DICTIONARY LEARNING FOR GAUSSIAN KERNEL ADAPTIVE FILTERING WITH VARIABLE KERNEL CENTER AND WIDTH

Tomoya Wada, Kosuke Fukumori, and Toshihisa Tanaka

Department of Electrical and Electronic Engineering, Tokyo University of Agriculture and Technology
2–14–16 Nakacho, Koganei-shi, Tokyo, 184–8588, Japan
Email: {wada15, fukumori17}@sip.tuat.ac.jp, tanakat@cc.tuat.ac.jp

ABSTRACT

This paper establishes an adaptive update method for the Gaussian kernel parameters in the application to the kernel adaptive filtering (KAF). In this method, the kernel parameters are all adaptive and data-driven, although they should be given or estimated by cross-validation. In terms of the Gaussian KAF, every input sample or signal has its own width and center, which are updated at each iteration based on the proposed least-square-type rules to minimize the estimation error. In particular, the proposed update rule keeps the width in the manifold of the positive numbers. Together with the ℓ_1-regularized least squares, the overall KAF algorithm can avoid the overfitting and the increase of dimensionality. Experimental results support the validity of the method.

Index Terms— Nonlinear adaptive filtering, kernel methods, reproducing kernel Hilbert space, dictionary learning.

1. INTRODUCTION

An adaptive filter or adaptive filtering is a system or technique that updates its parameters at every time step to approximate a static or dynamic unknown system [1]. Although traditional adaptive filters assume linear models, many situations require nonlinear adaptive filters, since systems in the real environments can be modeled nonlinear. Several types of nonlinear adaptive filters have been reported. Among them, the kernel adaptive filtering (KAF) as an efficient nonlinear approximation approach with online manner developed in a reproducing kernel Hilbert space (RKHS) has attracted much more interests [2, 3].

The system model of the KAF is represented by the superposition of the kernels corresponding to the observed signals (or samples), where the adaptive algorithm is intended to estimate coupling coefficients of kernels. Typical KAF algorithms include the kernel least mean square (KLMS) [4–7], the kernel normalized least mean square (KNLMS), the kernel affine projection algorithms (KAPA) [8, 9], and the kernel recursive least squares (KRLS) [10], etc. The main bottleneck of the KAF algorithms is their linearly growing structure with each new input signal, which poses both computational issues and overfitting. A straightforward but practical approach to this problem is to limit the number of observed signals. This set of observed signals is called a *dictionary*. Typical criteria for the dictionary learning include the novelty criterion [11], the approximate linear dependency (ALD) criterion [10], the surprise criterion [12], and the coherence-based criterion [13]. They accept only the novel and informative input signals as the dictionary members. Another approach is the ℓ_1-regularization [14, 15]. In this approach, the filter coefficients are regularized by the ℓ_1-norm, which makes some of coefficients zero, and then the corresponding entries in the dictionary are eliminated. Even when the system model dynamically changes, the number of dictionary members can be suppressed.

Another implementation of the KAF is to update the parameters of kernels to decrease the estimation error of the output. In standard KAF, kernel centers are given as observed signals. Some related works proposed to adaptively move all the kernel centers in the dictionary to minimize the square error [16–18]. The kernel width is another important parameter to govern the performance kernel machines [19–22]. An attempt to adaptively estimate the kernel width has been reported recently [22]. In a recent work [23], Wada and Tanaka have also proposed a novel update method for the kernel width, which can efficiently find a proper width in the manifold of the positive numbers. Although the above methods are efficient strategies for increasing performance of the KAF, all kernel functions in each system have a common width parameter, since a RKHS only has one single kernel function. The use of a RKHS is mathematically simple but the model is not flexible.

This paper proposes an adaptive update method for both the Gaussian centers and widths of all kernel functions that contribute to the KAF model. Thus, the dictionary in the proposed KAF consists of a couple of the kernel center and width. For each input signal, all entries in the dictionary are updated to minimize the estimation error. Least-square-type rules are adopted for this strategy. In particular, the developed update rule with a logarithm map can search kernel widths on the manifold of positive real numbers. This double adaptation strategy for the kernel center and the kernel width is incorporated with the ℓ_1-regularized least squares for updating the filter coefficients, which lead to avoiding the overfitting and the increase of dimensionality.

2. KERNEL NONLINEAR FILTERING MODEL

Let $\boldsymbol{u}^{(n)} \in \mathcal{U} \subset \mathbb{R}^L$ be a input signal at time instance n and $d^{(n)} \in \mathbb{R}$ be the corresponding desired signal. The goal is to learn a continuous input-output mapping, $f : \mathcal{U} \to \mathbb{R}$, based on the incoming sequence, $\{\boldsymbol{u}^{(i)}, d^{(i)}\}_{i=1}^N$, in the reproducing kernel Hilbert space (RKHS), $\mathcal{H}$, induced by the positive-definite kernel, $\kappa(\cdot,\cdot) : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$. In the KAF, $f(\cdot)$ is modeled a linear form in RKHS: $f(\cdot) = \langle \Omega, \phi(\cdot) \rangle$, where $\phi(\boldsymbol{u}) = \kappa(\cdot, \boldsymbol{u})$ is a nonlinear function which transfers the input signal from the initial space to RKHS and Ω is the weight vector in RKHS. According to the representer theorem [13], Ω can be described as

$$\Omega = \sum_{j=1}^{r} h_j \kappa(\cdot, \boldsymbol{c}_j). \tag{1}$$

This work is supported by JSPS KAKENHI Grant Number 17H01760.

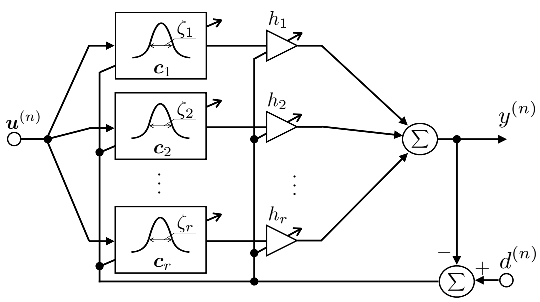


Fig. 1: Conceptual diagram of the proposed V-CAW.

where $\{h_j\}_{j=1}^{r} \in \mathbb{R}$ are the filter coefficients to be estimated and $\mathcal{D} = \{\boldsymbol{c}_j\}_{j=1}^{r}$ is a set of input signals accepted by a criterion. This set is called the dictionary. The number of dictionary is restricted to r ($r^{(n)} \ll n$) in the sparse representation [10–15]; therefore, the dictionary, $\mathcal{D}^{(n)}$, is updated at every time instance. Then, the filter output is represented as

$$y^{(n)} = \langle \Omega, \kappa(\cdot, \boldsymbol{u}^{(n)}) \rangle = \sum\nolimits_{j=1}^{r} h_j \kappa(\boldsymbol{u}^{(n)}, \boldsymbol{c}_j). \tag{2}$$

A widely-used kernel function is the Gaussian kernel, which is a celebrated example of positive definite kernels, defined as

$$\kappa(\cdot, \boldsymbol{c}; \zeta) = \exp\left(-\zeta \|\cdot - \boldsymbol{c}\|^2\right), \tag{3}$$

where $\boldsymbol{c}$ and ζ are the parameters called the *center* and the *width* of the Gaussian kernel, respectively. In the traditional KAF algorithms, the centers are not moved once they are added to the dictionary and the width is fixed common value among the centers. In a word, they have been static parameters. Unlike this, we propose to regard them as dynamic parameters. In the rest of the paper, the proposed method for data-driven adaptation of the parameters is described.

3. UPDATE METHOD FOR VARIABLE KERNEL CENTERS AND WIDTHS (V-CAW)

The proposed model is the superposition of the Gaussian kernels with time-variable width $\zeta_j^{(n)}$ and center $\boldsymbol{c}_j^{(n)}$ given as:

$$\begin{aligned} y^{(n)} &= \sum\nolimits_{j=1}^{r^{(n)}} h_j^{(n)} \kappa(\boldsymbol{u}^{(n)}, \boldsymbol{c}_j^{(n)}; \zeta_j^{(n)}) \\ &= \sum\nolimits_{j=1}^{r^{(n)}} h_j^{(n)} \exp(-\zeta_j^{(n)} \|\boldsymbol{u}^{(n)} - \boldsymbol{c}_j^{(n)}\|^2) \\ &= \boldsymbol{h}^{(n)\top} \boldsymbol{\kappa}^{(n)}, \end{aligned} \tag{4}$$

where

$$\boldsymbol{h}^{(n)} := [h_1^{(n)} h_2^{(n)}, \ldots, h_r^{(n)}]^\top \in \mathbb{R}^r, \tag{5}$$

$$\boldsymbol{\kappa}^{(n)} := [\kappa(\boldsymbol{u}^{(n)}, \boldsymbol{c}_1^{(n)}; \zeta_1^{(n)}), \kappa(\boldsymbol{u}^{(n)}, \boldsymbol{c}_2^{(n)}; \zeta_2^{(n)}), \ldots, \kappa(\boldsymbol{u}^{(n)}, \boldsymbol{c}_r^{(n)}; \zeta_r^{(n)})]^\top \in \mathbb{R}^r. \tag{6}$$

Fig. 1 shows a conceptual diagram of the proposed V-CAW. It should be noted that the dictionary at time n is a time-variable of a couple the center and the width, which is described as

$$\mathcal{D}^{(n)} = \{(\boldsymbol{c}_1^{(n)}, \zeta_1^{(n)}), (\boldsymbol{c}_2^{(n)}, \zeta_2^{(n)}), \ldots, (\boldsymbol{c}_r^{(n)}, \zeta_r^{(n)})\}. \tag{7}$$

To adaptively find the proper parameters, we adopt the instantaneous square error as the loss function:

$$J^{(n)}(\mathcal{D}^{(n)}) := e^{(n)^2} = |d^{(n)} - y^{(n)}|^2. \tag{8}$$

Remark. We describe the sum space of RKHS [24] in order to discuss a space in which multikernel adaptive filters [25, 26], including the proposed filter, exist. We consider the case of sum space of two RKHS, for the sake of ease without loss of generality.

Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be Hilbert spaces. Moreover, $\mathcal{H}_1 \oplus \mathcal{H}_2$ denotes the direct sum of RKHS of $\mathcal{H}_1$ and $\mathcal{H}_2$, represented as H. In this case, the norm of direct sum of $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$, $f = (f_1, f_2) \in H$, is represented as follows [24]:

$$\|f\|_H^2 := \|f_1\|_{\mathcal{H}_1}^2 + \|f_2\|_{\mathcal{H}_2}^2. \tag{9}$$

In particular, if $\mathcal{H}_1 \cap \mathcal{H}_2 = \{0\}$, the sum space, $\mathcal{H} := \{f = f_1 + f_2 \mid f_1 \in \mathcal{H}_1, f_2 \in \mathcal{H}_2\}$, is isomorphic to the direct space, H [24]. Consequently, the norm in $\mathcal{H}$ is represented as

$$\|f\|_{\mathcal{H}}^2 := \|f_1\|_{\mathcal{H}_1}^2 + \|f_2\|_{\mathcal{H}_2}^2. \tag{10}$$

Also, let the kernel of $\mathcal{H}_1$ and the kernel of $\mathcal{H}_2$ denote κ_1 and κ_2, respectively. The value of any $f \in \mathcal{H}$ can be evaluated by the kernel, $\kappa = \kappa_1 + \kappa_2$ [24]:

$$f(\boldsymbol{u}) = \langle f, \kappa(\cdot, \boldsymbol{u}) \rangle_{\mathcal{H}} = \langle f_1, \kappa_1(\cdot, \boldsymbol{u}) \rangle_{\mathcal{H}_1} + \langle f_2, \kappa_2(\cdot, \boldsymbol{u}) \rangle_{\mathcal{H}_2}. \tag{11}$$

Assume that M different kernels, $\{\kappa_m(\cdot, \cdot)\}_{m=1}^{M}$, are given. Also, let $\mathcal{H}_m$ and $\mathcal{H}$ denote RKHS determined by the mth kernel and the corresponding sum space, respectively. In this case, from (11), the output is represented with the filter, $\Omega \in \mathcal{H}$, and nonlinear mapping of input, $\phi(\boldsymbol{u}^{(n)}) = \kappa(\cdot, \boldsymbol{u}^{(n)}) \in \mathcal{H}$, as

$$y^{(n)} = \langle \Omega, \kappa(\cdot, \boldsymbol{u}^{(n)}) \rangle_{\mathcal{H}} = \sum\nolimits_{m=1}^{M} \langle \Omega_m, \kappa_m(\cdot, \boldsymbol{u}^{(n)}) \rangle_{\mathcal{H}_m}, \tag{12}$$

where Ω_m is constructed in each $\mathcal{H}_m$ and Ω is the sum of Ω_m. It should be noted that there is no need for the index set of the dictionary in each RKHS to equalize [26]. Therefore, the output of our filter in (4) can be rewritten as a multikernel adaptive filter with r^n different kernels:

$$y^{(n)} = \langle \Omega, \kappa(\cdot, \boldsymbol{u}^{(n)}) \rangle_{\mathcal{H}} = \sum\nolimits_{j=1}^{r} \langle \Omega_j, \kappa_j(\cdot, \boldsymbol{u}^{(n)}) \rangle_{\mathcal{H}_j}, \tag{13}$$

where $\Omega_j = h_j \kappa_j(\cdot, \boldsymbol{c}_j)$.

3.1. Updating the Kernel Centers

The update rule for each kernel center can be derived by using least mean square (LMS) algorithm [1]:

$$\begin{aligned} \boldsymbol{c}_j^{(n+1)} &= \boldsymbol{c}_j^{(n)} - \eta_c \left. \frac{\partial J^{(n)}(\boldsymbol{c}_j)}{\partial \boldsymbol{c}_j} \right|_{\boldsymbol{c}_j = \boldsymbol{c}_j^{(n)}} \\ &= \boldsymbol{c}_j^{(n)} + 4\eta_c \zeta_j e^{(n)} h_j \exp\left(-\zeta_j \|\boldsymbol{u}^{(n)} - \boldsymbol{c}_j^{(n)}\|^2\right) (\boldsymbol{u}^{(n)} - \boldsymbol{c}_j^{(n)}), \end{aligned} \tag{14}$$

where η_c is a step size. Although the update methods for the centers proposed in [16–18] are efficient strategies for increasing performance of the KAF, the generalization capability depends on the selection of the kernel width. Therefore, in addition to that, we propose to update the kernel widths in Section 3.2.

3.2. Updating the Kernel Widths

The width is a positive parameter, which is an element of a manifold of the positive real numbers, denoted by $\mathbb{R}^+$. On the other hand, a standard LMS is an algorithm for parameters in $\mathbb{R}$. This paper proposes to employ a one-to-one map from $\mathbb{R}^+$ to $\mathbb{R}$ given by $\xi(\zeta) = \log(\zeta/\zeta^{(n)})$, where $\zeta^{(n)}$ is the value at the nth iteration. Then, a standard LMS is applied to ξ in $\mathbb{R}$. Following this scenario, the LMS update for $\xi_j \in \mathbb{R}$ can be derived as

$$\begin{aligned}\xi_j^{(n+1)} &= \underbrace{\xi_j(\zeta_j^{(n)})}_{=0} - \eta_{\mathrm{w}} \left.\frac{\partial J^{(n)}(\zeta_j)}{\partial \xi_j}\right|_{\zeta_j=\zeta_j^{(n)}} = -\eta_{\mathrm{w}} \left.\frac{\partial J^{(n)}(\zeta_j)}{\partial \zeta_j}\frac{\partial \zeta_j(\xi_j)}{\partial \xi_j}\right|_{\zeta_j=\zeta_j^{(n)}} \\ &= -\eta_{\mathrm{w}}\zeta_j^{(n)} \left.\frac{\partial J^{(n)}(\zeta_j)}{\partial \zeta_j}\right|_{\zeta_j=\zeta_j^{(n)}}, \qquad (15)\end{aligned}$$

where η_{w} is a step size. Thanks to the normalization by $\zeta_j^{(n)}$, we can update them stably on the manifold of positive real numbers, even when ζ_j is very small. Then, the $(n+1)$th width is obtained by applying the inverse map, $\zeta(\xi) = \zeta^{(n)}\exp(\xi)$. Therefore, the following update rule is derived:

$$\begin{aligned}\zeta_j^{(n+1)} &= \zeta_j^{(n)}\exp(\xi_j^{(n+1)}) = \zeta^{(n)}\exp\left(-\eta_{\mathrm{w}}\zeta_j^{(n)} \left.\frac{\partial J^{(n)}(\zeta_j)}{\partial \zeta_j}\right|_{\zeta_j=\zeta_j^{(n)}}\right) \\ &= \zeta_j^{(n)}\exp\left(-2\eta_{\mathrm{w}}\zeta_j^{(n)}e^{(n)}h_j^{(n)}\exp\left(-\zeta_j^{(n)}\|\boldsymbol{u}^{(n)}-\boldsymbol{c}_j^{(n)}\|^2\right)\|\boldsymbol{u}^{(n)}-\boldsymbol{c}_j^{(n)}\|^2\right). \qquad (16)\end{aligned}$$

4. ℓ_1-REGULARIZED KNLMS INCORPORATED WITH V-CAW

To avoid the overfitting and the increase of dimensionality, the proposed update method for the parameters in Section 3 is incorporated with the ℓ_1-regularized least squares for updating the filter coefficients. In this scenario, the weighted ℓ_1-norm is added to the square error:

$$\Theta^{(n)} := |d^{(n)} - \boldsymbol{h}^{(n)\top}\boldsymbol{\kappa}^{(n)}|^2 + \lambda \underbrace{\sum_{j=1}^{r^{(n)}} w_j^{(n)}|h_j^{(n)}|}_{:=\psi^{(n)}}, \qquad (17)$$

where $\psi^{(n)}$ and λ are a weighted ℓ_1-norm and a regularization parameter, respectively. Moreover, $w_j^{(n)} = 1/(|h_j^{(n)}| + \beta)$ is a dynamically adjusted weight [15]. To minimizing the cost function, we can apply the forward-backward splitting [27], since $\Theta^{(n)}$ is a convex function. The update rule is then given as follows [15]:

$$\boldsymbol{h}^{(n+1)} = \mathrm{prox}_{\mu\lambda\psi^{(n)}}\left[\overline{\boldsymbol{h}}^{(n)} + \frac{\mu\left(d^{(n)} - \overline{\boldsymbol{h}^{(n)}}^{\top}\overline{\boldsymbol{\kappa}^{(n)}}\right)\overline{\boldsymbol{\kappa}^{(n)}}}{\rho + \|\overline{\boldsymbol{\kappa}^{(n)}}\|^2}\right], \qquad (18)$$

where μ is a step size parameter and ρ is a stabilization parameter. Also, $\overline{\boldsymbol{\kappa}^{(n)}} := [\boldsymbol{\kappa}^{(n)\top}, \kappa(\boldsymbol{u}^{(n)}, \boldsymbol{u}^{(n)})]^\top$ and $\overline{\boldsymbol{h}^{(n)}} := [\boldsymbol{h}^{(n)\top}, 0]^\top$. Besides, $\mathrm{prox}_{\mu\lambda\psi^{(n)}}(\cdot)$ denotes the proximal operator [27] of $\lambda\psi^{(n)}$. For a vector $\boldsymbol{\alpha} := [\alpha_1, \alpha_2, \ldots, \alpha_r]^\top \in \mathbb{R}^r$, $\mathrm{prox}_{\mu\lambda\psi^{(n)}}(\boldsymbol{\alpha})$ is given as

$$\left(\mathrm{prox}_{\mu\lambda\psi^{(n)}}(\boldsymbol{\alpha})\right)_j = \mathrm{sgn}\{\alpha_j\}\max\{|\alpha_j| - \mu\lambda w_j^{(n)}, 0\}, \qquad (19)$$

where $(\cdot)_j$ denotes the jth element of a vector. This update rule can promote the sparsity of h_j, and then some of the coefficients become almost zero. If $h_j \approx 0$, remove $(\boldsymbol{c}_j, \zeta_j)$ from the dictionary. The overall algorithm (KNLMS-ℓ_1 + V-CAW) is summarized in Algorithm 1, where Steps 9 and 10 are the main steps of V-CAW.

Algorithm 1 KNLMS-ℓ_1 + V-CAW

1: Set the initial kernel width, $\zeta_{\mathrm{init.}}$.
2: Add $(\boldsymbol{u}^{(0)}, \zeta_{\mathrm{init.}})$ into the dictionary as the 1st member, $\mathcal{D}^{(0)} = \{(\boldsymbol{c}_0, \zeta_0)\}$.
3: **while** $\{\boldsymbol{u}^{(n)}, d^{(n)}\}$ $(n > 1)$ available **do**
4: Add $(\boldsymbol{u}^{(n)}, \zeta_{\mathrm{init.}})$ into the dictionary, $\mathcal{D}^{(n)}$.
5: Update the filter coefficients using (18).
6: **if** $h_j \approx 0$ **then**
7: Remove $(\boldsymbol{c}_j, \zeta_j)$ from the dictionary, $\mathcal{D}^{(n)}$.
8: **end if**
9: Update the kernel centers using (14)
10: Update the kernel widths using (15).
11: **end while**

5. NUMERICAL EXAMPLES

To illustrate the performance of the proposed algorithm, numerical examples including dynamic system identification and real world Internet traffic prediction are presented.

5.1. Dynamic System Identification

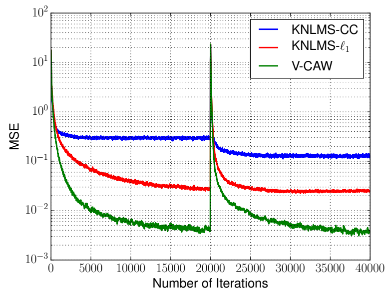
We consider the nonstationary nonlinear system [18] as follows:

$$d^{(n)} := \begin{cases} 10\{e^{-(u^{(n)}-3)^2} + e^{-(u^{(n)}-7)^2}\} & (0 \le n \le 20,000) \\ 10\{e^{-a(u^{(n)}-13)^2} + e^{-b(u^{(n)}-17)^2}\} & (20,000 < n \le 40,000) \end{cases} \qquad (20)$$

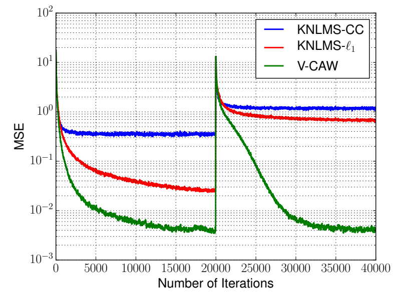
which is corrupted by noise sampled from a zero-mean Gaussian distribution with standard deviation equal to 0.3. In the above system, a and b are constants. Input signals $u^{(n)}$ are sampled from uniform distribution on the interval $[0, 10]$ when $0 \le n \le 20,000$ and the interval $[10, 20]$ when $20,000 < n \le 40,000$. For comparison purpose, we test the KNLMS with coherence-based criterion (KNLMS-CC) [13], the KNLMS with ℓ_1-regularization (KNLMS-ℓ_1) [14,15], and the proposed method in Algorithm 1 (V-CAW). The parameters of them are set as: KNLMS-CC ($\mu = 0.09$, $\rho = 0.03$, $\zeta = 1.0$, $\delta = 0.5$), KNLMS-ℓ_1 ($\mu = 0.09$, $\rho = 0.03$, $\zeta = 1.0$, $\lambda = 5.0\times10^{-3}, \beta = 0.1$), and V-CAW ($\mu = 0.09, \rho = 0.03, \zeta_{\mathrm{init.}} = 1.0$, $\eta_{\mathrm{c}} = 1.0\times10^{-3}$, $\eta_{\mathrm{w}} = 1.0\times10^{-3}$, $\lambda = 5.0\times10^{-3}$, $\beta = 0.1$), where δ in KNLMS-CC is a threshold to determine the level of sparsity of the model [13]. We adopt mean square error (MSE) as the evaluation criteria. The MSE is calculated by taking an arithmetic average over 100 independent realizations. We test three different sets of values of (a, b): $(a, b) = \{(0.5, 0.5), (2, 2), (0.5, 2)\}$. Figs. 2 (a), (b), and (c) show that the V-CAW achieves lower MSE than the others when $0 \le n \le 20,000$. This implies the efficacy of updating the centers. In addition, it is observed that the V-CAW flexibly responds to the system switches at $n = 20,000$. It should be noted that the V-CAW can adapt the system featured by the sum of different Gaussian kernels such as the case of that $(a, b) = (0.5, 2)$ since the proposed method has different widths for each kernel. That is to say, the proposed V-CAW can flexibly adapt for the unknown systems. Moreover, Figs. 2 (d), (e), and (f) exhibit that the V-CAW has the smallest dictionary size after the switch of the nonlinear system. The above results support the efficacy of adaptation for the centers and the widths.

5.2. Internet Traffic Prediction

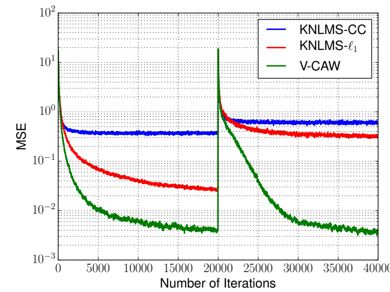
The second example is a real-world Internet traffic dataset [21]. The task is to predict the current value of the sample using the previous ten consecutive samples. For the convenience of computation, all



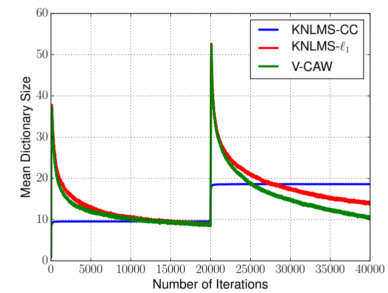
(a) MSE when $(a, b) = (0.5, 0.5)$



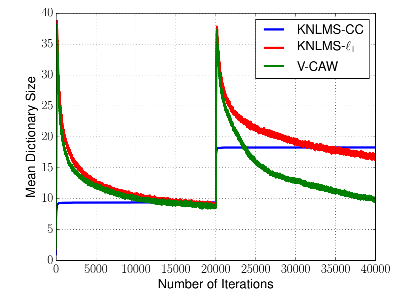
(b) MSE when $(a, b) = (2, 2)$



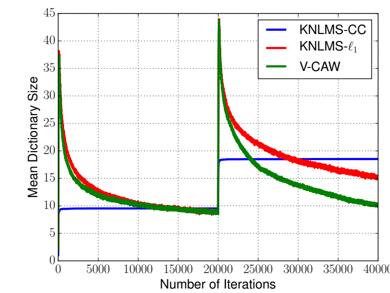
(c) MSE when $(a, b) = (0.5, 2)$



(d) Mean dictionary size when $(a, b) = (0.5, 0.5)$

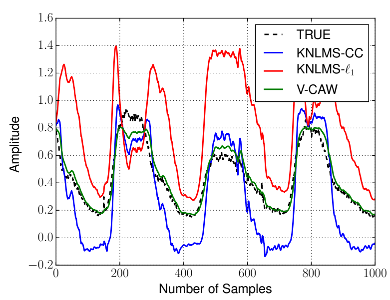


(e) Mean dictionary size when $(a, b) = (2, 2)$

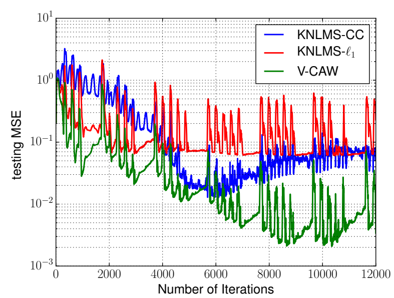


(f) Mean dictionary size when $(a, b) = (0.5, 2)$

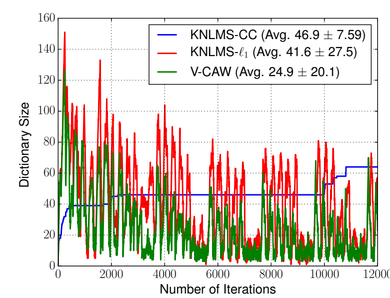
Fig. 2: Performance comparison: The convergence curves of MSE ((a), (b), and (c)) and mean dictionary size ((e), (f), and (g)) for three different parameters (a, b). These results are calculated by taking an arithmetic average over 100 independent realizations.



(a) The tracking of each filter



(b) Convergence curves



(c) Dictionary size evolutions

Fig. 3: Internet traffic time series prediction. (a) The tracking of each filter in terms of the test data, (b) the convergence curves of the trained filters, and (c) Dictionary size evolutions in terms of training.

samples are normalized into the interval $[0, 1]$. We again compare the performance of the KNLMS-CC ($\mu = 0.5$, $\rho = 0.03$, $\zeta = 1.0$, $\delta = 0.97$), the KNLMS-ℓ_1 ($\mu = 0.5$, $\rho = 0.03$, $\zeta = 1.0$, $\lambda = 5.0 \times 10^{-5}$, $\beta = 0.1$), and the V-CAW ($\mu = 0.5$, $\rho = 0.03$, $\zeta_{\text{init.}} = 1.0$, $\eta_c = 0.5$, $\eta_w = 0.07$, $\lambda = 5.0 \times 10^{-5}$, $\beta = 0.1$). In this example, 12,000 samples are used as the training data and the other 1,000 samples as the test data. Fig. 3 (a) shows that the V-CAW has the highest tracking ability. Besides, the convergence cures in terms of the testing MSE are demonstrated in Fig. 3 (b). At each iteration, the testing MSE is computed on the test set using the filter resulting from the training set. Fig. 3 (b) demonstrates that the V-CAW achieves the best performance. Finally, Fig. 3 (c) shows the dictionary size evolutions in terms of training. It is observed in Fig. 3 (c) that the V-CAW suppresses the increase of the dictionary size.

6. CONCLUSION

This paper proposed an update method for variable kernel centers and widths (V-CAW) in the kernel adaptive filtering. In the V-CAW, the dictionary consists of a couple of the kernel center and width. For each input signal, all entries in the dictionary are updated by the proposed least-square type rules. In particular, the developed update rule with a logarithm map can search kernel widths on the manifold of positive real numbers. The proposed V-CAW is incorporated with the ℓ_1-regularized least squares to avoid the overfitting and the increase of dimensionality. Numerical examples showed that the proposed method exhibits higher performance in terms of the MSE and the dictionary size in dynamic systems.

7. REFERENCES

[1] S. Haykin, *Adaptive Filter Theory*. Upper Saddle River, NJ: Prentice-Hall, 2002.

[2] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, 2004.

[3] W. Liu, J. Principe, and S. Haykin, *Kernel Adaptive Filtering*. Hoboken, NJ: Wiley, 2010.

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

[5] P. Bouboulis and S. Theodoridis, "Extension of Wirtinger's calculus to reproducing kernel Hilbert spaces and the complex kernel LMS," *IEEE Trans. Signal Process.*, vol. 59, no. 3, pp. 964–978, 2011.

[6] B. Chen, S. Zhao, P. Zhu, and J. C. Principe, "Quantized kernel least mean square algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 1, pp. 22–32, 2012.

[7] F. Tobar, S.-Y. Kung, and D. Mandic, "Multikernel least mean square algorithm," *IEEE Trans. Neural Netw.*, vol. 25, no. 2, pp. 265–277, 2014.

[8] W. Liu and J. C. Príncipe, "Kernel affine projection algorithms," *EURASIP J. Adv. Signal Process.*, vol. 2008, no. 1, pp. 1–13, 2008.

[9] J. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. Jensen, "Nonlinear acoustic echo cancellation based on a parallel-cascade kernel affine projection algorithm," in *Proc. of 2012 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2012)*, 2012, pp. 33–36.

[10] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, 2004.

[11] J. Platt, "A resource-allocating network for function interpolation," *Neural computation*, vol. 3, no. 2, pp. 213–225, 1991.

[12] W. Liu, I. Park, Y. Wang, and J. C. Príncipe, "Extended kernel recursive least squares algorithm," *IEEE Trans. Signal Process.*, vol. 57, no. 10, pp. 3801–3814, 2009.

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

[14] W. Gao, J. Chen, C. Richard, J. Huang, and R. Flamary, "Kernel LMS algorithm with forward-backward splitting for dictionary learning," in *Proc. of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2013)*, 2013, pp. 5735–5739.

[15] W. Gao, J. Chen, C. Richard, and J. Huang, "Online dictionary learning for kernel LMS," *IEEE Trans. Signal Process.*, vol. 62, no. 11, pp. 2765–2777, 2014.

[16] C. Saide, R. Lengelle, P. Honeine, C. Richard, and R. Achkar, "Dictionary adaptation for online prediction of time series data with kernels," in *Proc. of 2012 IEEE Statistical Signal Processing Workshop (SSP)*, 2012, pp. 604–607.

[17] C. Saide, R. Lengelle, P. Honeine, and R. Achkar, "Online kernel adaptive algorithms with dictionary adaptation for MIMO models," *IEEE Signal Process. Lett.*, vol. 20, no. 5, pp. 535–538, 2013.

[18] T. Ishida and T. Tanaka, "Efficient construction of dictionaries for kernel adaptive filtering in a dynamic environment," in *Proc. of 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, 2015, pp. 3536–3540.

[19] N. Benoudjit and M. Verleysen, "On the kernel widths in radial-basis function networks," *Neural Process. Lett.*, vol. 18, no. 2, pp. 139–154, 2003.

[20] A. K. Ghosh, "Kernel discriminant analysis using case-specific smoothing parameters," *IEEE Trans. Syst., Man, Cybern. B*, vol. 38, no. 5, pp. 1413–1418, 2008.

[21] B. Chen, J. Liang, N. Zheng, and J. C. Principe, "Kernel least mean square with adaptive kernel size," *Neurocomputing*, vol. 191, pp. 95–106, 2016.

[22] H. Fan, Q. Song, and S. B. Shrestha, "Kernel online learning with adaptive kernel width," *Neurocomputing*, vol. 175, pp. 233–242, 2016.

[23] T. Wada and T. Tanaka, "Doubly adaptive kernel filtering," in *Proc. of 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA 2017)*, no. TA-P3.6, 2017.

[24] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, no. 9, pp. 337–404, 1950.

[25] M. Yukawa, "Multikernel adaptive filtering," *IEEE Trans. Signal Process.*, vol. 60, no. 9, pp. 4672–4682, 2012.

[26] T. Ishida and T. Tanaka, "Multikernel adaptive filters with multiple dictionaries and regularization," in *Proc. of 2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA 2013)*, 2013, pp. 1–6.

[27] Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada, "A sparse adaptive filtering using time-varying soft-thresholding techniques," in *Proc. of 2010 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2010)*, 2010, pp. 3734–3737.