for mathematical tractability. The simulation results will show that the resulting analytical models have low sensitivity to this assumption, as they accurately predict the behavior of the three algorithms.

B. Normalized NNLMS algorithm

Using (22) with the appropriate subscript in (15) and $e(n) = y(n) - \boldsymbol{\alpha}^\top_{\mathrm{N}}(n)\,\boldsymbol{x}(n) = z(n) - (\boldsymbol{v}_{\mathrm{N}}(n) - \boldsymbol{\xi}(n))^\top\,\boldsymbol{x}(n)$ yields

$$\begin{aligned}
\boldsymbol{v}_{\mathrm{N}}(n+1) = \boldsymbol{v}_{\mathrm{N}}(n) &+ \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\, z(n)\, \boldsymbol{D}_x(n)\, \boldsymbol{v}_{\mathrm{N}}(n) \\
&+ \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\, z(n)\, \boldsymbol{D}_x(n)\, \boldsymbol{\alpha}^*_{\mathrm{o}}(n) \\
&- \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\, \boldsymbol{D}_x(n)\, \boldsymbol{\alpha}^*_{\mathrm{o}}(n)\, \boldsymbol{x}^\top(n)\, \boldsymbol{v}_{\mathrm{N}}(n) \\
&- \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\, \boldsymbol{D}_x(n)\, \boldsymbol{v}_{\mathrm{N}}(n)\, \boldsymbol{v}^\top_{\mathrm{N}}(n)\, \boldsymbol{x}(n) \\
&+ \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\, \boldsymbol{D}_x(n)\, \boldsymbol{v}_{\mathrm{N}}(n)\, \boldsymbol{\xi}^\top(n)\, \boldsymbol{x}(n) \\
&+ \frac{\eta}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\, \boldsymbol{D}_x(n)\, \boldsymbol{\alpha}^*_{\mathrm{o}}(n)\, \boldsymbol{\xi}^\top(n)\, \boldsymbol{x}(n) - \boldsymbol{\Delta}_{\mathrm{N}}(n).
\end{aligned} \quad (24)$$

where $\boldsymbol{\Delta}_{\mathrm{N}}(n) = \boldsymbol{\alpha}^*_{\mathrm{o}}(n+1) - \boldsymbol{\alpha}^*_{\mathrm{o}}(n)$ is a deterministic vector proportional to the derivative of the mean unconstrained optimal solution.

Taking the expected value of (24) and noting that the expectations of the second, third, sixth and seventh terms on

TABLE I
COMPUTATIONAL COMPLEXITY

Algorithm	Recursion	Computational cost per iteration				Main property
		$+$	$\times$	sgn	$(\cdot)^\gamma$	
NNLMS	Equation (13)	$2N$	$3N+1$			Original one, simplicity
Norm. NNLMS	Equation (15)	$3N$	$4N+1$			Insensitivity to input power
Exp. NNLMS	Equation (18)	$2N$	$3N+1$		N	Balance on weight convergence
S-S NNLMS	Equation (19)	N	$2N$	N		Reduced computational cost

the r.h.s. are equal to zero by virtue of the natures of $z(n)$ and $\boldsymbol{\xi}(n)$ yields

$$\begin{aligned} E\{\boldsymbol{v}_{\mathrm{N}}(n+1)\} &= E\{\boldsymbol{v}_{\mathrm{N}}(n)\} \\ &- \eta E\left\{\frac{1}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\,\boldsymbol{D}_x(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^*(n)\,\boldsymbol{x}^\top(n)\,\boldsymbol{v}_{\mathrm{N}}(n)\right\} \\ &- \eta E\left\{\frac{1}{\boldsymbol{x}^\top(n)\boldsymbol{x}(n)+\epsilon}\boldsymbol{D}_x(n)\boldsymbol{v}_{\mathrm{N}}(n)\boldsymbol{v}_{\mathrm{N}}^\top(n)\boldsymbol{x}(n)\right\} - \boldsymbol{\Delta}_{\mathrm{N}}(n). \end{aligned} \tag{25}$$

Using the independence assumption, the second expectation in the r.h.s. of (25) can be written as

$$\begin{aligned} &E\left\{\frac{1}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\,\boldsymbol{D}_x(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^*(n)\,\boldsymbol{x}^\top(n)\,\boldsymbol{v}_{\mathrm{N}}(n)\right\} \\ =&\boldsymbol{D}_{\alpha_{\mathrm{o}}^*}(n)\,E\left\{\frac{\boldsymbol{x}(n)\,\boldsymbol{x}^\top(n)}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\right\}E\{\boldsymbol{v}_{\mathrm{N}}(n)\} \end{aligned} \tag{26}$$

Evaluation of the first expected value in the r.h.s. of (26) requires approximations. Each numerator element is given by $x(n-i)x(n-j)$. The random part of the denominator is given by $\sum_{k=0}^{N-1} x^2(n-k)$. A common approximation that works well for reasonably large N is to neglect the correlation between these two variates, as the latter tends to vary much slower than the former [26], [27]. Moreover, given its slow variation we approximate $\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)$ by its mean value $N\sigma_x^2$, which is reasonable for large values of N. Using these approximations yields

$$\begin{aligned} &E\left\{\frac{1}{\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n)+\epsilon}\,\boldsymbol{D}_x(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^*(n)\,\boldsymbol{x}^\top(n)\,\boldsymbol{v}_{\mathrm{N}}(n)\right\} \\ \approx&\frac{1}{N\sigma_x^2+\epsilon}\boldsymbol{D}_{\alpha_{\mathrm{o}}^*}(n)\,\boldsymbol{R}_x\,E\{\boldsymbol{v}_{\mathrm{N}}(n)\}. \end{aligned} \tag{27}$$

Using again $\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n) \approx N\sigma_x^2$ and removing it from the expected value, the ith component of the second expectation in the r.h.s. of (25) is

$$\begin{aligned} &\Big[\boldsymbol{D}_x(n)\,\boldsymbol{v}_{\mathrm{N}}(n)\,\boldsymbol{v}_{\mathrm{N}}^\top(n)\,\boldsymbol{x}(n)\Big]_i \\ =&\sum_{j=1}^{N} x(n-i+1)\,v_{\mathrm{N}_i}(n)\,v_{\mathrm{N}_j}(n)\,x(n-j+1). \end{aligned} \tag{28}$$

Taking the expectation, using the independence assumption and defining $\boldsymbol{K}_{\mathrm{N}}(n) = E\{\boldsymbol{v}_{\mathrm{N}}(n)\,\boldsymbol{v}_{\mathrm{N}}^\top(n)\}$ we obtain

$$\begin{aligned} \Big[E\{\boldsymbol{D}_x(n)\,\boldsymbol{v}_{\mathrm{N}}(n)\,\boldsymbol{v}_{\mathrm{N}}^\top(n)\,\boldsymbol{x}(n)\}\Big]_i &= \sum_{j=1}^{N} r_x(j-i)\,[\boldsymbol{K}_{\mathrm{N}}(n)]_{ij} \\ &= [\boldsymbol{R}_x\,\boldsymbol{K}_{\mathrm{N}}(n)]_{ii} \end{aligned} \tag{29}$$

which yields

$$E\{\boldsymbol{D}_x(n)\,\boldsymbol{v}_{\mathrm{N}}(n)\,\boldsymbol{v}_{\mathrm{N}}^\top(n)\,\boldsymbol{x}(n)\} = \mathrm{diag}\{\boldsymbol{R}_x\,\boldsymbol{K}_{\mathrm{N}}(n)\}$$

where $\mathrm{diag}\{\cdot\}$ denotes the vector of diagonal entries in the matrix. Hence, (25) becomes

$$\begin{aligned} E\{\boldsymbol{v}_{\mathrm{N}}(n+1)\} = &\left(\boldsymbol{I} - \frac{\eta}{N\sigma_x^2+\epsilon}\,\boldsymbol{D}_{\alpha_{\mathrm{o}}^*}(n)\,\boldsymbol{R}_x\right)E\{\boldsymbol{v}_{\mathrm{N}}(n)\} \\ &- \frac{\eta}{N\sigma_x^2+\epsilon}\,\mathrm{diag}\{\boldsymbol{R}_x\boldsymbol{K}_{\mathrm{N}}(n)\} - \boldsymbol{\Delta}_{\mathrm{N}}(n). \end{aligned} \tag{30}$$

This recursion for $E\{\boldsymbol{v}_{\mathrm{N}}(n)\}$ requires a model for $\boldsymbol{K}_{\mathrm{N}}(n)$. A recursive model will be derived for $\boldsymbol{K}_{\mathrm{N}}(n)$ in Section V, see (44). That model can be used along with (30) to predict the mean weight behavior of the Normalized NNLMS algorithm. Nevertheless, we have found that a sufficiently accurate and more intuitive mean behavior model can be obtained by neglecting the weight error fluctuations and using the following separation approximation

$$\boldsymbol{K}_{\mathrm{N}}(n) \approx E\{\boldsymbol{v}_{\mathrm{N}}(n)\}\,E\{\boldsymbol{v}_{\mathrm{N}}^\top(n)\}. \tag{31}$$

This approximation has been successfully used in [20] to study the mean behavior of the NNLMS algorithm. A discussion about the validity of the approximation can be found in [20]. Extensive simulation results have shown that this approximation achieves adequate accuracy in modeling the mean behavior of the adaptive weights. We thus obtain the following model

$$\begin{aligned} E\{\boldsymbol{v}_{\mathrm{N}}(n+1)\} = &\left(\boldsymbol{I} - \frac{\eta}{N\sigma_x^2+\epsilon}\,\boldsymbol{D}_{\alpha_{\mathrm{o}}^*}(n)\,\boldsymbol{R}_x\right)E\{\boldsymbol{v}_{\mathrm{N}}(n)\} \\ &- \frac{\eta}{N\sigma_x^2+\epsilon}\,\mathrm{diag}\Big\{\boldsymbol{R}_x\,E\{\boldsymbol{v}_{\mathrm{N}}(n)\}\,E\{\boldsymbol{v}_{\mathrm{N}}^\top(n)\}\Big\} - \boldsymbol{\Delta}_{\mathrm{N}}(n). \end{aligned} \tag{32}$$

C. Exponential NNLMS algorithm

Using (22) with the appropriate subscript in (18), $e(n) = y(n) - \boldsymbol{\alpha}_{\mathrm{E}}^\top(n)\,\boldsymbol{x}(n) = z(n) - (\boldsymbol{v}_{\mathrm{E}}(n) - \boldsymbol{\xi}(n))^\top\,\boldsymbol{x}(n)$, and considering that $\mathrm{sign}\{\alpha_{\mathrm{E}_i}\}\,|\alpha_{\mathrm{E}_i}|^\gamma$ is equal to the real solution of $\alpha_{\mathrm{E}_i}^\gamma$, the Exponential NNLMS weight error update equation can be written as

$$\begin{aligned} &\boldsymbol{v}_{\mathrm{E}}(n+1) \\ =&\boldsymbol{v}_{\mathrm{E}}(n) + \eta\,e(n)\,\boldsymbol{D}_x(n)\left(\boldsymbol{v}_{\mathrm{E}}(n) + \boldsymbol{\alpha}_{\mathrm{o}}^*(n)\right)^{(\gamma)} \\ =&\boldsymbol{v}_{\mathrm{E}}(n) + \eta\,z(n)\,\boldsymbol{D}_x(n)\,\left(\boldsymbol{v}_{\mathrm{E}}(n) + \boldsymbol{\alpha}_{\mathrm{o}}^*(n)\right)^{(\gamma)} \\ &- \eta\,\boldsymbol{D}_x(n)\left(\boldsymbol{v}_{\mathrm{E}}(n) + \boldsymbol{\alpha}_{\mathrm{o}}^*(n)\right)^{(\gamma)}\boldsymbol{v}_{\mathrm{E}}^\top(n)\,\boldsymbol{x}(n) \\ &+\eta\,\boldsymbol{\xi}^\top(n)\boldsymbol{x}(n)\left(\boldsymbol{v}_{\mathrm{E}}(n) + \boldsymbol{\alpha}_{\mathrm{o}}^*(n)\right)^{(\gamma)} - \boldsymbol{\Delta}_{\mathrm{E}}(n). \end{aligned} \tag{33}$$

where $\left(\boldsymbol{v}_{\mathrm{E}}(n) + \boldsymbol{\alpha}_{\mathrm{o}}^*(n)\right)^{(\gamma)}$ is a real vector.

The nonlinear term $(\boldsymbol{v}_{\mathrm{E}}(n) + \boldsymbol{\alpha}_{\mathrm{o}}^*(n)(n))^{(\gamma)}$ on the r.h.s. complicates the evaluation of the expected value of (33) because the statistics of the weight error vector are unknown. We have again found out that using a zero-th order approximation

of $\boldsymbol{v}_{\mathrm{E}}(n)$ is sufficient to provide a reasonably good model for the mean weight error behavior. Thus, we make

$$(v_{\mathrm{E}_i}(n)+\alpha^*_{\mathrm{o}_i}(n))^{\gamma} \approx (E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma}. \tag{34}$$

Using (34) in (33), taking the expected value and considering the statistical properties of $z(n)$ and $\boldsymbol{\xi}(n)$ yields

$$\begin{aligned} &E\{\boldsymbol{v}_{\mathrm{E}}(n+1)\} \\ \approx &E\{\boldsymbol{v}_{\mathrm{E}}(n)\} - \eta\, E\{\boldsymbol{D}_x(n)\big(E\{\boldsymbol{v}_{\mathrm{E}}(n)\}+\boldsymbol{\alpha}^*_{\mathrm{o}}(n)\big)^{(\gamma)} \\ &\cdot \boldsymbol{x}^{\top}(n)\boldsymbol{v}_{\mathrm{E}}(n)\}-\boldsymbol{\Delta}_{\mathrm{E}}(n) \\ = &\Big(\boldsymbol{I}_N - \eta\, \boldsymbol{D}_r(n)\boldsymbol{R}_x\Big)E\{\boldsymbol{v}_{\mathrm{E}}(n)\}-\boldsymbol{\Delta}_{\mathrm{E}}(n) \end{aligned} \tag{35}$$

where $\boldsymbol{I}_N$ is the $N \times N$ identity matrix and $\boldsymbol{D}_r(n)$ is an $N \times N$ diagonal matrix defined as $\boldsymbol{D}_r(n) = \mathrm{diag}\{\boldsymbol{r}(n)\}$ with $\boldsymbol{r}(n)$ being the $N \times 1$ vector whose ith component is $r_i(n) = (E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma}$. It is simple to verify that this model collapses to the NNLMS model derived in [20] for $p = q = 1$.

D. Sign-Sign NNLMS algorithm

The statistical analysis of the Sign-Sign NNLMS algorithm behavior is complicated by the fact that the weight update term is discontinuous in both the input vector $\boldsymbol{x}(n)$ and the error $e(n)$ [28]. To make that analysis tractable, we consider the case of input signal $x(n)$ zero-mean and Gaussian [23], [28].

Using (22) with the appropriate subscript in (19) and $e(n) = y(n)-\boldsymbol{\alpha}^{\top}_{\mathrm{S}}(n)\,\boldsymbol{x}(n) = z(n)-(\boldsymbol{v}_{\mathrm{S}}(n)-\boldsymbol{\xi}(n))^{\top}\,\boldsymbol{x}(n)$, the Sign-Sign NNLMS weight error update equation can be written as

$$\begin{aligned} \boldsymbol{v}_{\mathrm{S}}(n+1) =&\boldsymbol{v}_{\mathrm{S}}(n)+\eta\,\mathrm{sgn}\{z(n)-\boldsymbol{v}^{\top}_{\mathrm{S}}(n)\boldsymbol{x}(n)+\boldsymbol{\xi}^{\top}(n)\boldsymbol{x}(n)\} \\ &\cdot \mathrm{sgn}\{\boldsymbol{D}_x(n)\}\,(\boldsymbol{v}_{\mathrm{S}}(n)+\boldsymbol{\alpha}^*_{\mathrm{o}}(n))-\boldsymbol{\Delta}_{\mathrm{S}}(n). \end{aligned} \tag{36}$$

Note that, unlike the former two variants, the non-stationarity effect appears in the weight error update equation (36) as a nonlinear function of $\boldsymbol{\xi}(n)$. The ith component of (36) is given by

$$\begin{aligned} v_{\mathrm{S}_i}(n+1) = &\, v_{\mathrm{S}_i}(n) + \eta\,\mathrm{sgn}\big\{z(n) - \boldsymbol{v}^{\top}_{\mathrm{S}}(n)\boldsymbol{x}(n) \\ &+ \boldsymbol{\xi}^{\top}(n)\boldsymbol{x}(n)\big\}\mathrm{sgn}\big\{x_i(n)\big\}\,(v_{\mathrm{S}_i}(n)+\alpha^*_{\mathrm{o}_i}(n))-\Delta_{\mathrm{S}_i}(n). \end{aligned} \tag{37}$$

To determine the expected value of (37), we first note that it has been demonstrated in [28], [29] using Price's theorem [30] that

$$E\Big\{\mathrm{sgn}\{\theta_1\}\,\mathrm{sgn}\{\theta_2\}\Big\} = \frac{2}{\pi}\,\sin^{-1}\left(\frac{E\{\theta_1\theta_2\}}{\sigma_{\theta_1}\sigma_{\theta_2}}\right) \tag{38}$$

for θ_1 and θ_2 two zero-mean jointly Gaussian variables with variances $\sigma^2_{\theta_1}$ and $\sigma^2_{\theta_2}$, respectively. Then, noting that $z(n) - \boldsymbol{v}^{\top}_{\mathrm{S}}(n)\boldsymbol{x}(n)+\boldsymbol{\xi}^{\top}(n)\boldsymbol{x}(n)$ and $x_i(n)$ are zero-mean Gaussian when conditioned on $\boldsymbol{v}_{\mathrm{S}}(n)$ and $\boldsymbol{\xi}(n)$, we assume them jointly Gaussian[2] and use the result in (38) to obtain

$$\begin{aligned} &E\Big\{\mathrm{sgn}\{z(n) - \boldsymbol{v}^{\top}_{\mathrm{S}}(n)\,\boldsymbol{x}(n)+\boldsymbol{\xi}^{\top}(n)\boldsymbol{x}(n)\} \\ &\qquad\cdot \mathrm{sgn}\{x_i(n)\}|\boldsymbol{v}_{\mathrm{S}}(n),\boldsymbol{\xi}(n)\Big\} \\ &\approx \frac{2}{\pi}\sin^{-1}\left(-\frac{\boldsymbol{R}_i^{\top}\,\boldsymbol{v}_{\mathrm{S}}(n) - \boldsymbol{R}_i^{\top}\,\boldsymbol{\xi}(n)}{\sigma_x\,\sigma_{e|\boldsymbol{v}_{\mathrm{S}}(n),\boldsymbol{\xi}(n)}}\right) \end{aligned} \tag{39}$$

where $\boldsymbol{R}_i$ the i-th column of $\boldsymbol{R}$ and $\sigma^2_{e|\boldsymbol{v}_{\mathrm{S}}(n),\boldsymbol{\xi}(n)}$ is the variance of $e(n)$ when conditioned on $\boldsymbol{v}_{\mathrm{S}}(n)$ and $\boldsymbol{\xi}(n)$.

Now, since $\sin^{-1}(\cdot)$ is a nonlinear function and the distribution of its argument is unknown, we proceed as we did for the Exponential NNLMS algorithm and replace the nonlinear function by its zero-th order approximation

$$\begin{aligned} &E\Big\{\mathrm{sgn}\{z(n) - \boldsymbol{v}^{\top}_{\mathrm{S}}(n)\,\boldsymbol{x}(n) + \boldsymbol{\xi}^{\top}(n)\boldsymbol{x}(n)\}\,\mathrm{sgn}\{x_i(n)\}\Big\} \\ \approx &\frac{2}{\pi}\sin^{-1}\left(-\frac{\boldsymbol{R}_i^{\top}\,E\{\boldsymbol{v}_{\mathrm{S}}(n)\}}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}}\right) \end{aligned} \tag{40}$$

with

$$\begin{aligned} &\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}} \\ &= \sqrt{\sigma^2_z + \mathrm{tr}\Big\{\boldsymbol{R}_x\,E\{\boldsymbol{v}_{\mathrm{S}}(n)\}E\{\boldsymbol{v}^{\top}_{\mathrm{S}}(n)\}\Big\} + \mathrm{trace}\{\boldsymbol{R}_x\boldsymbol{\Xi}\}} \end{aligned} \tag{41}$$

Taking the expected value of (37), using the results (39) and (40) and expressing the result in vector form yields the mean weight error vector behavior model

$$\begin{aligned} E\{\boldsymbol{v}_{\mathrm{S}}(n+1)\} =&\Big(\boldsymbol{I}_N + \eta\,\boldsymbol{D}_p(n)\Big)E\{\boldsymbol{v}_{\mathrm{S}}(n)\} \\ &+ \eta\,\boldsymbol{D}_p(n)\,\boldsymbol{\alpha}^*_{\mathrm{o}}(n)-\boldsymbol{\Delta}_{\mathrm{S}}(n) \end{aligned} \tag{42}$$

where $\boldsymbol{D}_p(n)$ is the $N \times N$ diagonal matrix $\boldsymbol{D}_p(n) = \mathrm{diag}\{\boldsymbol{p}(n)\}$ with $\boldsymbol{p}(n)$ being the $N \times 1$ vector whose ith entry is given by (40).

V. Second Moment Analysis

We now study the behavior of the second-order moments of the adaptive weights for the three algorithms proposed in Section III. The analysis is performed under the same statistical hypotheses used in the previous section. The following additional assumptions are used in the subsequent analysis:

- A1 : The input vector $\boldsymbol{x}(n)$ is Gaussian.
- A2 : The weight error vector $\boldsymbol{v}(n)$ is statistically independent of $\boldsymbol{x}(n)\,\boldsymbol{x}^{\top}(n)$. The reasoning for this approximation has been discussed in detail in [31].
- A3 : The statistical dependence of $\boldsymbol{v}(n)\,\boldsymbol{v}^{\top}(n)$ and $\boldsymbol{v}(n)$ is neglected, following the same reasoning valid for A2, see [31].
- A4 : $\boldsymbol{v}(n)$ and $(\boldsymbol{x}^{\top}(n)\,\boldsymbol{v}(n))^2$ are statistically independent given A2. This is because $(\boldsymbol{x}^{\top}(n)\,\boldsymbol{v}(n))^2$ is a linear combination of the entries in $\boldsymbol{v}(n)\,\boldsymbol{v}^{\top}(n)$. Hence, this approximation follows the same reasoning discussed in [31] to justify A2.

These assumptions are typical in the study of adaptive algorithms. They are sometimes used for simplification and sometimes required for mathematical feasibility. The simulation results will show that these assumptions lead to analytical

[2] As $\boldsymbol{x}(n)$ and $z(n)$ are independent and both Gaussian, $[\boldsymbol{x}(n), z(n)]$ is jointly Gaussian. When conditioned on $\boldsymbol{v}_{\mathrm{S}}(n)$ and $\boldsymbol{\xi}(n)$, $[\boldsymbol{x}^{\top}(n), z(n) - \boldsymbol{v}^{\top}_{\mathrm{S}}(n)\boldsymbol{x}(n)+\boldsymbol{\xi}^{\top}(n)\boldsymbol{x}(n)]$ is jointly Gaussian as a linear transformation of $[\boldsymbol{x}^{\top}(n), z(n)]$.

models which are accurate enough in predicting the behavior of the algorithms for design purposes.

The excess means square estimation error (EMSE) is given by $\zeta(n) = E\left\{\tilde{\boldsymbol{v}}^\top(n)\boldsymbol{x}(n)\boldsymbol{x}^\top(n)\tilde{\boldsymbol{v}}(n)\right\}$. Using the relation between $\tilde{\boldsymbol{v}}(n)$ and $\boldsymbol{v}(n)$, the properties of $\boldsymbol{\xi}(n)$, and noting from (24), (33) and (36) that $\boldsymbol{v}(n)$ and $\boldsymbol{\xi}(n)$ are independent, we can write $\zeta(n)$ as

$$\begin{aligned}\zeta(n) &= E\left\{(\boldsymbol{v}(n)-\boldsymbol{\xi}(n))^\top\boldsymbol{x}(n)\boldsymbol{x}^\top(n)(\boldsymbol{v}(n)-\boldsymbol{\xi}(n))\right\}\\ &= \text{trace}\{\boldsymbol{R}_x\,\boldsymbol{K}(n)\} + \text{trace}\{\boldsymbol{R}_x\,\boldsymbol{\Xi}\}\end{aligned} \tag{43}$$

with $\boldsymbol{K}(n) = E\left\{\boldsymbol{v}(n)\boldsymbol{v}^\top(n)\right\}$. The term $\text{trace}\{\boldsymbol{R}_x\,\boldsymbol{\Xi}\}$ is the contribution of the random non-stationarity of the system to the EMSE. In the following, we derive recursive models for $\boldsymbol{K}(n)$ for each of the algorithms.

A. Normalized NNLMS algorithm

Post-multiplying (24) by its transpose, taking the expectation, using the approximation $\boldsymbol{x}^\top(n)\,\boldsymbol{x}(n) \approx N\sigma_x^2$ in (24), defining $\tilde{\eta} = \eta/(N\sigma_x^2+\epsilon)$, using assumptions A1–A4 and proceeding as in [20] leads to

$$\begin{aligned}&\boldsymbol{K}_{\text{N}}(n+1) = \boldsymbol{K}_{\text{N}}(n)\\ &-\tilde{\eta}\Big(\boldsymbol{P}_{1_{\text{N}}}(n)\boldsymbol{K}_{\text{N}}(n) + \boldsymbol{K}_{\text{N}}(n)\boldsymbol{P}_{1_{\text{N}}}^\top(n) + \boldsymbol{P}_{5_{\text{N}}}(n) + \boldsymbol{P}_{5_{\text{N}}}^\top(n)\Big)\\ &+\tilde{\eta}^2\left(\boldsymbol{P}_{6_{\text{N}}}(n) + \boldsymbol{P}_{7_{\text{N}}}(n) + \boldsymbol{P}_{7_{\text{N}}}^\top(n) + \boldsymbol{P}_{8_{\text{N}}}(n)\right)\\ &+\tilde{\eta}^2\sigma_z^2\left(\boldsymbol{P}_{2_{\text{N}}}(n) + \boldsymbol{P}_{3_{\text{N}}}(n) + \boldsymbol{P}_{3_{\text{N}}}^\top(n) + \boldsymbol{P}_{4_{\text{N}}}(n)\right)\\ &+\tilde{\eta}^2\left(\boldsymbol{P}_{9_{\text{N}}}(n) + \boldsymbol{P}_{10_{\text{N}}}(n) + \boldsymbol{P}_{11_{\text{N}}}(n) + \boldsymbol{P}_{11_{\text{N}}}^\top(n)\right)\\ &+\boldsymbol{K}_{\boldsymbol{\Delta}_{\text{N}}}(n)\end{aligned} \tag{44}$$

with

$$\boldsymbol{P}_{1_{\text{N}}}(n) = E\{\boldsymbol{D}_x(n)\,\boldsymbol{\alpha}_{\text{o}}^*(n)\,\boldsymbol{x}(n)\} = \boldsymbol{D}_{\alpha_{\text{o}}^*}(n)\,\boldsymbol{R}_x. \tag{45}$$

$$\begin{aligned}\boldsymbol{P}_{2_{\text{N}}}(n) &= E\{\boldsymbol{D}_x(n)\,\boldsymbol{\alpha}_{\text{o}}^*(n)\,\boldsymbol{\alpha}_{\text{o}}^*(n)^\top\boldsymbol{D}_x(n)\}\\ &= \boldsymbol{D}_{\alpha_{\text{o}}^*}(n)\,\boldsymbol{R}_x\,\boldsymbol{D}_{\alpha_{\text{o}}^*}(n).\end{aligned} \tag{46}$$

$$\begin{aligned}\boldsymbol{P}_{3_{\text{N}}}(n) &= E\{\boldsymbol{D}_x(n)\,\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{\alpha}_{\text{o}}^*(n)^\top\boldsymbol{D}_x(n)\}\\ &\approx E\{\boldsymbol{D}_{v_{\text{N}}}(n)\}\,\boldsymbol{R}_x\,\boldsymbol{D}_{\alpha_{\text{o}}^*}(n).\end{aligned} \tag{47}$$

$$\boldsymbol{P}_{4_{\text{N}}}(n) = E\{\boldsymbol{D}_x(n)\,\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{v}_{\text{N}}^\top(n)\,\boldsymbol{D}_x(n)\} \approx \boldsymbol{R}_x \circ \boldsymbol{K}_{\text{N}}(n) \tag{48}$$

where $\circ$ denotes the so-called Hadamard entry-wise product,

$$\begin{aligned}\boldsymbol{P}_{5_{\text{N}}}(n) &= E\{\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{x}^\top(n)\,\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{v}_{\text{N}}^\top(n)\,\boldsymbol{D}_x(n)\}\\ &\approx \boldsymbol{K}_{\text{N}}(n)\,\boldsymbol{R}_x\,E\{\boldsymbol{D}_{v_{\text{N}}}(n)\}.\end{aligned} \tag{49}$$

$$\begin{aligned}&\boldsymbol{P}_{6_{\text{N}}}(n)\\ &= E\{\boldsymbol{D}_x(n)\,\boldsymbol{\alpha}_{\text{o}}^*(n)\,\boldsymbol{x}^\top(n)\,\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{v}_{\text{N}}^\top(n)\,\boldsymbol{x}(n)\,\boldsymbol{\alpha}_{\text{o}}^*(n)^\top\boldsymbol{D}_x(n)\}\\ &= \boldsymbol{D}_{\alpha_{\text{o}}^*}(n)\,\boldsymbol{Q}_{\text{N}}(n)\,\boldsymbol{D}_{\alpha_{\text{o}}^*}(n).\end{aligned} \tag{50}$$

where the matrix $\boldsymbol{Q}_{\text{N}}(n)$ is defined by $\boldsymbol{Q}_{\text{N}}(n) = \boldsymbol{D}_{\alpha_{\text{o}}^*}(n)\,(2\,\boldsymbol{R}_x\,\boldsymbol{K}_{\text{N}}(n)\,\boldsymbol{R}_x + \text{trace}\{\boldsymbol{R}_x\,\boldsymbol{K}_{\text{N}}(n)\}\,\boldsymbol{R}_x)$,

$$\begin{aligned}&\boldsymbol{P}_{7_{\text{N}}}(n)\\ &= E\{\boldsymbol{D}_x(n)\,\boldsymbol{\alpha}_{\text{o}}^*(n)\boldsymbol{x}^\top(n)\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{x}^\top(n)\,\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{v}_{\text{N}}^\top(n)\,\boldsymbol{D}_x(n)\}\\ &\approx \boldsymbol{D}_{\alpha_{\text{o}}^*}(n)\,\boldsymbol{Q}_{\text{N}}(n)\,E\{\boldsymbol{D}_{v_{\text{N}}}(n)\}.\end{aligned} \tag{51}$$

$$\begin{aligned}&\boldsymbol{P}_{8_{\text{N}}}(n)\\ &= E\{\boldsymbol{D}_x(n)\,\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{v}_{\text{N}}^\top(n)\,\boldsymbol{x}(n)\,\boldsymbol{x}^\top(n)\,\boldsymbol{v}_{\text{N}}(n)\,\boldsymbol{v}_{\text{N}}^\top(n)\,\boldsymbol{D}_x(n)\}\\ &= \boldsymbol{Q}_{\text{N}}(n) \circ \boldsymbol{K}_{\text{N}}(n).\end{aligned} \tag{52}$$

$$\begin{aligned}\boldsymbol{P}_{9_{\text{N}}}(n) =& E\{\boldsymbol{\xi}^\top(n)\boldsymbol{x}(n)\boldsymbol{\xi}^\top(n)\boldsymbol{x}(n)\boldsymbol{D}_x(n)\\ &\cdot\boldsymbol{v}_{\text{N}}(n)\boldsymbol{v}_{\text{N}}^\top(n)\boldsymbol{D}_x(n)\}\end{aligned} \tag{53}$$

$$\begin{aligned}\boldsymbol{P}_{10_{\text{N}}}(n) =& E\{\boldsymbol{\xi}^\top(n)\boldsymbol{x}(n)\boldsymbol{\xi}^\top(n)\boldsymbol{x}(n)\boldsymbol{D}_x(n)\\ &\cdot\boldsymbol{\alpha}_{\text{o}}^*(n)\boldsymbol{\alpha}_{\text{o}}^{*\top}(n)\boldsymbol{D}_x(n)\}\end{aligned} \tag{54}$$

and

$$\begin{aligned}\boldsymbol{P}_{11_{\text{N}}}(n) =& E\{\boldsymbol{\xi}^\top(n)\boldsymbol{x}(n)\boldsymbol{\xi}^\top(n)\boldsymbol{x}(n)\boldsymbol{D}_x(n)\\ &\cdot\boldsymbol{v}_{\text{N}}(n)\boldsymbol{\alpha}_{\text{o}}^{*\top}(n)\boldsymbol{D}_x(n)\}\end{aligned} \tag{55}$$

In obtaining (44), it was considered that the products of the last two term of (24) by the other terms lead to zero mean values due to the properties of $\boldsymbol{\xi}(n)$.

Expected values $\boldsymbol{P}_{1_{\text{N}}}(n)$ through $\boldsymbol{P}_{8_{\text{N}}}(n)$ correspond to the terms of the weight error vector recursive equation derived for the NNLMS algorithm in [20] with $\tilde{\eta}$ substituted for η. Thus, we use the results from [20] and indicate their values directly in (45) through (52). We now derive expressions for $\boldsymbol{P}_{9_{\text{N}}}(n)$ through $\boldsymbol{P}_{11_{\text{N}}}(n)$. These terms convey the effect of the random part of the environment non-stationarity.

Computing (i,j)th entry of $\boldsymbol{P}_{9_{\text{N}}}(n)$ yields

$$\begin{aligned}&[\boldsymbol{P}_{9_{\text{N}}}]_{ij}(n)\\ &= E\left\{\sum_k\sum_l \xi_k(n)\xi_l(n)x_k(n)x_l(n)x_i(n)v_{\text{N}_i}(n)v_{\text{N}_j}(n)x_j(n)\right\}\\ &= \sum_k\sum_l E\left\{\xi_k(n)\xi_l(n)\right\}\,E\left\{v_{\text{N}_i}(n)v_{\text{N}_j}(n)\right\}\\ &\quad\cdot E\left\{x_k(n)x_l(n)x_i(n)x_j(n)\right\}\end{aligned} \tag{56}$$

As $E\{\xi_k(n)\xi_l(n)\} \neq 0$ only for $k=l$, $[\boldsymbol{P}_{9_{\text{N}}}]_{ij}(n) = \sigma_\xi^2\,\sum_k[\boldsymbol{K}_{\text{N}}(n)]_{ij}\,E\left\{x_k^2(n)x_i(n)x_j(n)\right\}$. Using the Gaussian moment factorizing theorem yields $\sum_k E\left\{x_k^2(n)x_i(n)x_j(n)\right\}$ = $\left([\boldsymbol{R}_x]_{ij}[\boldsymbol{R}_x]_{kk} + [\boldsymbol{R}_x]_{ik}[\boldsymbol{R}_x]_{jk}\right)$ = $\left[\boldsymbol{R}_x\,\text{trace}\{\boldsymbol{R}_x\} + 2\,\boldsymbol{R}_x\,\boldsymbol{R}_x\right]_{ij}$. This enables us to write the result in matrix form

$$\boldsymbol{P}_{9_{\text{N}}}(n) = \sigma_\xi^2\,\boldsymbol{K}_{\text{N}}(n) \circ (\boldsymbol{R}_x\,\text{trace}\{\boldsymbol{R}_x\} + 2\,\boldsymbol{R}_x\,\boldsymbol{R}_x) \tag{57}$$

Similarly, we have

$$\boldsymbol{P}_{10_{\text{N}}}(n) = \sigma_\xi^2\,\left(\boldsymbol{\alpha}_{\text{o}}^*(n)\,\boldsymbol{\alpha}_{\text{o}}^{*\top}(n)\right) \circ (\boldsymbol{R}_x\,\text{trace}\{\boldsymbol{R}_x\} + 2\,\boldsymbol{R}_x\,\boldsymbol{R}_x) \tag{58}$$

$$\begin{aligned}\boldsymbol{P}_{11_{\text{N}}}(n) =& \sigma_\xi^2\,\left(E\left\{\boldsymbol{v}_{\text{N}}(n)\right\}\boldsymbol{\alpha}_{\text{o}}^{*\top}(n)\right)\\ &\circ (\boldsymbol{R}_x\,\text{trace}\{\boldsymbol{R}_x\} + 2\,\boldsymbol{R}_x\,\boldsymbol{R}_x)\end{aligned} \tag{59}$$

The last term $\boldsymbol{K}_{\boldsymbol{\Delta}_{\text{N}}}(n)$ conveys the effect of deterministic variation of the mean of system weights. Observing the terms multiplied with $\boldsymbol{\Delta}_{\text{N}}(n)$, we have

$$\begin{aligned}&\boldsymbol{K}_{\boldsymbol{\Delta}_{\text{N}}}(n)\\ &= \boldsymbol{\Delta}_{\text{N}}(n)\boldsymbol{\Delta}_{\text{N}}^\top(n) - \boldsymbol{\Delta}_{\text{N}}(n)(E\left\{\boldsymbol{v}_{\text{N}}(n+1)\right\} + \boldsymbol{\Delta}_{\text{N}}(n))^\top\\ &\quad - (E\left\{\boldsymbol{v}_{\text{N}}(n+1)\right\} + \boldsymbol{\Delta}_{\text{N}}(n))\boldsymbol{\Delta}_{\text{N}}^\top(n)\\ &= -\boldsymbol{\Delta}_{\text{N}}(n)\boldsymbol{\Delta}_{\text{N}}^\top(n) - \boldsymbol{\Delta}_{\text{N}}(n)E\left\{\boldsymbol{v}_{\text{N}}(n+1)\right\}^\top\\ &\quad - E\left\{\boldsymbol{v}_{\text{N}}(n+1)\right\}\boldsymbol{\Delta}_{\text{N}}^\top(n)\end{aligned} \tag{60}$$

B. Exponential NNLMS algorithm

The second order moment analysis of the Exponential NNLMS algorithm requires an improvement on approximation (34) for the nonlinearity $(\boldsymbol{v}_{\mathrm{E}}(n)+\boldsymbol{\alpha}^*_{\mathrm{o}}(n))^{(\gamma)}$ in (33). We use instead the following first order approximation for the real-valued solution of $(v_{\mathrm{E}_i}(n)+\alpha^*_{\mathrm{o}_i}(n))^{\gamma}$:

$$\begin{aligned}&v_{\mathrm{E}_i}(n)+\alpha^*_{\mathrm{o}_i}(n))^{\gamma}\\ \approx&(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma}+\gamma\, g(E\{v_{\mathrm{E}_i}(n)\})\\ &\cdot(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma-1}(v_{\mathrm{E}_i}(n)-E\{v_{\mathrm{E}_i}(n)\})\end{aligned} \tag{61}$$

where

$$\begin{aligned}g(E\{v_{\mathrm{E}_i}(n)\})=&1-\Big[u\big(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n)+\varepsilon\big)\\ &-u\big(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n)-\varepsilon\big)\Big]\end{aligned} \tag{62}$$

with $u(\cdot)$ being the unit step function and ε a small constant. The reason to include the gate function g about $E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n)$ in the regular Taylor series is that the derivative of $(v_{\mathrm{E}_i}(n)+\alpha^*_{\mathrm{o}_i}(n))^{\gamma}$ tends to infinity if $v_{\mathrm{E}_i}(n)$ approaches $-\alpha^*_{\mathrm{o}_i}(n)$. It is simple to verify that $\lim_{v_{\mathrm{E}_i}(n)\to-\alpha^*_{\mathrm{o}_i}(n)} g(E\{v_{\mathrm{E}_i}(n)\})\,(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma-1}=0$. The zero-th order approximation is sufficient about the point where the function is equal to zero. With this new approximation, the term on $v_{\mathrm{E}_i}(n)$ in (61) will include moments of the weight error vector which are necessary to proper modeling its fluctuations.

To use vector notation, we define two deterministic vectors $\boldsymbol{r}(n)$ and $\boldsymbol{s}(n)$ whose ith entries are respectively

$$\begin{aligned}r_i=&(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma}\\ &-\gamma g(E\{v_{\mathrm{E}_i}(n)\})(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma-1}E\{v_{\mathrm{E}_i}(n)\}\\ s_i=&\gamma\, g(E\{v_{\mathrm{E}_i}(n)\})\,(E\{v_{\mathrm{E}_i}(n)\}+\alpha^*_{\mathrm{o}_i}(n))^{\gamma-1}\end{aligned}$$

respectively. We define also the corresponding diagonal matrices $\boldsymbol{D}_r(n)=\mathrm{diag}\{\boldsymbol{r}(n)\}$ and $\boldsymbol{D}_s(n)=\mathrm{diag}\{\boldsymbol{s}(n)\}$. With these new definitions, the linear approximation can be written in vector form as

$$\big(\boldsymbol{v}_{\mathrm{E}}(n)+\boldsymbol{\alpha}^*_{\mathrm{o}}(n)\big)^{(\gamma)}\approx\boldsymbol{r}(n)+\boldsymbol{D}_s(n)\,\boldsymbol{v}_{\mathrm{E}}(n) \tag{63}$$

Post-multiplying (33) by its transpose, using (63), taking the expected value, using assumptions A1–A4 and defining matrix $\boldsymbol{D}_{v_{\mathrm{E}}}(n)=\mathrm{diag}\{\boldsymbol{v}_{\mathrm{E}}(n)\}$ yields, after simple algebraic manipulations as done in [20]

$$\begin{aligned}&\boldsymbol{K}_{\mathrm{E}}(n+1)\\ =&\boldsymbol{K}_{\mathrm{E}}(n)-\eta\left(\boldsymbol{P}_{1_{\mathrm{E}}}(n)\,\boldsymbol{K}_{\mathrm{E}}(n)+\boldsymbol{K}_{\mathrm{E}}(n)\,\boldsymbol{P}^{\top}_{1_{\mathrm{E}}}(n)\right)\\ &-\eta\left(\boldsymbol{P}_{5_{\mathrm{E}}}(n)\boldsymbol{K}_{\mathrm{E}}(n)+\boldsymbol{K}_{\mathrm{E}}(n)\boldsymbol{P}^{\top}_{5_{\mathrm{E}}}(n)\right)\\ &+\eta^2\left(\boldsymbol{P}_{6_{\mathrm{E}}}(n)+\boldsymbol{P}_{7_{\mathrm{E}}}(n)+\boldsymbol{P}^{\top}_{7_{\mathrm{E}}}(n)+\boldsymbol{P}_{8_{\mathrm{E}}}(n)\right)\\ &+\eta^2\sigma_z^2\left(\boldsymbol{P}_{2_{\mathrm{E}}}(n)+\boldsymbol{P}_{3_{\mathrm{E}}}(n)+\boldsymbol{P}^{\top}_{3_{\mathrm{E}}}(n)+\boldsymbol{P}_{4_{\mathrm{E}}}(n)\right)\\ &+\eta^2\boldsymbol{P}_{9_{\mathrm{E}}}(n)+\boldsymbol{K}_{\boldsymbol{\Delta}_{\mathrm{E}}}(n)\end{aligned} \tag{64}$$

with the eight moments $\boldsymbol{P}_{1_{\mathrm{E}}}(n)-\boldsymbol{P}_{8_{\mathrm{E}}}(n)$ given by

$$\boldsymbol{P}_{1_{\mathrm{E}}}(n)=E\Big\{\boldsymbol{D}_x(n)\,\boldsymbol{r}(n)\,\boldsymbol{x}^{\top}(n)\Big\}=\boldsymbol{D}_r(n)\,\boldsymbol{R}_x. \tag{65}$$

$$\boldsymbol{P}_{2_{\mathrm{E}}}(n)=E\{\boldsymbol{D}_x(n)\boldsymbol{r}(n)\boldsymbol{r}^{\top}(n)\boldsymbol{D}_x(n)\}\approx\boldsymbol{D}_r(n)\boldsymbol{R}_x\boldsymbol{D}_r(n). \tag{66}$$

$$\begin{aligned}\boldsymbol{P}_{3_{\mathrm{E}}}(n)=&\;E\{\boldsymbol{D}_x(n)\,\boldsymbol{r}(n)\,\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\,\boldsymbol{D}_s(n)\,\boldsymbol{D}_x(n)\}\\ \approx&\;\boldsymbol{D}_r(n)\,\boldsymbol{R}_x\,E\{\boldsymbol{D}_{v_{\mathrm{E}}}(n)\}\,\boldsymbol{D}_s(n)\end{aligned} \tag{67}$$

$$\begin{aligned}\boldsymbol{P}_{4_{\mathrm{E}}}(n)=&E\{\boldsymbol{D}_x(n)\,\boldsymbol{D}_s(n)\,\boldsymbol{v}_{\mathrm{E}}(n)\,\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\,\boldsymbol{D}_s(n)\,\boldsymbol{D}_x(n)\}\\ \approx&\;\boldsymbol{D}_s(n)\big(\boldsymbol{R}_x\circ\boldsymbol{K}_{\mathrm{E}}(n)\big)\,\boldsymbol{D}_s(n).\end{aligned} \tag{68}$$

$$\begin{aligned}\boldsymbol{P}_{5_{\mathrm{E}}}(n)=&\left\{\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\,\boldsymbol{x}(n)\,\boldsymbol{D}_x(n)\,\boldsymbol{D}_s(n)\right\}\\ =&\;\mathrm{diag}\Big\{\boldsymbol{R}_x\,E\{\boldsymbol{v}_{\mathrm{E}}(n)\}\Big\}\boldsymbol{D}_s(n)\end{aligned} \tag{69}$$

$$\begin{aligned}&\boldsymbol{P}_{6_{\mathrm{E}}}(n)\\ =&E\{\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\,\boldsymbol{x}(n)\,\boldsymbol{D}_x(n)\,\boldsymbol{r}(n)\,\boldsymbol{r}^{\top}(n)\,\boldsymbol{D}_x(n)\,\boldsymbol{x}^{\top}(n)\,\boldsymbol{v}_{\mathrm{E}}(n)\}\\ \approx&\boldsymbol{D}_r(n)\,\boldsymbol{Q}_{\mathrm{E}}(n)\,\boldsymbol{D}_r(n).\end{aligned} \tag{70}$$

where the matrix $\boldsymbol{Q}_{\mathrm{E}}(n)$ in above equations is defined by $\boldsymbol{Q}_{\mathrm{E}}(n)=2\,\boldsymbol{R}_x\,\boldsymbol{K}_{\mathrm{E}}(n)\,\boldsymbol{R}_x+\mathrm{trace}\{\boldsymbol{R}_x\,\boldsymbol{K}_{\mathrm{E}}(n)\}\,\boldsymbol{R}_x$

$$\begin{aligned}&\boldsymbol{P}_{7_{\mathrm{E}}}(n)\\ =&\,E\{\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\boldsymbol{x}(n)\boldsymbol{D}_x(n)\boldsymbol{r}(n)\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\boldsymbol{D}_s(n)\boldsymbol{D}_x(n)\boldsymbol{x}^{\top}(n)\boldsymbol{v}_{\mathrm{E}}(n)\}\\ \approx&\,\boldsymbol{D}_r(n)\,\boldsymbol{Q}_{\mathrm{E}}(n)\,E\{\boldsymbol{D}_{v_{\mathrm{E}}}(n)\}\,\boldsymbol{D}_s(n).\end{aligned} \tag{71}$$

and

$$\begin{aligned}\boldsymbol{P}_{8_{\mathrm{E}}}(n)=&E\{\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\,\boldsymbol{x}(n)\,\boldsymbol{D}_x(n)\,\boldsymbol{D}_s(n)\boldsymbol{v}_{\mathrm{E}}(n)\,\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\\ &\cdot\boldsymbol{D}_s(n)\,\boldsymbol{D}_x(n)\boldsymbol{x}^{\top}(n)\,\boldsymbol{v}_{\mathrm{E}}(n)\}\\ \approx&\boldsymbol{D}_s(n)\left(\boldsymbol{Q}_{\mathrm{E}}(n)\circ\boldsymbol{K}_{\mathrm{E}}(n)\right)\boldsymbol{D}_s(n).\end{aligned} \tag{72}$$

The expectation $\boldsymbol{P}_{9_{\mathrm{E}}}(n)$ conveys the non-stationarity effects and is given by

$$\begin{aligned}\boldsymbol{P}_{9_{\mathrm{E}}}(n)=&E\Big\{(\boldsymbol{\xi}^{\top}(n)\boldsymbol{x}(n))^2\left(\boldsymbol{v}_{\mathrm{E}}(n)+\boldsymbol{\alpha}^*_{\mathrm{o}}(n)\right)^{(\gamma)}\\ &\cdot\left(\boldsymbol{v}_{\mathrm{E}}(n)+\boldsymbol{\alpha}^*_{\mathrm{o}}(n)\right)^{(\gamma)\top}\Big\}\end{aligned} \tag{73}$$

Using the first order approximation (63) and simple manipulations yields

$$\begin{aligned}&\boldsymbol{P}_{9_{\mathrm{E}}}(n)\\ &=\mathrm{trace}\{\boldsymbol{\Xi}\boldsymbol{R}_x\}\Big\{\boldsymbol{r}(n)\boldsymbol{r}^{\top}(n)+\boldsymbol{D}_s(n)\boldsymbol{K}_{\mathrm{E}}(n)\boldsymbol{D}_s(n)\\ &+\boldsymbol{r}(n)E\big\{\boldsymbol{v}^{\top}_{\mathrm{E}}(n)\big\}\boldsymbol{D}_s(n)+\boldsymbol{D}_s(n)E\left\{\boldsymbol{v}_{\mathrm{E}}(n)\right\}\boldsymbol{r}^{\top}(n)\Big\}\end{aligned} \tag{74}$$

The last term $\boldsymbol{K}_{\boldsymbol{\Delta}_{\mathrm{E}}}(n)$ is obtained in the same form of (60)

$$\begin{aligned}\boldsymbol{K}_{\boldsymbol{\Delta}_{\mathrm{E}}}(n)=&-\boldsymbol{\Delta}_{\mathrm{E}}(n)\boldsymbol{\Delta}^{\top}_{\mathrm{E}}(n)-\boldsymbol{\Delta}_{\mathrm{E}}(n)E\left\{\boldsymbol{v}_{\mathrm{E}}(n+1)\right\}^{\top}\\ &-E\left\{\boldsymbol{v}_{\mathrm{E}}(n+1)\right\}\boldsymbol{\Delta}^{\top}_{\mathrm{E}}(n)\end{aligned} \tag{75}$$

C. Sign-Sign NNLMS algorithm

Using the weight error vector definition $\boldsymbol{v}_{\mathrm{S}}(n)=\boldsymbol{\alpha}_{\mathrm{S}}(n)-\boldsymbol{\alpha}^*_{\mathrm{o}}(n)$ in (19) and $\mathrm{sgn}\{e(n)\}\mathrm{sgn}\{\boldsymbol{D}_x(n)\}=\mathrm{sgn}\{\boldsymbol{D}_x(n)e(n)\}$ yields

$$\begin{aligned}\boldsymbol{v}_{\mathrm{S}}(n+1)=&\boldsymbol{v}_{\mathrm{S}}(n)+\eta\,\mathrm{sgn}\{\boldsymbol{D}_x(n)\,e(n)\}\,\boldsymbol{v}_{\mathrm{S}}(n)\\ &+\eta\,\mathrm{sgn}\{\boldsymbol{D}_x(n)\,e(n)\}\,\boldsymbol{\alpha}^*_{\mathrm{o}}(n)-\boldsymbol{\Delta}_{\mathrm{S}}(n)\end{aligned} \tag{76}$$

Post-multiplying (76) by its transpose, taking the expected value and rearranging the terms leads to

$$\begin{aligned}&\boldsymbol{K}_{\mathrm{S}}(n+1)\\&=\boldsymbol{K}_{\mathrm{S}}(n)+\eta\left(\boldsymbol{P}_{1_{\mathrm{S}}}(n)+\boldsymbol{P}_{1_{\mathrm{S}}}^{\top}(n)\right)+\eta\left(\boldsymbol{P}_{2_{\mathrm{S}}}(n)+\boldsymbol{P}_{2_{\mathrm{S}}}^{\top}(n)\right)\\&+\eta^2\left(\boldsymbol{P}_{3_{\mathrm{S}}}(n)+\boldsymbol{P}_{4_{\mathrm{S}}}(n)+\boldsymbol{P}_{4_{\mathrm{S}}}^{\top}(n)+\boldsymbol{P}_{5_{\mathrm{S}}}(n)\right)+\boldsymbol{K}_{\boldsymbol{\Delta}_{\mathrm{S}}}(n)\end{aligned} \tag{77}$$

where

$$\boldsymbol{P}_{1_{\mathrm{S}}}(n)=E\{\boldsymbol{v}_{\mathrm{S}}(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}\,\mathrm{sgn}\{\boldsymbol{D}_x(n)\,e(n)\}\} \tag{78}$$

$$\boldsymbol{P}_{2_{\mathrm{S}}}(n)=E\{\boldsymbol{v}_{\mathrm{S}}(n)\,\boldsymbol{v}_{\mathrm{S}}^{\top}(n)\,\mathrm{sgn}\{\boldsymbol{D}_x(n)\,e(n)\}\} \tag{79}$$

$$\begin{aligned}\boldsymbol{P}_{3_{\mathrm{S}}}(n)=&E\{\mathrm{sgn}\{\boldsymbol{D}_x(n)e(n)\}\boldsymbol{\alpha}_{\mathrm{o}}^{*}(n)\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}\\&\cdot\mathrm{sgn}\{\boldsymbol{D}_x(n)e(n)\}\}\end{aligned} \tag{80}$$

$$\begin{aligned}\boldsymbol{P}_{4_{\mathrm{S}}}(n)=&E\{\mathrm{sgn}\{\boldsymbol{D}_x(n)e(n)\}\boldsymbol{\alpha}_{\mathrm{o}}^{*}(n)\boldsymbol{v}_{\mathrm{S}}^{\top}(n)\\&\cdot\mathrm{sgn}\{\boldsymbol{D}_x(n)e(n)\}\}\end{aligned} \tag{81}$$

$$\begin{aligned}\boldsymbol{P}_{5_{\mathrm{S}}}(n)=&E\{\mathrm{sgn}\{\boldsymbol{D}_x(n)e(n)\}\boldsymbol{v}(n)\boldsymbol{v}_{\mathrm{S}}^{\top}(n)\\&\cdot\mathrm{sgn}\{\boldsymbol{D}_x(n)e(n)\}\}\end{aligned} \tag{82}$$

These expected values are calculated in the following for $\boldsymbol{x}(n)$ Gaussian.

Expected value $\boldsymbol{P}_{1_{\mathrm{S}}}(n)$:

Using the properties of statistical expectation $\boldsymbol{P}_{1_{\mathrm{S}}}(n)$ can be written as

$$\begin{aligned}\boldsymbol{P}_{1_{\mathrm{S}}}(n)=&E_v\Big\{\boldsymbol{v}_{\mathrm{S}}(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}(n)\\&E\big\{\mathrm{sgn}\{\boldsymbol{D}_x(n)\,e(n)|\boldsymbol{v}_{\mathrm{S}}(n),\boldsymbol{\xi}(n)\}\big\}\Big\}.\end{aligned} \tag{83}$$

The conditional expectation in (83) is given by (39), which must be approximated. Approximation (40) for the ith element of (39) is too simple to predict the weight error fluctuations. A more suitable approximation is given by a first order Taylor series expansion:

$$\begin{aligned}&\frac{2}{\pi}\sin^{-1}\left(-\frac{\boldsymbol{R}_i^{\top}\,\boldsymbol{v}_{\mathrm{S}}(n)}{\sigma_x\sigma_{e|\boldsymbol{v}_{\mathrm{S}}(n),\boldsymbol{\xi}(n)}}\right)\\\approx&\frac{2}{\pi}\sin^{-1}\left(-\frac{\boldsymbol{R}_i^{\top}\,E\{\boldsymbol{v}_{\mathrm{S}}(n)\}}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}}\right)\\&-\frac{2}{\pi}\frac{\boldsymbol{R}_i^{\top}\,(\boldsymbol{v}_{\mathrm{S}}(n)-E\{\boldsymbol{v}_{\mathrm{S}}(n)\})}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}\sqrt{1-\left(\dfrac{\boldsymbol{R}_i^{\top}\,E\{\boldsymbol{v}_{\mathrm{S}}(n)\})}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}}\right)^2}}\\=&q_i(n)+\boldsymbol{s}_i^{\top}(n)\boldsymbol{v}_{\mathrm{S}}(n).\end{aligned} \tag{84}$$

where the scalar $q_i(n)$ and the vector $\boldsymbol{s}_i(n)$ are deterministic variables defined respectively as

$$\begin{aligned}q_i(n)=&\frac{2}{\pi}\sin^{-1}\left(-\frac{\boldsymbol{R}_i^{\top}\,E\{\boldsymbol{v}_{\mathrm{S}}(n)\}}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}}\right)\\&+\frac{2}{\pi}\frac{\boldsymbol{R}_i^{\top}E\{\boldsymbol{v}_{\mathrm{S}}(n)\}}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}\sqrt{1-\left(\dfrac{\boldsymbol{R}_i^{\top}\,E\{\boldsymbol{v}_{\mathrm{S}}(n)\})}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}}\right)^2}}\end{aligned} \tag{85}$$

$$\boldsymbol{s}_i(n)=-\frac{2}{\pi}\frac{\boldsymbol{R}_i}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}\sqrt{1-\left(\dfrac{\boldsymbol{R}_i^{\top}\,E\{\boldsymbol{v}_{\mathrm{S}}(n)\})}{\sigma_x\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}}\right)^2}} \tag{86}$$

with $\sigma_{e|E\{\boldsymbol{v}_{\mathrm{S}}(n)\},\boldsymbol{\Xi}}$ defined in (41).

Using (84) in (83) and defining the $N\times 1$ vector $\boldsymbol{q}(n)$ with ith element $q_i(n)$ and the $N\times N$ matrix $\boldsymbol{S}(n)$ with ith column $\boldsymbol{s}_i(n)$, (83) becomes, after simple manipulations,

$$\begin{aligned}\boldsymbol{P}_{1_{\mathrm{S}}}(n)\approx&E\{\boldsymbol{v}_{\mathrm{S}}(n)\}\,\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}(n)\;\mathrm{diag}\{\boldsymbol{q}(n)\}\\&+\boldsymbol{K}_{\mathrm{S}}(n)\,\boldsymbol{S}(n)\,\boldsymbol{D}_{\alpha_{\mathrm{o}}^{*}}(n)\end{aligned} \tag{87}$$

Expected value $\boldsymbol{P}_{2_{\mathrm{S}}}(n)$:

Similar to $\boldsymbol{P}_{1_{\mathrm{S}}}(n)$, we first express $\boldsymbol{P}_{2_{\mathrm{S}}}(n)$ in the form

$$\boldsymbol{P}_{2_{\mathrm{S}}}(n)=E_v\{\boldsymbol{v}_{\mathrm{S}}(n)\,\boldsymbol{v}_{\mathrm{S}}^{\top}(n)\,E\{\mathrm{sgn}\{\boldsymbol{D}_x(n)\,e(n)|\boldsymbol{v}_{\mathrm{S}}(n)\}\}\} \tag{88}$$

Then, using (39) and (84) we obtain

$$\begin{aligned}\boldsymbol{P}_{2_{\mathrm{S}}}(n)\approx&\boldsymbol{K}_{\mathrm{S}}(n)\,\mathrm{diag}\{\boldsymbol{q}(n)\}\\&+E\{\boldsymbol{v}_{\mathrm{S}}(n)\,\boldsymbol{v}_{\mathrm{S}}^{\top}(n)\,\mathrm{diag}\{\boldsymbol{S}^{\top}(n)\boldsymbol{v}_{\mathrm{S}}(n)\}\}.\end{aligned} \tag{89}$$

The (i,j)th element of the expectation in (89) is given by

$$\begin{aligned}&E\{[\boldsymbol{v}_{\mathrm{S}}(n)\boldsymbol{v}_{\mathrm{S}}^{\top}(n)\mathrm{diag}\{\boldsymbol{S}^{\top}(n)\boldsymbol{v}(n)\}]_{ij}\}\\&=\sum_{k=1}^{N}S_{kj}\,E\{v_{\mathrm{S}_k}(n)v_{\mathrm{S}_i}(n)v_{\mathrm{S}_j}(n)\}.\end{aligned} \tag{90}$$

Evaluation of the third order moment in (90) requires further approximation, as the distribution of $\boldsymbol{v}_{\mathrm{S}}(n)$ is unknown. We assume that the distribution of $\boldsymbol{v}(n)$ can be approximated by a Gaussian distribution about its mean value. Then, using the properties of Gaussian variables [32] and defining the centered variable $\bar{v}_p(n)=v_p(n)-E\{v_p(n)\}$ for $p=i,j,k$, we have

$$\begin{aligned}&E\{v_i(n)v_j(n)v_k(n)\}\\\approx&E\{\bar{v}_i(n)\bar{v}_j(n)\}\,E\{v_k(n)\}+E\{\bar{v}_i(n)\bar{v}_k(n)\}\,E\{v_j(n)\}\\&+E\{\bar{v}_j(n)\bar{v}_k(n)\}\,E\{v_i(n)\}+E\{v_i(n)\}E\{v_j(n)\}E\{v_k(n)\}\\=&K_{ij}(n)\,E\{v_k(n)\}+K_{ik}(n)\,E\{v_j(n)\}+K_{jk}(n)\,E\{v_i(n)\}\\&-2\,E\{v_i(n)\}E\{v_j(n)\}E\{v_k(n)\}\end{aligned} \tag{91}$$

which completes the derivation of (89).

Expected value $\boldsymbol{P}_{3_{\mathrm{S}}}$:

The (i,j)-th entry of matrix $\boldsymbol{P}_{3_{\mathrm{S}}}(n)$ is given by

$$\begin{aligned}&[\boldsymbol{P}_{3_{\mathrm{S}}}(n)]_{ij}\\=&E\{\mathrm{sgn}\{x_i(n)\,e(n)\}\,[\boldsymbol{\alpha}_{\mathrm{o}}^{*}(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}(n)]_{ij}\,\mathrm{sgn}\{x_j(n)\,e(n)\}^{\top}\}\\=&E\{[\boldsymbol{\alpha}_{\mathrm{o}}^{*}(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}(n)]_{ij}\,\mathrm{sgn}\{x_i(n)\,x_j(n)\}\}\end{aligned} \tag{92}$$

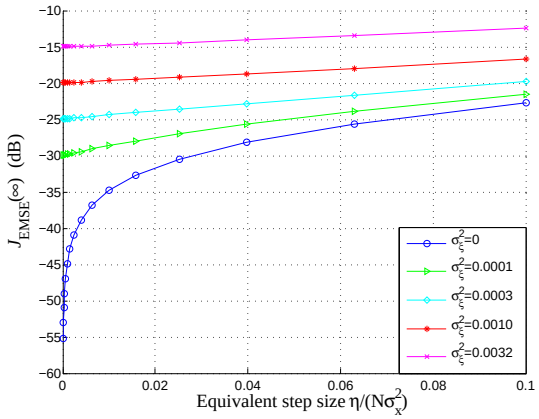
Using (38), $E\{\mathrm{sgn}\{x_i(n)\,x_j(n)\}\}=(2/\pi)\sin^{-1}([\boldsymbol{R}_x]_{ij}/\sigma_x^2)$ and

$$[\boldsymbol{P}_{3_{\mathrm{S}}}(n)]_{ij}=[\boldsymbol{\alpha}_{\mathrm{o}}^{*}(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}(n)]_{ij}\,\frac{2}{\pi}\sin^{-1}\left(\frac{[\boldsymbol{R}_x]_{ij}}{\sigma_x^2}\right) \tag{93}$$

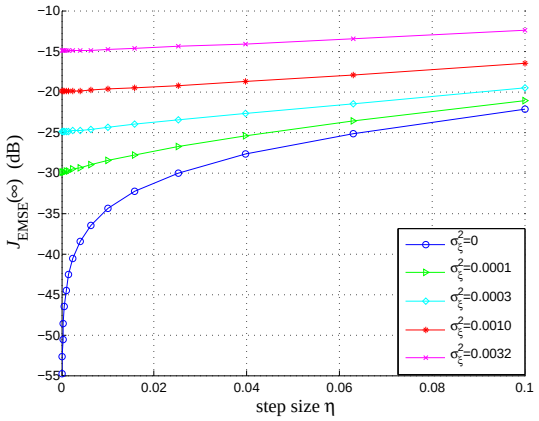
Finally, expressing the result in the matrix form yields

$$\boldsymbol{P}_{3_{\mathrm{S}}}(n)=\frac{2}{\pi}\,(\boldsymbol{\alpha}_{\mathrm{o}}^{*}(n)\,\boldsymbol{\alpha}_{\mathrm{o}}^{*\top}(n))\circ\boldsymbol{T} \tag{94}$$

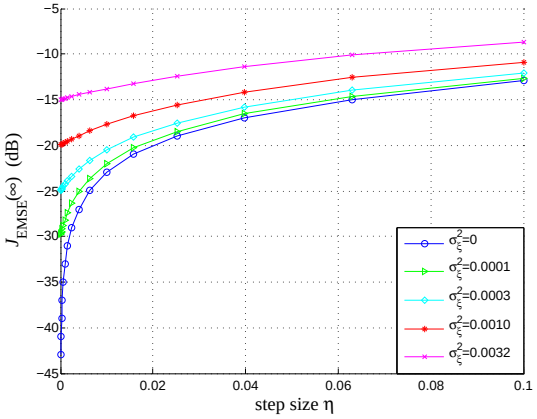
where the (i,j)th elements of the $N\times N$ matrix $\boldsymbol{T}$ is given by $[\boldsymbol{T}]_{ij}=\sin^{-1}([\boldsymbol{R}_x]_{ij}/\sigma_x^2)$.



(a) Normalized NNLMS algorithm



(b) Exponential NNLMS algorithm



(c) Sign-Sign NNLMS algorithm

Fig. 2. Steady-State EMSE behavior for the Normalized NNLMS, Exponential NNLMS and Sign-Sign NNLMS algorithms.

Expected values $\boldsymbol{P}_{4_{\mathrm{S}}}(n)$ and $\boldsymbol{P}_{5_{\mathrm{S}}}(n)$:
Using the same reasoning and approximations used to evaluate $\boldsymbol{P}_{1_{\mathrm{S}}}(n)$ to $\boldsymbol{P}_{3_{\mathrm{S}}}(n)$ yields

$$\boldsymbol{P}_{4_{\mathrm{S}}}(n) = \frac{2}{\pi}\left(\boldsymbol{\alpha}_{\mathrm{o}}^{*}(n)\, E\{\boldsymbol{v}_{\mathrm{S}}^{\top}(n)\}\right) \circ \boldsymbol{T} \tag{95}$$

$$\boldsymbol{P}_{5_{\mathrm{S}}}(n) = \frac{2}{\pi}\left(\boldsymbol{K}_{\mathrm{S}}(n)\right) \circ \boldsymbol{T} \tag{96}$$

The last term $\boldsymbol{K}_{\boldsymbol{\Delta}_{\mathrm{S}}}(n)$ writes

$$\begin{aligned}\boldsymbol{K}_{\boldsymbol{\Delta}_{\mathrm{S}}}(n) = &- \boldsymbol{\Delta}_{\mathrm{S}}(n)\boldsymbol{\Delta}_{\mathrm{S}}^{\top}(n) - \boldsymbol{\Delta}_{\mathrm{S}}(n) E\left\{\boldsymbol{v}_{\mathrm{S}}(n+1)\right\}^{\top} \\ &- E\left\{\boldsymbol{v}_{\mathrm{S}}(n+1)\right\}\boldsymbol{\Delta}_{\mathrm{S}}^{\top}(n)\end{aligned} \tag{97}$$

VI. Steady-State Behavior

The recursive transient models derived in Section IV and in Section V are nonlinear in the weight error correlation matrix. Hence, it is not possible to derive closed form expressions for their steady-state values. Nevertheless, these models can also provide information about the steady-state behavior of the algorithms in the limit as $n \to \infty$. Fig. 2 shows examples of plots of the steady-state EMSE for the three new algorithms as functions of both the step size and the nonstationarity parameter σ_ξ^2. These plots were obtained from the theoretical models for an unknown response of order 10 with samples drawn from a uniform distribution in [0,1].

VII. Simulation Results and Discussion

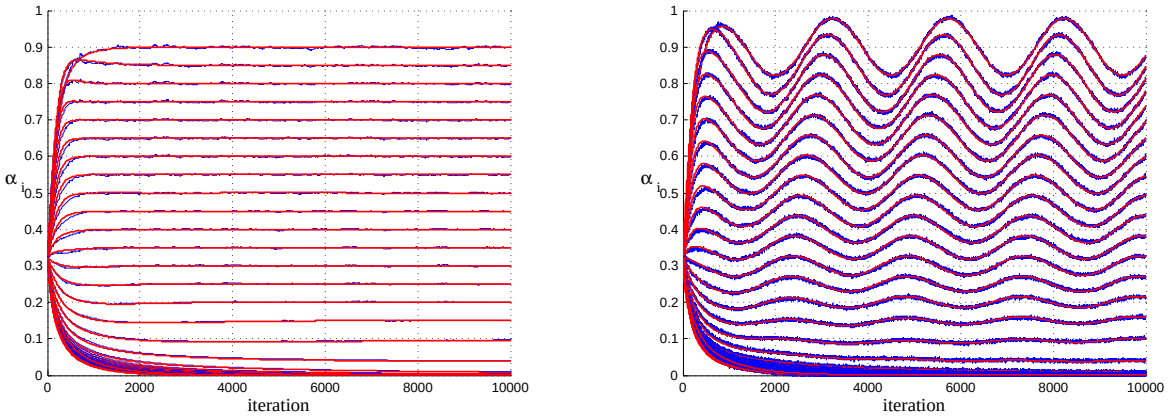
We now present simulation examples to illustrate the properties of the three algorithms and the accuracy of the derived models. The parameters for these examples were chosen to illustrate several properties of the three algorithms while conserving space. Similar results have been obtained using a variety of parameter sets. For all examples, $N = 31$. The unknown stationary system is defined as

$$\alpha_{\mathrm{o}_i}^{*\,(\mathrm{stat.})} = \begin{cases} 0.9 - 0.05\, i, & i = 0, \ldots, 18 \\ -0.01\,(i-18) & i = 19, \ldots, 31 \end{cases} \tag{98}$$

For the non-stationary case, we consider an unknown response defined by

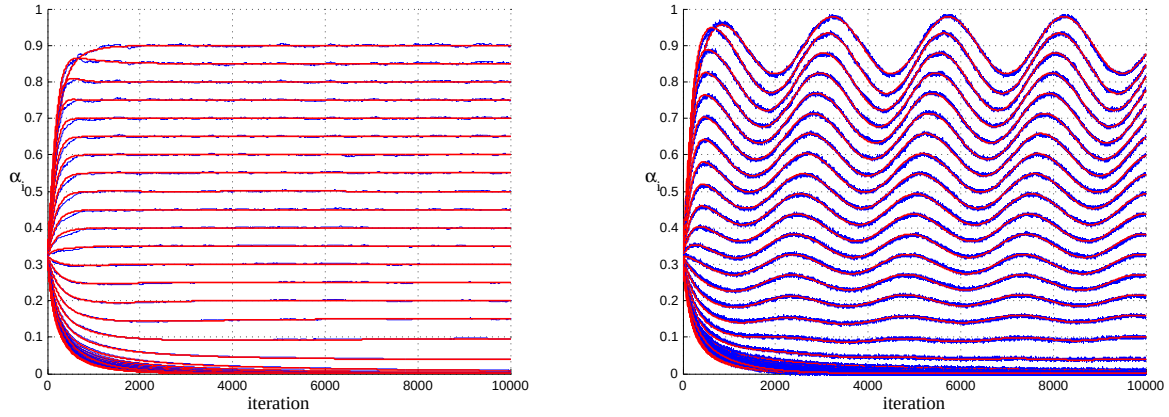
$$\begin{aligned}&\alpha_{\mathrm{o}_i}^{*\,(\mathrm{nonstat.})}(n) \\ &= \alpha_{\mathrm{o}_i}^{*\,(\mathrm{stat.})} + \frac{\left|\alpha_{\mathrm{o}_i}^{*\,(\mathrm{stat.})}\right|}{10} \sin\left(\frac{2\,\pi}{T}\, n + 2\,\pi \frac{i-1}{N}\right) + \xi_i(n)\end{aligned} \tag{99}$$

where the period T of the deterministic sinusoidal component was set to 2500. $\boldsymbol{\xi}(n)$ is a zero-mean Gaussian random vector with correlation matrix $\sigma_\xi^2 \boldsymbol{I}$ with $\sigma_\xi^2 = 5 \times 10^{-4}$. The input



(a) stationary case, $\sigma_x^2 = 1$ (b) nonstationary case, $\sigma_x^2 = 1$

Fig. 3. Evolution of the coefficients $\alpha_i(n)$ for the normalized NNLMS algorithm in stationary and nonstationary environments for $\sigma_x^2 = 1$. Theoretical curves are from (32).



(a) stationary case, $\sigma_x^2 = 0.5$ (b) nonstationary case, $\sigma_x^2 = 0.5$

Fig. 4. Evolution of the coefficients $\alpha_i(n)$ for the normalized NNLMS algorithm in stationary and nonstationary environments for $\sigma_x^2 = 0.5$. Theoretical curves are from (32).

signal is a first-order AR process given by $x(n) = 0.5\,x(n-1) + w(n)$, with $w(n)$ i.i.d. zero-mean Gaussian with variance σ_w^2, adjusted to obtain the desired input power $\sigma_x^2 = 1$. The noise $z(n)$ is zero-mean i.i.d. Gaussian with variance $\sigma_z^2 = 10^{-2}$. The adaptive weights in $\boldsymbol{\alpha}_i(0)$ were all initialized at $10/N$ for all realizations. The step size was always set to $\eta = 0.005$ for all but the normalized variant. For the latter we used $\eta = 0.005\,N\sigma_x^2$, which leads to an equivalent step size $\tilde{\eta} = 0.005$. Monte Carlo simulations were obtained by averaging 100 runs.

1) Example 1: Figs. 3 and 4 show the results for the Normalized NNLMS. The parameter ϵ was set to 0. Blue curves show simulation results and red curves show the theoretical predictions from (32). Fig. 3 is for $\sigma_x^2 = 1$ and Fig. 4 is for $\sigma_x^2 = 0.5$. It can be verified that the model (32) accurately predicts the algorithm behavior, and that normalization has made the algorithm performance basically independent of the input power.

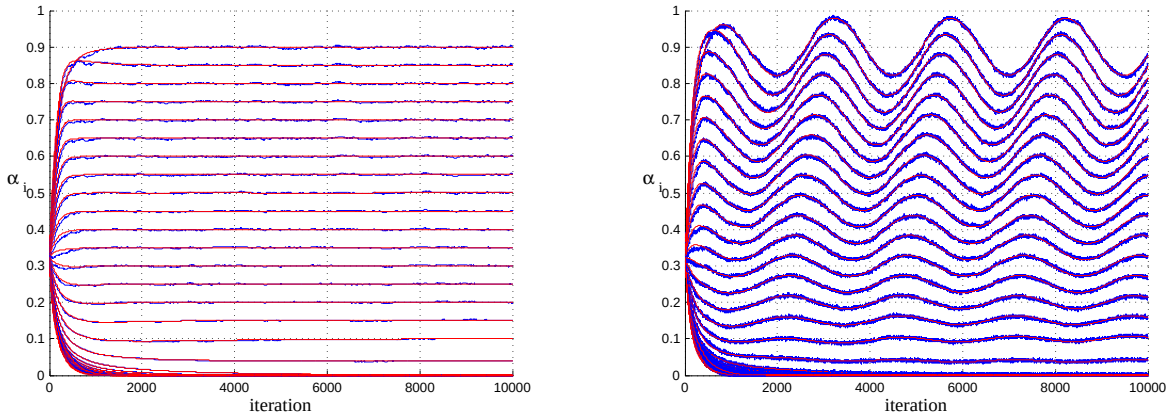
2) Example 2: Fig. 5 illustrates the results for the Exponential NNLMS algorithm. The parameter $(p, q) = (5, 7)$ was used. Compared with Fig. 3, these figures clearly show that the coefficients that tend to zero in steady-state had their convergence rate significantly improved by the Exponential NNLMS algorithm. Also, the accuracy of the theoretical model (35) can be verified.

3) Example 3: Fig. 6 illustrate the result of the Sign-Sign NNLMS under stationary and nonstationary environment. These figures illustrate the accuracy of the model (42). It is also clear that the Sign-Sign NNLMS coefficients converge much slower than those for the NNLMS algorithm as expected.

A. Second moment behavior

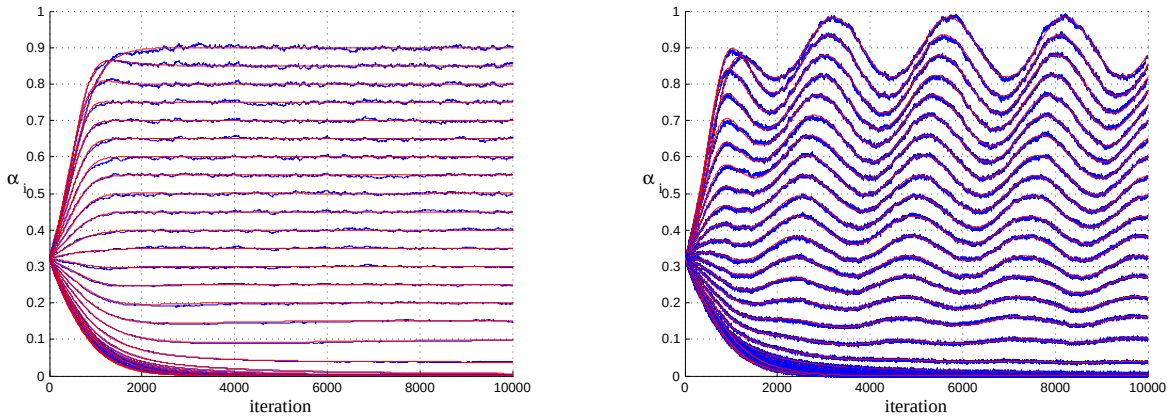
We now illustrate the EMSE behavior of the NNLMS variants. Again, blue curves were obtained from Monte Carlo simulation and red curves show the theoretical predictions.

1) Example 4: Fig. 7 illustrates the EMSE behavior of the Normalized NNLMS algorithm. The accuracy of model (44) can be easily verified. Two more curves are added to each plot to illustrate the effect of the random non-stationarity parameter σ_ξ^2. Random perturbations with different variances were also added to $\alpha_{o_i}^{*\,(\text{stat.})}$ (Fig. 7(a)) and to the nonstationary case (Fig. 7(b)). The light blue (dash-dot) lines show the theoretical



(a) stationary case (b) nonstationary case

Fig. 5. Evolution of the coefficients $\alpha_i(n)$ for the Exponential NNLMS algorithm in stationary and nonstationary environments. Theoretical curves are from (35).



(a) stationary case (b) nonstationary case

Fig. 6. Evolution of the coefficients $\alpha_i(n)$ for the Sign-Sign NNLMS algorithm in stationary and nonstationary environments. Theoretical curves are from (42).

EMSE for $\sigma_\xi^2 = 10^{-3}$, while the green (dash) lines show the theoretical EMSE for $\sigma_\xi^2 = 5 \times 10^{-3}$. These curves illustrate the expected extra EMSE due to tracking of the random optimal solution variations. Simulation curves coincide with the theoretical ones, but are not shown to preserve the visibility of the other curves.

2) Example 5: Fig. 8 illustrates the EMSE behavior of the Exponential NNLMS algorithm. The blue and red curves show again the simulation results and the accurate theoretical predictions using (64) for $(p, q) = (5, 7)$. The light blue (dash-dot) and the green (dash) curves show the theoretical predictions of the EMSE behavior for $(p, q) = (3, 5)$ and $(p, q) = (1, 1)$ (original NNLMS), respectively. The simulation results agree with these curves but are not shown for clarity. These results confirm that the Exponential NNLMS algorithm accelerates the convergence of the adaptive weights when compared to NNLMS.

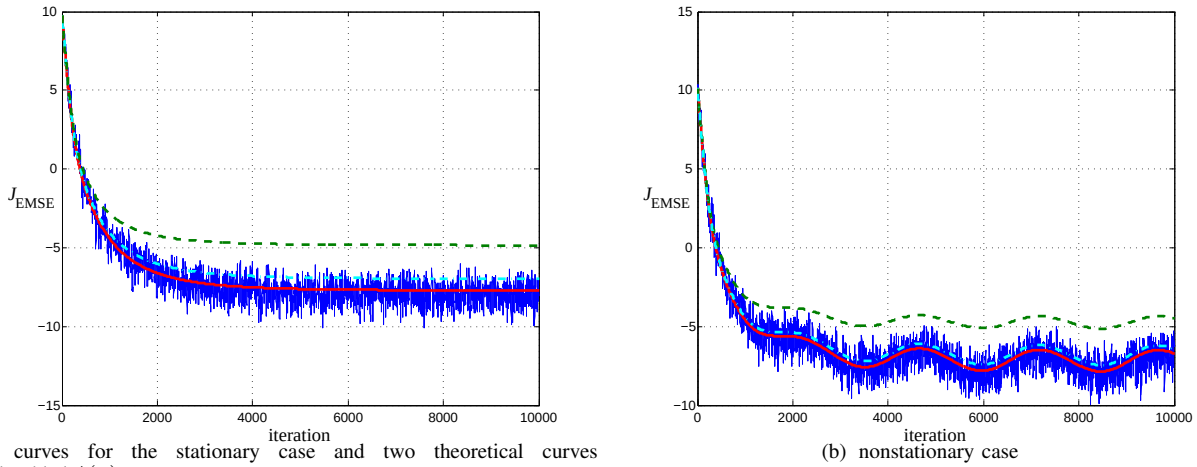
3) Example 6: Fig. 9 illustrates EMSE behavior of the Sign-Sign NNLMS algorithm. Once more, the red curves and blue curves illustrate the accuracy of the model (77). The green (dashed) curves show the performance of the original NNLMS in the same conditions. These curves illustrate the slower convergence rate of Sign-Sign NNLMS when compared to NNLMS, the price paid for a reduced computational complexity.

B. A comparative example

This example compares the performance of the NNLMS algorithm and its variants with that of unconstrained algorithms in solving the unconstrained solution problem of identifying an unknown weight vector $\boldsymbol{\alpha}^*$ with non-negative coefficients. This is an interesting application, as in this case the unconstrained algorithm will converge in the mean to the optimal solution. Though the problem description may guarantee that the optimal weights are positive, often in practice one do not have accurate information about the number of coefficients in the optimal solution. A common approach is to set the adaptive filter with a sufficient number of coefficients, usually larger than the actual unknown number. This examples illustrates the performance of the different algorithms in this case.

Consider a non-negative unknown optimal solution

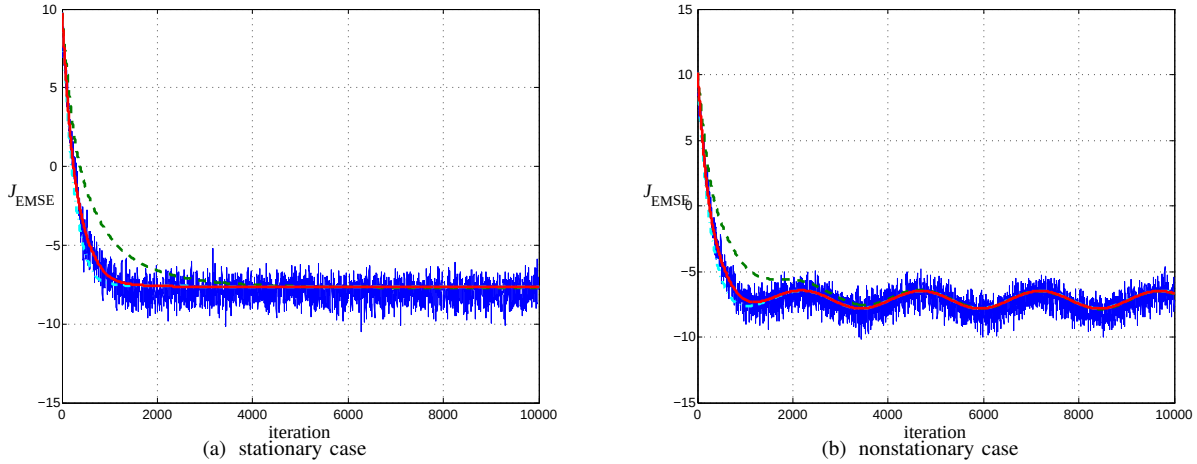
$$\boldsymbol{\alpha}_i^* = \exp(-0.6\, i) \tag{100}$$



(a) curves for the stationary case and two theoretical curves with added $\boldsymbol{\xi}(n)$

(b) nonstationary case

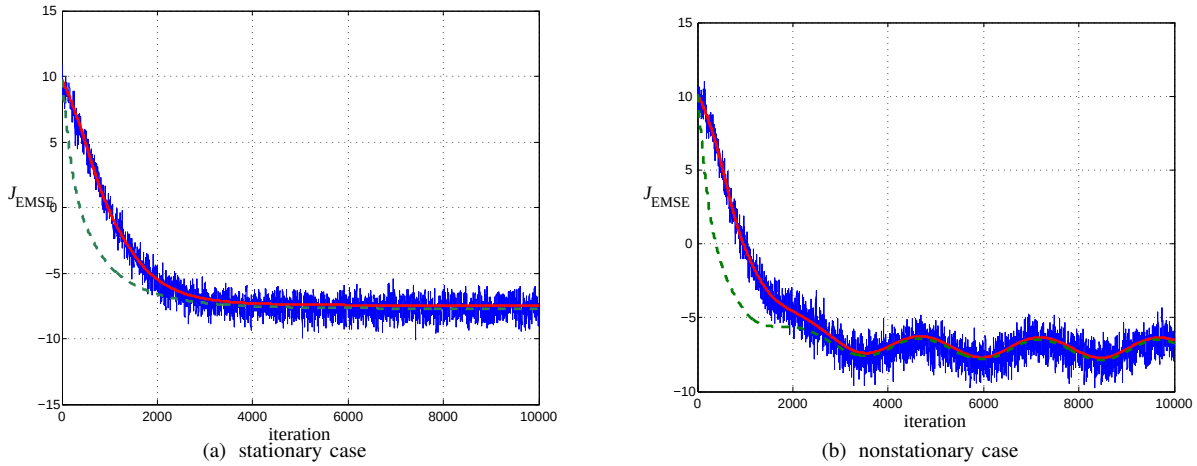
Fig. 7. Evolution of the EMSE for the Normalized NNLMS algorithm in stationary and nonstationary environment. Light blue dash-dot line and green dashed line show the theoretical results for $\sigma_\xi^2 = 10^{-3}$ and 5×10^{-3}, respectively.



(a) stationary case

(b) nonstationary case

Fig. 8. Evolution of the EMSE for the Exponential NNLMS algorithm in stationary and nonstationary environment. Light blue dash-dot line and green dashed line show the theoretical results for $(p, q) = (3, 5)$ and $(p, q) = (1, 1)$ (original NNLMS), respectively.



(a) stationary case

(b) nonstationary case

Fig. 9. Evolution of the EMSE for the Sign-Sign NNLMS algorithm in stationary and nonstationary environment. Theoretical evolution of original NNLMS is represented by the green dashed line.

with $i = 1, \ldots, 10$ and adaptive filters with $N = 30$ coefficients. Five algorithms were tested: NLMS [24], Projected Gradient NLMS [33], Normalized NNLMS, Exponential NNLMS and Sign-Sign NNLMS. In Projected Gradient NLMS, the coefficients which activate the non-negativity constraints are projected into the feasible region, i.e. set to 0, at each iteration. The input signal was given by $x(n) = 0.5\,x(n-1) + w(n)$ with $\sigma_w^2 = 3/4$ so that $\sigma_x^2 = 1$. The

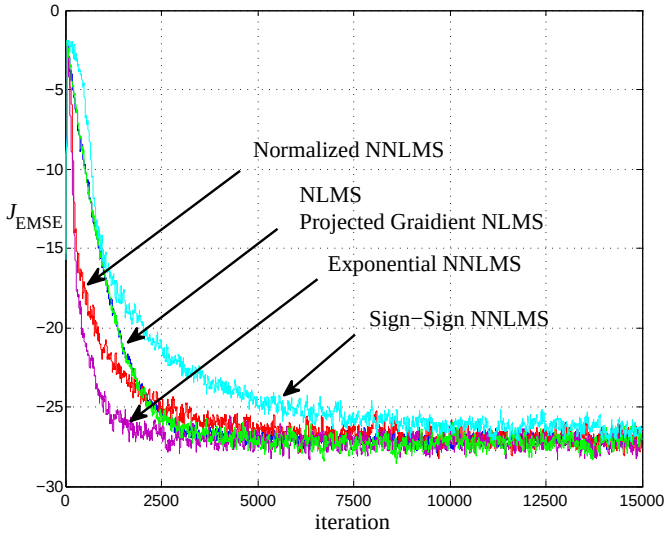


Fig. 10. EMSE (db) for the four algorithms compared.

initial weights $\boldsymbol{\alpha}(0)$ were drawn from the uniform distribution $\mathcal{U}(0,1)$. The additive noise $z(n)$ was i.i.d. Gaussian with $\sigma_z^2 = 0.1$. The step sizes were chosen for each algorithm by experimentation so that all would reach approximately the same steady-state EMSE with the value of 2×10^{-3}. The step sizes were $\eta = 0.035$ for both NLMS and Projected Gradient NLMS, $\eta = 0.8750$ for Normalized NNLMS, $\eta = 0.022$ for Exponential NNLMS and $\eta = 0.007$ for Sign-Sign NNLMS. Figure 10 shows the EMSE evolution for the five algorithms (Monte Carlo simulation averaged over 100 realizations). Figure 11 shows the estimated weights for a single realization of the input signal at $n = 15000$. Although the unconstrained NLMS algorithm is able to converge to the optimal solution in the mean sense, it does not provide a good estimation of the zero-valued coefficients in a single realization. NNLMS-type algorithms, including the Sign-Sign algorithm (which has not even converged to the steady-state at $n = 15000$) do a better job in determining the support of the actual response.

VIII. Conclusion

Many real-life systems require non-negative coefficients when their physical behavior is parameterized. In such cases, a non-negativity constraint should be imposed on the parameters to estimate in order to avoid physically absurd and uninterpretable results. The Non-Negative Least-Mean-Square (NNLMS) algorithm has been recently proposed to solve such a constrained Wiener problem online. In this paper, we proposed three variants of NNLMS, each addressing a different issue that may affect NNLMS under given circumstances. The performances of the Normalized NNLMS, Exponential NNLMS and Sign-Sign NNLMS algorithms were studied for nonstationary environments. The optimal unconstrained solution was modeled by a time-variant mean plus a random fluctuation. The derived analytical models were shown to accurately predict both the mean and the mean-square behavior of the algorithms. Their performances were compared and their advantages in potential applications discussed. Future research efforts will further explore these NNLMS variants properties and apply them in practical situations where efficient adaptive solutions to non-negative filtering problem are required.

References

[1] F. Benvenuto, R. Zanella, L. Zanni, and M. Bertero, "Nonnegative least-squares image deblurring: improved gradient projection approaches," *Inverse Problems*, vol. 26, no. 1, Feb. 2010.

[2] Y Lin and D. D. Lee, "Bayesian regularization and nonnegative deconvolution for room impulse response estimation," *IEEE Transactions on Signal Processing*, vol. 54, no. 3, pp. 839–847, Mar. 2006.

[3] A. Cont and S. Dubinov, "Realtime multiple pitch and multiple-instrument recognition for music signals unsing sparse non-negative constraints," in *Proc. of the 10th Intl. Conference on Digital Audio Effects (DAFx-07)*, Bordeaux, France, Sep. 2007.

[4] D.D. Lee and H.S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999.

[5] D.D. Lee and H.S. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, pp. 556–562, Apr. 2001.

[6] J. Mairal, F. Bach, J. Ponce, and G. Saphiro, "Online learning for matrix factorization and sparse coding," *Journal of Machine Learning Research*, , no. 11, pp. 19–60, Jan. 2010.

[7] A. Cichocki, R. Zdunek, and A.H. Phan, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*, Wiley, 2009.

[8] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Computational Statistics and Data Analysis*, vol. 52, no. 1, pp. 155–173, Sep. 2007.

[9] M. D. Plumbley, "Algorithms for nonnegative independent component analysis," *IEEE Transactions on Neural Networks*, vol. 14, no. 3, pp. 534–543, Mar. 2003.

[10] S. Moussaoui, D. Brie, A. Mohammad-Djafari, and C. Carteret, "Separation of non-negative mixture of non-negative sources using a bayesian approach and MCMC sampling," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4133–4145, Nov. 2006.

[11] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, Society for Industrial and Applied Mathematics, 1995.

[12] R. Bro and S. De Jong, "A fast non-negativity-constrained least squares algorithm," *Journal of Chemometrics*, vol. 11, no. 5, pp. 393–401, Sep./Oct. 1997.

[13] J. B. Rosen, "The gradient projection method for nonlinear programming. part 1: Linear constraints," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 1, pp. 181–217, Mar. 1960.

[14] P. H. Calamai and J. J. Moré, "Projected gradient methods for linearly constrained problems," *Mathematical Programming*, vol. 39, no. 1, pp. 93–116, Oct. 1987.

[15] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, Jan. 1988.

[16] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 97 –123, Jan. 2011.

[17] C. J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, Oct. 2007.

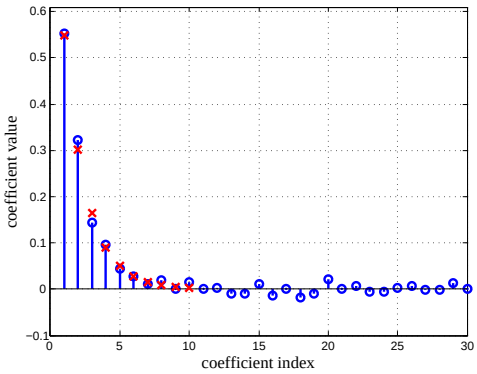
[18] C. J. Lin, "On the convergence of multiplicative update algorithms for nonnegative matrix factorization," *IEEE Transactions on Neural Networks*, vol. 18, no. 6, pp. 1589–1596, Nov. 2007.

[19] H. Lantéri, M. Roche, O. Cuevas, and C. Aime, "A general method to devise maximum-likelihood signal restoration multiplicative algorithms with non-negativity constraints," *Signal Processing*, vol. 81, no. 5, pp. 945–974, May 2001.

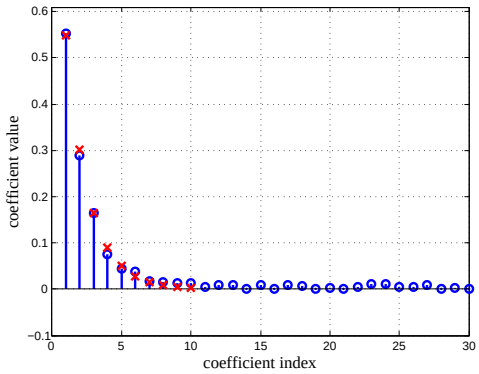
[20] J. Chen, C. Richard, J. C. M. Bermudez, and P. Honeine, "Non-negative least-mean square algorithm," *IEEE Transactions on Signal Processing*, vol. 59, no. 11, pp. 5225–5235, Nov. 2011.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*, University Press, Cambridge, 2004.

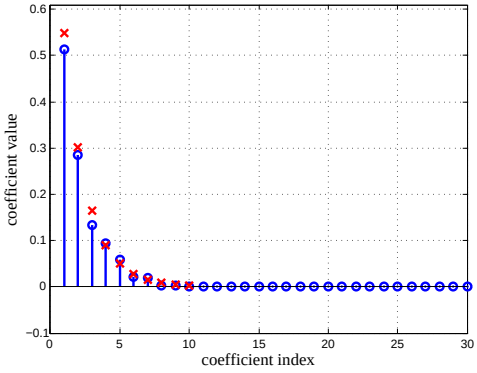
[22] "Itu-t recommendation g.726 (former ccltt recommendation g.721)," 1994.



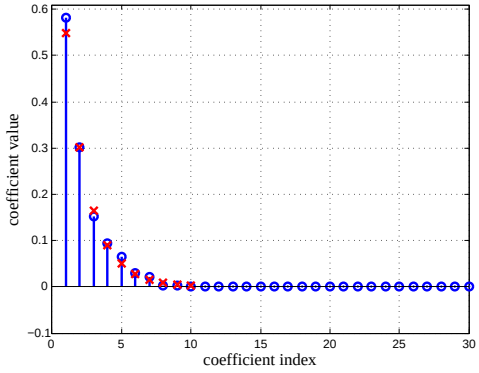
(a) NLMS algorithm.



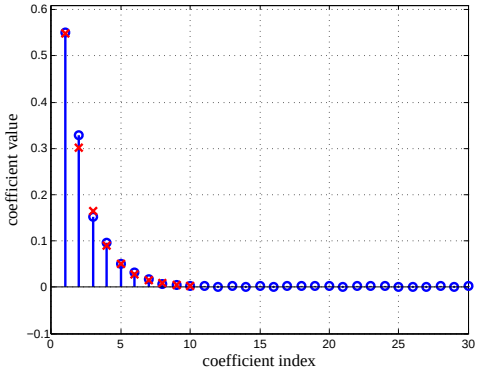
(b) Projected NLMS algorithm.



(c) Normalized NNLMS algorithm.



(d) Exponential NNLMS algorithm.



(e) Sign-Sign NNLMS algorithm.

Fig. 11. Weights estimated by NLMS, Projected NLMS, Normalized NNLMS, and Exponential NNLMS at $n = 15000$ for a single realization. Real weights are marked by $\times$. The NNLMS variants determine clearly the support of the response.

[23] S. Koike, "Analysis of the sign-sign algorithm based on gaussian distributed tap weights," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Seattle, USA, May 1998.

[24] Ali Sayed, *Adaptive Filters*, Wiley-Interscience, New York, 2008.

[25] V. Solo and X. Kong, *Adaptive Signal Processing Algorithms: Stability and Performance*, Prentice Hall, 1994.

[26] C. Samson and V. U. Reddy, "Fixed point error analysis of the normalized ladder algorithms," *IEEE Transactions on Acoustics, Speech, Signal Processing*, vol. 31, no. 10, pp. 1177–1191, Oct. 1983.

[27] S. J. M Almeida, J. C. M. Bermudez, and N. J. Bershad, "A statistical analysis of the affine projection algorithm for unity step size and autoregressive inputs," *IEEE Transactions on Circuits and Systems Part I: Fundamental Theory and Applications*, vol. 52, no. 7, pp. 1394–1405, Jul. 2005.

[28] E. Eweda, "Transient and tracking performance bounds of the sign-sign algorithm," *IEEE Transactions on Signal Processing*, vol. 47, no. 8, pp. 2200–2210, Aug. 1999.

[29] S. Dasgupta, CR Johnson Jr, and A.M. Baksho, "Sign-sign LMS convergence with independent stochastic inputs," *IEEE Transactions on Information Theory*, vol. 36, no. 1, pp. 197–201, Jan. 1990.

[30] R. Price, "A useful theorem for nonlinear devices having gaussian inputs," *IRE Transactions on Information Theory*, vol. 4, no. 2, pp. 69–72, Jun. 1958.

[31] J. Minkoff, "On the unnecessary assumption of statistical independence between reference signal and filter weights in feedforward adaptive systems," *IEEE Transactions on Signal Processing*, vol. 49, no. 5, pp. 1109, May 2001.

[32] K. S. Miller, *Multidimensional Gaussian Distributions*, Wiley, New York, 1964.

[33] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 97–123, Jan. 2011.