Variable-Regularized Low-Complexity RLS Adaptive Algorithms for Bilinear Forms

Ionuţ-Dorinel Fîciu, Camelia Elisei-Iliescu, Cristian-Lucian Stanciu, and Constantin Paleologu
University Politehnica of Bucharest, Romania
Email: {cristian, pale}@comm.pub.ro

***Abstract*—This work in progress targets the identification of bilinear forms using fast converging adaptive algorithms. In this framework, the bilinear term is defined with respect to the impulse responses of a spatiotemporal model. Recently, the recursive least-squares algorithm tailored for bilinear forms (RLS-BF) was introduced in this context, together with its low-complexity version based on dichotomous coordinate descent (DCD) iterations, namely RLS-DCD-BF. Aiming to improve the robustness of the RLS-DCD-BF algorithm in noisy conditions, a variable-regularized version is presented in this paper. Simulation results indicate the appealing performance of this solution and motivate our future works on multilinear forms.**

Index Terms*—*adaptive filter; bilinear forms; recursive least-squares; variable regularization; dichotomous coordinate descent

I. Introduction

The recursive least-squares (RLS) is the algorithm of choice in many system identification scenarios [1]. The main reason behind this popularity is its fast converging feature, while the drawback consists of a high computational complexity. However, there are several efficient solutions to reduce the computational amount of the RLS algorithm, like the dichotomous coordinate descent (DCD) method [2].

Nevertheless, the system identification problems become challenging in case of a larger parameter space [3]. Such approaches can be formulated in terms of the identification of multilinear forms, while the solutions are based on multi-dimensional adaptive algorithms. Currently, we focus on the identification of bilinear forms, where the bilinear term is defined with respect to the impulse responses of a spatiotemporal model [4]. Recently, the RLS algorithm tailored for the identification of bilinear forms (RLS-BF) was developed, together with its DCD-based version (RLS-DCD-BF) [5].

This work in progress is focused on two main directions. First, in order to improve the robustness in noisy environments, we present a variable-regularized (VR) version of the RLS-BF algorithm (namely VR-RLS-BF), where the time-dependent regularization parameters are adjusted so that the algorithm can perform well in noisy environments. Second, we develop low-complexity versions of the VR-RLS-BF algorithm based on the DCD method. Simulation results show the good behavior of the VR-based algorithms and open the path toward the generalization of these solutions for multilinear forms.

The paper is organized as follows. Section II introduces the system model and briefly presents the RLS-DCD-BF algorithm [5]. Section III is dedicated to the variable regularized versions of this algorithm. Simulation results are provided in Section IV, while Section V concludes the paper.

II. RLS-DCD-BF Algorithm

In the framework of a real-valued bilinear model [4], the reference signal (at discrete-time index n) is defined as

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

where $\mathbf{h}$ and $\mathbf{g}$ are the two impulse responses of the system (of lengths L and M, respectively), $\mathbf{X}(n) = \begin{bmatrix} \mathbf{x}_1(n) & \mathbf{x}_2(n) & \cdots & \mathbf{x}_M(n) \end{bmatrix}$ is the zero-mean multiple-input signal matrix of size $L \times M$, $\mathbf{x}_m(n) = \begin{bmatrix} x_m(n) & x_m(n-1) & \cdots & x_m(n-L+1) \end{bmatrix}^T$ is a vector containing the L most recent time samples of the mth ($m = 1, 2, \ldots, M$) input signal, $w(n)$ is the zero-mean additive noise [which is uncorrelated with $\mathbf{X}(n)$], and the superscript ^{T} denotes the transpose operator. From (1), we can define the signal-to-noise ratio (SNR) as SNR $= \sigma_y^2/\sigma_w^2$, where $\sigma_y^2 = E\left[y^2(n)\right]$ and $\sigma_w^2 = E\left[w^2(n)\right]$ are the variances of $y(n)$ and $w(n)$, respectively, with $E[\cdot]$ denoting mathematical expectation. The goal is to identify the two impulse responses of the bilinear system (i.e., $\mathbf{h}$ and $\mathbf{g}$) with two adaptive filters, denoted by $\widehat{\mathbf{h}}(n)$ and $\widehat{\mathbf{g}}(n)$.

Following this approach, the estimated signal results in $\widehat{y}(n) = \widehat{\mathbf{h}}^T(n-1)\mathbf{X}(n)\widehat{\mathbf{g}}(n-1)$, while the error signal is

$$\begin{aligned} e(n) &= d(n) - \widehat{y}(n) = d(n) - \widehat{\mathbf{h}}^T(n-1)\mathbf{X}(n)\widehat{\mathbf{g}}(n-1) \\ &= d(n) - \widehat{\mathbf{h}}^T(n-1)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n) = d(n) - \widehat{\mathbf{g}}^T(n-1)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n), \end{aligned} \qquad (2)$$

with the notation $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n) = \left[\widehat{\mathbf{g}}(n-1) \otimes \mathbf{I}_L\right]^T \widetilde{\mathbf{x}}(n)$ and $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n) = \left[\mathbf{I}_M \otimes \widehat{\mathbf{h}}(n-1)\right]^T \widetilde{\mathbf{x}}(n)$, where $\otimes$ is the Kronecker product, $\mathbf{I}_L$ and $\mathbf{I}_M$ are the identity matrices of sizes $L \times L$ and $M \times M$, respectively, and $\widetilde{\mathbf{x}}(n) = \begin{bmatrix} \mathbf{x}_1^T(n) & \mathbf{x}_2^T(n) & \cdots & \mathbf{x}_M^T(n) \end{bmatrix}^T$. Following the least-squares (LS) error criterion, the normal equations are [4]

$$\mathbf{R}_{\widehat{\mathbf{g}}}(n)\widehat{\mathbf{h}}(n) = \mathbf{p}_{\widehat{\mathbf{g}}}(n), \qquad (3)$$

$$\mathbf{R}_{\widehat{\mathbf{h}}}(n)\widehat{\mathbf{g}}(n) = \mathbf{p}_{\widehat{\mathbf{h}}}(n), \qquad (4)$$

where the terms from (3)–(4) can be recursively updated as

$$\mathbf{R}_{\widehat{\mathbf{g}}}(n) = \lambda_{\widehat{\mathbf{h}}}\mathbf{R}_{\widehat{\mathbf{g}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n), \qquad (5)$$

$$\mathbf{p}_{\widehat{\mathbf{g}}}(n) = \lambda_{\widehat{\mathbf{h}}}\mathbf{p}_{\widehat{\mathbf{g}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)d(n), \qquad (6)$$

$$\mathbf{R}_{\widehat{\mathbf{h}}}(n) = \lambda_{\widehat{\mathbf{g}}}\mathbf{R}_{\widehat{\mathbf{h}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n), \qquad (7)$$

$$\mathbf{p}_{\widehat{\mathbf{h}}}(n) = \lambda_{\widehat{\mathbf{g}}}\mathbf{p}_{\widehat{\mathbf{h}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)d(n), \qquad (8)$$

Table I. RLS-DCD-BF algorithm.

```
Initialization:
$\widehat{\mathbf{h}}(0) = \left[\begin{array}{cccc} 1 & 0 & \cdots & 0 \end{array}\right]^T, \ \widehat{\mathbf{g}}(0) = \frac{1}{M}\left[\begin{array}{cccc} 1 & 1 & \cdots & 1 \end{array}\right]^T$
$\mathbf{R}_{\widehat{\mathbf{g}}}(0) = \delta\mathbf{I}_L, \ \mathbf{R}_{\widehat{\mathbf{h}}}(0) = \delta\mathbf{I}_M, \ \mathbf{r}_{\widehat{\mathbf{h}}}(0) = \mathbf{0}_{L\times 1}, \ \mathbf{r}_{\widehat{\mathbf{g}}}(0) = \mathbf{0}_{M\times 1}$
For $n = 1, 2, \ldots$
Step 1:   $\mathbf{R}_{\widehat{\mathbf{g}}}(n) = \lambda_{\widehat{\mathbf{h}}}\mathbf{R}_{\widehat{\mathbf{g}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n)$
          $\mathbf{R}_{\widehat{\mathbf{h}}}(n) = \lambda_{\widehat{\mathbf{g}}}\mathbf{R}_{\widehat{\mathbf{h}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n)$
Step 2:   $e(n) = d(n) - \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n)\widehat{\mathbf{h}}(n-1) = d(n) - \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n)\widehat{\mathbf{g}}(n-1)$
Step 3:   $\widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(n) = \lambda_{\widehat{\mathbf{h}}}\mathbf{r}_{\widehat{\mathbf{h}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)e(n)$
          $\widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(n) = \lambda_{\widehat{\mathbf{g}}}\mathbf{r}_{\widehat{\mathbf{g}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)e(n)$
Step 4:   $\mathbf{R}_{\widehat{\mathbf{g}}}(n)\triangle\widehat{\mathbf{h}}(n) = \widetilde{\mathbf{p}}_{\widehat{\mathbf{g}}}(n) \ \overset{\text{DCD}}{\Longrightarrow} \ \triangle\widehat{\mathbf{h}}(n), \ \mathbf{r}_{\widehat{\mathbf{h}}}(n)$
          $\mathbf{R}_{\widehat{\mathbf{h}}}(n)\triangle\widehat{\mathbf{g}}(n) = \widetilde{\mathbf{p}}_{\widehat{\mathbf{h}}}(n) \ \overset{\text{DCD}}{\Longrightarrow} \ \triangle\widehat{\mathbf{g}}(n), \ \mathbf{r}_{\widehat{\mathbf{g}}}(n)$
Step 5:   $\widehat{\mathbf{h}}(n) = \widehat{\mathbf{h}}(n-1) + \triangle\widehat{\mathbf{h}}(n)$
          $\widehat{\mathbf{g}}(n) = \widehat{\mathbf{g}}(n-1) + \triangle\widehat{\mathbf{g}}(n)$
```

with $\lambda_{\widehat{\mathbf{h}}}$ $(0 \ll \lambda_{\widehat{\mathbf{h}}} < 1)$ and $\lambda_{\widehat{\mathbf{g}}}$ $(0 \ll \lambda_{\widehat{\mathbf{g}}} < 1)$ denoting the forgetting factors. Using the matrix inversion lemma [1] to update $\mathbf{R}_{\widehat{\mathbf{g}}}^{-1}(n)$ and $\mathbf{R}_{\widehat{\mathbf{h}}}^{-1}(n)$, the RLS algorithm for bilinear forms (RLS-BF) was developed in [5]. This algorithm has a computational complexity proportional to $\mathcal{O}(L^2 + M^2)$.

In order to reduce the computational amount of the RLS-BF algorithm, an efficient approach is based on transforming the sequences of the normal equations (3)–(4) into a sequence of auxiliary normal equations. These auxiliary normal equations can be solved by using an efficient iterative technique, like the DCD algorithm [2]. The resulting RLS-DCD algorithm for bilinear forms (RLS-DCD-BF) is resumed in Table I.

The evaluation of the correlation matrices [i.e., $\mathbf{R}_{\widehat{\mathbf{g}}}(n)$ and $\mathbf{R}_{\widehat{\mathbf{h}}}(n)$ in step 1] represents a computationally expensive step of the RLS-DCD-BF algorithm. Exploiting the structure of these matrices, two approximate versions of the RLS-DCD-BF algorithm were proposed in [5]. The first one, namely RLS-DCD-BF-v1, takes into account that $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)$ owns (to some extent) the time-shift property, especially in the steady-state, when $\widehat{\mathbf{g}}(n) \approx \widehat{\mathbf{g}}(n-1)$. Consequently, since the matrix $\mathbf{R}_{\widehat{\mathbf{g}}}(n)$ is symmetric, only its first column could be computed, i.e.,

$$\mathbf{R}_{\widehat{\mathbf{g}}}^{(1)}(n) = \lambda_{\widehat{\mathbf{h}}}\mathbf{R}_{\widehat{\mathbf{g}}}^{(1)}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)\widetilde{x}_{\widehat{\mathbf{g}}}^{(1)}(n), \tag{9}$$

where $\widetilde{x}_{\widehat{\mathbf{g}}}^{(1)}(n)$ denotes the first element of the vector $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)$. Moreover, the lower-right $(L-1)\times(L-1)$ block of $\mathbf{R}_{\widehat{\mathbf{g}}}(n)$ can be approximated by the $(L-1)\times(L-1)$ upper-left block of the matrix $\mathbf{R}_{\widehat{\mathbf{g}}}(n-1)$. Thus, the computational complexity of the RLS-DCD-BF-v1 algorithm is proportional to $\mathcal{O}(L+M^2)$.

Similarly, under some strong assumptions [i.e., i) the covariance matrices of the inputs are close to a diagonal one and ii) the input signals are independent and have the same power], we can also assume that $\mathbf{R}_{\widehat{\mathbf{h}}}(n)$ tends to a diagonal matrix, so that it could be efficiently updated similar to (9). Hence, a second version of RLS-DCD-BF algorithm results (namely RLS-DCD-BF-v2), with a computational complexity proportional to $\mathcal{O}(L + M)$.

III. Variable Regularized RLS-BF Algorithms

The regularization process is essential in practice, taking into account the presence of additive noise [6]. Here, we consider the regularized RLS-BF algorithm, with the updates:

$$\widehat{\mathbf{h}}(n) = \widehat{\mathbf{h}}(n-1) + \left[\mathbf{R}_{\widehat{\mathbf{g}}}(n) + \delta_{\widehat{\mathbf{h}}}\mathbf{I}_L\right]^{-1}\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)e(n), \tag{10}$$

$$\widehat{\mathbf{g}}(n) = \widehat{\mathbf{g}}(n-1) + \left[\mathbf{R}_{\widehat{\mathbf{h}}}(n) + \delta_{\widehat{\mathbf{g}}}\mathbf{I}_M\right]^{-1}\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)e(n), \tag{11}$$

where $\delta_{\widehat{\mathbf{h}}}$ and $\delta_{\widehat{\mathbf{g}}}$ are the regularization terms. Alternatively,

$$\widehat{\mathbf{h}}(n) = \mathbf{P}_{\widehat{\mathbf{g}}}(n)\widehat{\mathbf{h}}(n-1) + \widetilde{\mathbf{h}}(n), \tag{12}$$

$$\widehat{\mathbf{g}}(n) = \mathbf{P}_{\widehat{\mathbf{h}}}(n)\widehat{\mathbf{g}}(n-1) + \widetilde{\mathbf{g}}(n), \tag{13}$$

where $\mathbf{P}_{\widehat{\mathbf{g}}}(n) = \mathbf{I}_L - \left[\mathbf{R}_{\widehat{\mathbf{g}}}(n) + \delta_{\widehat{\mathbf{h}}}\mathbf{I}_L\right]^{-1}\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n)$, $\mathbf{P}_{\widehat{\mathbf{h}}}(n) = \mathbf{I}_M - \left[\mathbf{R}_{\widehat{\mathbf{h}}}(n) + \delta_{\widehat{\mathbf{g}}}\mathbf{I}_M\right]^{-1}\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n)$, and

$$\widetilde{\mathbf{h}}(n) = \left[\mathbf{R}_{\widehat{\mathbf{g}}}(n) + \delta_{\widehat{\mathbf{h}}}\mathbf{I}_L\right]^{-1}\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)d(n), \tag{14}$$

$$\widetilde{\mathbf{g}}(n) = \left[\mathbf{R}_{\widehat{\mathbf{h}}}(n) + \delta_{\widehat{\mathbf{g}}}\mathbf{I}_M\right]^{-1}\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)d(n). \tag{15}$$

The vectors $\widetilde{\mathbf{h}}(n)$ and $\widetilde{\mathbf{g}}(n)$ are the correction components of the algorithm, due to the dependency on $d(n)$. Let us define

$$\widetilde{e}_{\widehat{\mathbf{g}}}(n) = d(n) - \widetilde{\mathbf{h}}^T(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n), \tag{16}$$

$$\widetilde{e}_{\widehat{\mathbf{h}}}(n) = d(n) - \widetilde{\mathbf{g}}^T(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n), \tag{17}$$

the error signals between the desired signal and the estimated signals obtained from the filters optimized in (14)–(15). In the context of system identification, the purpose is to recover the noise signal from the error of the adaptive filter [6]. As a consequence, we could find $\delta_{\widehat{\mathbf{h}}}$ and $\delta_{\widehat{\mathbf{g}}}$ in such a way that

$$E\left[\widetilde{e}_{\widehat{\mathbf{g}}}^2(n)\right] = E\left[\widetilde{e}_{\widehat{\mathbf{h}}}^2(n)\right] = \sigma_w^2. \tag{18}$$

At this point, let us assume that the covariance matrices of the inputs are close to a diagonal one, i.e., $E\left[\mathbf{x}_m(n)\mathbf{x}_m^T(n)\right] \approx \sigma_{x_m}^2\mathbf{I}_L, \ m = 1, 2, \ldots, M$; also, let us consider that the input signals are independent and have the same power, i.e., $\sigma_{x_m}^2 \approx \sigma_x^2, \ m = 1, 2, \ldots, M$. In this context, we can show that $\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n) \approx M\sigma_x^2\left\|\widehat{\mathbf{h}}(n-1)\right\|^2$, where $\|\cdot\|$ denotes the Euclidean norm. Also, $E\left[\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n)\right] \approx \sigma_x^2\left\|\widehat{\mathbf{h}}(n-1)\right\|^2\mathbf{I}_M$. Similarly, we obtain $\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n) \approx L\sigma_x^2\left\|\widehat{\mathbf{g}}(n-1)\right\|^2$ and $E\left[\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n)\right] \approx \sigma_x^2\left\|\widehat{\mathbf{g}}(n-1)\right\|^2\mathbf{I}_L$. Therefore, for $\lambda_{\widehat{\mathbf{h}}} \approx 1 - 1/L$ and $\lambda_{\widehat{\mathbf{g}}} \approx 1 - 1/M$, we obtain (for n large enough):

$$\mathbf{R}_{\widehat{\mathbf{g}}}(n) + \delta_{\widehat{\mathbf{h}}}\mathbf{I}_L \approx \left[L\sigma_x^2\left\|\widehat{\mathbf{g}}(n-1)\right\|^2 + \delta_{\widehat{\mathbf{h}}}\right]\mathbf{I}_L, \tag{19}$$

$$\mathbf{R}_{\widehat{\mathbf{h}}}(n) + \delta_{\widehat{\mathbf{g}}}\mathbf{I}_M \approx \left[M\sigma_x^2\left\|\widehat{\mathbf{h}}(n-1)\right\|^2 + \delta_{\widehat{\mathbf{g}}}\right]\mathbf{I}_M. \tag{20}$$

Next, squaring and taking the expectations on both sides of (16)–(17), then developing (18) based on (19)–(20), we obtain two quadratic equations with the following solutions:

$$\delta_{\widehat{\mathbf{h}}} = \frac{LE\left[\|\widehat{\mathbf{g}}(n-1)\|^2\right]\left(1+\sqrt{1+\text{SNR}}\right)}{\text{SNR}}\sigma_x^2, \tag{21}$$

$$\delta_{\widehat{\mathbf{g}}} = \frac{ME\left[\left\|\widehat{\mathbf{h}}(n-1)\right\|^2\right]\left(1+\sqrt{1+\text{SNR}}\right)}{\text{SNR}}\sigma_x^2. \tag{22}$$

The most problematic term in (21)–(22) is the SNR, taking into account that the additive noise could be nonstationary in real-world applications. However, in (1), the output signal $y(n)$ can be considered uncorrelated with the noise $w(n)$. Therefore, (1) can be expressed in terms of power estimates as $\sigma_d^2 = \sigma_y^2 + \sigma_w^2$, where $\sigma_d^2 = E\left[d^2(n)\right]$ is the variance of the reference signal. Consequently, $\sigma_w^2 = \sigma_d^2 - \sigma_y^2$. At this point, let us assume that the adaptive filters have converged to a certain degree, so that we can use the approximation $\sigma_y^2 \approx \sigma_{\widehat{y}}^2$, where $\sigma_{\widehat{y}}^2 = E\left[\widehat{y}^2(n)\right]$ is the variance of the estimated signal $\widehat{y}(n)$. Using this approximation, it results that $\sigma_w^2 \approx \sigma_d^2 - \sigma_{\widehat{y}}^2$. The power estimates can be recursively estimated as $\widehat{\sigma}_d^2(n) = \gamma\widehat{\sigma}_d^2(n-1) + (1-\gamma)d^2(n)$ and $\widehat{\sigma}_{\widehat{y}}^2(n) = \gamma\widehat{\sigma}_{\widehat{y}}^2(n-1) + (1-\gamma)\widehat{y}^2(n)$, where $0 \ll \gamma < 1$. Therefore, an estimate of the SNR results in $\widehat{\text{SNR}}(n) = \widehat{\sigma}_{\widehat{y}}^2(n)/|\widehat{\sigma}_d^2(n) - \widehat{\sigma}_{\widehat{y}}^2(n)|$. Next, since $\widehat{\mathbf{h}}(n-1)$ and $\widehat{\mathbf{g}}(n-1)$ are available at time index n, the expectation operator can be omitted in (21)–(22). Thus, using the notation $s(n) = \left[1+\sqrt{1+\widehat{\text{SNR}}(n)}\right]/\widehat{\text{SNR}}(n)$, the regularization terms from (21)–(22) result in

$$\delta_{\widehat{\mathbf{h}}}(n) = L\,\|\widehat{\mathbf{g}}(n-1)\|^2\,s(n)\sigma_x^2, \tag{23}$$

$$\delta_{\widehat{\mathbf{g}}}(n) = M\left\|\widehat{\mathbf{h}}(n-1)\right\|^2 s(n)\sigma_x^2. \tag{24}$$

Consequently, we obtain a variable-regularized RLS-BF (VR-RLS-BF) algorithm, with the updates similar to (10)–(11), but using the variable regularization parameters from (23)–(24).

Furthermore, the problem can be interpreted in terms of solving the normal equations [1]:

$$\underline{\mathbf{R}}_{\widehat{\mathbf{g}}}(n)\widehat{\mathbf{h}}(n) = \mathbf{p}_{\widehat{\mathbf{g}}}(n), \tag{25}$$

$$\underline{\mathbf{R}}_{\widehat{\mathbf{h}}}(n)\widehat{\mathbf{g}}(n) = \mathbf{p}_{\widehat{\mathbf{h}}}(n), \tag{26}$$

where $\underline{\mathbf{R}}_{\widehat{\mathbf{g}}}(n) = \widehat{\mathbf{R}}_{\widehat{\mathbf{g}}}(n) + \delta_{\widehat{\mathbf{h}}}(n)\mathbf{I}_L$, $\underline{\mathbf{R}}_{\widehat{\mathbf{h}}}(n) = \widehat{\mathbf{R}}_{\widehat{\mathbf{h}}}(n) + \delta_{\widehat{\mathbf{g}}}(n)\mathbf{I}_M$, and $\mathbf{p}_{\widehat{\mathbf{g}}}(n)$ and $\mathbf{p}_{\widehat{\mathbf{h}}}(n)$ are given in (6) and (8), respectively. The normal equations (25)–(26) can also be recursively solved using the DCD method [2]. Therefore, following a similar approach (as presented in Section II), it results a low-complexity version of the VR-RLS-BF algorithm based on the DCD iterations, namely VR-RLS-DCD-BF; this algorithm is summarized in Table II. Also, the computational complexity in step 1 of the VR-RLS-DCD-BF algorithm can be reduced based on (9). Similarly, two versions of the VR-RLS-DCD-BF algorithm can be obtained, which are also indexed as v1 and v2. The first one uses (7) and (9) in step 1 [i.e., an amount of $\mathcal{O}(L+M^2)$ operations], while the second one further reduces the computational amount up to $\mathcal{O}(L+M)$, similar to the RLS-DCD-BF-v2 algorithm.

Table II. VR-RLS-DCD-BF algorithm.

```
Initialization:
$\widehat{\mathbf{h}}(0) = [\ 1 \quad 0 \quad \cdots \quad 0\ ]^T,\ \widehat{\mathbf{g}}(0) = \frac{1}{M}[\ 1 \quad 1 \quad \cdots \quad 1\ ]^T$
$\mathbf{R}_{\widehat{\mathbf{g}}}(0) = \mathbf{0}_{L\times L},\ \mathbf{R}_{\widehat{\mathbf{h}}}(0) = \mathbf{0}_{M\times M}$
$\underline{\mathbf{r}}_{\widehat{\mathbf{h}}}(0) = \mathbf{0}_{L\times 1},\ \underline{\mathbf{r}}_{\widehat{\mathbf{g}}}(0) = \mathbf{0}_{M\times 1}$
For $n = 1, 2, \ldots$
Step 1:  $\mathbf{R}_{\widehat{\mathbf{g}}}(n) = \lambda_{\widehat{\mathbf{h}}}\mathbf{R}_{\widehat{\mathbf{g}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n)$
         $\mathbf{R}_{\widehat{\mathbf{h}}}(n) = \lambda_{\widehat{\mathbf{g}}}\mathbf{R}_{\widehat{\mathbf{h}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)\widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n)$
Step 2:  Compute $\delta_{\widehat{\mathbf{h}}}(n)$ and $\delta_{\widehat{\mathbf{g}}}(n)$ using (23)–(24)
Step 3:  $\underline{\mathbf{R}}_{\widehat{\mathbf{g}}}(n) = \mathbf{R}_{\widehat{\mathbf{g}}}(n) + \delta_{\widehat{\mathbf{h}}}(n)\mathbf{I}_L$
         $\underline{\mathbf{R}}_{\widehat{\mathbf{h}}}(n) = \mathbf{R}_{\widehat{\mathbf{h}}}(n) + \delta_{\widehat{\mathbf{g}}}(n)\mathbf{I}_M$
Step 4:  $e(n) = d(n) - \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}^T(n)\widehat{\mathbf{h}}(n-1) = d(n) - \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}^T(n)\widehat{\mathbf{g}}(n-1)$
Step 5:  $\underline{\widetilde{\mathbf{p}}}_{\widehat{\mathbf{g}}}(n) = \lambda_{\widehat{\mathbf{h}}}\underline{\mathbf{r}}_{\widehat{\mathbf{h}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{g}}}(n)e(n)$
         $\underline{\widetilde{\mathbf{p}}}_{\widehat{\mathbf{h}}}(n) = \lambda_{\widehat{\mathbf{g}}}\underline{\mathbf{r}}_{\widehat{\mathbf{g}}}(n-1) + \widetilde{\mathbf{x}}_{\widehat{\mathbf{h}}}(n)e(n)$
Step 6:  $\underline{\mathbf{R}}_{\widehat{\mathbf{g}}}(n)\triangle\widehat{\mathbf{h}}(n) = \underline{\widetilde{\mathbf{p}}}_{\widehat{\mathbf{g}}}(n) \overset{\text{DCD}}{\Longrightarrow} \triangle\widehat{\mathbf{h}}(n),\ \underline{\mathbf{r}}_{\widehat{\mathbf{h}}}(n)$
         $\underline{\mathbf{R}}_{\widehat{\mathbf{h}}}(n)\triangle\widehat{\mathbf{g}}(n) = \underline{\widetilde{\mathbf{p}}}_{\widehat{\mathbf{h}}}(n) \overset{\text{DCD}}{\Longrightarrow} \triangle\widehat{\mathbf{g}}(n),\ \underline{\mathbf{r}}_{\widehat{\mathbf{g}}}(n)$
Step 7:  $\widehat{\mathbf{h}}(n) = \widehat{\mathbf{h}}(n-1) + \triangle\widehat{\mathbf{h}}(n)$
         $\widehat{\mathbf{g}}(n) = \widehat{\mathbf{g}}(n-1) + \triangle\widehat{\mathbf{g}}(n)$
```

IV. SIMULATION RESULTS

Simulations are performed in the framework of the bilinear model described in Section II, in the context of a system identification scenario. The two impulse responses of the spatiotemporal system ($\mathbf{h}$ and $\mathbf{g}$) are randomly generated, with Gaussian distribution; the lengths are set to $L = 64$ and $M = 8$. An abrupt change of the system is simulated in the first two experiments, by generating two new random impulse responses in the middle of simulations, in order to evaluate the tracking capabilities of the algorithms. In the experiments, we compare the performance of the RLS-DCD-BF, VR-RLS-BF, and VR-RLS-DCD-BF algorithms. The same values of the forgetting factors are used for all the algorithms, by setting $\lambda_{\widehat{\mathbf{h}}} = \lambda_{\widehat{\mathbf{g}}} = 1 - 1/(2ML)$. As a performance measure, the normalized projection misalignment (NPM) [7] is used to evaluate the identification of the individual impulse responses.

In the first experiment (Figure 1), we compare the performance of the VR-based algorithms in moderate SNR conditions, using SNR $= 10$ dB. The input signals are AR(1) processes, where each one of them is generated by filtering a white Gaussian noise through a first-order system with the transfer function $1/\left(1-0.8z^{-1}\right)$. All the DCD-based algorithms use only one "successful" DCD iteration [2], which is a significant gain in terms of computational complexity. As we can notice in Figure 1(a), both VR-RLS-DCD versions (v1 and v2) perform very similar to the VR-RLS-BF algorithm. However, it can be noticed a slower tracking reaction of the VR-RLS-DCD-v2 [Figure 1(b)], due to the approximation related to the matrix $\mathbf{R}_{\widehat{\mathbf{h}}}(n)$, which is related to the spatial filter $\widehat{\mathbf{g}}(n)$. On the other hand, since the temporal filter $\widehat{\mathbf{h}}(n)$

Copyright (c) IARIA, 2021. ISBN: 978-1-61208-848-8

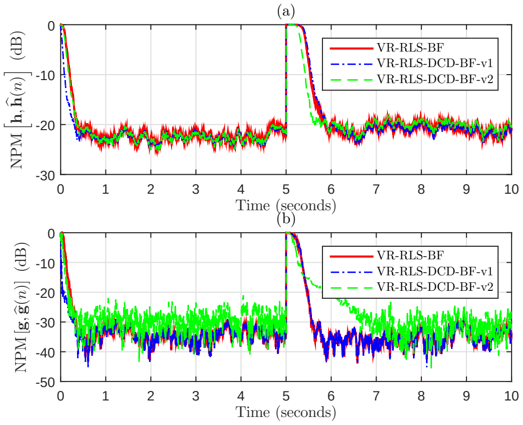

Figure 1. Comparison of the VR-based algorithms in terms of (a) NPM $\left[\mathbf{h}, \widehat{\mathbf{h}}(n)\right]$ and (b) NPM $[\mathbf{g}, \widehat{\mathbf{g}}(n)]$. The system changes after 5 seconds. The input signals are AR(1) processes and SNR $= 10$ dB.

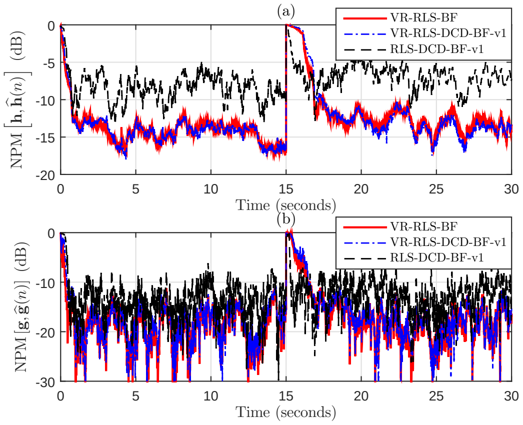

Figure 2. Comparison of the VR-based algorithms in terms of (a) NPM $\left[\mathbf{h}, \widehat{\mathbf{h}}(n)\right]$ and (b) NPM $[\mathbf{g}, \widehat{\mathbf{g}}(n)]$. The system changes after 15 seconds. The input signals are speech sequences and SNR $= 0$ dB.

is not directly influence by this approximation, it behaves well in case of the VR-RLS-DCD-BF-v2 algorithm [Figure 1(a)].

In the following, we focus on the performance of the VR-RLS-DCD-BF algorithm, which is compared to the RLS-DCD-BF algorithm (i.e., the "non-regularized" version). The input signals are speech sequences. Thus, we do not include in comparisons the v2 versions, since the assumption related to the matrix $\mathbf{R}_{\widehat{\mathbf{h}}}(n)$ is biased for nonstationary inputs.

In Figure 2, we consider low SNR conditions, by setting SNR $= 0$ dB. It can be noticed that the VR-based algorithms perform very similar. Nevertheless, the RLS-DCD-BF-v1 algorithm achieves a slightly better tracking reaction, while reaching a higher misalignment level.

Finally, in Figure 3, we assess the performance of the algorithms in case of SNR variations. In the first 12 seconds, we set SNR $= 0$ dB; then, the SNR drops to -25 dB for the next 6 seconds. Nevertheless, the VR-based algorithms are very robust in this case. Also, the VR-RLS-BF and VR-RLS-DCD-BF-v1 perform very similar. As expected, the RLS-DCD-BF-v1 algorithm is affected by the noise variation.

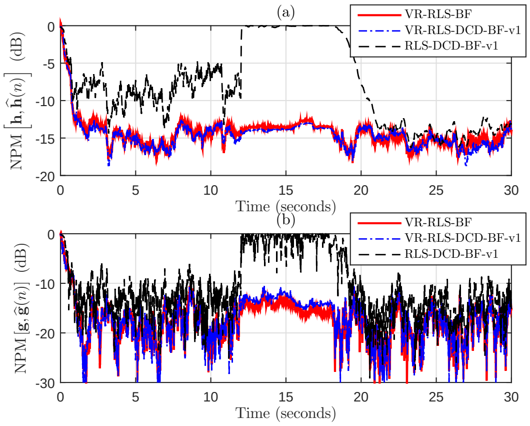

Figure 3. Comparison of the VR-RLS-BF, VR-RLS-DCD-BF-v1, and RLS-DCD-v1 algorithms in terms of (a) NPM $\left[\mathbf{h}, \widehat{\mathbf{h}}(n)\right]$ and (b) NPM $[\mathbf{g}, \widehat{\mathbf{g}}(n)]$. The SNR decreases from 0 dB to -25 dB between times 12 and 18 seconds.

V. Conclusions and Future Works

In this work in progress, we have focused on the regularization terms of the RLS algorithm for bilinear forms. First, a method for finding the optimal regularization parameters (depending on the SNR) was presented. Second, using a proper evaluation of the SNR, a variable-regularized algorithm was developed, together with two low-complexity versions based on the DCD method. Simulations have shown that the VR-based algorithms outperform their non-regularized counterpart, mainly in terms of robustness against SNR variations.

Future works will focus on the extension of these solutions in case of multilinear forms, by exploiting tensor-based adaptive algorithms. In this context, the decomposition methods can be combined with low-rank approximations, aiming the identification of more general forms of impulse responses.

Acknowledgement

This work was supported by a grant of the Romanian Ministry of Education and Research, CNCS – UEFISCDI, project number: PN-III-P1-1.1-TE-2019-0529, within PNCDI III.

References

[1] A. H. Sayed, *Adaptive Filters*. New York, NY: Wiley, 2008.

[2] 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.

[3] M. Rupp and S. Schwarz, "A tensor LMS algorithm," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015, pp. 3347–3351.

[4] 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.

[5] C. Elisei-Iliescu et al., "Efficient recursive least-squares algorithms for the identification of bilinear forms," *Digital Signal Processing*, vol. 83, pp. 280–296, Dec. 2018.

[6] J. Benesty, C. Paleologu, and S. Ciochină, "Regularization of the RLS algorithm," *IEICE Trans. Fundamentals*, vol. E94-A, pp. 1628–1629, Aug. 2011.

[7] 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.