Variants of Partial Update Augmented CLMS Algorithm and Their Performance Analysis

Vahid Vahidpour, Amir Rastegarnia, Azam Khalili, Wael M. Bazzi, and Saeid Sanei,

***Abstract*—Naturally complex-valued information or those presented in complex domain are effectively processed by an augmented complex least-mean-square (ACLMS) algorithm. In some applications, the ACLMS algorithm may be too computationally- and memory-intensive to implement. In this paper, a new algorithm, termed partial-update ACLMS (PU-ACLMS) algorithm is proposed, where only a fraction of the coefficient set is selected to update at each iteration. Doing so, two types of partial-update schemes are presented referred to as the sequential and stochastic partial-updates, to reduce computational load and power consumption in the corresponding adaptive filter. The computational cost for full-update PU-ACLMS and its partial-update implementations are discussed. Next, the steady-state mean and mean-square performance of PU-ACLMS for non-circular complex signals are analyzed and closed-form expressions of the steady-state excess mean-square error (EMSE) and mean-square deviation (MSD) are given. Then, employing the weighted energy-conservation relation, the EMSE and MSD learning curves are derived. The simulation results are verified and compared with those of theoretical predictions through numerical examples.**

***Index Terms*—Augmented CLMS, energy-conservation, non-circular, partial-update, sequential algorithm, stochastic-algorithm, widely linear model.**

I. Introduction

COMPLEX-valued adaptive filters are exploited in many practical signal processing applications such as channel estimation [1], [2], frequency estimation [3]–[5], and self-interference cancellation [6], [7]. The standard complex least-mean-square (CLMS), as the generic extension of LMS algorithm in the complex domain $\mathbb{C}$ [8], [9], is one of the widely-used adaptive signal processing algorithms, because of its simplicity and ease of implementation [10], [11]. In some scenarios, often the source signals are non-circular or improper. Recent advances have put this assumptions under scrutiny [12], [13]. Specifically, adaptive filtering techniques developed with non-circularity or impropriety in mind have been manifested to possess superior performance in an expanding number of applications [12]–[15].

The augmented complex statistics have provided the possibility to adequately use the complementary information of signal non-circularity [13], [16]–[19]. This has served as a basis for the evolution of the class of augmented adaptive filtering algorithms. These adaptive algorithms are usually known as widely linear algorithms, e.g., ACLMS algorithms [9], widely linear LMS (Wl-LMS) [20], augmented affine projection algorithm (AAPA) [21], widely linear recursive least-squares (WL-RLS) [22], regularized normalized augmented complex LMS (RN-ACLMS) [23], and augmented extended Kalman filter (AEKF) algorithms [24].

Some adaptive filtering applications, like channel equalization, echo cancellation, and multi-user detection, require an adaptive filter with a very large coefficient vector. In such applications, the ACLMS algorithm may be too computationally- and memory-intensive to implement. In order to overcome the mentioned constraints, one might allow a subset of the adaptive filter coefficients to be updated at each iteration, rather than the entire coefficient vectors. Such a process is called partial coefficient update or briefly partial update [25]. Availability of a finite number of hardware multipliers often driven by cost, space and power consumption considerations, is the main reason for partial updating.

To reduce the computational costs and power consumption, various types of partial update schemes, such as periodic and sequential LMS algorithm [26], and stochastic partial updating [27] have been applied to LMS algorithm. In the periodic LMS algorithm, all the filter coefficients are updated every P-th iteration. The sequential LMS algorithm updates only a portion of coefficients at each iteration. The stochastic partial-update LMS [27] is a randomized version of sequential LMS algorithm in that the coefficient subsets are chosen in a random instead of deterministic fashion. Another approach referred to as max partial-update LMS algorithm has been proposed in [28]–[30]. Diniz and Werner [31] proposed another variant known as set-member-ship partial-update NLMS algorithm based on data-selective updating. The performance analysis of time-domain adaptive filters in the under-modeling situation is established in [8], where a deficient length ACLMS in $\mathbb{C}$ has been considered for second order improper signals. Some distributed versions of partial-update adaptive filters such as [32]–[37] have been developed in the literature.

In this paper, a reduced complexity ACLMS algorithm employing partial updating for improper complex signals is proposed. The algorithm, referred to as partial-update ACLMS (PU-ACLMS) algorithm involves selection of a fraction of the coefficients at every iteration. To this end, we consider two types partial-update schemes namely sequential and stochastic partial-update.

Manuscript received 2018. V. Vahidpour, A. Rastegarnia, and A. Khalili are with the Department of Electrical Engineering, Malayer University, Malayer 65719-95863, Iran (email: v.vahidpour@stu.malayeru.ac.ir; rastegarnia@malayeru.ac.ir; a-khalili@tabrizu.ac.ir).

W. M. Bazzi is with the Electrical and Computer Engineering Department, American University in Dubai Dubai, United Arab Emirates, (email: wbazzi@aud.edu).

S. Sanei is with the School of Science and Technology, Nottingham Trent University, Clifton Lane, Nottingham, U.K. (e-mail: saeid.sanei@ntu.ac.uk)

Digital Object Identifier 2018/XX

TABLE I
SYMBOLS AND THEIR DESCRIPTIONS

Symbol	Description
$(\cdot)^{\mathsf{T}}$	Matrix transposition
$(\cdot)^{*}$	Complex conjugate
$(\cdot)^{\mathsf{H}}$	Hermitian transposition
$\|\cdot\|^{2}$	Squared Euclidean norm
$\|\mathbf{x}\|^{2}_{\mathbf{\Sigma}}$	Weighted norm, $\mathbf{x}^{\mathsf{H}}\mathbf{\Sigma}\mathbf{x}$
$\mathsf{tr}(\mathbf{X})$	Trace of matrix $\mathbf{X}$
$\mathsf{E}[\cdot]$	Statistical expectation
$\otimes$	Kronecker product

The main contributions of this paper are summarized as follows:

- A new algorithm, called PU-ACLMS is proposed to control the computational complexity;
- The computational complexity for full-update ACLMS and its partial-update implementations are examined thoroughly. For large filter lengths the PU-ACLMS algorithms are able to lower the full-complexity by approximately a factor of two;
- In the absence of exact performance analysis the concept of energy-conservation is utilized to derive approximate closed-form expressions for mean-square-error (MSE) and excess mean-square-error (EMSE) of the proposed algorithm;
- The closed-form expressions enable us to find a monotonically increasing relationship between these quantities and the step-size parameter μ;
- The stability conditions for PU-ACLMS algorithm are derived both in mean and mean-square senses for non-circular signals scenarios. Conditions on step-size μ are established to guarantee the mean and mean-square stability of PU-ACLMS algorithms;
- Employing the energy-conservation approach, closed-form expressions to describe the EMSE and MSD learning curves are derived;
- The convergence rates of PU-ACLMS algorithms and the full-update ACLMS are investigated.

Throughout the paper, we adopt normal lowercase letters for scalars, bold lowercase letters for column vectors and bold uppercase letters for matrices, while $\mathbf{I}$ denotes an identity matrix of appropriate size. The real and complex domains are denoted by $\mathbb{R}$ and $\mathbb{C}$. $\mathbb{R}^{\geq 0}$ denotes the set of positive real numbers. For ease of reference, a list of main symbols used throughout the text are collected in Table I.

The rest of the paper is organized as follows: In Section II, the widely linear model and the ACLMS algorithm are briefly introduced. The PU-ACLMS algorithm and its computational complexity is provided in Section III. Section IV investigates the different aspects of PU-ACLMS algorithm including its steady-state performance, stability conditions, transient performance and convergence analysis. Performance evaluations are illustrated in Section V. The paper is finally concluded in Section VI.

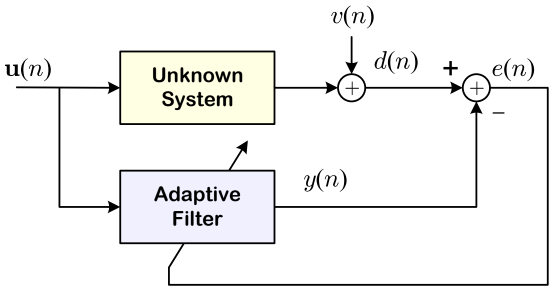

Fig. 1. Schematic diagram for a conventional adaptive filter parameter estimation.

II. AUGMENTED CLMS ALGORITHM

Consider the parameter estimation problem as depicted in Fig. 1. Here, $d(n) \in \mathbb{C}$ denotes a second order non-circular desired signal generated by a widely linear model as [18] ;

$$d(n) = \mathbf{u}^{\mathsf{T}}(n)\mathbf{h}^{o} + \mathbf{u}^{\mathsf{H}}(n)\mathbf{g}^{o} + v(n) \qquad (1)$$

where $\mathbf{h}^{o}$ and $\mathbf{g}^{o}$ denote the optimal standard and conjugate weight vectors respectively. Moreover, $v(n) \in \mathbb{C}$ is the measurement noise and $\mathbf{u}(n) = [u(n), \ldots, u(n-N+1)]^{\mathsf{T}} \in \mathbb{C}^{N\times 1}$ is the input vector. The following assumptions are considered for the data:

Assumption 1.

(i) The input vectors $\{\mathbf{u}(n)\}$ and the additive noise $\{v(n)\}$ are stationary and zero-mean.
(ii) The noise sequence $\{v(n)\}$ is independent and identically distributed (i.i.d.) with variance $\sigma_v^2 = \mathsf{E}\big[|v(n)|^2\big]$.
(iii) The noise sequence $\{v(n)\}$ is statistically independent of $\mathbf{u}(\ell)$ for all $n \neq \ell$.
(iv) The regressor covariance matrix is positive-definite $\mathbf{C}_{\mathbf{u}} = \mathsf{E}[\mathbf{u}(n)\mathbf{u}^{\mathsf{H}}(n)] > 0$.

The problem of estimating the model parameters can be formulated as follows:

$$\underset{\{\mathbf{h},\mathbf{g}\}}{\arg\min}\, J(\mathbf{h},\mathbf{g}) = \mathsf{E}\big[|e(n)|^2\big] = \mathsf{E}\big[|d(n) - y(n)|^2\big] \qquad (2)$$

where $\mathbf{h}(n)$ and $\mathbf{g}(n)$ are adjustable filter weight vectors, called standard and conjugate weight vectors, respectively. In addition, using the augmented statistics, the output $y(n)$ can be written as [9]

$$y(n) = \mathbf{u}^{\mathsf{T}}(n)\mathbf{h}(n) + \mathbf{u}^{\mathsf{H}}(n)\mathbf{g}(n) \qquad (3)$$

In order to solve (2), the ACLMS algorithm updates its weight vectors according to

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu e(n)\mathbf{u}^{*}(n) \qquad (4)$$
$$\mathbf{g}(n+1) = \mathbf{g}(n) + \mu e(n)\mathbf{u}(n) \qquad (5)$$

A detailed study of this algorithm can be found in [9].

III. PARTIAL-UPDATE ACLMS ALGORITHM

A. Algorithm Derivation

In the partial-update schemes, only M out of N weights are allowed to be updated at each iteration. This can be achieved by modifying the adaptation recursions in (4) and (5) as:

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu e(n)\boldsymbol{\mathcal{I}}_{\rm M}(n)\mathbf{u}^{*}(n) \tag{6}$$

$$\mathbf{g}(n+1) = \mathbf{g}(n) + \mu e(n)\boldsymbol{\mathcal{I}}_{\rm M}(n)\mathbf{u}(n) \tag{7}$$

where $\boldsymbol{\mathcal{I}}_{\rm M}(n)$ is an $N \times N$ diagonal coefficient selection matrix defined as:

$$\boldsymbol{\mathcal{I}}_{\rm M}(n) = \mathrm{diag}\left\{i_1(n), i_2(n), \cdots, i_N(n)\right\}$$

The $\boldsymbol{\mathcal{I}}_{\rm M}(n)$ entries satisfy the following constraints:

$$i_k(n) \in \{0,1\}, \quad \sum_{k=1}^{N} i_k(n) = M$$

In the *sequential* partial update $i_k(n)$ is given by

$$i_k(n) = \begin{cases} 1 & \text{if}\, k \in \boldsymbol{\mathcal{S}}_{\mathrm{mod}(n,\beta)+1} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where $\beta = \lceil N/M \rceil$ and the operator, mod(n, β), returns the reminder of the Euclidean division of n by β. Briefly, at a given iteration n, on of the coefficient subsets $\boldsymbol{\mathcal{S}}_t,\ t = \{1, \ldots, \beta\}$ is selected deterministically in a round-robin fashion [25], and the update is performed.

Remark 1. The coefficient subsets $\boldsymbol{\mathcal{S}}_t$ are not uniquely specified if they meet the following conditions [25]:

1) $\bigcup_{t=1}^{\beta} \boldsymbol{\mathcal{S}}_t = \mathbf{S}$, where $\mathbf{S} = \{1, 2, \ldots, N\}$;
2) $\boldsymbol{\mathcal{S}}_t \cap \boldsymbol{\mathcal{S}}_\ell = \phi,\ \forall t, \ell \in \{1, \ldots, \beta\}$ and $t \neq \ell$.

In the stochastic PU-ACLMS, at a given iteration n one of the sets $\boldsymbol{\mathcal{S}}_t, t = \{1, \ldots, \beta\}$, is selected in random form $\{\boldsymbol{\mathcal{S}}_1, \ldots, \boldsymbol{\mathcal{S}}_\beta\}$ with equal probability, whereas for sequential PU-ACLMS one of the sets $\boldsymbol{\mathcal{S}}_t$ is selected in a deterministic fashion.

Remark 2. Let's define the partial-augmented weight vector $\mathbf{w}(n)$, the augmented system input vectors $\mathbf{z}(n)$ and diagonal matrix $\boldsymbol{\mathcal{J}}_{\rm M}(n)$ as follows

$$\mathbf{w}(n) \triangleq [\mathbf{h}^{\mathsf{T}}(n), \mathbf{g}^{\mathsf{T}}(n)]^{\mathsf{T}} \tag{9}$$

$$\mathbf{z}(n) \triangleq [\mathbf{u}^{\mathsf{H}}(n), \mathbf{u}^{\mathsf{T}}(n)]^{\mathsf{T}} \tag{10}$$

$$\boldsymbol{\mathcal{J}}_{\rm M}(n) \triangleq \mathsf{Blkdiag}\left\{\boldsymbol{\mathcal{I}}_{\rm M}(n), \boldsymbol{\mathcal{I}}_{\rm M}(n)\right\} \tag{11}$$

Using the above definitions, the augmented partial-coefficient-update equations (6) and (7) become:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e(n)\boldsymbol{\mathcal{J}}_{\rm M}(n)\mathbf{z}(n) \tag{12}$$

Remark 3. It should be noted that (6) and (7) can be rewritten in an equivalent form as

$$\mathbf{h}_{\rm M}(n+1) = \mathbf{h}_{\rm M}(n) + \mu e(n)\mathbf{u}_{\rm M}^{*}(n) \tag{13}$$

$$\mathbf{g}_{\rm M}(n+1) = \mathbf{g}_{\rm M}(n) + \mu e(n)\mathbf{u}_{\rm M}(n) \tag{14}$$

where $\mathbf{h}_{\rm M}(n)$ denotes $M \times 1$ sub-vector of $\mathbf{h}(n)$ which is formed at time n by stacking the elements of $\mathbf{h}(n)$ with $i_k(n) = 1$. The vectors $\mathbf{g}_{\rm M}(n)$ and $\mathbf{u}_{\rm M}(n)$ are respectively the $M \times 1$ sub-vector of $\mathbf{g}(n)$ and $\mathbf{u}(n)$ defined similarly to $\mathbf{h}_{\rm M}(n)$. It is noteworthy to mention that (13) and (14) represent a reduced-size adaptive filter in accordance with the partial update.

The summary of PU-ACLMS algorithm can be seen in Algorithm 1.

Algorithm 1 The pseudocodes of PU-ACLMS Algorithm.

```
Set a small $\mu$ value and initialize $\mathbf{h}_0$ and $\mathbf{g}_0$ randomly. Then perform the following steps for $n \geq 1$:
1. Evaluate the output $y(n)$ via (3);
2. Compute the error signal $e(n) = d(n) - y(n)$;
3. Select $\boldsymbol{\mathcal{S}}_t$ (based on the sequential or stochastic rules).
4. Update $\mathbf{h}(n)$ via (6);
5. Update $\mathbf{g}(n)$ via (7);
6. If convergence is achieved stop, otherwise go to 1.
```

B. Computational Complexity

The total computational complexity of the ACLMS algorithm is $16N + 2$ real multiplications and $16N$ real additions per iteration. The computational complexity for the *sequential* PU-ACLMS comprises of the following steps at each iteration:

1) $8N$ real multiplications and $8N - 2$ real additions to compute the current output $y(n)$.
2) Two real additions are required to evaluate the error signal, $e(n)$.
3) Two real multiplications are needed to compute $\mu e(n)$.
4) $8M$ real multiplications and $4M$ real additions are needed to compute $\mu e\left(i\right)\mathbf{u}_{\rm M}^{*}\left(i\right)$ and $\mu e(n)\mathbf{u}_{\rm M} e(n)$.
5) Finally, the update of $\mathbf{h}_{\rm M}$ and $\mathbf{g}_{\rm M}$ requires $4M$ real additions in total.

Therefore, each iteration of sequential PU-ACLMS requires $8(N+M)+2$ real multiplications and $8(N+M)$ real additions.

Compared to sequential PU-ACLMS, the computational cost of the stochastic PU-ACLMS algorithm requires evaluation of two additional quantities:

1) One real multiplication and one real addition for the implementation of random coefficient selection, employing a simple random generator, e.g., linear congruential generator defied by [38]:
$$x(n+1) = \mathrm{mod}\{(ax(n) + b), c\}, \ n \geq 0 \tag{15}$$
where a and b are some positive integers, c is the modulus, and $x(0)$ is known as a seed for the random number generator.
2) One real multiplication and one real addition for the random integer
$$\pi(n) = \frac{\beta - 1}{c - 1}x(n) + 1 \tag{16}$$
where $\pi(n)$ is an independent random variable with discrete uniform distribution:
$$\mathrm{Pr}\{\pi(n) = t\} = \frac{1}{\beta} \tag{17}$$

Consequently, the stochastic PU-ACLMS algorithm has the computational complexity of $8(N+M)+4$ real multiplications and $8(N + M) + 2$ real additions per iteration. Table II summarizes the respective computational complexity of the ACLMS algorithm and its partial-update implementations, per iteration for complex-valued data.

Remark 4. For large values of N, the PU-ACLMS algorithm is capable of lowering the full-complexity approximately by a factor of two. It can, however, lead to some performance deterioration.

TABLE II
COMPUTATIONAL COMPLEXITY OF ACLMS AND PU-ACLMS ALGORITHMS AT EACH ITERATION.

Algorithm	$\times$	$+$
ACLMS	$16N+2$	$16N$
Sequential PU-ACLMS	$8(N+M)+2$	$8(N+M)$
Stochastic PU-ACLMS	$8(N+M)+4$	$8(N+M)+2$

Remark 5. Clearly, partial update algorithms are likely to reduce the computational cost. Their implementation require some additional computations for selecting the sub-set of weights to be updated. To address this issue, some available methods such as short-sort approach [39] that use efficient weight selection procedures to minimize the processing overhead can be applied. However, they are effective where the unknown system's impulse response is sparse [40] or the weights are significantly different in value (e.g. amplitude of some weights are much larger than the others).

IV. Performance Analysis

In this section, the performance of PU-ACLMS algorithm is studied in detail. More specifically, we investigate the steady-state performance, stability conditions, transient behavior and the algorithm convergence.

A. Steady-state Analysis

In this sub-section, we use PU-ACLMS given by (13) and (14) to obtain a closed form for EMSE measure defined as

$$\zeta(\infty) \triangleq \lim_{n\to\infty} \mathsf{E}\left[|e_a(n)|^2\right] \tag{18}$$

where $e_a(n)$ is defined as

$$e_a(n) = \mathbf{u}^{\mathsf{T}}(n)\widetilde{\mathbf{h}}(n) + \mathbf{u}^{\mathsf{H}}(n)\widetilde{\mathbf{g}}(n) \tag{19}$$

with

$$\widetilde{\mathbf{h}}(n) = \mathbf{h}^o - \mathbf{h}(n), \quad \widetilde{\mathbf{g}}(n) = \mathbf{g}^o - \mathbf{g}(n) \tag{20}$$

Let's also denote by $\mathbf{h}^o_{\mathrm{M}}(n)$ and $\mathbf{g}^o_{\mathrm{M}}(n)$ $M\times 1$ vectors which are formed at any time instant n by stacking the elements of $\mathbf{h}^o$ and $\mathbf{g}^o$ with $i_k(n)=1$, respectively. Subtracting (13) and (14) from $\mathbf{h}^o_{\mathrm{M}}(n)$ and $\mathbf{g}^o_{\mathrm{M}}(n)$ gives

$$\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1) = \widetilde{\mathbf{h}}_{\mathrm{M}}(n,n) - \mu e(n)\mathbf{u}^*_{\mathrm{M}}(n) \tag{21}$$

$$\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1) = \widetilde{\mathbf{g}}_{\mathrm{M}}(n,n) - \mu e(n)\mathbf{u}_{\mathrm{M}}(n) \tag{22}$$

where $\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1) = \mathbf{h}^o_{\mathrm{M}}(n) - \mathbf{h}_{\mathrm{M}}(n+1)$ and $\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1) = \mathbf{g}^o_{\mathrm{M}}(n) - \mathbf{g}_{\mathrm{M}}(n+1)$ are the partial-update coefficient errors at time instant n. Pre-multiplying both sides of (21) by $\mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)$ and both sides of (22) by $\mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)$ to yield

$$\mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1) = \mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n) - \mu\|\mathbf{u}_{\mathrm{M}}(n)\|^2 e(n) \tag{23}$$

$$\mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1) = \mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n) - \mu\|\mathbf{u}_{\mathrm{M}}(n)\|^2 e(n) \tag{24}$$

The partial-update *a priori* estimation error, $\varepsilon_a(n)$ and the partial-update *a posteriori* estimation error, $\varepsilon_p(n)$ are defined as:

$$\varepsilon_a(n) = \mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n) + \mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n) \tag{25}$$

$$\varepsilon_p(n) = \mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1) + \mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1) \tag{26}$$

It is easy to show that

$$\varepsilon_p(n) = \varepsilon_a(n) - 2\mu e(n)\|\mathbf{u}_{\mathrm{M}}(n)\|^2 \tag{27}$$

Solving for $e(n)$ we obtain

$$e(n) = \frac{1}{2\mu\|\mathbf{u}_{\mathrm{M}}(n)\|^2}\left[\varepsilon_a(n) - \varepsilon_p(n)\right] \tag{28}$$

Substituting (28) into (21) and (22) yields

$$\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1) + \frac{\mathbf{u}^*_{\mathrm{M}}(n)}{2\|\mathbf{u}_{\mathrm{M}}(n)\|^2}\varepsilon_a(n) = \widetilde{\mathbf{h}}_{\mathrm{M}}(n,n) + \frac{\mathbf{u}^*_{\mathrm{M}}(n)}{2\|\mathbf{u}_{\mathrm{M}}(n)\|^2}\varepsilon_p(n) \tag{29}$$

$$\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1) + \frac{\mathbf{u}_{\mathrm{M}}(n)}{2\|\mathbf{u}_{\mathrm{M}}(n)\|^2}\varepsilon_a(n) = \widetilde{\mathbf{g}}_{\mathrm{M}}(n,n) + \frac{\mathbf{u}_{\mathrm{M}}(n)}{2\|\mathbf{u}_{\mathrm{M}}(n)\|^2}\varepsilon_p(n) \tag{30}$$

Taking the squared Euclidean norm of both sides of (29) and (30) the following energy conservation relations are obtained which describe the evolution of the weight error vectors:

$$\begin{aligned}
&\left\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1)\right\|^2 + \widetilde{\mathbf{h}}^{\mathsf{H}}_{\mathrm{M}}(n,n+1)\frac{\mathbf{u}^*_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\varepsilon_a(n) \\
&\quad + \varepsilon^*_a(n)\frac{\mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1) + \frac{|\varepsilon_a(n)|^2}{4\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2} \\
&\quad = \left\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n)\right\|^2 + \widetilde{\mathbf{h}}^{\mathsf{H}}_{\mathrm{M}}(n,n)\frac{\mathbf{u}^*_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\varepsilon_p(n) \\
&\quad + \varepsilon^*_p(n)\frac{\mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n) + \frac{|\varepsilon_p(n)|^2}{4\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}
\end{aligned} \tag{31}$$

and

$$\begin{aligned}
&\|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1)\|^2 + \widetilde{\mathbf{g}}^{\mathsf{H}}_{\mathrm{M}}(n,n+1)\frac{\mathbf{u}_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\varepsilon_a(n) \\
&\quad + \varepsilon^*_a(n)\frac{\mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1) + \frac{|\varepsilon_a(n)|^2}{4\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2} \\
&\quad = \|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n)\|^2 + \widetilde{\mathbf{g}}^{\mathsf{H}}_{\mathrm{M}}(n,n)\frac{\mathbf{u}_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\varepsilon_p(n) \\
&\quad + \varepsilon^*_p(n)\frac{\mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n) + \frac{|\varepsilon_p(n)|^2}{4\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}
\end{aligned} \tag{32}$$

Adding up (31) and (32) gives

$$\begin{aligned}
&\left\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1)\right\|^2 + \|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1)\|^2 + \frac{|\varepsilon_a(n)|^2}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2} \\
&+ \frac{\varepsilon_a(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\left[\underbrace{\widetilde{\mathbf{h}}^{\mathsf{H}}_{\mathrm{M}}(n,n+1)\mathbf{u}^*_{\mathrm{M}}(n) + \widetilde{\mathbf{g}}^{\mathsf{H}}_{\mathrm{M}}(n,n+1)\mathbf{u}_{\mathrm{M}}(n)}_{\varepsilon^*_p(n)}\right] \\
&+ \frac{\varepsilon^*_a(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\left[\underbrace{\mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1) + \mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1)}_{\varepsilon_p(n)}\right] \\
&= \left\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n)\right\|^2 + \|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n)\|^2 + \frac{|\varepsilon_p(n)|^2}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}
\end{aligned}$$

$$+\frac{\varepsilon_p(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\left[\underbrace{\widetilde{\mathbf{h}}^{\mathsf{H}}_{\mathrm{M}}(n,n)\mathbf{u}^{*}_{\mathrm{M}}(n)+\widetilde{\mathbf{g}}^{\mathsf{H}}_{\mathrm{M}}(n,n)\mathbf{u}_{\mathrm{M}}(n)}_{\varepsilon_a^*(n)}\right]$$

$$+\frac{\varepsilon_p^*(n)}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\left[\underbrace{\mathbf{u}^{\mathsf{T}}_{\mathrm{M}}(n)\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n)+\mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n)}_{\varepsilon_a(n)}\right] \quad (33)$$

Consequently, the weight error energy conservation equation for the widely linear modeling case becomes

$$\left\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1)\right\|^2+\left\|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1)\right\|^2+\frac{\left|\varepsilon_a(n)\right|^2}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}$$
$$=\left\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n)\right\|^2+\left\|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n)\right\|^2+\frac{\left|\varepsilon_p(n)\right|^2}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2} \quad (34)$$

The expression (34) delivers an exact relation between the partial-update a priori and a posteriori estimation errors and partial-update weight errors. At the steady-state we have [1]

$$\mathsf{E}\left[\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n+1)\|^2\right]=\mathsf{E}\left[\|\widetilde{\mathbf{h}}_{\mathrm{M}}(n,n)\|^2\right]<\infty \quad (35)$$
$$\mathsf{E}\left[\|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n+1)\|^2\right]=\mathsf{E}\left[\|\widetilde{\mathbf{g}}_{\mathrm{M}}(n,n)\|^2\right]<\infty \quad (36)$$

Taking the statistical expectation of both sides of (34), at steady-state (34) becomes

$$\mathsf{E}\left[\frac{\left|\varepsilon_a(n)\right|^2}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\right]=\mathsf{E}\left[\frac{\left|\varepsilon_p(n)\right|^2}{2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2}\right] \quad \text{as } n\to\infty \quad (37)$$

Substituting (27) into the right-hand-side of above expression, we obtain

$$\mu\mathsf{E}\left[\left|e(n)\right|^2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right]=\mathsf{E}\left[\left|\varepsilon_a(n)\right|\left|e(n)\right|\right] \quad \text{as } n\to\infty \quad (38)$$

In the sequel, the steady-state variance relation (38) is used to obtain a closed-form expression for the EMSE measure. First, the following assumption is considered which is commonly used to study adaptive filters [1]:

Assumption 2.

(i) The a priori estimation error $\varepsilon_a(n)$ is statistically independent of the input regressor vector $\mathbf{u}_{\mathrm{M}}(m)$ for all $n\neq m$.

(ii) (separation assumption [1]) $\mathbf{u}_{\mathrm{M}}(n)$ is statistically independent of $e_a(n)$ and alternatively. This means that $\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2$ is also statistically independent of $e(n)$.

Since $e(n)=e_a(n)+v(n)$, (38) can be rewritten as

$$\mu\mathsf{E}\left[\left|e_a(n)\right|^2+2\left|e_a(n)\right|\left|v(n)\right|+\left|v(n)\right|^2\right]$$
$$=\mathsf{E}\left[\left|\varepsilon_a(n)\right|\left|e_a(n)\right|+\left|\varepsilon_a(n)\right|\left|v(n)\right|\right] \quad (39)$$

Under *Assumptions* 1 and 2.i, the above expression simplifies to

$$\mu\mathsf{E}\left[\left|e_a(n)\right|^2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right]+\mu\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})=\mathsf{E}\left[\varepsilon_a(n)e_a(n)\right] \quad (40)$$

where

$$\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})=\mathsf{tr}(\mathsf{E}[\mathbf{u}_{\mathrm{M}}(n)\mathbf{u}^{\mathsf{H}}_{\mathrm{M}}(n)])=\mathsf{E}\left[\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right] \quad (41)$$

For the first term in the RHS of (39) we have

$$\mathsf{E}[\varepsilon_a(n)e_a(n)]=\rho_{\mathrm{M}}\mathsf{E}\left[\left|e_a(n)\right|^2\right]=\rho_{\mathrm{M}}\zeta(\infty) \quad (42)$$

where $0<\rho_{\mathrm{M}}\leq 1$ is a constant that shows how much $e_a(n)$ is lessened as a consequence of partial coefficient updates. Substituting (41) into (40) gives:

$$\zeta(\infty)=\frac{\mu}{\rho_{\mathrm{M}}}\left(\mathsf{E}\left[\left|e_a(n)\right|^2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right]+\mu\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})\right) \quad (43)$$

Employing *Assumption* 2.ii, $\mathsf{E}\left[\left|e_a(n)\right|^2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right]$ can be separated into the product of two expectations as follows:

$$\mathsf{E}\left[\left|e_a(n)\right|^2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right]=\mathsf{E}\left[\left|e_a(n)\right|^2\right]\mathsf{E}\left[\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right] \quad (44)$$

Now substituting (44) into (43) gives the following steady-state EMSE expression:

$$\zeta(\infty)=\frac{\mu\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})}{\rho_{\mathrm{M}}-\mu\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})} \quad (45)$$

Remark 6. For sufficiently small step-size μ, at steady-state we expect to have:

$$\mathsf{E}\left[\left|e_a(n)\right|^2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right]\ll\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}}) \quad (46)$$

and we neglect the term $\mathsf{E}\left[\left|e_a(n)\right|^2\left\|\mathbf{u}_{\mathrm{M}}(n)\right\|^2\right]$ to derive an closed-form for EMSE as

$$\zeta(\infty)\approx\frac{\mu\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})}{\rho_{\mathrm{M}}} \quad (47)$$

Corollary 1. If all the filter coefficients are updated (i.e. $M=N$, $\boldsymbol{\mathcal{I}}_{\mathrm{M}}(n)=\mathbf{I}_N$, $\rho_{\mathrm{M}}=1$ and $\mathbf{C}_{\mathbf{u}_M}=\mathbf{C}_{\mathbf{u}}$), then (45) changes to

$$\zeta_{\mathrm{ACLMS}}(\infty)=\frac{\mu\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}})}{1-\mu\mathsf{tr}(\mathbf{C}_{\mathbf{u}})} \quad (48)$$

For small step sizes this can be approximated as

$$\zeta_{\mathrm{ACLMS}}(\infty)\approx\mu\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}}) \quad (49)$$

This expression is the steady-state EMSE for a *full-update* PU-ACLMS algorithm.

Corollary 2. For the PU-ACLMS algorithm we have[1]

$$\mathbf{C}_{\mathbf{u}_{\mathrm{M}}}=\frac{M}{N}\mathbf{C}_{\mathbf{u}},\quad \rho_{\mathrm{M}}=\frac{M}{N} \quad (50)$$

Replacing (50) in (48) we obtain

$$\zeta(\infty)=\frac{\mu\sigma_v^2\frac{M}{N}\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})}{\frac{M}{N}-\mu\mathsf{tr}(\mathbf{C}_{\mathbf{u}_{\mathrm{M}}})}=\frac{\mu\sigma_v^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}})}{1-\mu\mathsf{tr}(\mathbf{C}_{\mathbf{u}})} \quad (51)$$

which is the same steady-state EMSE as for the *full-update* PU-ACLMS algorithm.

According to corollaries 1 and 2, the steady-state performance of the PU-ACLMS algorithm can be summarized as the following Proposition.

Proposition 1. *Let Assumptions 1 and 2 hold. Then, regardless of the values of M, both sequential and stochastic partial-update schemes have the same steady-state EMSE as that of the full-update ACLMS algorithm.*

[1] This relation holds for stationary input signals.

B. Stability Analysis

To gain further insight into the performance of partial-update PU-ACLMS algorithms, we shall continue to examine the stability of PU-ACLMS algorithm. To this end, we consider the update equations (6) and (7). Let us also define the partial-augmented weight vector $\widetilde{\mathbf{w}}(n)$ as

$$\widetilde{\mathbf{w}}(n) \triangleq \mathbf{w}^{o} - \mathbf{w}(n) \tag{52}$$

where $\mathbf{w}^{o} = [\mathbf{h}^{o\mathsf{T}}, \mathbf{g}^{o\mathsf{T}}]^{\mathsf{T}}$. Substracting both sides of (12) from the optimum solution $\mathbf{w}^{o}$ gives

$$\widetilde{\mathbf{w}}\,(i+1) = \widetilde{\mathbf{w}}(n) + \mu e(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n) \tag{53}$$

It is useful to mention that the expression (53) leaves non-updated subsets of $\tilde{\mathbf{w}}(n)$ and $\tilde{\mathbf{w}}(n+1)$ identical. From (53), the output-error $e(n)$ can be rewritten as

$$e(n) = \mathbf{z}^{\mathsf{H}}(n)\tilde{\mathbf{w}}(n) + \upsilon(n) \tag{54}$$

Then, the weight-error-recursion in (53) becomes

$$\begin{aligned}\widetilde{\mathbf{w}}(n+1) = \left[\mathbf{I} - \mu\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\right]\widetilde{\mathbf{w}}(n)&\\ - \mu\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\upsilon(n)&\end{aligned} \tag{55}$$

In the sequel, recursion (55) is used to obtain the mean and mean-square stability conditions.

1) Mean Stability: Applying statistical expressions to the both sides of (55) and employing *Assumption* 1, yields

$$\begin{aligned}\mathsf{E}[\widetilde{\mathbf{w}}(n+1)] &= \left(\mathbf{I} - \mu\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)]\right)\mathsf{E}[\widetilde{\mathbf{w}}(n)]\\ &= \left(\mathbf{I} - \mu\mathbf{C}^{*}_{\mathbf{z}_{\mathrm{M}}}\right)\mathsf{E}[\widetilde{\mathbf{w}}(n)]\end{aligned} \tag{56}$$

where $\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}$ is the partial-augmented covariance matrix of $\mathbf{z}(n)$, given by

$$\begin{aligned}\mathbf{C}_{\mathbf{z}_{\mathrm{M}}} &\triangleq \mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)]\\ &= \mathsf{E}\left[\begin{bmatrix}\boldsymbol{\mathcal{I}}_{\mathrm{M}}(n) & 0\\ 0 & \boldsymbol{\mathcal{I}}_{\mathrm{M}}(n)\end{bmatrix}\begin{bmatrix}\mathbf{u}(n)\mathbf{u}^{\mathsf{H}}(n) & \mathbf{u}(n)\mathbf{u}^{\mathsf{T}}(n)\\ \mathbf{u}^{*}(n)\mathbf{u}^{\mathsf{H}}(n) & \mathbf{u}^{*}(n)\mathbf{u}^{\mathsf{T}}(n)\end{bmatrix}\right]\\ &= \begin{bmatrix}\mathbf{C}_{\mathbf{u}_{\mathrm{M}}} & \mathbf{D}_{\mathbf{u}_{\mathrm{M}}}\\ \mathbf{D}^{*}_{\mathbf{u}_{\mathrm{M}}} & \mathbf{C}^{*}_{\mathbf{u}_{\mathrm{M}}}\end{bmatrix}\end{aligned} \tag{57}$$

The matrices $\mathbf{C}_{\mathbf{u}_{\mathrm{M}}}$ and $\mathbf{D}_{\mathbf{u}_{\mathrm{M}}}$ are respectively denoted as the partial covariance and partial complementary covariance matrices of $\mathbf{u}_{\mathrm{M}}(n)$.

We establish the following Proposition which guarantees asymptotic unbiasedness of the PU-ACLMS algorithm.

Proposition 2. (Mean Stability) *Let the widely-linear model* (1) *and Assumption 1 hold. Then, the PU-ACLMS algorithm is asymptotically unbiased for any initial condition if, and only if, the positive step-size parameter μ satisfies*

$$0 < \mu < \frac{2}{\lambda_{\max}\left(\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\right)} \tag{58}$$

where $\lambda_{\max}\left(\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\right)$ is the largest eigenvalue of the partial-augmented matrix $\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}$.

Proof: See Appendix A. ■

2) Mean-square Stability: By computing the weighted norm of both sides of (55), for an arbitrary Hermitioan weighting matrix $\boldsymbol{\Sigma} > 0$, and applying the expectation operator together with employing Assumption 1, we arrive at the following weighted variance relation:

$$\begin{aligned}\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n+1)\|^{2}_{\boldsymbol{\Sigma}}\Big] = \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^{2}_{\boldsymbol{\Sigma}'}\Big]&\\ + \mu^{2}\sigma^{2}_{\upsilon}\mathsf{E}[\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\boldsymbol{\Sigma}\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)]&\end{aligned} \tag{59}$$

$$\begin{aligned}\boldsymbol{\Sigma}' = \boldsymbol{\Sigma} - \mu\boldsymbol{\Sigma}\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n) - \mu\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\boldsymbol{\Sigma}&\\ + \mu^{2}\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\boldsymbol{\Sigma}\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)&\end{aligned} \tag{60}$$

Under *Assumptions* 1 and 2, $\widetilde{\mathbf{w}}(n)$ is independent of both $\boldsymbol{\Sigma}^{'}$ and $\mathbf{z}(n)$. Thus, we can split $\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^{2}_{\boldsymbol{\Sigma}'}\Big]$ into

$$\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^{2}_{\boldsymbol{\Sigma}'}\Big] = \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^{2}_{\mathsf{E}[\boldsymbol{\Sigma}']}\Big] \tag{61}$$

with the weighting matrix $\boldsymbol{\Sigma}^{'}$ replaced by its mean, denoted by $\boldsymbol{\Gamma}$, i.e. $\boldsymbol{\Gamma} \triangleq \mathsf{E}\big[\boldsymbol{\Sigma}'\big]$. In this way, the recursions (59) and (60) are rewritten as follows:

$$\begin{aligned}\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n+1)\|^{2}_{\boldsymbol{\Sigma}}\Big] = \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^{2}_{\boldsymbol{\Gamma}}\Big]&\\ + \mu^{2}\sigma^{2}_{\upsilon}\mathsf{E}\Big[\|\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\|^{2}_{\boldsymbol{\Sigma}}\Big]&\end{aligned} \tag{62}$$

$$\begin{aligned}\boldsymbol{\Gamma} = \boldsymbol{\Sigma} - \mu\boldsymbol{\Sigma}\mathbf{C}_{\mathbf{z}_{\mathrm{M}}} - \mu\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\boldsymbol{\Sigma}&\\ + \mu^{2}\mathsf{E}\Big[\|\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\|^{2}_{\boldsymbol{\Sigma}}\,\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\Big]&\end{aligned} \tag{63}$$

By defining $\boldsymbol{\gamma} \triangleq \mathrm{vec}\left\{\mathsf{E}[\boldsymbol{\Gamma}]\right\}$ and $\boldsymbol{\sigma} \triangleq \mathrm{vec}\left\{\mathsf{E}[\boldsymbol{\Sigma}]\right\}$ in addition to applying the Kronecker product notation property[2] we can modify (63) as

$$\begin{aligned}\mathrm{vec}\left\{\boldsymbol{\Gamma}\right\} = \mathrm{vec}\left\{\boldsymbol{\Sigma}\right\} - \mu\mathrm{vec}\left\{\boldsymbol{\Sigma}\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\right\} - \mu\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\boldsymbol{\Sigma}\right\}&\\ + \mu^{2}\mathsf{E}\Big[\mathrm{vec}\Big\{\|\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\|^{2}_{\boldsymbol{\Sigma}}\,\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\Big\}\Big]&\end{aligned} \tag{64}$$

After vectorization, the relations (62) and (63) become:

$$\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n+1)\|^{2}_{\boldsymbol{\sigma}}\Big] = \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^{2}_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}\Big] + \mu^{2}\sigma^{2}_{\upsilon}\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\sigma} \tag{65}$$

where the coefficient matrix $\boldsymbol{\mathcal{F}}$ is given by

$$\begin{aligned}\boldsymbol{\mathcal{F}} = \mathbf{I}_{(2N)^{2}} - \mu\left(\mathbf{C}^{\mathsf{T}}_{\mathbf{z}_{\mathrm{M}}} \otimes \mathbf{I}_{2N} + \mathbf{I}_{2N} \otimes \mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\right)&\\ + \mu^{2}\mathsf{E}\big[(\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n))^{\mathsf{T}} \otimes (\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n))\big]&\end{aligned} \tag{66}$$

and the vector $\mathbf{c}_{\mathrm{M}}$ is defined as

$$\mathbf{c}_{\mathrm{M}} = \mathrm{vec}\left\{\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)]\right\} \tag{67}$$

which is derived from

$$\begin{aligned}\mathsf{E}\Big[\|\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\|^{2}_{\boldsymbol{\Sigma}}\Big] &= \mathsf{E}[\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\boldsymbol{\Sigma}\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)]\\ &= \mathrm{tr}(\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\boldsymbol{\Sigma}])\\ &= \mathrm{tr}(\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)]\,\boldsymbol{\Sigma})\\ &= \mathrm{vec}^{\mathsf{T}}\left\{\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)]\right\}\boldsymbol{\sigma}\\ &= \mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\sigma}\end{aligned} \tag{68}$$

[2] For any matrices $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\Sigma}$ with compatible dimensions we have

$$\mathrm{vec}\left\{\mathbf{A}\boldsymbol{\Sigma}\mathbf{B}\right\} = \left(\mathbf{B}^{\mathsf{T}} \otimes \mathbf{A}\right)\mathrm{vec}\left\{\boldsymbol{\Sigma}\right\},$$

Note that, the $(2N)^2 \times (2N)^2$ coefficient matrix $\boldsymbol{\mathcal{F}}$ can be expressed as:

$$\boldsymbol{\mathcal{F}}(\mu) = \mathbf{I} - \mu \mathbf{P} + \mu^2 \mathbf{Q} \tag{69}$$

where $(2N)^2 \times (2N)^2$ matrices $\{\mathbf{P}, \mathbf{Q}\}$ are given by

$$\mathbf{P} = \mathbf{C}^{\mathsf{T}}_{\mathbf{z}_{\mathrm{M}}} \otimes \mathbf{I}_{2N} + \mathbf{I}_{2N} \otimes \mathbf{C}_{\mathbf{z}_{\mathrm{M}}} \tag{70}$$

$$\mathbf{Q} = \mathsf{E}\big[(\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n))^{\mathsf{T}} \otimes (\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n))\big] \tag{71}$$

To derive the stability conditions for the step-size parameter μ, we employ the state-space model developed in [1]. Here, we avoid to expose the corresponding argument, except to state that the stability of matrix $\boldsymbol{\mathcal{F}}$ ensures the mean-square stability of the PU-ACLMS algorithms. In this way, the following proposition holds.

Proposition 3. (Mean-square Stability) *Let the data $\{d(n), \mathbf{u}(n)\}$ satisfy the widely-linear model* (1) *and the independence assumption 1 holds. Then, the PU-ACLMS algorithm is mean-square stable if, and only if, the step-size μ is chosen to satisfy*

$$0 < \mu < \min\left\{\frac{1}{\lambda_{\max}\left(\mathbf{P}^{-1}\mathbf{Q}\right)}, \frac{1}{\max\left\{\lambda\left\{\mathbf{G}\right\} \in \mathbb{R}^{>0}\right\}}\right\} \tag{72}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are defined by (70) *and* (71), *and $\mathbf{G} \in \mathbb{C}^{M^2 \times M^2}$ is defined by the following block matrix*

$$\mathbf{G} = \begin{bmatrix} \mathbf{P}/2 & -\mathbf{Q}/2 \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \tag{73}$$

Proof: See Appendix B. ■

C. Transient Analysis

1) Learning Curves: Evolution of the EMSE and MSD over time is described by the learning curves which are defined as

$$\zeta(n) = \mathsf{E}\big[|e_a(n)|^2\big], \quad \eta(n) = \mathsf{E}\big[\|\mathbf{w}(n)\|^2\big] \tag{74}$$

First, the learning curve for EMSE is obtained. Iterating (65) starting from $n = 0$ we achieve

$$\begin{aligned}\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n+1)\|^2_{\boldsymbol{\sigma}}\Big] &= \|\mathbf{w}^o\|^2_{\boldsymbol{\mathcal{F}}^{n+1}\boldsymbol{\sigma}} + \mu^2\sigma_\upsilon^2 \sum_{j=0}^{n} \mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\mathcal{F}}^k\boldsymbol{\sigma} \\ &= \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^2_{\boldsymbol{\sigma}}\Big] + \|\mathbf{w}^o\|^2_{\boldsymbol{\mathcal{F}}^{n+1}\boldsymbol{\sigma}} - \|\mathbf{w}^o\|^2_{\boldsymbol{\mathcal{F}}^{n}\boldsymbol{\sigma}} + \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\mathcal{F}}^n\boldsymbol{\sigma} \\ &= \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^2_{\boldsymbol{\sigma}}\Big] + \|\mathbf{w}^o\|^2_{\boldsymbol{\mathcal{F}}^{n}(\boldsymbol{\mathcal{F}}-\mathbf{I})\boldsymbol{\sigma}} + \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\mathcal{F}}^n\boldsymbol{\sigma} \end{aligned} \tag{75}$$

where $\mathbf{w}(0) = 0$. Using $e_a(n) = \mathbf{z}^{\mathsf{H}}(n)\widetilde{\mathbf{w}}(n)$ the EMSE at time n can be expressed alternatively as

$$\begin{aligned}\zeta(n) &= \mathsf{E}[\widetilde{\mathbf{w}}^{\mathsf{H}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\widetilde{\mathbf{w}}(n)] \\ &= \mathsf{E}[\mathsf{E}[\widetilde{\mathbf{w}}^{\mathsf{H}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\widetilde{\mathbf{w}}(n)]\,|\widetilde{\mathbf{w}}(n)] \\ &= \mathsf{E}[\widetilde{\mathbf{w}}^{\mathsf{H}}(n)\mathbf{C}_{\mathbf{z}}\widetilde{\mathbf{w}}(n)] \\ &= \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(n)\|^2_{\mathbf{C}_{\mathbf{z}}}\Big]\end{aligned} \tag{76}$$

where $\mathbf{C}_{\mathbf{z}} = \mathsf{E}[\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)]$. Thus, by setting $\boldsymbol{\sigma} = \mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\right\}$ in (75), the time evolution of EMSE can be obtained.

Proposition 4. (Learning Curve) *Consider the same setting of Proposition 3. Let $\zeta(n)$ denote the time evolution of EMSE, as defined by* (76). *Then, the EMSE learning curve of PU-ACLMS is given by the following recursion over $n \geq 0$:*

$$\begin{aligned}\zeta(n+1) = \zeta(n) &+ \|\mathbf{w}^o\|^2_{\boldsymbol{\mathcal{F}}^{n}(\boldsymbol{\mathcal{F}}-\mathbf{I})\mathrm{vec}\{\mathbf{C}_{\mathbf{z}}\}} \\ &+ \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\mathcal{F}}^n\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\right\}\end{aligned} \tag{77}$$

Similarly, using (74) *the MSD learning curve can be evaluated by setting $\boldsymbol{\sigma} = \mathrm{vec}\left\{\mathbf{I}\right\}$ in* (75) *which gives*

$$\begin{aligned}\eta(n+1) = \eta(n) &+ \|\mathbf{w}^o\|^2_{\boldsymbol{\mathcal{F}}^{n}(\boldsymbol{\mathcal{F}}-\mathbf{I})\mathrm{vec}\{\mathbf{I}\}} \\ &+ \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\mathcal{F}}^n\mathrm{vec}\left\{\mathbf{I}\right\}\end{aligned} \tag{78}$$

2) Alternative Steady-state Analysis: Recursion (65) can be used to derive the steady-state values for EMSE and MSD by setting $n \to \infty$. In this case recursion (65) becomes

$$\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(\infty)\|^2_{\boldsymbol{\sigma}}\Big] = \mathsf{E}\Big[\|\widetilde{\mathbf{w}}(\infty)\|^2_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}\Big] + \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\sigma} \tag{79}$$

which is equivalent to

$$\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(\infty)\|^2_{(\mathbf{I}-\boldsymbol{\mathcal{F}})\boldsymbol{\sigma}}\Big] = \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\boldsymbol{\sigma} \tag{80}$$

To specify the EMSE, it is required to evaluate $\mathsf{E}\Big[\|\widetilde{\mathbf{w}}(\infty)\|^2_{\boldsymbol{\sigma}}\Big]$ by setting $\boldsymbol{\sigma} = (\mathbf{I} - \boldsymbol{\mathcal{F}})^{-1}\,\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\right\}$. In this way, the left-hand side of (80) is the steady-state EMSE and we have

$$\zeta(\infty) = \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\left(\mathbf{I} - \boldsymbol{\mathcal{F}}\right)^{-1}\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\right\} \tag{81}$$

For sufficiently small step-size μ, expression (66) can be approximated by

$$\boldsymbol{\mathcal{F}} \approx \mathbf{I} - 2\mu(\mathbf{C}_{\mathbf{z}_{\mathrm{M}}} \otimes \mathbf{I}) \tag{82}$$

Substituting (82) into (81) leads to

$$\begin{aligned}\zeta(\infty) &= \mu^2\sigma_\upsilon^2\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\left(\mathbf{I} - \boldsymbol{\mathcal{F}}\right)^{-1}\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\right\} \\ &\approx \frac{\mu\sigma_\upsilon^2}{2}\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\left(\mathbf{C}_{\mathbf{z}_{\mathrm{M}}} \otimes \mathbf{I}\right)^{-1}\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\right\} \\ &\approx \frac{\mu\sigma_\upsilon^2}{2}\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\left(\mathbf{C}^{-1}_{\mathbf{z}_{\mathrm{M}}} \otimes \mathbf{I}\right)\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\right\} \\ &\approx \frac{\mu\sigma_\upsilon^2}{2}\mathbf{c}^{\mathsf{T}}_{\mathrm{M}}\mathrm{vec}\left\{\mathbf{C}_{\mathbf{z}}\mathbf{C}^{-1}_{\mathbf{z}_{\mathrm{M}}}\right\} \\ &\approx \frac{\mu\sigma_\upsilon^2}{2}\mathsf{tr}\big(\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)]\,\mathbf{C}_{\mathbf{z}}\mathbf{C}^{-1}_{\mathbf{z}_{\mathrm{M}}}\big)\end{aligned} \tag{83}$$

Under Assumptions 1 and 2, we have

$$\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)] = \mathbf{C}_{\mathbf{z}_{\mathrm{M}}} \tag{84}$$

$$\mathsf{tr}\big(\mathsf{E}[\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)\mathbf{z}(n)\mathbf{z}^{\mathsf{H}}(n)\boldsymbol{\mathcal{J}}_{\mathrm{M}}(n)]\,\mathbf{C}_{\mathbf{z}}\mathbf{C}^{-1}_{\mathbf{z}_{\mathrm{M}}}\big) = \mathsf{tr}(\mathbf{C}_{\mathbf{z}}) \tag{85}$$

replacing (84) and (85) in (83) yields

$$\zeta(\infty) \approx \frac{\mu\sigma_\upsilon^2}{2}\mathsf{tr}(\mathbf{C}_{\mathbf{z}}) \tag{86}$$

Since $\mathsf{tr}(\mathbf{C}_{\mathbf{z}}) = 2\mathsf{tr}(\mathbf{C}_{\mathbf{u}})$, the steady-state EMSE is given by

$$\zeta(\infty) \approx \mu\sigma_\upsilon^2\mathsf{tr}(\mathbf{C}_{\mathbf{u}}) \tag{87}$$

This is the same result as that obtained for the full-update ACLMS algorithm (see (49)), which according to Proposition 1 is also valid for partial update schemes. The steady-state MSD can be calculated in a similar way as

$$\eta(\infty) \approx \mu\sigma_\upsilon^2 N \tag{88}$$

D. Convergence Rate Analysis

It is useful to compare the convergence rate of PU-ACLMS algorithms to the full-update ACLMS algorithm. For this, we examine the decay rates of sequential and stochastic schemes under the following assumptions.

Assumption 3.

(i) The filter length N is a multiple of β, i.e. $\frac{N}{M}$ is an integer.

(ii) The augmented covariance matrix of $\mathbf{z}(n)$, $\mathbf{C}_{\mathbf{z}}$, is block-diagonal such that $\sum_{t=1}^{\beta} \boldsymbol{\mathcal{T}}_t \mathbf{C}_{\mathbf{z}} \boldsymbol{\mathcal{T}}_t = \mathbf{C}_{\mathbf{z}}$. The matrix $\boldsymbol{\mathcal{T}}_t$ is defined by zeroing out some rows in the identity matrix $\mathbf{I}$ such that $\sum_{t=1}^{\beta} \boldsymbol{\mathcal{T}}_t = \mathbf{I}$.

Note that, *Assumption* 3-ii is used to make the analysis tractabe.

First, it is required to restate the evolution equations of the existing algorithms. Thus, for the regular full-update ACLMS algorithm the recursion (56) is rewritten as follows:

$$\mathsf{E}[\widetilde{\mathbf{w}}(n+1)] = (\mathbf{I} - \mu \mathbf{C}_{\mathbf{z}}^{*})\,\mathsf{E}[\widetilde{\mathbf{w}}(n)] \tag{89}$$

Combining β-iterations of the recursion (89), yields the mean of the coefficient error vector update given by

$$\begin{aligned}\mathsf{E}[\widetilde{\mathbf{w}}\,(n+\beta)] &= \prod_{t=1}^{\beta} (\mathbf{I} - \mu \mathbf{C}_{\mathbf{z}}^{*})\,\mathsf{E}[\widetilde{\mathbf{w}}\,(n+t-1)] \\ &= (\mathbf{I} - \mu \mathbf{C}_{\mathbf{z}}^{*})^{\beta}\,\mathsf{E}[\widetilde{\mathbf{w}}(n)] \end{aligned}\tag{90}$$

For the sequential PU-ACLMS, the mean-error-update equation is given by

$$\mathsf{E}[\widetilde{\mathbf{w}}\,(n+1)] = \left(\mathbf{I} - \mu \boldsymbol{\mathcal{T}}_{\mathrm{mod}(n,\beta)+1} \mathbf{C}_{\mathbf{z}}^{*}\right) \mathsf{E}[\widetilde{\mathbf{w}}(n)] \tag{91}$$

For any number of optional fractions, the update equation (91) can be written as

$$\mathsf{E}[\widetilde{\mathbf{w}}\,(n+\beta)] = \prod_{t=1}^{\beta} \left(\mathbf{I} - \mu \boldsymbol{\mathcal{T}}_{\mathrm{mod}(n+t,\beta)+1} \mathbf{C}_{\mathbf{z}}^{*}\right) \mathsf{E}[\widetilde{\mathbf{w}}(n)] \tag{92}$$

Combining β updates of (92), the following relation for evolution of mean-error-update is obtained:

$$\mathsf{E}[\widetilde{\mathbf{w}}\,(n+\beta)] = \left(\mathbf{I} - \frac{\mu}{\beta} \mathbf{C}_{\mathbf{z}}^{*}\right)^{\beta} \mathsf{E}[\widetilde{\mathbf{w}}(n)] \tag{93}$$

In the stochastic scheme, the following evolution equation conditioned on a choice $\boldsymbol{\mathcal{S}}_t$, $t = 1, \ldots, \beta$, results in:

$$\begin{aligned}\mathsf{E}[\widetilde{\mathbf{w}}\,(n+1)\,|\boldsymbol{\mathcal{S}}_t] &= (\mathbf{I} - \mu \mathbf{T}_n \mathbf{C}_{\mathbf{z}}^{*})\,\mathsf{E}[\widetilde{\mathbf{w}}(n)|\boldsymbol{\mathcal{S}}_t] \\ &= (\mathbf{I} - \mu \boldsymbol{\mathcal{T}}_t \mathbf{C}_{\mathbf{z}}^{*})\,\mathsf{E}[\widetilde{\mathbf{w}}(n)|\boldsymbol{\mathcal{S}}_t] \end{aligned}\tag{94}$$

where the matrix $\mathbf{T}_n$ is chosen randomly from $\boldsymbol{\mathcal{T}}_t$, $t = 1, \ldots, \beta$, with equal probability. Averaging (94) over all choices of $\boldsymbol{\mathcal{S}}_k$, we get

$$\mathsf{E}[\widetilde{\mathbf{w}}\,(n+1)] = \left(\mathbf{I} - \frac{\mu}{\beta} \mathbf{C}_{\mathbf{z}}^{*}\right) \mathsf{E}[\widetilde{\mathbf{w}}(n)] \tag{95}$$

Collecting β updates of this equation, the mean of coefficient error vector update is

$$\begin{aligned}\mathsf{E}[\widetilde{\mathbf{w}}\,(n+\beta)] &= \prod_{t=1}^{\beta} \left(\mathbf{I} - \frac{\mu}{\beta} \mathbf{C}_{\mathbf{z}}^{*}\right) \mathsf{E}[\widetilde{\mathbf{w}}\,(n+t-1)] \\ &= \left(\mathbf{I} - \frac{\mu}{\beta} \mathbf{C}_{\mathbf{z}}^{*}\right)^{\beta} \mathsf{E}[\widetilde{\mathbf{w}}(n)] \end{aligned}\tag{96}$$

In order to gain further insight into the convergence behavior of the proposed algorithms, we used the method in [1, pp. 148–150] to obtain the modes of convergence for diferent algorithms. To continue, since $\mathbf{C}_{\mathbf{z}}^{*}$ is Hermitian and positive semi-definite, we can write the eigen-decomposition of it as

$$\mathbf{C}_{\mathbf{z}}^{*} = \boldsymbol{\mathcal{U}} \Lambda \boldsymbol{\mathcal{U}}^{\mathsf{H}}$$

where $\Lambda = \mathrm{diag}\{\lambda_1, \ldots, \lambda_N\}$, $\lambda_k \geq \lambda_{k+1}$, is the diagonal matrix of the eigenvalues of $\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}^{*}$ and $\boldsymbol{\mathcal{U}}$ is the matrix of the corresponding eigenvectors [3].

To express the convergence modes [12] solely in terms of the corresponding eigenvalues of the partial correlation matrix, we rotate the partial weight error vector $\widetilde{\mathbf{w}}(n)$ by the eigenmatrix $\boldsymbol{\mathcal{U}}$, that is, $\widetilde{\mathbf{w}}^{'}(n) = \boldsymbol{\mathcal{U}}\widetilde{\mathbf{w}}(n)$. This linear transformation changes the evolution of the partial weighted error vectors in different algorithms as follows:

$$\mathsf{E}[\widetilde{\mathbf{w}}'\,(n+1)] = (\mathbf{I} - \mu\Lambda)\,\mathsf{E}[\widetilde{\mathbf{w}}'(n)]\,, \quad \text{(full)} \tag{97}$$

$$\mathsf{E}[\widetilde{\mathbf{w}}'\,(n+1)] = (\mathbf{I} - \mu\Lambda)^{\beta}\,\mathsf{E}[\widetilde{\mathbf{w}}'(n)]\,, \quad \text{(seq)} \tag{98}$$

$$\mathsf{E}[\widetilde{\mathbf{w}}'\,(n+1)] = (\mathbf{I} - \mu\Lambda)^{\beta}\,\mathsf{E}[\widetilde{\mathbf{w}}'(n)]\,, \quad \text{(stoch)} \tag{99}$$

From (97), (98), and (99), the modes of convergence for different algorithms are given by

$$r_{\text{full},m} = \Big(1 - \mu\lambda_m\Big)^{\beta} \tag{100}$$

$$r_{\text{seq},m} = \left(1 - \frac{\mu\lambda_m}{\beta}\right)^{\beta} \tag{101}$$

$$r_{\text{stoch},m} = \left(1 - \frac{\mu\lambda_m}{\beta}\right)^{\beta} \tag{102}$$

Fig. 2 shows the modes of convergence for a special case $\mathbf{C}_{\mathbf{z}} = \mathbf{I}$, for $\beta = \{2, 4\}$, where $\bar{\mu} \triangleq \mu\lambda$. The smaller magnitude modes of convergence results in a faster decay to zero. Thus, from Fig. 2 it can be observe that the full-update ACLMS algorithm is faster than sequential and stochastic PUACLMS algorithms, regardless of the values of step-size parameter.

V. Simulation Results

In order to evaluate the performance of proposed PU-ACLMS algorithms the parameter estimation problem as shown in Fig. 1 is considered. The non-circular input vectors $\mathbf{u}(n)$ are generated ;

$$\mathbf{u}(n) = \sqrt{1 - \delta^2}\,\mathbf{q}_r(n) + \jmath\delta\mathbf{q}_i(n) \tag{103}$$

where $\jmath^2 = -1$ and $\delta = 0.95$. Moreover, $\mathbf{q}_r(n)$ and $\mathbf{q}_i(n)$ are zero-mean doubly-white circular Gaussian noise process with covariance matrix $\sigma_{\mathbf{q}}^2\mathbf{I}$ with $\sigma_{\mathbf{q}}^2 = 0.9$. The measurement noise $v(n)$ is generated from a zero-mean doubly-white circular Gaussian noise process with unit variance. The weights were initialized randomly. Te required matrices $\mathbf{C}_{\mathbf{z}}$ and $\mathbf{C}_{\mathbf{u}}$ are approximated with Mont Carlo (MC) simulation. All graphs are obtained by averaging over 100 independent trials. Fig. 3 show the scatter plot of desired signal $d(n)$ and measurement noise $\mathbf{u}(n)$.

[3] The eigenvectors, $\boldsymbol{\mathcal{U}}$, may be chosen to be orthonormal in which cases $\boldsymbol{\mathcal{U}}$ is unitary matrix; $\boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{U}}^{\mathsf{H}} = \boldsymbol{\mathcal{U}}^{\mathsf{H}}\boldsymbol{\mathcal{U}} = \mathbf{I}$

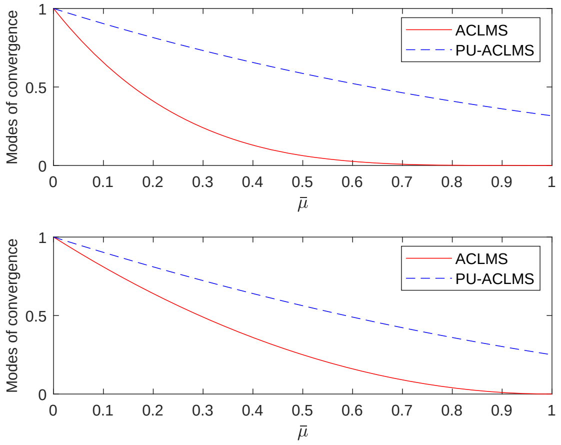

Fig. 2. The modes of convergence for different algorithms, $\beta = 2$ (top) and $\beta = 4$ (bottom).

Figs. 4 and 5 illustrate the resulting theoretical and simulated learning curves, EMSE, and MSD curves of PU-ACLMS algorithm using sequential and stochastic schemes for different choices of coefficients update M. From these figures it can be observed that the theoretical analysis precisely specify the EMSE and MSD evolutions, in both transient and steady-state stages.

Evidently, the full-update cases have the fastest convergence rate in comparison to PU-ACLMS algorithms. Generally speaking, the speed of convergence reduces proportionally for both sequential and stochastic schemes as the number of updated coefficients per iteration is decreased. Fig. 6 provides a comparison between MSD curves of sequential and stochastic schemes. As can be seen, the transient and steady-state behavior of of these two schemes are quite similar throughout all stage of adaptation.

It should be noted that, although the proposed algorithm has slower convergence rate compared with the full update ACLMS algorithm, due to its lower computational complexity, its execution time (per iteration) is lower than that of the full update ACLMS algorithm. For example, in our system[4] the execution time (per iteration) for $M = 2,\ 4,$ and 6 is 0.2304 mSec, 0.2381 mSec, and 0.2414 mSec, respectively. This time for the full update ACLMS algorithm is 0.2469 mSec.

In the next scenario, the performance of PU-ACLMS algorithm for different values of step-sizes, μ is evaluated. Fig. 7 shows the steady-state EMSE and MSD values of PU-ACLMS algorithm as a function of step-size using the theoretical expressions and simulated ones. It can be seen from 7 that the theoretical steady-state EMSE and MSD can well follow the simulated ones. Again in all figures, a good agreement between analytical and empirical results is achieved with maximum performance difference of only 0.002 dB.

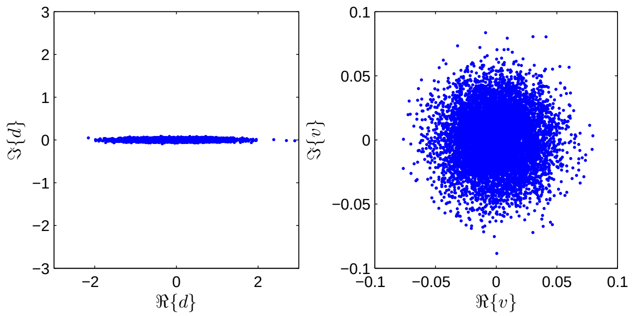

Fig. 3. The scatter plot of desired signal $d(n)$ and measurement noise $v(n)$.

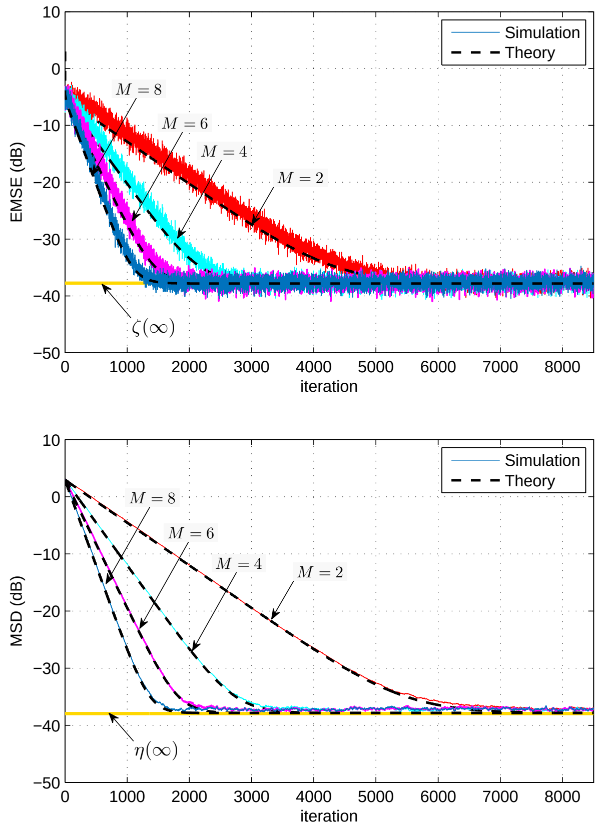

Fig. 4. Theoretical and experimental EMSE (top) and MSD (bottom) of PU-ACLMS algorithm using sequential scheme for different values of coefficients update M, when the step-size is $\mu = 0.02$.

Finally, we define the following performance loss to compare the performance of different algorithms:

$$P(\%) = \frac{\eta(n_s) - \eta(\infty)}{\eta(0) - \eta(\infty)}$$

where n_s denotes the number of iterations that the full update ACLMS algorithm reaches its steady-state value. Moreover, $\eta(0)$ and $\eta(\infty)$ denote the MSD values for the PU-ACLMS algorithm for $n = 0$ and $n \to \infty$, respectively. This quantity for the PU-ACLMS algorithm with sequential scheme with $M = 2,\ 4,$ and 6 are 3.79%, 0.21%, and 0.01%, respectively.

[4] The simulations are done in a PC Laptop 64-bit Intel Core i7-4510U CPU @ 2.002.60 GHz RAM 8.00 GB with MATLAB R2017b and its CVX toolbox working with Gurobi.

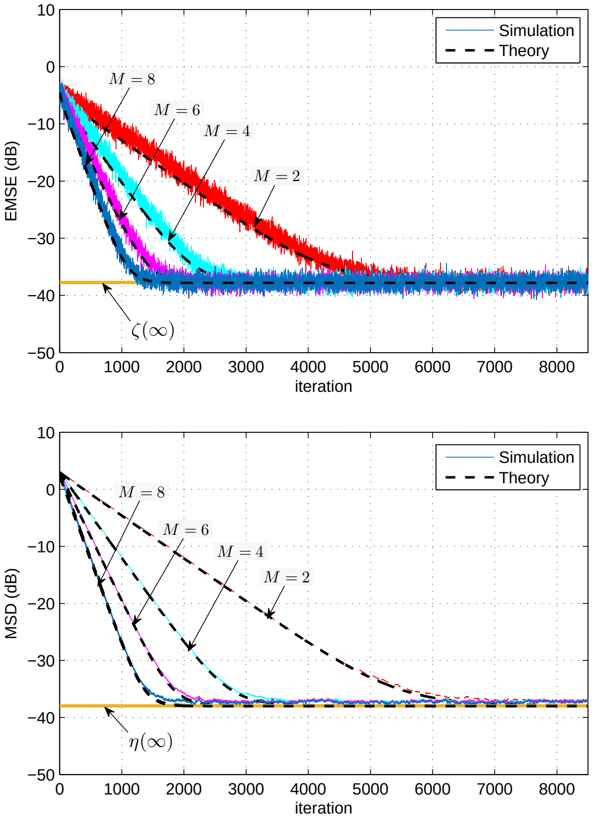

Fig. 5. Theoretical and experimental EMSE (top) and MSD (bottom) of PU-ACLMS algorithm using stochastic scheme for different values of coefficients update M, when the step-size is $\mu=0.02$.

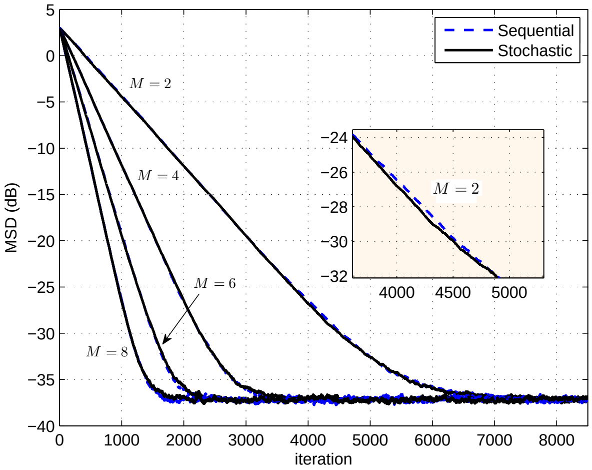

Fig. 6. MSD learning curves of PU-ACLMS algorithm for both sequential and stochastic schemes for different values of coefficients update M, when the step-size is $\mu=0.02$.

VI. Conclusion

In this paper, a new algorithm, termed briefly PU-ACLMS algorithm has been proposed, where only a portion of coefficient weights are selected to update at each iteration. Moreover, the computational complexity for full-update ACLMS and its partial-update implementations have been discussed. It has been concluded that for large filter lengths the PU-ACLMS algorithm is able to decrease the maximum complexity approximately by a factor of two. The steady-state performances of the proposed PU-ACLMS algorithm have been analyzed and the mean and mean-square convergence behavior of the PU-ACLMS algorithm studied for a second-order non-circular Gaussian input regressor. In addition, three cases of coefficient weight updates, i.e., full-update, sequential and stochastic partial update have been analyzed. The EMSE and MSD learning curves have been derived. The provided analysis demonstrated that at a given step-size μ the level of steady-state EMSE of full-update case and PU-ACLMS algorithm are identical. However, their convergence speeds are decreased in proportion to the number of coefficients weights updated per iteration divided by the filter length. The simulation results support the theoretical derivations.

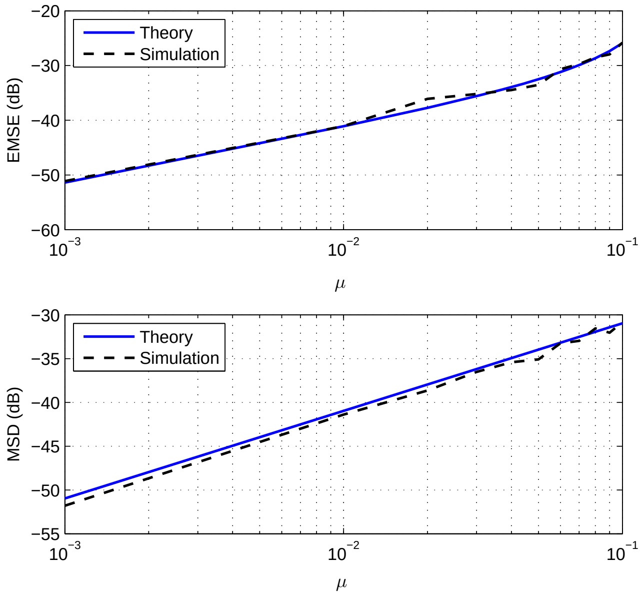

Fig. 7. Theoretical and experimental steady-state EMSE (top) and MSD (bottom) of sequential PU-ACLMS algorithm in terms of the step-size parameter μ.

Appendix A
Proof of Proposition 2

Proof: Consider the update equation (56). Using the eigen-decomposition of $\mathbf{C}^*_{\mathbf{z}_{\mathrm{M}}}$ (see (97)) the evolution of the partial weighted error vectors as follows:

$$\mathsf{E}[\widetilde{\mathbf{w}}'\,(n+1)] = (\mathbf{I}-\mu\Lambda)\,\mathsf{E}[\widetilde{\mathbf{w}}'(n)] \tag{A.2}$$

As the maximum eigenvalue, $\lambda_{\max}$, yields the fastest mode of convergence, the condition for convergence in mean of the PU-ACLMS algorithm becomes

$$|\lambda_{\max}\,(\mathbf{I}-\mu\Lambda)| \tag{A.3}$$

Using the fact that $\mathbf{C}^*_{\mathbf{z}_{\mathrm{M}}}$ and $\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}$ have the same eigenvalues, the asymptotic unbiasedness and mean stability of the PU-

ACLMS are guaranteed if

$$\left|1-\mu\lambda_{\max}\left(\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\right)\right|<1 \tag{A.4}$$

This inequality determines the stability bounds for step-size μ as

$$0<\mu<\frac{2}{\lambda_{\max}\left(\mathbf{C}_{\mathbf{z}_{\mathrm{M}}}\right)} \tag{A.5}$$

■

Appendix B
Proof of Proposition 3

Proof: Consider the matrix $\boldsymbol{\mathcal{F}}$ of the form (69) with positive-definite matrix $\mathbf{P}>0$, non-negative definite matrix $\mathbf{Q}\geq 0$, and positive step-size μ. For mean-square stability, we need to determine a bound on μ such that all eigenvalues of $\boldsymbol{\mathcal{F}}$ are strictly inside the unite disc, i.e. $|\lambda(\boldsymbol{\mathcal{F}})|<1$. This condition is satisfied if the step-size parameter μ satisfy

$$\max_{\|\mathbf{x}\|=1}\left\{\mathbf{x}\boldsymbol{\mathcal{F}}(\mu)\,\mathbf{x}^{H}\right\}<1 \tag{B.1}$$

or equivalently, $\mathbf{P}-\mu\mathbf{Q}>0$, and

$$\min\left\{\mathbf{x}\mathbf{H}(\mu)\,\mathbf{x}^{H}\right\}>-1 \tag{B.2}$$

or, equivalently, $\mathbf{H}(\mu)=2\mathbf{I}-\mu\mathbf{P}+\mu^{2}\mathbf{Q}>0$. These inequalities satisfy the stability conditions $\lambda(\boldsymbol{\mathcal{F}})<1$ and $\lambda(\boldsymbol{\mathcal{F}})>-1$, respectively.

In the light of Appendix 25.A of [1], the inequality (B.1) holds if, and only if,

$$\mu<\frac{1}{\lambda_{\max}\left(\mathbf{P}^{-1}\mathbf{Q}\right)} \tag{B.3}$$

Since the eigenvalues of $\mathbf{H}$ vary continuously with step-size parameter μ, an upper-bound on μ, μ_{max}, such that $\mathbf{H}(\mu)>0$ is obtained from the roots of $\det[\mathbf{H}(\mu)]=0$. Employing the block-matrix determinant principle, the determinant of $\mathbf{H}(\mu)$ is equal to the determinant of the block-matrix

$$\mathbf{K}(\mu)=\begin{bmatrix}2\mathbf{I}-\mu\mathbf{P} & \mu\mathbf{Q}\\ -\mu\mathbf{I} & \mathbf{I}\end{bmatrix}, \tag{B.4}$$

Moreover, since

$$\mathbf{K}(\mu)=\begin{bmatrix}2\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\left(\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}-\mu\begin{bmatrix}\mathbf{P}/2 & -\mathbf{Q}/2\\ \mathbf{I} & \mathbf{0}\end{bmatrix}\right) \tag{B.5}$$

the condition $\det[\mathbf{K}(\mu)]=0$ and $\det(\mathbf{I}-\mu\mathbf{G})=0$ are identical, where

$$\mathbf{G}\triangleq\begin{bmatrix}\mathbf{P}/2 & -\mathbf{Q}/2\\ \mathbf{I} & \mathbf{0}\end{bmatrix} \tag{B.6}$$

In order to guarantee $\lambda(\boldsymbol{\mathcal{F}})>-1$, the step-size μ must satisfy the following inequality

$$\mu<\frac{1}{\max\left\{\lambda\left\{\mathbf{G}\right\}\in\mathbb{R}^{>0}\right\}} \tag{B.7}$$

It follows that the aforementioned results can be merged together to yield the following condition

$$0<\mu<\min\left\{\frac{1}{\lambda_{\max}\left(\mathbf{P}^{-1}\mathbf{Q}\right)},\frac{1}{\max\left\{\lambda\left\{\mathbf{G}\right\}\in\mathbb{R}^{>0}\right\}}\right\} \tag{B.8}$$

The above range of step-size μ guarantees the stability of $\boldsymbol{\mathcal{F}}$.

■

References

[1] A. H. Sayed, "Adaptive Filters. Hoboken," 2008.

[2] C. Jahanchahi, S. Kanna, and D. Mandic, "Complex dual channel estimation: Cost effective widely linear adaptive filtering," *Signal Processing*, vol. 104, pp. 33–42, 2014.

[3] Y. Xia and D. P. Mandic, "Widely linear adaptive frequency estimation of unbalanced three-phase power systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 1, pp. 74–83, 2011.

[4] Y. Xia, L. Qiao, Q. Yang, W. Pei, and D. P. Mandic, "Widely linear adaptive frequency estimation for unbalanced three-phase power systems with multiple noisy measurements," in *2017 22nd International Conference on Digital Signal Processing (DSP)*. IEEE, 2017, pp. 1–5.

[5] Y. Xia, S. C. Douglas, and D. P. Mandic, "Adaptive frequency estimation in smart grid applications: Exploiting noncircularity and widely linear adaptive estimators," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 44–54, 2012.

[6] Z. Li, Y. Xia, W. Pei, K. Wang, and D. P. Mandic, "An augmented nonlinear LMS for digital self-interference cancellation in full-duplex direct-conversion transceivers," *IEEE Transactions on Signal Processing*, vol. 66, no. 15, pp. 4065–4078, 2018.

[7] Z. Li, Y. Xia, W. Pei, and D. P. Mandic, "A cost-effective nonlinear self-interference canceller in full-duplex direct-conversion transceivers," *Signal Processing*, vol. 158, pp. 4–14, 2019.

[8] Y. Xia, S. C. Douglas, and D. P. Mandic, "Performance analysis of the deficient length augmented CLMS algorithm for second order noncircular complex signals," *Signal Processing*, vol. 144, pp. 214–225, 2018.

[9] D. P. Mandic, Y. Xia, and S. C. Douglas, "Steady state analysis of the CLMS and augmented CLMS algorithms for noncircular complex signals," in *2010 Conference Record of the Forty Fourth Asilomar Conference on Signals, Systems and Computers*. IEEE, 2010, pp. 1635–1639.

[10] B. Widrow, J. McCool, and M. Ball, "The complex LMS algorithm," *Proceedings of the IEEE*, vol. 63, no. 4, pp. 719–720, 1975.

[11] Y. Wu and J. Ni, "Steady-state mean-square deviation analysis of the zero-attracting CLMS algorithm with circular gaussian input," *IEEE Access*, vol. 7, pp. 52 331–52 338, 2019.

[12] D. P. Mandic and V. S. L. Goh, *Complex valued nonlinear adaptive filters: noncircularity, widely linear and neural models*. John Wiley & Sons, 2009, vol. 59.

[13] T. Adali, P. J. Schreier, and L. L. Scharf, "Complex-valued signal processing: The proper way to deal with impropriety," *IEEE Transactions on Signal Processing*, vol. 59, no. 11, pp. 5101–5125, 2011.

[14] P. Clark, I. Kirsteins, and L. Atlas, "Multiband analysis for colored amplitude-modulated ship noise," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2010, pp. 3970–3973.

[15] S. Zarei, W. H. Gerstacker, J. Aulin, and R. Schober, "I/Q imbalance aware widely-linear receiver for uplink multi-cell massive mimo systems: Design and sum rate analysis," *IEEE Transactions on Wireless Communications*, vol. 15, no. 5, pp. 3393–3408, 2016.

[16] P. J. Schreier and L. L. Scharf, "Second-order analysis of improper complex random vectors and processes," *IEEE Transactions on Signal Processing*, vol. 51, no. 3, pp. 714–725, 2003.

[17] B. Picinbono and P. Chevalier, "Widely linear estimation with complex data," *IEEE Transactions on Signal Processing*, vol. 43, no. 8, pp. 2030–2033, 1995.

[18] B. Picinbono and P. Bondon, "Second-order statistics of complex signals," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 411–420, 1997.

[19] A. Khalili, A. Rastegarnia, and S. Sanei, "Quantized augmented complex least-mean square algorithm: Derivation and performance analysis," *Signal Processing*, vol. 121, pp. 54–59, 2016.

[20] R. Schober, W. H. Gerstacker, and L.-J. Lampe, "Data-aided and blind stochastic gradient algorithms for widely linear mmse mai suppression for ds-cdma," *IEEE Transactions on Signal Processing*, vol. 52, no. 3, pp. 746–756, 2004.

[21] Y. Xia, C. C. Took, and D. P. Mandic, "An augmented affine projection algorithm for the filtering of noncircular complex signals," *Signal Processing*, vol. 90, no. 6, pp. 1788–1799, 2010.

[22] S. C. Douglas, "Widely-linear recursive least-squares algorithm for adaptive beamforming," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 2041–2044.

[23] Y. Xia, S. Javidi, and D. P. Mandic, "A regularised normalised augmented complex least mean square algorithm," in *2010 7th International*

Symposium on Wireless Communication Systems. IEEE, 2010, pp. 355–359.

[24] S. L. Goh and D. P. Mandic, "An augmented extended Kalman filter algorithm for complex-valued recurrent neural networks," *Neural Computation*, vol. 19, no. 4, pp. 1039–1055, 2007.

[25] K. Dogancay, *Partial-update adaptive signal processing: Design Analysis and Implementation*. Academic Press, 2008.

[26] S. C. Douglas, "Adaptive filters employing partial updates," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 44, no. 3, pp. 209–216, 1997.

[27] M. Godavarti and A. O. Hero, "Partial update lms algorithms," *IEEE Transactions on signal processing*, vol. 53, no. 7, pp. 2382–2399, 2005.

[28] T. Aboulnasr and K. Mayyas, "Selective coefficient update of gradient-based adaptive algorithms," in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3. IEEE, 1997, pp. 1929–1932.

[29] S. Douglas, "Analysis and implementation of the max-NLMS adaptive filter," in *Conference Record of The Twenty-Ninth Asilomar Conference on Signals, Systems and Computers*, vol. 1. IEEE, 1996, pp. 659–663.

[30] S. C. Douglas, "A family of normalized LMS algorithms," *IEEE signal processing letters*, vol. 1, no. 3, pp. 49–51, 1994.

[31] S. Werner, M. L. De Campos, and P. S. Diniz, "Partial-update NLMS algorithms with data-selective updating," *IEEE Transactions on Signal Processing*, vol. 52, no. 4, pp. 938–949, 2004.

[32] R. Arablouei, K. Dogancay, S. Werner, and Y.-F. Huang, "Adaptive distributed estimation based on recursive least-squares and partial diffusion," *Signal Processing, IEEE Transactions on*, vol. 62, no. 14, pp. 3510–3522, 2014.

[33] R. Arablouei, S. Werner, Y.-F. Huang, and K. Dogancay, "Distributed least mean-square estimation with partial diffusion," *Signal Processing, IEEE Transactions on*, vol. 62, no. 2, pp. 472–484, 2014.

[34] V. Vahidpour, A. Rastegarnia, A. Khalili, W. M. Bazzi, and S. Sanei, "Analysis of partial diffusion LMS for adaptive estimation over networks with noisy links," *IEEE Transactions on Network Science and Engineering*, vol. 5, no. 2, pp. 101–112, 2018.

[35] V. Vahidpour, A. Rastegarnia, A. Khalili, and S. Sanei, "Analysis of partial diffusion recursive least squares adaptation over noisy links," *IET Signal Processing*, vol. 11, no. 6, pp. 749–757, 2017.

[36] ——, "Partial diffusion Kalman filtering for distributed state estimation in multiagent networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

[37] V. Vahidpour, A. Rastegarnia, M. Latifi, A. Khalili, and S. Sanei, "Performance analysis of distributed Kalman filtering with partial diffusion over noisy network," *IEEE Transactions on Aerospace and Electronic Systems*, 2019.

[38] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[39] P. A. Naylor and W. Sherliker, "A short-sort M-Max NLMS partial-update adaptive filter with applications to echo cancellation," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, vol. 5, April 2003, pp. V–373.

[40] K. Doanay and P. A. Naylor, "Recent advances in partial update and sparse adaptive filters," in *2005 13th European Signal Processing Conference*, Sep. 2005, pp. 1–4.