Low-Complexity RLS Algorithms for the Identification of Bilinear Forms

Camelia Elisei-Iliescu, Cristian Stanciu,
Constantin Paleologu, Cristian Anghel, Silviu Ciochină
University Politehnica of Bucharest, Romania
Email: {cristian,pale,canghel,silviu}@comm.pub.ro

Jacob Benesty
INRS-EMT
University of Quebec, Montreal, Canada
Email: benesty@emt.inrs.ca

***Abstract*—The identification of bilinear forms is a challenging problem since its parameter space may be very large and the adaptive filters should be able to cope with this aspect. Recently, the recursive least-squares tailored for bilinear forms (namely RLS-BF) was developed in this context. In order to reduce its computational complexity, two versions based on the dichotomous coordinate descent (DCD) method are proposed in this paper. Simulation results indicate the good performance of these algorithms, with appealing features for practical implementations.**

***Index Terms*—Adaptive filter, recursive least-squares algorithm, dichotomous coordinate descent, bilinear forms, system identification, multiple-input/single-output system.**

I. Introduction

The recursive least-squares (RLS) algorithm [1], [2] represents an appealing choice for system identification problems. Its popularity is mainly related to its fast convergence rate, which outperforms by far the family of least-mean-square (LMS) algorithms. However, the price to pay is a higher computational complexity. Nevertheless, several solutions to reduce the complexity of the RLS algorithm were developed [1]. More recently, the dichotomous coordinate descent (DCD) method proposed by Zakharov and co-authors [3]– [5] proved to be one of the most attractive approaches. Consequently, the resulting RLS-DCD algorithm [4] was successfully applied for system identification problems, e.g., [6], [7].

The system identification problems are more challenging when the parameter space becomes larger [8], [9]. Such frameworks can be found in conjunction with different applications, e.g., [10]– [13]. Most of these approaches are related to the identification of bilinear/trilinear forms, based on tensor decomposition and modelling. In this manner, different problems of high dimension can be reformulated, so that low-dimension techniques are “tensored” together [14].

In this paper, we focus on the identification of bilinear forms; in this context, the bilinear term is defined with respect to the impulse responses of a spatiotemporal model, which resembles a multiple-input/single-output (MISO) system. In [15], an iterative Wiener filter was developed for the identification of such bilinear forms, while [16] provides an analysis of the conventional adaptive algorithms designed for this purpose. The RLS algorithm tailored for the identification of bilinear forms (namely RLS-BF) was also introduced in [16].

Since the complexity of the RLS-BF algorithm could be quite large for practical implementations, we further develop in this paper two low-complexity versions based on the DCD method. Simulation results indicate the good performance of the proposed algorithms, which could represent appealing solutions for bilinear system identification problems.

Camelia Elisei-Iliescu would like to acknowledge the support of ROMATSA. The work of Cristian Stanciu and Cristian Anghel was supported by European Space Agency under Grant no. 4000121222/17/NL/CBi. The work of Constantin Paleologu and Silviu Ciochină was supported by UEFISCDI Romania and MCI through projects PARFAIT, SOLOMON and VIRTUOSE, funded in part by European Unions Horizon 2020 under grant agreements No. 787002 (SAFECARE project) and No. 777996 (SealedGRID project).

II. Bilinear Model and the RLS-BF Algorithm

In this work, the reference signal is defined in the context of a simplified MISO system as

$$d(t) = \mathbf{h}^T\mathbf{X}(t)\mathbf{g} + w(t) = y(t) + w(t), \tag{1}$$

where $\mathbf{h}$ and $\mathbf{g}$ are the two impulse responses of the system of lengths L and M, respectively,

$$\mathbf{X}(t) = \begin{bmatrix} \mathbf{x}_1(t) & \mathbf{x}_2(t) & \cdots & \mathbf{x}_M(t) \end{bmatrix} \tag{2}$$

is the zero-mean multiple-input signal matrix of size $L \times M$, $\mathbf{x}_m(t) = \begin{bmatrix} x_m(t) & x_m(t-1) & \cdots & x_m(t-L+1) \end{bmatrix}^T$ is a vector containing the L most recent time samples of the mth ($m = 1, 2, \ldots, M$) input signal, and $w(t)$ is the zero-mean additive noise. It is assumed that all the signals are real valued, and $\mathbf{X}(t)$ and $w(t)$ are uncorrelated. The output signal $y(t)$ represents the bilinear form (in $\mathbf{h}$ and $\mathbf{g}$).

On the other hand, based on the vectorization operation (i.e., conversion of a matrix into a vector [17]), the matrix $\mathbf{X}(t)$ of size $L \times M$ can be rewritten as a vector of length ML, i.e., $\text{vec}\left[\mathbf{X}(t)\right] = \widetilde{\mathbf{x}}(t)$. Therefore, the output signal $y(t)$ results in

$$\begin{aligned} y(t) &= \text{tr}\left[\left(\mathbf{h}\mathbf{g}^T\right)^T \mathbf{X}(t)\right] = \text{vec}^T\left(\mathbf{h}\mathbf{g}^T\right)\text{vec}\left[\mathbf{X}(t)\right] \\ &= \left(\mathbf{g} \otimes \mathbf{h}\right)^T \widetilde{\mathbf{x}}(t) = \mathbf{f}^T\widetilde{\mathbf{x}}(t), \end{aligned} \tag{3}$$

where $\text{tr}[\cdot]$ denotes the trace of a square matrix, $\otimes$ is the Kronecker product, and $\mathbf{f} = \mathbf{g} \otimes \mathbf{h}$ is the global impulse response of length ML, which is the Kronecker product between the individual impulse responses $\mathbf{g}$ and $\mathbf{h}$. Consequently, the reference signal in (1) becomes

$$d(t) = \mathbf{f}^T\widetilde{\mathbf{x}}(t) + w(t). \quad (4)$$

The goal is to identify the temporal and spatial impulse responses $\mathbf{h}$ and $\mathbf{g}$ with two adaptive filters $\widehat{\mathbf{h}}(t)$ and $\widehat{\mathbf{g}}(t)$ of lengths L and M, respectively. Consequently, the spatiotemporal impulse response $\mathbf{f} = \mathbf{g}\otimes\mathbf{h}$ can be identified with a long filter $\widehat{\mathbf{f}}(t) = \widehat{\mathbf{g}}(t)\otimes\widehat{\mathbf{h}}(t)$ of length ML. It is clear from (1) that $y(t) = \mathbf{h}^T\mathbf{X}(t)\mathbf{g} = (\mathbf{h}/\eta)^T\,\mathbf{X}(t)\,(\eta\mathbf{g})$, where $\eta \neq 0$ is a real-valued number. Hence, the pair $\mathbf{h}/\eta$ and $\eta\mathbf{g}$ is equivalent to the pair $\mathbf{h}$ and $\mathbf{g}$ in the bilinear form. This implies that we can only identify $\widehat{\mathbf{h}}(t)$ and $\widehat{\mathbf{g}}(t)$ up to a scaling factor. On the other hand, since $\mathbf{f} = \mathbf{g}\otimes\mathbf{h} = (\eta\mathbf{g})\otimes(\mathbf{h}/\eta)$, the global impulse response will be identified with no scaling ambiguity. Therefore, to evaluate the identification of the individual impulse responses, we should use the normalized projection misalignment, as defined in [18], while the identification of the global filter should be evaluated with the usual normalized misalignment, which is defined as $\left\|\mathbf{f} - \widehat{\mathbf{f}}(t)\right\|^2 / \left\|\mathbf{f}\right\|^2$, where $\|\cdot\|$ denotes the Euclidean norm.

In this bilinear context, the estimated signal is $\widehat{y}(t) = \widehat{\mathbf{h}}^T(t-1)\mathbf{X}(t)\widehat{\mathbf{g}}(t-1)$, so that the error signal results in

$$\begin{aligned} e(t) &= d(t) - \widehat{y}(t) = d(t) - \widehat{\mathbf{h}}^T(t-1)\mathbf{X}(t)\widehat{\mathbf{g}}(t-1) \\ &= d(t) - \left[\widehat{\mathbf{g}}(t-1)\otimes\widehat{\mathbf{h}}(t-1)\right]^T\widetilde{\mathbf{x}}(t) \\ &= d(t) - \widehat{\mathbf{f}}^T(t-1)\widetilde{\mathbf{x}}(t) \\ &= d(t) - \widehat{\mathbf{h}}^T(t-1)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t) = d(t) - \widehat{\mathbf{g}}^T(t-1)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t), \quad (5)\end{aligned}$$

with the notation $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t) = \left[\widehat{\mathbf{g}}(t-1)\otimes\mathbf{I}_L\right]^T\widetilde{\mathbf{x}}(t)$ and $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t) = \left[\mathbf{I}_M\otimes\widehat{\mathbf{h}}(t-1)\right]^T\widetilde{\mathbf{x}}(t)$, where $\mathbf{I}_L$ and $\mathbf{I}_M$ are the identity matrices of sizes $L\times L$ and $M\times M$, respectively.

Following the least-squares (LS) error criterion [1], we define the cost functions $J_{\widehat{\mathbf{g}}}\left[\widehat{\mathbf{h}}(t)\right] = \sum_{i=1}^{t}\lambda_{\widehat{\mathbf{h}}}^{t-i}\left[d(i) - \widehat{\mathbf{h}}^T(t)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(i)\right]^2$ and $J_{\widehat{\mathbf{h}}}\left[\widehat{\mathbf{g}}(t)\right] = \sum_{i=1}^{t}\lambda_{\widehat{\mathbf{g}}}^{t-i}\left[d(i) - \widehat{\mathbf{g}}^T(t)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(i)\right]^2$, where $\lambda_{\widehat{\mathbf{h}}}$ $(0 \ll \lambda_{\widehat{\mathbf{h}}} < 1)$ and $\lambda_{\widehat{\mathbf{g}}}$ $(0 \ll \lambda_{\widehat{\mathbf{g}}} < 1)$ are the forgetting factors. The minimization of $J_{\widehat{\mathbf{g}}}\left[\widehat{\mathbf{h}}(t)\right]$ and $J_{\widehat{\mathbf{h}}}\left[\widehat{\mathbf{g}}(t)\right]$ with respect to $\widehat{\mathbf{h}}(t)$ and $\widehat{\mathbf{g}}(t)$, respectively, lead to the normal equations [1]:

$$\mathbf{R}_{\widehat{\mathbf{g}}}(t)\widehat{\mathbf{h}}(t) = \mathbf{p}_{\widehat{\mathbf{g}}}(t), \quad (6)$$
$$\mathbf{R}_{\widehat{\mathbf{h}}}(t)\widehat{\mathbf{g}}(t) = \mathbf{p}_{\widehat{\mathbf{h}}}(t), \quad (7)$$

where

$$\mathbf{R}_{\widehat{\mathbf{g}}}(t) = \lambda_{\widehat{\mathbf{h}}}\mathbf{R}_{\widehat{\mathbf{g}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(t), \quad (8)$$
$$\mathbf{p}_{\widehat{\mathbf{g}}}(t) = \lambda_{\widehat{\mathbf{h}}}\mathbf{p}_{\widehat{\mathbf{g}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)d(t), \quad (9)$$
$$\mathbf{R}_{\widehat{\mathbf{h}}}(t) = \lambda_{\widehat{\mathbf{g}}}\mathbf{R}_{\widehat{\mathbf{h}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(t), \quad (10)$$
$$\mathbf{p}_{\widehat{\mathbf{h}}}(t) = \lambda_{\widehat{\mathbf{g}}}\mathbf{p}_{\widehat{\mathbf{h}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)d(t). \quad (11)$$

Using the matrix inversion lemma [1] to update $\mathbf{R}_{\widehat{\mathbf{g}}}^{-1}(t)$ and $\mathbf{R}_{\widehat{\mathbf{h}}}^{-1}(t)$, the RLS algorithm for bilinear forms, namely RLS-BF, was developed in [16]. Simulations in [16] show that the RLS-BF algorithm outperforms its LMS-based counterparts especially in terms of convergence rate. The price to pay is a significant increase in the computational complexity.

III. Low Complexity RLS-BF Algorithms

The RLS-BF algorithm [16] has a computational complexity proportional to $\mathcal{O}(L^2+M^2)$. In order to reduce this computational amount, the approach presented in this section is based on transforming the sequences of the normal equations (6) and (7) into a sequence of auxiliary normal equations [4]. Further, these auxiliary normal equations are solved by using efficient iterative techniques, like the DCD algorithm [3]– [5].

Let us assume that the normal equations from (6) and (7) are approximately solved at time instant $t-1$ and the approximate solutions $\widehat{\mathbf{h}}(t-1)$ and $\widehat{\mathbf{g}}(t-1)$ are available. In this framework, the residual vectors of the solutions can be defined as

$$\mathbf{r}_{\widehat{\mathbf{h}}}(t-1) = \mathbf{p}_{\widehat{\mathbf{g}}}(t-1) - \mathbf{R}_{\widehat{\mathbf{g}}}(t-1)\widehat{\mathbf{h}}(t-1), \quad (12)$$
$$\mathbf{r}_{\widehat{\mathbf{g}}}(t-1) = \mathbf{p}_{\widehat{\mathbf{h}}}(t-1) - \mathbf{R}_{\widehat{\mathbf{h}}}(t-1)\widehat{\mathbf{g}}(t-1). \quad (13)$$

Next, based on the notation $\triangle\widehat{\mathbf{h}}(t) = \widehat{\mathbf{h}}(t) - \widehat{\mathbf{h}}(t-1)$ and $\triangle\widehat{\mathbf{g}}(t) = \widehat{\mathbf{g}}(t) - \widehat{\mathbf{g}}(t-1)$, the normal equations (6)–(7) become

$$\mathbf{R}_{\widehat{\mathbf{g}}}(t)\left[\widehat{\mathbf{h}}(t-1) + \triangle\widehat{\mathbf{h}}(t)\right] = \mathbf{p}_{\widehat{\mathbf{g}}}(t), \quad (14)$$
$$\mathbf{R}_{\widehat{\mathbf{h}}}(t)\left[\widehat{\mathbf{g}}(t-1) + \triangle\widehat{\mathbf{g}}(t)\right] = \mathbf{p}_{\widehat{\mathbf{h}}}(t). \quad (15)$$

Furthermore, using the notation

$$\triangle\mathbf{R}_{\widehat{\mathbf{g}}}(t) = \mathbf{R}_{\widehat{\mathbf{g}}}(t) - \mathbf{R}_{\widehat{\mathbf{g}}}(t-1), \quad (16)$$
$$\triangle\mathbf{p}_{\widehat{\mathbf{g}}}(t) = \mathbf{p}_{\widehat{\mathbf{g}}}(t) - \mathbf{p}_{\widehat{\mathbf{g}}}(t-1), \quad (17)$$
$$\triangle\mathbf{R}_{\widehat{\mathbf{h}}}(t) = \mathbf{R}_{\widehat{\mathbf{h}}}(t) - \mathbf{R}_{\widehat{\mathbf{h}}}(t-1), \quad (18)$$
$$\triangle\mathbf{p}_{\widehat{\mathbf{h}}}(t) = \mathbf{p}_{\widehat{\mathbf{h}}}(t) - \mathbf{p}_{\widehat{\mathbf{h}}}(t-1), \quad (19)$$

and taking (12) and (13) into account, the systems of equations (14)–(15) can be rewritten as

$$\mathbf{R}_{\widehat{\mathbf{g}}}(t)\triangle\widehat{\mathbf{h}}(t) = \widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(t), \quad (20)$$
$$\mathbf{R}_{\widehat{\mathbf{h}}}(t)\triangle\widehat{\mathbf{g}}(t) = \widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(t), \quad (21)$$

where

$$\widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(t) = \mathbf{r}_{\widehat{\mathbf{h}}}(t-1) + \triangle\mathbf{p}_{\widehat{\mathbf{g}}}(t) - \triangle\mathbf{R}_{\widehat{\mathbf{g}}}(t)\widehat{\mathbf{h}}(t-1), \quad (22)$$
$$\widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(t) = \mathbf{r}_{\widehat{\mathbf{g}}}(t-1) + \triangle\mathbf{p}_{\widehat{\mathbf{h}}}(t) - \triangle\mathbf{R}_{\widehat{\mathbf{h}}}(t)\widehat{\mathbf{g}}(t-1). \quad (23)$$

The systems from (20) and (21) represent the auxiliary normal equations, which have to be solved instead of the conventional systems from (6) and (7), respectively.

Based on the notation from (16) and (18), the updates of the correlation matrices become

$$\triangle\mathbf{R}_{\widehat{\mathbf{g}}}(t) = (\lambda_{\widehat{\mathbf{h}}} - 1)\mathbf{R}_{\widehat{\mathbf{g}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(t), \quad (24)$$
$$\triangle\mathbf{R}_{\widehat{\mathbf{h}}}(t) = (\lambda_{\widehat{\mathbf{g}}} - 1)\mathbf{R}_{\widehat{\mathbf{h}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(t), \quad (25)$$

By multiplying (24) to the right with $\widehat{\mathbf{h}}(t-1)$, then taking (12) and (22) into account, and finally using the update (9) together with the error signal, we get

$$\widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(t) = \lambda_{\widehat{\mathbf{h}}}\mathbf{r}_{\widehat{\mathbf{h}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)e(t). \quad (26)$$

Relation (26) is used to evaluate the right-hand term of (20). Similarly, we obtain

$$\widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(t) = \lambda_{\widehat{\mathbf{g}}}\mathbf{r}_{\widehat{\mathbf{g}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)e(t), \qquad (27)$$

which should be used to evaluate the right-hand term of (21).

In the framework of (20)–(21), we need to use an iterative method to obtain $\triangle\widehat{\mathbf{h}}(t)$ and $\triangle\widehat{\mathbf{g}}(t)$. In addition, it would be useful if this method could also provide the residual vectors $\mathbf{r}_{\widehat{\mathbf{h}}}(t)$ and $\mathbf{r}_{\widehat{\mathbf{g}}}(t)$ in a more computationally efficient manner as compared to (12) and (13). To this purpose, the line search methods can be considered [20], [21]. In this context, solving the auxiliary normal equations (20)–(21) is equivalent to minimizing the quadratic functions $\widetilde{J}_{\widehat{\mathbf{g}}}\left[\triangle\widehat{\mathbf{h}}(t)\right] = 0.5\triangle\widehat{\mathbf{h}}^T(t)\mathbf{R}_{\widehat{\mathbf{g}}}(t)\triangle\widehat{\mathbf{h}}(t) - \triangle\widehat{\mathbf{h}}^T(t)\widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(t)$ and $\widetilde{J}_{\widehat{\mathbf{h}}}\left[\triangle\widehat{\mathbf{g}}(t)\right] = 0.5\triangle\widehat{\mathbf{g}}^T(t)\mathbf{R}_{\widehat{\mathbf{h}}}(t)\triangle\widehat{\mathbf{g}}(t) - \triangle\widehat{\mathbf{g}}^T(t)\widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(t)$.

The basic procedure of a line search method can be summarized as follows. At each iteration, the solutions $\triangle\widehat{\mathbf{h}}(t)$ and $\triangle\widehat{\mathbf{g}}(t)$ are updated in the directions $\mathbf{d}_{\mathbf{r}_{\widehat{\mathbf{h}}}}(t)$ and $\mathbf{d}_{\mathbf{r}_{\widehat{\mathbf{h}}}}(t)$, respectively, such as $\mathbf{d}^T_{\mathbf{r}_{\widehat{\mathbf{h}}}}(t)\mathbf{r}_{\widehat{\mathbf{h}}}(t) \neq \mathbf{0}$ and $\mathbf{d}^T_{\mathbf{r}_{\widehat{\mathbf{g}}}}(t)\mathbf{r}_{\widehat{\mathbf{g}}}(t) \neq \mathbf{0}$ [i.e., the solutions are chosen to be non-orthogonal to the residual vectors $\mathbf{r}_{\widehat{\mathbf{h}}}(t)$ and $\mathbf{r}_{\widehat{\mathbf{g}}}(t)$, respectively]. In this case, the functions to be minimized are $\widetilde{J}_{\widehat{\mathbf{g}}}\left[\triangle\widehat{\mathbf{h}}(t) + \alpha\mathbf{d}_{\mathbf{r}_{\widehat{\mathbf{h}}}}(t)\right]$ and $\widetilde{J}_{\widehat{\mathbf{h}}}\left[\triangle\widehat{\mathbf{g}}(t) + \alpha\mathbf{d}_{\mathbf{r}_{\widehat{\mathbf{g}}}}(t)\right]$, where the step size α is selected according to the chosen method.

In this work, the auxiliary normal equations from (20)–(21) are solved by using the DCD algorithm with a leading element [4]. The step-size α takes one of M_{b} predefined values within an amplitude range $[-H, H]$ [3]. If the value of H is properly chosen, the values of the step-size α correspond to the powers of two and are associated with the bits comprising the binary representation of each computed value in the solution vectors $\triangle\widehat{\mathbf{h}}(t)$ and $\triangle\widehat{\mathbf{g}}(t)$. Due to the quantized step-size α (i.e., powers of two), the DCD algorithm does not need multiplications or divisions (these operations are simply replaced by bit-shifts), but only additions; thus, it is well suited for hardware implementation. The method has "unsuccessful" iterations (i.e., without updates of the solution vectors) and "successful" iterations, when $\triangle\widehat{\mathbf{h}}(t)$ and $\triangle\widehat{\mathbf{g}}(t)$ are updated, and the residual vectors are adjusted to reflect the changes in the solution vectors. The maximum number of allowed updates or "successful" iterations, N_{u}, has to be a priori selected. In practice, the value of N_{u} is much smaller as compared to the length of the filter.

In the bilinear context, the arithmetic complexity of the DCD algorithm is proportional to $N_{\text{u}}(L+M)$ additions. Consequently, the complexity associated to the matrix inversion is greatly reduced as compared to the RLS-BF algorithm, which is based on the matrix inversion lemma and needs $\mathcal{O}(L^2+M^2)$ operations. Due to the lack of space, we do not further detail the DCD algorithm; implementation aspects are given in [5].

The resulting RLS-DCD algorithm for bilinear forms, namely RLS-DCD-BF (basic version), is summarized in Table I. In terms of computational complexity, an expensive step of the algorithm is related to the evaluation of the correlation matrices, i.e., $\mathbf{R}_{\widehat{\mathbf{g}}}(t)$ and $\mathbf{R}_{\widehat{\mathbf{h}}}(t)$ in step 1. In case of the conventional RLS-DCD algorithm [4], the input data vector has the time-shift property, which further allows to reduce the complexity of this step (by computing only the first column of the correlation matrix). In the bilinear context, the vectors $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)$ and $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)$ do not have such a property, which brings additional challenges.

TABLE I
RLS-DCD-BF ALGORITHM (BASIC VERSION).

```
Initialization:
$\widehat{\mathbf{h}}(0) = [\ 1\ \ 0\ \ \cdots\ \ 0\ ]^T,\ \widehat{\mathbf{g}}(0) = \frac{1}{M}[\ 1\ \ 1\ \ \cdots\ \ 1\ ]^T$
$\mathbf{R}_{\widehat{\mathbf{g}}}(0) = \delta\mathbf{I}_L,\ \mathbf{R}_{\widehat{\mathbf{h}}}(0) = \delta\mathbf{I}_M,\ \ \delta > 0$ (regularization constant)
$\mathbf{r}_{\widehat{\mathbf{h}}}(0) = \mathbf{0}_{L\times 1},\ \mathbf{r}_{\widehat{\mathbf{g}}}(0) = \mathbf{0}_{M\times 1}$
For $t = 1, 2, \ldots$
Step 1:  $\mathbf{R}_{\widehat{\mathbf{g}}}(t) = \lambda_{\widehat{\mathbf{h}}}\mathbf{R}_{\widehat{\mathbf{g}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)\widetilde{\mathbf{x}}^T_{\widehat{\mathbf{g}}}(t)$
         $\mathbf{R}_{\widehat{\mathbf{h}}}(t) = \lambda_{\widehat{\mathbf{g}}}\mathbf{R}_{\widehat{\mathbf{h}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)\widetilde{\mathbf{x}}^T_{\widehat{\mathbf{h}}}(t)$
Step 2:  $e(t) = d(t) - \widetilde{\mathbf{x}}^T_{\widehat{\mathbf{g}}}(t)\widehat{\mathbf{h}}(t-1) = d(t) - \widetilde{\mathbf{x}}^T_{\widehat{\mathbf{h}}}(t)\widehat{\mathbf{g}}(t-1)$
Step 3:  $\widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(t) = \lambda_{\widehat{\mathbf{h}}}\mathbf{r}_{\widehat{\mathbf{h}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)e(t)$
         $\widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(t) = \lambda_{\widehat{\mathbf{g}}}\mathbf{r}_{\widehat{\mathbf{g}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)e(t)$
Step 4:  $\mathbf{R}_{\widehat{\mathbf{g}}}(t)\triangle\widehat{\mathbf{h}}(t) = \widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(t)\ \Rightarrow\ \triangle\widehat{\mathbf{h}}(t),\ \mathbf{r}_{\widehat{\mathbf{h}}}(t)$
         $\mathbf{R}_{\widehat{\mathbf{h}}}(t)\triangle\widehat{\mathbf{g}}(t) = \widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(t)\ \Rightarrow\ \triangle\widehat{\mathbf{g}}(t),\ \mathbf{r}_{\widehat{\mathbf{g}}}(t)$
         (to be solved with DCD iterations)
Step 5:  $\widehat{\mathbf{h}}(t) = \widehat{\mathbf{h}}(t-1) + \triangle\widehat{\mathbf{h}}(t)$
         $\widehat{\mathbf{g}}(t) = \widehat{\mathbf{g}}(t-1) + \triangle\widehat{\mathbf{g}}(t)$
```

Let us first examine the structure of the vector $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)$. This can be expressed as

$$\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t) = \left[\widehat{\mathbf{g}}(t-1)\otimes\mathbf{I}_L\right]^T\widetilde{\mathbf{x}}(t) = \sum_{m=1}^{M}\widehat{g}_m(t-1)\mathbf{x}_m(t)$$
$$= \left[\ \textstyle\sum_{m=1}^{M}\widehat{g}_m(t-1)x_m(t)\ \ \cdots\ \ \sum_{m=1}^{M}\widehat{g}_m(t-1)x_m(t-L+1)\ \right]^T,$$

where $\widehat{\mathbf{g}}(t) = \left[\ \widehat{g}_1(t)\ \ \widehat{g}_2(t)\ \ \cdots\ \ \widehat{g}_M(t)\ \right]^T$ and the input signals are defined in (2). In the steady-state, when $\widehat{\mathbf{g}}(t) \approx \widehat{\mathbf{g}}(t-1)$, we can see that $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)$ owns (to some extent) the time-shift property. Consequently, since the matrix $\mathbf{R}_{\widehat{\mathbf{g}}}(t)$ is symmetric, only its first column could be computed, i.e.,

$$\mathbf{R}^{(1)}_{\widehat{\mathbf{g}}}(t) = \lambda_{\widehat{\mathbf{h}}}\mathbf{R}^{(1)}_{\widehat{\mathbf{g}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)\widetilde{x}^{(1)}_{\widehat{\mathbf{g}}}(t), \qquad (28)$$

where $\widetilde{x}^{(1)}_{\widehat{\mathbf{g}}}(t)$ denotes the first element of the vector $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)$. Moreover, the lower-right $(L-1)\times(L-1)$ block of $\mathbf{R}_{\widehat{\mathbf{g}}}(t)$ can be approximated by the $(L-1)\times(L-1)$ upper-left block of the matrix $\mathbf{R}_{\widehat{\mathbf{g}}}(t-1)$. Using this approach for evaluating $\mathbf{R}_{\widehat{\mathbf{g}}}(t)$ in step 1 of the RLS-DCD-BF algorithm, we can obtain an approximate version, namely RLS-DCD-BF-v1, with lower computational complexity, i.e., $\mathcal{O}(L+M^2)$.

Next, let us examine the vector $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)$, which is

$$\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t) = \left[\mathbf{I}_M\otimes\widehat{\mathbf{h}}(t-1)\right]^T\widetilde{\mathbf{x}}(t)$$
$$= \left[\ \widehat{\mathbf{h}}^T(t-1)\mathbf{x}_1(t)\ \ \cdots\ \ \widehat{\mathbf{h}}^T(t-1)\mathbf{x}_M(t)\ \right]^T.$$

Clearly, the vector $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)$ does not have the time-shift property. However, the matrix $\mathbf{R}_{\widehat{\mathbf{h}}}(t)$ is symmetric. In fact, this matrix represents an estimate of the correlation matrix $\widetilde{\mathbf{R}}_{\widehat{\mathbf{h}}}(t) = E[\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)\widetilde{\mathbf{x}}^T_{\widehat{\mathbf{h}}}(t)]$, where $E[\cdot]$ denotes mathematical expectation. The terms on the main diagonal of this matrix are $E\left[\widehat{\mathbf{h}}^T(t-1)\mathbf{x}_m(t)\mathbf{x}^T_m(t)\widehat{\mathbf{h}}(t-1)\right]$, $m = 1, 2, \ldots, M$. Let us assume that the covariance matrices of the inputs are close to a diagonal one, i.e., $E\left[\mathbf{x}_m(t)\mathbf{x}^T_m(t)\right] \approx \sigma^2_{x_m}\mathbf{I}_L$, $m = 1, 2, \ldots, M$. This is a fairly restrictive assumption on the input signals; however, it has been widely used to simplify the development in different scenarios [1], [22]. Also, let us consider that the input signals are independent and have the same power, i.e., $\sigma^2_{x_m} \approx \sigma^2_x$, $m = 1, 2, \ldots, M$. Consequently,

$$\begin{aligned}
&E\left[\widehat{\mathbf{h}}^T(t-1)\mathbf{x}_m(t)\mathbf{x}^T_m(t)\widehat{\mathbf{h}}(t-1)\right] \\
&= \mathrm{tr}\left\{E\left[\mathbf{x}_m(t)\mathbf{x}^T_m(t)\widehat{\mathbf{h}}(t-1)\widehat{\mathbf{h}}^T(t-1)\right]\right\} \\
&= \mathrm{tr}\left\{E\left[\mathbf{x}_m(t)\mathbf{x}^T_m(t)\right]E\left[\widehat{\mathbf{h}}(t-1)\widehat{\mathbf{h}}^T(t-1)\right]\right\} \\
&\approx \sigma^2_x E\left[\left\|\widehat{\mathbf{h}}(t-1)\right\|^2\right].
\end{aligned}$$

Hence, $\widetilde{\mathbf{R}}_{\widehat{\mathbf{h}}}(t) \approx \sigma^2_x E\left[\left\|\widehat{\mathbf{h}}(t-1)\right\|^2\right]\mathbf{I}_M$. Under these circumstances, we can also assume that $\mathbf{R}_{\widehat{\mathbf{h}}}(t)$ tends to a diagonal matrix, so that it could be efficiently updated similar to (28). In other words, only its first column could be computed, i.e.,

$$\mathbf{R}^{(1)}_{\widehat{\mathbf{h}}}(t) = \lambda_{\widehat{\mathbf{g}}}\mathbf{R}^{(1)}_{\widehat{\mathbf{h}}}(t-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)\widetilde{x}^{(1)}_{\widehat{\mathbf{h}}}(t), \tag{29}$$

where $\widetilde{x}^{(1)}_{\widehat{\mathbf{h}}}(t)$ denotes the first element of the vector $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)$. Also, the lower-right $(M-1)\times(M-1)$ block of $\mathbf{R}_{\widehat{\mathbf{h}}}(t)$ can be approximated by the $(M-1)\times(M-1)$ upper-left block of the matrix $\mathbf{R}_{\widehat{\mathbf{h}}}(t-1)$. Using both (28) and (29) in step 1 of the RLS-DCD-BF algorithm from Table I, we obtain a second approximate version, namely RLS-DCD-BF-v2, with a computational complexity proportional to $\mathcal{O}(L+M)$.

The complexity orders reported before refer to steps 1–5 in Table I. In addition, the computation of $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(t)$ and $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(t)$ requires $2ML$ multiplications and $2ML-(L+M)$ additions; this amount is proportional to the length of the global filter.

IV. Simulation Results

Simulations are performed from a system identification perspective, in the framework of the MISO system described in Section II. The two impulse responses are randomly generated (with Gaussian distribution). The length of the first impulse response $\mathbf{h}$ is $L = 64$, while the length of $\mathbf{g}$ is $M = 8$ (in Figs. 1 and 3) or $M = 2$ (in Fig. 2). In order to evaluate the tracking capabilities of the algorithms, an abrupt change of the system is simulated in the middle of all the experiments (by generating two new random impulse responses). The input signals $x_m(t)$, $m = 1, 2, \ldots, M$ are either AR(1) processes [each one of them is generated by filtering a white Gaussian noise through a first-order system $1/\left(1-0.8z^{-1}\right)$] or speech sequences; the sampling rate is 8 kHz. The additive noise $v(t)$ is white and Gaussian. The signal-to-noise ratio (SNR) is defined as σ^2_y/σ^2_w, where σ^2_y and σ^2_w are the variances of $y(t)$ and $w(t)$, respectively; in our experiments, we set SNR $= 30$ dB. The performance measure is the normalized misalignment (in dB), to evaluate the identification of the global impulse response $\mathbf{f} = \mathbf{g}\otimes\mathbf{h}$ (of length ML).

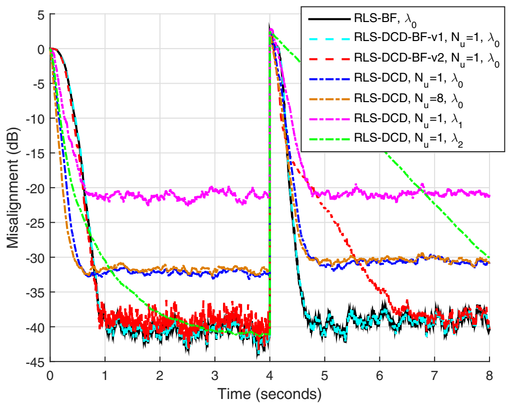

Fig. 1. Normalized misalignment of the RLS-based algorithms; the values of the forgetting factors are $\lambda_{\widehat{\mathbf{h}}} = \lambda_{\widehat{\mathbf{g}}} = \lambda_0 = 1-1/(2ML)$, $\lambda_1 = 1-1/(3L)$, and $\lambda_2 = 1-1/(16ML)$. The input signals are AR(1) processes, SNR $=$ 30 dB, and $ML = 512$.

The algorithms involved in the experiments are the RLS-BF [16], the proposed DCD-based versions (i.e., RLS-DCD-BF-v1 and RLS-DCD-BF-v2), and the regular RLS-DCD algorithm [4] (which can also be used for the identification of the global impulse response $\mathbf{f}$). The RLS-DCD algorithm [4] results based on the input signal $\widetilde{\mathbf{x}}(t)$, the reference signal from (4), and the error signal from the second line of (5). Its derivation is similar to the one presented in [4]; however, since $\widetilde{\mathbf{x}}(t)$ does not have an exact time-shift property, an approximation similar to (28) should be used. In addition, we should note that the solution based on the regular RLS-DCD algorithm involves an adaptive filter of length ML, while the proposed RLS-DCD-BF versions use two shorter filters of lengths L and M, respectively, which is more convenient in practice. Different values of the forgetting factors are used in the experiment; also, the DCD-based algorithms use $M_\mathrm{b} = 16$ and different values of N_u (details are provided in each experiment).

In the first experiment, the input signals are AR(1) processes. As we can notice in Fig. 1, the performance of RLS-BF and RLS-DCD-BF-v1 algorithms are almost identical, while the RLS-DCD-BF-v2 algorithm has a slightly higher misalignment and a slower tracking reaction [due to the approximation in (29)]. The regular RLS-DCD algorithm needs a high value of the forgetting factor to obtain a good tracking reaction, paying with a significant increase of the misalignment. Most important, the DCD-based algorithms obtain good performance even for $N_\mathrm{u} = 1$, which represents a significant gain in terms of computational complexity.

Next, we consider that the input signals are dependent (similar to a Hammerstein model [23]). Here, we choose $M = 2$, i.e., $x_m(t) = x^m(t)$, $m = 1, 2$, where $x(t)$ is an AR(1) process. The results are presented in Fig. 2. As we can

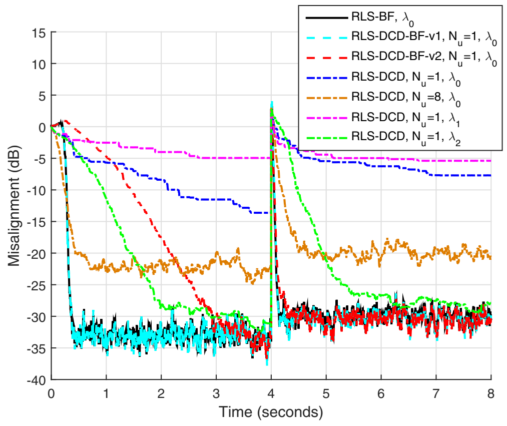

Fig. 2. Normalized misalignment of the RLS-based algorithms; the values of the forgetting factors are $\lambda_{\widehat{\mathbf{h}}} = \lambda_{\widehat{\mathbf{g}}} = \lambda_0 = 1-1/(2ML)$, $\lambda_1 = 1-1/(3L)$, and $\lambda_2 = 1 - 1/(16ML)$. The input signals are $x_m(t) = x^m(t)$, with $m = 1, 2$ [where $x(t)$ is an AR(1) process], SNR = 30 dB, and $ML = 128$.

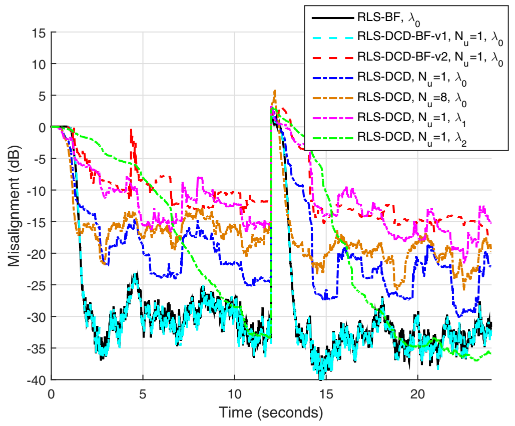

Fig. 3. Normalized misalignment of the RLS-based algorithms; the values of the forgetting factors are $\lambda_{\widehat{\mathbf{h}}} = \lambda_{\widehat{\mathbf{g}}} = \lambda_0 = 1-1/(2ML)$, $\lambda_1 = 1-1/(3L)$, and $\lambda_2 = 1 - 1/(16ML)$. The input signals are speech sequences, SNR = 30 dB, and $ML = 512$.

notice, the RLS-DCD-BF-v2 algorithm has a slower initial convergence rate as compared to the other algorithms, due to the approximation used to evaluate $\mathbf{R}_{\widehat{\mathbf{h}}}(t)$ (which is more suitable for independent input signals); however, it has a good tracking capability. In this scenario, the regular RLS-DCD algorithm needs a high value of the forgetting or a higher value of N_{u} to improve the performance.

Finally, in Fig. 3, the input signals are speech sequences. In this case, the approximation behind the RLS-DCD-BF-v2 is biased, so that its performance is deteriorating. On the other hand, the performance of the RLS-DCD-BF-v1 version is similar to the RLS-BF algorithm, both outperforming the regular RLS-DCD algorithm.

V. Conclusions and Perspectives

In this paper, we have proposed two versions of the RLS-DCD algorithm tailored for the identification of bilinear forms, namely RLS-DCD-BF-v1 and RLS-DCD-BF-v2. Their computational complexity are lower as compared to the RLS-BF algorithm (which is based on the matrix inversion lemma). The performance of the RLS-DCD-BF-v1 is basically identical to the RLS-BF algorithm, while the RLS-DCD-BF-v2 behaves well in most of the scenarios (owning the lowest computational complexity). Future work will focus on variable regularized versions of these algorithms, aiming to improve their robustness in different noisy conditions.

References

[1] S. Haykin, *Adaptive Filter Theory*. Fourth Edition, Upper Saddle River, NJ: Prentice-Hall, 2002.
[2] J. Benesty and Y. Huang, Eds., *Adaptive Signal Processing–Applications to Real-World Problems*. Berlin, Germany: Springer-Verlag, 2003.
[3] Y. V. Zakharov and T. C. Tozer, "Multiplication-free iterative algorithm for LS problem," *IEE Electronics Lett.*, vol. 40, pp. 567–569, Apr. 2004.
[4] Y. V. Zakharov, G. P. White, and J. Liu, "Low-complexity RLS algorithms using dichotomous coordinate descent iterations," *IEEE Trans. Signal Processing*, vol. 56, pp. 3150–3161, July 2008.
[5] J. Liu, Y. V. Zakharov, and B. Weaver, "Architecture and FPGA design of dichotomous coordinate descent algorithms," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 56, pp. 2425–2438, Nov. 2009.
[6] C. Stanciu, J. Benesty, C. Paleologu, T. Gänsler, and S. Ciochină, "A widely linear model for stereophonic acoustic echo cancellation," *Signal Processing*, vol. 93, pp. 511–516, Feb. 2013.
[7] Y. V. Zakharov and V. H. Nascimento, "DCD-RLS adaptive filters with penalties for sparse identification," *IEEE Trans. Signal Processing*, vol. 61, pp. 3198–3213, June 2013.
[8] M. Rupp and S. Schwarz, "A tensor LMS algorithm," in *Proc. IEEE ICASSP*, 2015, pp. 3347–3351.
[9] M. Rupp and S. Schwarz, "Gradient-based approaches to learn tensor products," in *Proc. EUSIPCO*, 2015, pp. 2486–2490.
[10] D. Gesbert and P. Duhamel, "Robust blind joint data/channel estimation based on bilinear optimization," in *Proc. IEEE WSSAP*, 1996, pp. 168–171.
[11] A. Stenger and W. Kellermann, "Adaptation of a memoryless preprocessor for nonlinear acoustic echo cancelling," *Signal Processing*, vol. 80, pp. 1747–1760, Sept. 2000.
[12] L. N. Ribeiro, S. Schwarz, M. Rupp, A. L. F. de Almeida, and J. C. M. Mota, "A low-complexity equalizer for massive MIMO systems based on array separability," in *Proc. EUSIPCO*, 2017, pp. 2522–2526.
[13] M. N. da Costa, G. Favier, and J. M. T. Romano, "Tensor modelling of MIMO communication systems with performance analysis and Kronecker receivers," *Signal Processing*, vol. 145, pp. 304–316, Apr. 2018.
[14] C. F. Van Loan, "The ubiquitous Kronecker product," *J. Computational Applied Mathematics*, vol. 123, pp. 85–100, 2000.
[15] J. Benesty, C. Paleologu, and S. Ciochină, "On the identification of bilinear forms with the Wiener filter," *IEEE Signal Processing Lett.*, vol. 24, pp. 653–657, May 2017.
[16] C. Paleologu, J. Benesty, and S. Ciochină, "Adaptive filtering for the identification of bilinear forms," *Digital Signal Processing*, vol. 75, pp. 153–167, Apr. 2018.
[17] D. A. Harville, *Matrix Algebra From a Statistician's Perspective*. New York: Springer-Verlag, 1997.
[18] D. R. Morgan, J. Benesty, and M. M. Sondhi, "On the evaluation of estimated impulse responses," *IEEE Signal Processing Lett.*, vol. 5, pp. 174–176, July 1998.
[19] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: The Johns Hopkins University Press, 1996.
[20] C. E. Davila, "Line search algorithms for adaptive filtering," *IEEE Trans. Signal Processing*, vol. 41, pp. 2490–2494, July 1993.
[21] J. H. Husoy, "Adaptive filters viewed as iterative linear equation solvers," in *Lecture Notes in Computer Science: Numerical Analysis and Its Applications*. Berlin, Germany: Springer-Verlag, 2005, vol. 3401/2005, pp. 320–327.
[22] A. I. Sulyman and A. Zerguine, "Convergence and steady-state analysis of a variable step-size NLMS algorithm," *Signal Processing*, vol. 83, pp. 1255–1273, June 2003.
[23] E.-W. Bai and D. Li, "Convergence of the iterative Hammerstein system identification algorithm," *IEEE Trans. Automatic Control*, vol. 49, pp. 1929–1940, Nov. 2004.