IMPROVING MULTIKERNEL ADAPTIVE FILTERING WITH SELECTIVE BIAS

*Magno T. M. Silva, Renato Candido**

Universidade de São Paulo
{magno, renatocan}@lps.usp.br

Jerónimo Arenas-García, Luis A. Azpicueta-Ruiz†

Univ. Carlos III de Madrid, Spain
{jarenas, azpicueta}@tsc.uc3m.es

ABSTRACT

In this paper, we propose a scheme to simplify the selection of kernel adaptive filters in a multikernel structure. By multiplying the output of each kernel filter by an adaptive biasing factor between zero and one, the degrading effects of poorly adjusted kernel filters can be minimized, increasing the robustness of the multikernel scheme. This approach is able to deal with the lack of the necessary statistical information for an optimal adjustment of the filter and its structure. The advantages of the proposed scheme with respect to other multikernel solutions are checked by means of numerical examples in the context of signal prediction and system identification.

***Index Terms*—** Nonlinear adaptive signal processing, kernel adaptive filtering, cooperative architectures

1. INTRODUCTION

Kernel adaptive filters (KAFs) are able to solve nonlinear problems by implicitly projecting the input vector to a high dimensional space, where a standard linear adaptive filter is employed (see, e.g., [1–3]). They have been mainly applied in system identification [4], echo cancellation [5], time series prediction [1, 6–8], and channel equalization [1, 9].

The kernel version of the least-mean-squares (LMS) algorithm, denoted as KLMS [9], is the most popular among the different KAFs. KLMS maps the input column-vector $\mathbf{x}(n) \in \mathbb{R}^N$ into a high dimensional feature space $\mathbb{F}$ as $\varphi(\mathbf{x}(n))$, using a Mercer's kernel, which is a continuous, symmetric, and positive-definite function $\kappa : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$, such that $\kappa(\mathbf{x}, \mathbf{x}') = \varphi(\mathbf{x})^T \varphi(\mathbf{x}')$, where the superscript T denotes transposition. The Gaussian kernel is one of the most used kernel functions in the literature. It is defined as $\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\|\mathbf{x} - \mathbf{x}'\|^2/(2\sigma^2)\right)$, where $\|\cdot\|$ denotes the Euclidean norm and $\sigma > 0$ the kernel width [1].

In the space $\mathbb{F}$, the LMS algorithm is used to update the filter weight column-vector $\mathbf{\Omega}(n-1)$ in order to estimate the desired signal $d(n)$. The Representer's Theorem states that $\mathbf{\Omega}(n-1)$ can be expressed as a linear combination of past input samples, so that the filter output can be obtained as

$$y(n) = \varphi(\mathbf{x}(n))^T \mathbf{\Omega}(n-1) = \mu \sum_{i=1}^{n-1} e(i)\kappa(\mathbf{x}(n), \mathbf{x}(i)), \quad (1)$$

where $e(i) = d(i) - y(i)$ and μ is a step size.

KAFs offer a powerful alternative to nonlinear filters, such as Volterra and functional link adaptive filters [10, 11]. However, their main drawbacks are the selection of an appropriate kernel and the high computational burden and memory, since the dictionary size (expansion in (1)) grows linearly with the incoming samples.

To avoid this linear growth, different sparsification techniques have been proposed to include in the dictionary only informative data (see, e.g., [12–16]). In this context, an alternative to these techniques is the quantized KLMS (QKLMS) algorithm [3]. If the minimum distance between the input vector $\mathbf{x}(n)$ and all the vectors contained in the dictionary is smaller than a threshold, the dictionary is kept unchanged and the coefficient of the closest vector is updated. Otherwise, the input vector is included in the dictionary and its coefficient must also be added in the memory. The use of "redundant" data to locally update the coefficient of the closest vector leads to an algorithm with better accuracy and a more compact dictionary [3].

To select an appropriate kernel, many adaptive multiple kernel approaches have been proposed in the literature. For instance, the single-input multikernel LMS (SI-KLMS) scheme [17], where the kernel function is a convex combination of kernels, i.e., $\kappa(\mathbf{x}, \mathbf{x}') = \sum_{\ell=1}^{L} \beta_\ell \kappa_\ell(\mathbf{x}, \mathbf{x}')$, where β_ℓ, $\ell = 1, \cdots, L$ are coefficients fixed beforehand with $\beta_\ell > 0$ and $\sum_{\ell=1}^{L} \beta_\ell = 1$. The component kernels $\kappa_\ell(\cdot, \cdot)$ can simply be usual kernels with different parameter settings. Since the sum of Mercer's kernels $\sum_{\ell=1}^{L} \beta_\ell \kappa_\ell(\cdot, \cdot)$ is also a valid Mercer's kernel, this approach is equivalent to the monokernel KLMS but considers a different feature space.

A convex combination of two KLMS filters was also proposed to select a proper kernel function [2, 6]. These works recur to the standard combination of two filters [18, 19], where the global output is computed as $y(n) = \eta(n)y_1(n) + [1 - \eta(n)]y_2(n)$ being $y_i(n)$, $i = 1, 2$ the outputs of the KLMS filters and $\eta(n)$ a mixing parameter. Depending on the value of $\eta(n)$, the combination may show a preference for one of the kernels. If the mixing parameter takes an intermediate value, both filters are used simultaneously, and the combination may perform better than each individual filter.

Finally, the multiple-input multikernel LMS (MI-KLMS) scheme [7, 8] augments the dimensionality of the feature space by mapping input samples to multiple feature elements, which in turn are linearly combined to yield the output estimate. This has been proven advantageous in some scenarios when compared with the standard kernel regression paradigm, allowing the learning of multiple nonlinear features presented in the data [8]. Considering the multidimensional mapping of $\mathbf{x}(n)$ to a feature space $\mathbb{M}$, given by $\mathbf{\Phi}(\mathbf{x}(n)) = [\varphi_1(\mathbf{x}(n)) \cdots \varphi_L(\mathbf{x}(n))]^T$, with $\varphi_\ell \in \mathbb{F}_\ell$ for $\ell = 1, \cdots, L$, and the inner product in $\mathbb{M}$ defined as $\langle \mathbf{\Phi}, \mathbf{\Phi}' \rangle_{\mathbb{M}} = \sum_{\ell=1}^{L} \langle \varphi_\ell, \varphi'_\ell \rangle$, it is possible to show that $\mathbb{M}$ is a Hilbert space and presents the reproducing properties [8]. Therefore, there is a kernel $\mathcal{K}$ associated with the inner product in $\mathbb{M}$, $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \langle \mathbf{\Phi}(\mathbf{x}), \mathbf{\Phi}(\mathbf{x}') \rangle_{\mathbb{M}}$. Thus, the filter output can be calculated by generalizing (1) to the new kernel,

$$y(n) = \sum_{i=1}^{n-1} \mu e(i) \sum_{\ell=1}^{L} \kappa_\ell(\mathbf{x}(i), \mathbf{x}(n)). \quad (2)$$

*The work of Silva has been partly supported by FAPESP under Grant 2017/20378-9 and CNPq under Grant 304715/2017-4.

†The work of Arenas-García has been partly supported by MINECO Projects TEC2014-52289-R, TEC2016-81900-REDT, and TEC2017-83838-R, and by Comunidad de Madrid Project PRICAM S2013/ICE-2933. The work of Azpicueta-Ruiz has been partly supported by MINECO Projects TEC2014-52289-R and TEC2017-83838-R.

This approach is equivalent to considering L KLMS filters in parallel and adapting them using a single error signal given by $e(n) = d(n) - y(n)$ [8]. In terms of performance, simulation results show that MI-KLMS generally tends to present a lower mean-square error (MSE) in comparison with SI-KLMS as a trade-off to its higher computational cost. In some applications, MI-KLMS may present lower MSE in comparison to the convex combination of two KLMS filters [2, 7]. However, in certain situations, if the parameters of one kernel component are poorly adjusted, then the convex combination is able to select the best component filter (making $\eta(n) \approx 0$ or $\eta(n) \approx 1$) and may outperform SI-KLMS and MI-KLMS, which have their output degraded by the poorly adjusted kernel. We notice that all these L-kernel approaches present similar computational cost considering the same sparsification techniques for the dictionaries.

In this paper, we propose a scheme to improve the selection of kernel filters in the MI-KLMS scheme. By multiplying the output of each kernel filter by an adaptive biasing factor between 0 and 1 [20], we can minimize the degrading effects of poorly adjusted kernel filters in the MI-KLMS scheme. Thus, our approach gets advantages of MI-KLMS and of the convex combination of KLMS filters. The paper is organized as follows. The proposed scheme is shown in Section 2, followed by some simulation results and conclusions, in Sections 3 and 4, respectively.

2. SELECTING KERNEL FILTERS IN MI-KLMS SCHEME

Fig. 1 depicts a block diagram of the proposed scheme, which is named as robust MI-KLMS (R-MI-KLMS) from now on. Our proposal is based on the inclusion of an upper layer on the MI-KLMS filter composed of different biasing factors with values between 0 and 1. At each branch, the factor $\lambda_\ell(n)$, $\ell = 1, 2, \ldots, L$ multiplies the output of the ℓ^{th} filter that composes MI-KLMS. In this way, the output of the proposed R-MI-KLMS can be calculated as

$$y_b(n) = \sum_{\ell=1}^{L} \lambda_\ell(n) y_\ell(n), \tag{3}$$

where $y_\ell(n)$ is the output of each individual adaptive kernel. This permits to weight the output of each kernel activating or deactivating the output of unnecessary kernels in the global filter output.

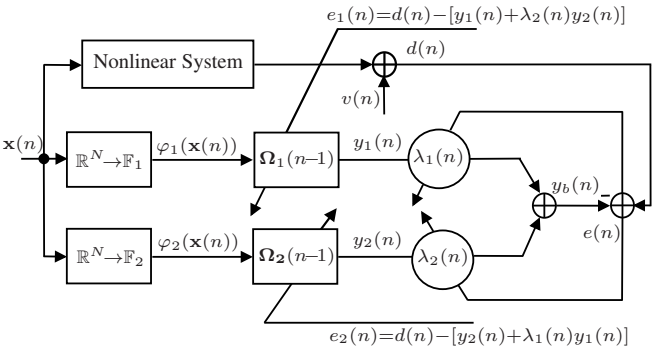

Fig. 1. R-MI-KLMS with $L = 2$ KAFs applied to nonlinear system identification, where $v(n)$ is a measurement noise.

All the biasing parameters are adapted to minimize the MSE of the global output, i.e., $\mathrm{E}\{e^2(n)\}$, where $\mathrm{E}\{\cdot\}$ stands for expectation and $e(n) = d(n) - y_b(n)$, as can be seen in Fig. 1 for $L = 2$ KAFs. Adapting these biasing factors will result in $\lambda_\ell(n) \to 1$ when the modeling capability of the kernel contributes to a reduction of the MSE, whereas $\lambda_\ell(n) \to 0$ otherwise. In the latter case, not only the output of the corresponding kernel will be suppressed, but also the gradient noise incurred by its adaptation, which is specially important in scenarios where the signal-to-noise ratio (SNR) is low [20].

An important difference between the MI-KLMS scheme and our proposal is related with the error employed to adapt each kernel. The update of all kernel filters in the MI-KLMS scheme use the same error $d(n) - \sum_{\ell=1}^{L} y_\ell(n)$, whereas the ℓ^{th} kernel of the R-MI-KLMS scheme updates its coefficients with [21]

$$e_\ell(n) = d(n) - \left[y_\ell(n) + \sum_{l=1, l\neq\ell}^{L} \lambda_l(n) y_l(n) \right] \tag{4}$$

in order to keep the best properties of each possible biased kernel of other branches. Thus, if $\lambda_l(n) \approx 0$ with $l \neq \ell$, Eq. (4) avoids incorporating in the update of the ℓ^{th} kernel information from kernels that are not useful or that incorporate a large amount of gradient noise.

Different strategies can be followed to adapt the biasing parameters of the R-MI-KLMS scheme. In our proposal, and following a similar strategy to that of [20], we reinterpret the output of each branch of R-MI-KLMS as a convex combination with a virtual kernel whose output is always zero. Making that, we get

$$y_b(n) = \sum_{\ell=1}^{L} \lambda_\ell(n) y_\ell(n) = \sum_{\ell=1}^{L} \lambda_\ell(n) y_\ell(n) + [1 - \lambda_\ell(n)] \cdot 0. \tag{5}$$

In this way, we can adapt the value of $\lambda_\ell(n)$ with $\ell = 1, \ldots, L$ employing well-known adaptive rules [19].

However, instead of adapting directly each biasing factor $\lambda_\ell(n)$, we adapt an auxiliary biasing parameter $\alpha_\ell(n)$ univocally related with $\lambda_\ell(n)$ by means of the sigmoidal function [19], i.e., $\lambda_\ell(n) = \mathrm{sgm}[\alpha_\ell(n-1)] = [1 + e^{-\alpha(n-1)}]^{-1}$. For this, we follow a popular scheme using a power-normalized stochastic gradient descent method to minimize $e^2(n)$, i.e.,

$$\alpha_\ell(n) = \alpha_\ell(n-1) + \frac{\mu_{\alpha_\ell}}{p_\ell(n)} e(n) y_\ell(n) \lambda_\ell(n) [1 - \lambda_\ell(n)], \tag{6}$$

where μ_{α_ℓ} is a step size and $p_\ell(n) = \beta p_\ell(n-1) + (1-\beta) y_\ell^2(n)$ represents a low-pass filtered version of $y_\ell^2(n)$, with $0 << \beta < 1$ being a smoothing factor. This scheme to adapt the biasing factors has two advantages:

- the use of the sigmoidal activation function is an easy way to keep $\lambda_\ell(n) \in [0, 1]$; and
- the factor $\lambda_\ell(n)[1 - \lambda_\ell(n)]$ in (6) reduces the gradient noise introduced in the adaptation of $\alpha_\ell(n)$ when the biasing factor is close to the limiting values 0 or 1, since the effective step size in (6) is reduced. It should be noted that $\lambda_\ell(n) = 1$ and $\lambda_\ell(n) = 0$ represent two important situations where a kernel is completely considered or discarded for the global output.

From a practical point of view, the adaptation of the biasing parameters has to be slightly modified [19]. To avoid the paralysis of the adaptation in (6), $\alpha_\ell(n)$ has to be restricted to a range of $[-\alpha^+, \alpha^+]$ (being $\alpha^+ = 4$ a common value used [18, 19]). As a consequence, to guarantee that $\lambda_\ell(n) \in [0, 1]$ under this restriction, the relation between $\lambda_\ell(n)$ and $\alpha_\ell(n)$ is given as [20]

$$\lambda_\ell(n) = \psi[\alpha(n-1)] \triangleq \frac{\mathrm{sgm}[\alpha_\ell(n-1)] - \mathrm{sgm}[-\alpha^+]}{\mathrm{sgm}[\alpha^+] - \mathrm{sgm}[-\alpha^+]}. \tag{7}$$

The definition of $\lambda_\ell(n)$ in (7) gives rise to a slight modification of Eq. (6), as can be seen in Table 1, which shows the pseudocode of the proposed R-MI-KLMS scheme considering L QKLMS filters. In this table, the dictionary for the ℓ^{th} KAF at time instant n is denoted as $\mathcal{C}_\ell(n) = \{\mathbf{x}(c_{j,\ell})\}_{j=1}^{N_{c,\ell}(n)}$, where $\mathbf{x}(c_{j,\ell})$ is the j^{th} element and $N_{c,\ell}(n)$ is its cardinality, which can vary from a time instant to another. The index $c_{j,\ell} \in \{1, 2, \cdots, n-1\}$ is used to distinguish the dictionary elements $\mathbf{x}(c_{1,\ell}), \cdots, \mathbf{x}(c_{N_c(n),\ell})$ from the input vector $\mathbf{x}(n)$. Analogously, the vector $\mathbf{a}_\ell(n) = [a_{c_{1,\ell}}(n) \cdots a_{c_{N_c(n),\ell}}(n)]^T$ contains the coefficients of the ℓ^{th} KAF.

Table 1. Summary of the R-MI-KLMS algorithm.

Input: $\{\mathbf{x}(n) \in \mathbb{R}^N, d(n)\}, \quad n = 1, 2, \cdots$
Initialization: Choose step sizes $\mu_\ell > 0$ and μ_{α_ℓ}, $\ell = 1 \cdots, L$, kernel widths $\sigma_\ell > 0$, thresholds $\varepsilon_\ell \geq 0$, $\ell = 1, \cdots, L$, forgetting factor β and initialize the dictionaries $\mathcal{C}_\ell(1) = \{\mathbf{x}(1)\}$, the parameters $\alpha_\ell(1) = \alpha^+$, $p_\ell(1) = 1$, and coefficient vectors $\mathbf{a}_\ell(1) = [\mu d(1)]$, $\ell = 1, \cdots, L$.

For $n = 2, 3, \cdots,$ do:
  For $\ell = 1, \cdots, L,$ do:

$$\lambda_\ell(n) = \psi[\alpha_\ell(n-1)] = \frac{\text{sgm}[\alpha_\ell(n-1)] - \text{sgm}[-\alpha^+]}{\text{sgm}[\alpha^+] - \text{sgm}[-\alpha^+]}$$

$$y_\ell(n) = \sum_{j=1}^{N_{c,\ell}(n)} a_{j,\ell}(n-1)\kappa_\ell(\mathbf{x}(n), \mathbf{x}(c_{j,\ell}))$$

$$\psi'[\alpha_\ell(n-1)] = \frac{\text{sgm}[\alpha_\ell(n-1)]\{1 - \text{sgm}[\alpha_\ell(n-1)]\}}{\text{sgm}[\alpha^+] - \text{sgm}[-\alpha^+]}$$

$$p_\ell(n) = \beta\, p_\ell(n-1) + (1-\beta)\, y_\ell^2(n)$$

  end
  $y_b(n) = \sum_{\ell=1}^{L} \lambda_\ell(n) y_\ell(n)$
  $e(n) = d(n) - y_b(n)$
  For $\ell = 1, \cdots, L,$ do:

$$e_\ell(n) = d(n) - \left[y_\ell(n) + \sum_{\ell'=1,\ \ell' \neq \ell}^{L} \lambda_{\ell'}(n) y_{\ell'}(n)\right]$$

$$\alpha_\ell(n) = \alpha_\ell(n-1) + \frac{\mu_{\alpha_\ell}}{p_\ell(n)} e(n) y_\ell(n) \psi'[\alpha_\ell(n-1)]$$

    if $|\alpha_\ell(n)| > \alpha^+$
      $\alpha_\ell(n) \leftarrow \text{sign}[\alpha_\ell(n)]\, \alpha^+$
    end

$$\text{dis}(\mathbf{x}(n), \mathcal{C}_\ell(n)) = \min_{1 \leq j \leq N_{c,\ell}(n)} \|\mathbf{x}(n) - \mathbf{x}(c_{j,\ell})\|$$

    If $\text{dis}(\mathbf{x}(n),\ \mathcal{C}_\ell(n)) \leq \varepsilon_\ell$, keep the dictionary unchanged: $\mathcal{C}_\ell(n+1) = \mathcal{C}_\ell(n)$, and quantize $\mathbf{x}(n)$ to the closest vector by updating the coefficient of that vector, i.e.,

$$a_{j^*,\ell}(n) = a_{j^*,\ell}(n-1) + \mu e_\ell(n),$$

    where

$$j^* = \arg \min_{1 \leq j \leq N_{c,\ell}(n)} \|\mathbf{x}(n) - \mathbf{x}(c_{j,\ell})\|$$

    Otherwise, assign a new center and corresponding new coefficient:

$$\mathcal{C}_\ell(n+1) = \{\mathcal{C}_\ell(n), \mathbf{x}(n)\},$$
$$\mathbf{a}_\ell(n) = [\mathbf{a}_\ell(n-1), \mu e_\ell(n)].$$

  end
end

3. SIMULATION RESULTS

In this section, we compare the performance of our proposal, with that of three multikernel solutions: MI-KLMS [7,8], SI-KLMS [17], and with the adaptive convex combination of two KLMS, denoted here as CC-KLMS [2,6]. Due to the inherent advantages of QKLMS, we consider this algorithm in the pseudocode of Table 1 as well as in all simulations of this section. Therefore, the letter "Q" appears in the acronyms of all schemes. We consider only the Gaussian kernel function and $L = 2$ filters for all schemes.

In the first set of simulations, we consider the problem of nonlinear prediction, where the desired sequence is computed as [7]

$$d(n) = \left[0.8 - 0.5\exp(-d^2(n-1))\right] d(n-1) - \left[0.3 + 0.9\exp(-d^2(n-1))\right] d(n-2) + 0.1\sin(d(n-1)\pi),$$

for $0 \leq n < N_{\text{it}}/2$, and

$$d(n) = \left[0.2 - 0.7\exp(-d^2(n-1))\right] d(n-1) - \left[0.8 + 0.8\exp(-d^2(n-1))\right] d(n-2) + 0.2\sin(d(n-1)\pi),$$

for $N_{\text{it}}/2 \leq n < N_{\text{it}}$ being N_{it} the number of iterations and $d(n) = 0.1$ for $n < 0$.

Considering $N = 2$, $N_{\text{it}} = 10^4$, and adjusting the parameters as in Table 2, we obtain the results shown in Figs. 2 and 3. Fig. 2 shows the learning curves, evolution of the mixing parameter $\eta(n)$ for CC-QKLMS, and evolution of the biasing parameters $\lambda_1(n)$ and $\lambda_2(n)$ for the R-MI-KLMS scheme, considering kernel widths as $\sigma_1 = 0.1$ and $\sigma_2 = 1$. We can observe that the CC-QKLMS scheme performs as its best component filter, which is confirmed by the mixing parameter evolution. SI-QKLMS$_1$ outperforms SI-QKLMS$_2$, which shows the importance of choosing adequately the step size and the dictionary threshold for this scheme. Finally, MI-QKLMS and its robust version present the same performance, since both component filters are considered to compute the overall output of R-MI-KLMS, which can be confirmed by the evolution of the biasing parameters. These schemes outperform all multiple kernel solutions considered in the simulation in terms of convergence rate and steady-state error.

Table 2. Parameters used in the simulations.

Algorithm	Parameters
QKLMS$_1$	$\mu_1 = 0.05$, σ_1, $\varepsilon_1 = 0.05$
QKLMS$_2$	$\mu_2 = 0.5$, σ_2, $\varepsilon_2 = 0.5$
CC-QKLMS	$\alpha^+ = 4$, $\beta = 0.9$, μ_α
SI-QKLMS$_1$	$\mu = 0.05$, $\beta_1 = \beta_2 = 0.5$, σ_1, σ_2, $\varepsilon = 0.05$
SI-QKLMS$_2$	$\mu = 0.5$, $\beta_1 = \beta_2 = 0.5$, σ_1, σ_2, $\varepsilon = 0.5$
MI-QKLMS	$\mu_1 = 0.05$, $\mu_2 = 0.5$, σ_1, σ_2, $\varepsilon_1 = 0.05$, $\varepsilon_2 = 0.5$
R-MI-QKLMS	$\mu_1 = 0.05$, $\mu_2 = 0.5$, σ_1, σ_2, $\varepsilon_1 = 0.05$, $\varepsilon_2 = 0.5$, $\alpha^+ = 4$, $\beta = 0.9$, μ_{α_1}, μ_{α_2}

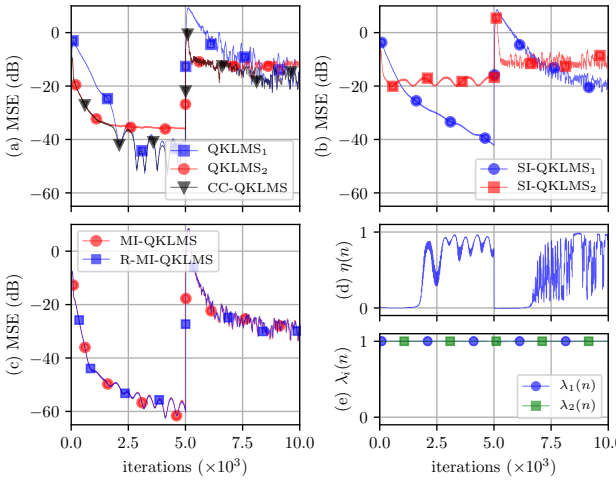

Fig. 2. MSE considering signal prediction performed with CC-QKMLS (a), with the mixing parameter (d); SI-QKLMS (b); MI-MQKLMS and R-MIQKLMS (c), with biasing parameters (e). Settings $\sigma_1 = 0.1$, $\sigma_2 = 1$, $\mu_\alpha = \mu_{\alpha_1} = \mu_{\alpha_2} = 1.5$ and other parameters as in Table 2. To facilitate the visualization, the MSE curves are smoothed using a moving average filter with 64 taps.

Changing the kernel widths to $\sigma_1 = 0.2$ and $\sigma_2 = 100$ and maintaining the other parameters as in Table 2, we obtain the results of Fig. 3. In this case, the kernel width $\sigma_2 = 100$ does not lead to good results for a monokernel filter as we can observe in the learning curve of the QKLMS$_2$ filter. Therefore, CC-QKLMS follows QKLMS$_1$ since its mixing parameter converges to one. The SI-QKLMS$_1$ scheme presents a lower convergence rate than that of the monokernel QKLMS$_1$. Finally, the kernel width $\sigma_2 = 100$ degrades the performance of MI-QKLMS for $1200 < n < 5000$, which is avoided by the proposed R-MI-QKLMS, since the biasing factor $\lambda_2(n) \approx 0$ until iteration $n = 6000$. After that, the kernel width $\sigma_2 = 100$ is considered in the overall output, since it does not degrade the performance of MI-QKLMS anymore.

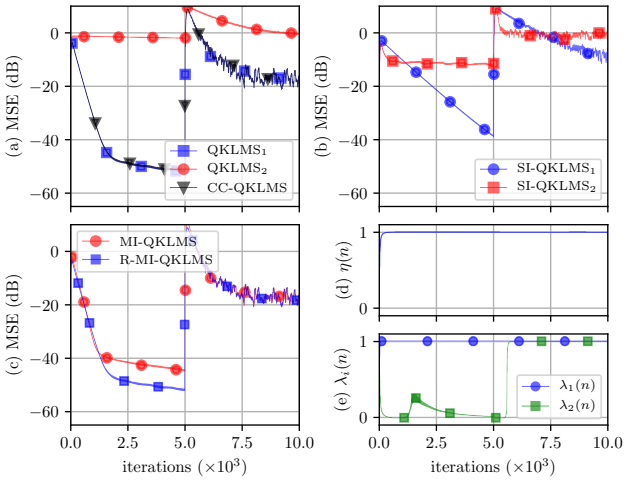

Fig. 3. Same as Fig. 2 but considering $\sigma_1 = 0.2$, $\sigma_2 = 100$, $\mu_\alpha = \mu_{\alpha_1} = \mu_{\alpha_2} = 0.3$ and other parameters as in Table 2.

In the second set of simulations, we consider a nonlinear system identification application with two different systems modified from [4]. For the first half of the experiment, we consider the first system, where the desired signal is $d_1(n) = s_1(n) + z_1(n)$, with

$$s_1(n) = \frac{s_1(n-1)}{1 + s_1^2(n-1)} + x^3(n-1),$$

where $x(n)$ is the input signal and $z_1(n)$ is a zero-mean Gaussian noise with variance $\sigma_{z_1}^2 = 10^{-4}$. At $n = 5 \times 10^4$, we consider an abrupt change that leads to a second nonlinear system, which generates a desired sequence computed as $d_2(n) = \psi(s_2(n)) + z_2(n)$, with $z_2(n)$ a zero mean Gaussian noise with variance $\sigma_{z_2}^2 = 10^{-6}$, $s_2(n) = x(n) + 0.5x(n-1) - 0.2s_2(n-1) + 0.35s_2(n-2)$, and

$$\psi(s) = \begin{cases} \dfrac{s}{3\sqrt{0.1 + 0.9s^2}} & \text{for } s \geq 0 \\ \dfrac{-s^2[1 - \exp(0.7s)]}{3} & \text{for } s < 0 \end{cases}.$$

Fig. 4 shows the learning curves in terms of excess MSE (EMSE), evolution of the mixing parameter $\eta(n)$ for CC-QKLMS, and evolution of the biasing parameters for R-MI-QKLMS. The input signal $x(n)$ was i.i.d. randomly generated from a zero-mean Gaussian distribution with standard deviation $\sigma_x = 0.15$. We can observe that CC-QKLMS performs as the best component filter. The performance of SI-QKLMS$_2$ is better than that of SI-QKLMS$_1$ in the identification of the first system but for the second system, SI-QKLMS$_1$ outperforms SI-QKLMS$_2$, emphasizing the importance of adjusting the step size and dictionary threshold adequately for this scheme. This result also shows that, even considering a smaller number of elements in the dictionary (by using a lower value for the dictionary threshold), it is possible to obtain an improved performance, depending on the other parameters. In this scenario, MI-QKLMS presents a poor performance and it is possible to observe the importance of using the improved version R-MI-QKLMS. This robust version is able to minimize the degrading effects of unnecessary kernels, achieving a learning curve with a convergence rate slightly higher than that of CC-QKLMS.

Table 3 shows the gain of R-MI-QKLMS in relation to other multikernel schemes in terms of steady-state EMSE in the identification of the second system for different values of SNR. We can observe that R-MI-QKLMS outperforms other multikernel schemes for the considered range of SNR. The advantage of R-MI-QKLMS is especially remarkable for low SNRs, which is useful in the frequent situations in which the SNR is unknown or changes over time. This is an important capability that the CC-QKLMS scheme lacks.

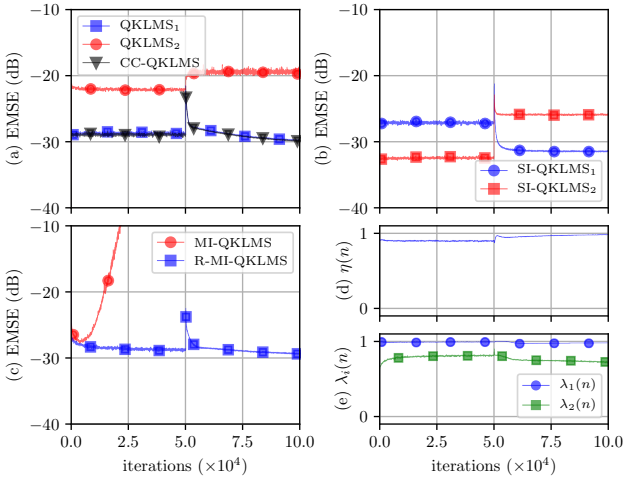

Fig. 4. EMSE considering system identification performed with CC-QKMLS (a), with the mixing parameter (d); SI-QKLMS (b); MI-MQKLMS and R-MIQKLMS (c), with biasing parameters (e). Settings $N = 2$, $\sigma_1 = 1$, $\sigma_2 = 0.1$, $\mu_\alpha = \mu_{\alpha_1} = \mu_{\alpha_2} = 1$ and other parameters as in Table 2. Average of 1000 runs. To facilitate the visualization, the EMSE curves are smoothed using a moving average filter with 64 taps.

Table 3. Gain of R-MI-QKLMS in the identification of the second system in relation to other schemes in terms of steady-state EMSE (dB), estimated averaging the last 1000 samples after filters convergence. Settings $\sigma_1 = 1$, $\sigma_2 = 0.1$, $\mu_\alpha = \mu_{\alpha_1} = \mu_{\alpha_2} = 0.1$ and other parameters as in Table 2.

SNR	CC-QKLMS	SI-QKLMS$_1$	SI-QKLMS$_2$	MI-QKLMS
−25	10.5	9.4	23.9	20.7
−20	6.7	6.8	15.3	16.6
−15	3.3	3.2	11.4	13.4
−10	1.4	0.9	8.5	9.9
−5	0.6	0.5	6.3	8.4
0	0.4	0.4	4.5	5.2
5	0.1	0.3	2.2	3.5
10	0.1	0.5	1.6	2.8
15	0	0.2	0.5	1.6

4. CONCLUSION

KAFs are important tools to solve nonlinear problems such as time series prediction, system identification, among others. Besides the high computational burden and memory, the selection of an appropriate kernel is essential to obtain a good performance in terms of MSE. To circumvent this problem, multikernel schemes that combine two or more KAFs with different kernel functions have been proposed in the literature. However, if the parameters of one kernel component are poorly adjusted, the performance of these schemes can be degraded. In this paper, we proposed a robust multikernel scheme that is able to activate or deactivate the output of unnecessary kernels in the global filter output. With a slight increase in the computational cost in comparison with MI-QKLMS, the proposed scheme can outperform other multikernel solutions, in scenarios where the settings of one or more kernels are not appropriate, and/or when the SNR is low. In a future work, we intend to theoretically analyze the proposed scheme and consider it in other applications.

5. REFERENCES

[1] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*, Wiley, 2010.

[2] W. Gao, C. Richard, J. C. M. Bermudez, and J. Huang, "Convex combinations of kernel adaptive filters," in *Proc. of IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Sep. 2014, pp. 1–5.

[3] B. Chen, S. Zhao, P. Zhu, and J. C. Principe, "Quantized kernel least mean square algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 22–32, Jan 2012.

[4] W. D. Parreira, J. C. M. Bermudez, C. Richard, and J. Y. Tourneret, "Stochastic behavior analysis of the gaussian kernel least-mean-square algorithm," *IEEE Transactions on Signal Processing*, vol. 60, no. 5, pp. 2208–2222, May 2012.

[5] S. Van Vaerenbergh, L. A. Azpicueta-Ruiz, and D. Comminiello, "A split kernel adaptive filtering architecture for nonlinear acoustic echo cancellation," in *Proc. of the European Signal Processing Conference (EUSIPCO)*, Budapest, Hungary, 2016, pp. 1768–1772.

[6] M. Scarpiniti, D. Comminiello, R. Parisi, and A. Uncini, "A collaborative approach to time-series prediction," in *Neural Nets WIRN11*, B. Apolloni, S. Bassis, A. Esposito, and F. C. Morabito, Eds., pp. 178–185. IOS Press, 2011.

[7] M. Yukawa, "Multikernel adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 60, no. 9, pp. 4672–4682, Sep. 2012.

[8] F. A. Tobar, S. Y. Kung, and D. P. Mandic, "Multikernel least mean square algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 2, pp. 265–277, Feb. 2014.

[9] W. Liu, P. P. Pokharel, and J. C. Principe, "The kernel least-mean-square algorithm," *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 543–554, Feb 2008.

[10] V. J. Mathews, "Adaptive polynomial filters," *IEEE Signal Processing Magazine*, vol. 8, pp. 10–26, July 1991.

[11] G. L. Sicuranza and A. Carini, "A generalized FLANN filter for nonlinear active noise control," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2412–2417, Nov 2011.

[12] J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 2, pp. 213–225, 1991.

[13] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[14] W. Liu, I. Park, and J. C. Principe, "An information theoretic approach of designing sparse kernel adaptive filters," *IEEE Transactions on Neural Networks*, vol. 20, no. 12, pp. 1950–1961, Dec. 2009.

[15] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 1058–1067, Mar. 2009.

[16] K. Slavakis, P. Bouboulis, and S. Theodoridis, "Online learning in reproducing kernel hilbert spaces," in *Academic Press Library in Signal Processing: Signal Processing Theory and Machine Learning*, R. Chellapa and S. Theodoridis, Eds., vol. 1, chapter 17, pp. 883–987. Academic Press, Chennai, 2014.

[17] G. R. G Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine Learning Research*, vol. 5, no. Jan, pp. 27–72, 2004.

[18] J. Arenas-García, A. R. Figueiras-Vidal, and A. H. Sayed, "Mean-square performance of a convex combination of two adaptive filters," *IEEE Transactions on Signal Processing*, vol. 54, pp. 1078–1090, Mar. 2006.

[19] J. Arenas-Garcia, L. A. Azpicueta-Ruiz, M. T. M. Silva, V. H. Nascimento, and A. H. Sayed, "Combinations of adaptive filters: Performance and convergence properties," *IEEE Signal Processing Magazine*, vol. 33, no. 1, pp. 120–140, Jan 2016.

[20] M. Lázaro-Gredilla, L. A. Azpicueta-Ruiz, A. R. Figueiras-Vidal, and J. Arenas-Garcia, "Adaptively biasing the weights of adaptive filters," *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3890–3895, Jul. 2010.

[21] L. A. Azpicueta-Ruiz, M. Zeller, A. R. Figueiras-Vidal, J. Arenas-Garcia, and W. Kellermann, "Adaptive combination of Volterra kernels and its application to nonlinear acoustic echo cancellation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, pp. 97–110, Jan. 2011.