ONLINE LEARNING WITH SELF-TUNED GAUSSIAN KERNELS: GOOD KERNEL-INITIALIZATION BY MULTISCALE SCREENING

Masa-aki Takizawa[1] *and Masahiro Yukawa*[1,2] *

1. Department of Electronics and Electrical Engineering, Keio University, Japan
2. Center for Advanced Intelligence Project, RIKEN, Japan

ABSTRACT

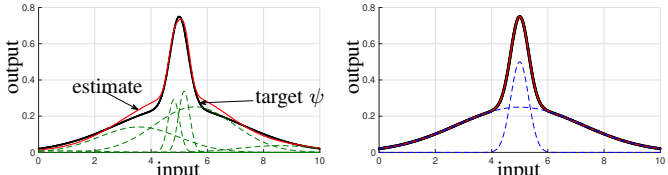
We propose an efficient adaptive update method for the kernel parameters: the kernel coefficients, scales and centers. The mirror descent and the steepest descent method for squared error cost function are employed to update the kernel scales and centers, respectively. Although the problem considered in this paper is nonconvex, we reduce the possibility of falling into local minima by using a novel multiple initialization scheme to grow the dictionary without great increases of the dictionary size. Through computer experiments, we show that the proposed algorithm enjoys a high adaptation-capability while maintaining a small dictionary size, without detailed tuning of the initial kernel parameters.

Index Terms— nonlinear adaptive estimation, automatic parameter tuning, Gaussian kernel, dictionary learning

1. INTRODUCTION

A variety of signal processing problems can be cast as adaptive estimation of nonlinear functions. The kernel adaptive filtering has attracted significant attention as an efficient online scheme for this task [1–13]. In the kernel adaptive filtering, the target nonlinear function is modeled as an element of a reproducing kernel Hilbert space [14, 15]. The major difficulty lies in finding an "efficient model" (e.g., the scale (variance) parameter of the Gaussian kernel) in the sense of reducing the estimation errors with low (or affordable at most) complexity. If improper models are used, the kernel adaptive filter may need a large-size dictionary, which causes slow convergence. Under the use of celebrated Gaussian kernel, the efficacy of models relies on the scales and centers of Gaussian functions. Let us present a brief review of the selection schemes for the kernel scales and centers. Center-selection schemes have been discussed in terms of novelty criteria which only pick up novel data from the input samples [3, 4]. An adaptive dictionary-refinement technique based on the proximity operator of a weighted (block) ℓ_1 norm has been proposed in [11, 12, 16]. Regarding the kernel scales, a reasonable scale parameter has been assumed available prior to adaptation in the early studies of kernel adaptive filtering. This assumption is however unrealistic, particularly when the data under consideration are nonstationary.

Motivation: The multikernel adaptive filtering has been proposed as a convex analytic approach with multiple different scales. The concept of *online model selection and learning* has been presented in [17, 18] based on the multikernel adaptive filtering framework, selecting appropriate scales (model parameters) from a hundred of possible scales by shrinking the coefficient vector for each scale while learning those parameters also to reduce the estimation errors simultaneously. Although those multikernel adaptive filtering approaches alleviate the difficulties of kernel design, there is still sufficient room for improvements in the sense of "efficiency" of the filter, as seen from the following illustrative example (see also Sec. 4). Let us consider an unknown function (the black curve in Fig. 1(a)) which



(a) estimaton with small-scale kernels (b) estimation with ideal kernels

Fig. 1. Estimating ψ by multikernel adaptive filter with Gaussian kernels which have different scales.

is a sum of two Gaussian functions with different scales, centers, and heights. It can be empirically shown that using many kernels (the green dotted curves in the figure) with smaller scales than the true ones gives a 'good' estimate (the red curve is the sum of those small-scale kernels). In view of Fig. 1(b), however, more efficient estimation with the smallest possible number of kernels (two kernels in the figure) can be achieved if those scales which perfectly match the target function in each local region are employed together with appropriately located centers. This observation motivates us to adapt the kernel scales and centers to enhance the efficiency of nonlinear estimation.

Contributions: In this paper, we propose an efficient adaptive method to update the three parameters: the kernel coefficients, scales, and centers. The problem addressed in this paper is nonconvex in terms of those three parameters, and therefore have the issue of local minima potentially. To reduce the possibility of falling into local minima, the proposed algorithm initializes the kernel scales to multiple values. To suppress sharp increases of the dictionary size, we propose an efficient dictionary growing scheme, named multiscale screening method, which checks the novelty of a new sample in terms of the estimation error and the coherence at different scales in sequence hierarchically in the large-to-small-scale order. The mirror descent [19] with the negative entropy is applied to the squared error cost function to update the scale parameters for guaranteeing the positivity of the scales, while the steepest descent method is employed for the centers. Through computer experiments in applications to online estimation of nonlinear systems and online time-series data prediction using some real data, we show that (i) the proposed algorithm enjoys high adaptation-capability while maintaining a small dictionary size, and (ii) it is also insensitive to the choice of initial scales.

2. PROBLEM STATEMENT

We address an adaptive estimation problem of a nonlinear system ψ with sequentially arriving input signals $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^L$, and its noisy output $d := \psi(\boldsymbol{u}) + \nu \in \mathbb{R}$, where $\boldsymbol{u}$ is assumed an i.i.d. random vector and ν is a zero-mean additive noise uncorrelated with any other signals. We consider the problem of estimating the function ψ by the following model: $\varphi^{(r)}(\boldsymbol{u}) := \sum_{j=1}^{r} h_j \exp\left(-\frac{\|\boldsymbol{u}-\boldsymbol{c}_j\|^2}{\xi_j}\right)$, where $r > 0$ is the expansion length, $\xi_j > 0$ are the scale parameters,

*This work was supported by KAKENHI Grant numbers 18J21595 and 18H01446.

$h_j \in \mathbb{R}$ are the coefficients and $\boldsymbol{c}_j \in \mathbb{R}^L$ are the centers of Gaussians. Here, we denote by $\|\cdot\|$ the norm induced from the canonical inner product $\langle \cdot, \cdot \rangle$ defined in $\mathbb{R}^L$. 'Efficiency' is considered in the present paper in terms of the expansion length r given a target precision $\epsilon > 0$. To obtain an efficient estimate of ψ, a possible problem formulation is given as follows:

$$\underset{r \in \{1,2,3,\cdots\}}{\text{minimize}} \; r \quad \text{s.t.} \; \min_{\varphi^{(r)} \in \mathcal{X}_r} E\left(\varphi^{(r)}(\boldsymbol{u}) - d\right)^2 \leq \epsilon, \quad (1)$$

where $\mathcal{X}_r := \{\varphi^{(r)} \mid h_j \in \mathbb{R},\ \xi_j > 0,\ \boldsymbol{c}_j \in \mathbb{R}^L,\ j = 1, \cdots, r\}$. Here, E denotes the expectation on $\mathcal{U}$. In practice, one may need to give an upper bound of r to keep the complexity and memory requirements reasonable when ϵ is too small (i.e., when high precision is required). In any case, to solve (1) directly, one needs to solve the minimization problem for many possible values of r. This implies that some practical remedy will be required from practical aspects. We therefore introduce an ℓ_1-based sparsification approach by extending the multikernel adaptive filtering framework.

3. PROPOSED PARAMETER ADAPTATION AND DICTIONARY CONSTRUCTION SCHEMES

First, the basic idea and the cost function of the proposed algorithm are presented. Second, a novel dictionary growing strategy which constructs an efficient multiscale structure in a hierarchical manner is presented. Third, the update schemes for the kernel coefficients, scales, and centers are presented in sequence. Finally, relations to prior works are presented.

3.1. Basic Idea and Cost Function

The dictionary is defined as the set

$$\mathcal{D}_n := \{\kappa(\cdot, \boldsymbol{c}_{j,n}; \xi_{j,n})\}_{j=1,\cdots,r_n},\ n \in \mathbb{N} \quad (2)$$

of the r_n-Gaussian kernels $\kappa(\boldsymbol{u}, \boldsymbol{c}_{j,n}; \xi_{j,n}) := \exp\left(-\frac{\|\boldsymbol{u} - \boldsymbol{c}_{j,n}\|^2}{\xi_{j,n}}\right)$, $\boldsymbol{u} \in \mathbb{R}^L$, associated with the scale parameter $\xi_{j,n} > 0$ and the center $\boldsymbol{c}_{j,n} \in \mathbb{R}^L$. Our filter is defined as

$$\varphi_n(\boldsymbol{u}) := \sum_{j=1}^{r_n} h_{j,n} \kappa(\boldsymbol{u}, \boldsymbol{c}_{j,n}; \xi_{j,n}),\ \boldsymbol{u} \in \mathbb{R}^L,\ j = 1 \cdots r_n,$$

where $h_{j,n} \in \mathbb{R}$ is the coefficient of the kernel. The filter output to the input $\boldsymbol{u}_n$ is given by $\varphi_n(\boldsymbol{u}_n) = \sum_{j=1}^{r_n} h_{j,n} \kappa(\boldsymbol{u}_n, \boldsymbol{c}_{j,n}; \xi_{j,n}) = \langle \boldsymbol{h}_n, \boldsymbol{\kappa}_n \rangle$, where $\boldsymbol{h}_n := [h_{1,n}, h_{2,n}, \cdots, h_{r_n,n}]^\mathsf{T}$ is the coefficient vector and $\boldsymbol{\kappa}_n := [\kappa(\boldsymbol{u}_n, \boldsymbol{c}_{1,n}; \xi_{1,n}), \cdots, \kappa(\boldsymbol{u}_n, \boldsymbol{c}_{r_n,n}; \xi_{r_n,n})]^\mathsf{T}$. Let $\alpha := \{(h_j,\ \xi_j, \boldsymbol{c}_j)\}_{j=1,\cdots,r_n} \in \mathbb{R}^{r_n} \times \mathbb{R}_+^{r_n} \times \mathbb{R}^{L \times r_n}$ be the set of kernel parameters. Finding those parameters which construct an efficient filter is formulated as follows:

$$\underset{\alpha}{\text{argmin}}\, E\left[\left(d_n - \varphi^{(r_n)}(\boldsymbol{u}_n)\right)^2\right] + \lambda \Omega_n(\boldsymbol{h}),\ \boldsymbol{h} \in \mathbb{R}^{r_n}, \quad (3)$$

where $\Omega_n(\boldsymbol{h}) := \sum_{j=1}^{r_n} \omega_{j,n} |h_j|$ is a weighted ℓ_1 norm with $\omega_{j,n} > 0$ and $\lambda > 0$ is the regularization parameter. The weighted ℓ_1 norm suppresses the size of dictionary by sparsifying the coefficient vector $\boldsymbol{h}_n$, i.e., small coefficients are enforced to be zero and their corresponding atoms are discarded (see Sec. 3.3 for more details about the role of the weighted ℓ_1 norm). To approach the problem (3) in an online manner, the cost function is defined as

$$J_n(\alpha) := \left(d_n - \varphi^{(r_n)}(\boldsymbol{u}_n)\right)^2 + \lambda \Omega_n(\boldsymbol{h}). \quad (4)$$

The nonconvexity of (3) implies that solutions derived by an iterative algorithm, such as the well-known LMS algorithm, depend on the initial values of the parameters α. To alleviate the sensitivity to the initial conditions, we propose a reasonable dictionary construction scheme which is based on the notion of the multikernel adaptive filtering in the following subsection.

3.2. Multiscale Screening: A Dictionary Growing Strategy

We empirically found that the initial kernel scales affected the performance and efficiency of the filter significantly when the selected scale was far from the scales of the target function. The idea to overcome this issue is to reduce the possibility of falling into local minima by using multiple initial values for the scale parameter. Here, we suppose that at least some of the initial scales are close to the correct scales. However, undesirable growths of the dictionary size due to the use of multiple initial values may cause high computational complexities and large memory size. To avoid this, we present an efficient dictionary growing strategy for the proposed algorithm, named *multiscale screening method*, which extracts global and local structures of the target function with the large- and small- scale kernels, respectively.

Let $\xi_{\text{init}}^{(q)},\ q \in \mathcal{Q} := \{1, 2, \cdots, Q\}$, be the initial kernel scales, where $\xi_{\text{init}}^{(1)} \geq \xi_{\text{init}}^{(2)} \geq \cdots \geq \xi_{\text{init}}^{(Q)} > 0$. Fig. 2 presents a flow chart of the multiscale screening method. The newly arriving datum $\boldsymbol{u}_n$ enters the first layer where it is judged whether the largest-scale kernel $\kappa(\cdot, \boldsymbol{u}_n; \xi_{\text{init}}^{(1)})$ needs to be added to the dictionary $\mathcal{D}_n$ or not by using a certain criterion presented in the next paragraph. If $\kappa(\cdot, \boldsymbol{u}_n; \xi_{\text{init}}^{(1)})$ is added into the dictionary, $\boldsymbol{u}_n$ skips all the following layers. If $\kappa(\cdot, \boldsymbol{u}_n; \xi_{\text{init}}^{(1)})$ is not added into the dictionary, $\boldsymbol{u}_n$ goes to the second layer and it is judged whether the second largest-scale kernel $\kappa(\cdot, \boldsymbol{u}_n; \xi_{\text{init}}^{(2)})$ needs to be added into the dictionary by using the criterion at the current scale, and so on. When the current datum $\boldsymbol{u}_n$ enters the dictionary at the qth scale, then we let $(h_{r_{n+1},n}, \xi_{r_{n+1},n}, \boldsymbol{c}_{r_{n+1},n}) := (0, \xi_{\text{init}}^{(q)}, \boldsymbol{u}_n)$ and $\kappa(\cdot, \boldsymbol{c}_{r_{n+1},n}; \xi_{r_{n+1},n})$ is added to $\mathcal{D}_n$.

The proposed criterion consists of the error and coherence conditions in a hierarchical manner. Roughly speaking, when the error is large, a large-scale kernel is needed to extract a global structure. When the error is small, on the other hand, a small-scale kernel is needed to extract a local structure. The error condition for the qth scale is given by $|e_n| > \epsilon^{(q)}, q \in \mathcal{Q}$ for some small constant $\epsilon^{(1)} \geq \epsilon^{(2)} \geq \cdots \geq \epsilon^{(q)} > 0$, where $e_n := d_n - \varphi_n(\boldsymbol{u}_n)$ is the instantaneous error. If the error condition is satisfied, the filter needs to be updated in the vicinity of $\boldsymbol{u}_n$ with the qth scale. To eliminate the redundancy from the dictionary, the coherence condition [4]

$$\text{coherence}^{(q)} := \max_{j=1,\cdots,r_n} \left|\kappa(\boldsymbol{u}_n, \boldsymbol{c}_j; \xi_{\text{init}}^{(q)})\right| \leq \delta^{(q)},\ q \in \mathcal{Q}$$

is used for some prespecified threshold $\delta^{(q)} \in (0, 1)$.

3.3. Adaptation of Kernel Coefficients and Dictionary Pruning

To minimize the time varying cost function (4) about the coefficients $\boldsymbol{h}_n$ in an online way, we employ the adaptive proximal forward backward splitting (APFBS) algorithm [20]. The basic idea is using the gradient to reduce the smooth term and using the proximity operator to reduce the nonsmooth one. The proximity operator $\text{prox}_{\lambda\Omega} : \mathbb{R}^{r_n} \to \mathbb{R}^{r_n}$ of Ω of index λ is defined as $\text{prox}_{\lambda\Omega_n}(\boldsymbol{x}) := \text{argmin}_{\boldsymbol{y} \in \mathbb{R}^{r_n}} \left(\lambda\Omega_n(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2\right),\ \boldsymbol{x} \in \mathbb{R}^{r_n}$ of which the ith component is $[\text{prox}_{\lambda\Omega_n}(\boldsymbol{x})]_i = \max\left\{1 - \frac{\lambda\omega_{i,n}}{|x_i|}, 0\right\} x_i$. The coefficient vector is updated as

$$\boldsymbol{h}_{n+1} := T\left\{\text{prox}_{\lambda\Omega}\left(\boldsymbol{h}_n + \mu^{(h)}(d_n - \boldsymbol{h}_n^\mathsf{T}\boldsymbol{\kappa}_n)\boldsymbol{\kappa}_n\right)\right\}, \quad (5)$$

where $\mu^{(h)} \in [0, 2]$ is the stepsize parameter, and the operator denoted by T is of decreasing the size of vector by removing zero components, i.e., size $(T(\boldsymbol{x})) = |\{i \in \{1, 2, \cdots, r_n\} \mid x_i \neq 0\}|$

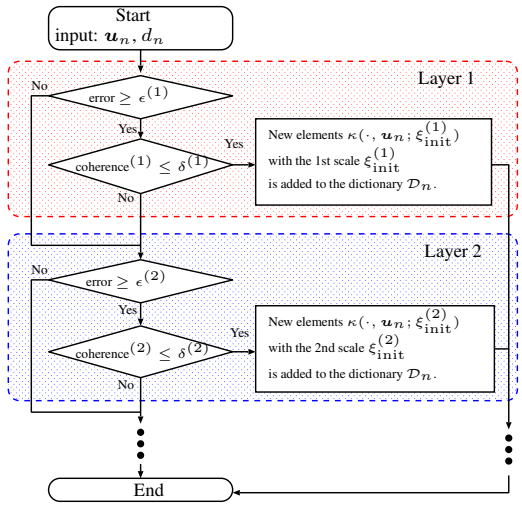


Fig. 2. A flow chart of the multiscale screening method for efficient dictionary growing.

for $\boldsymbol{x} := [x_1, x_2, \cdots, x_{r_n}]^{\mathsf{T}}$. The corresponding dictionary elements are also removed from the dictionary, i.e., $\mathcal{D}_{n+1} := \mathcal{D}_n / \kappa(\cdot, \boldsymbol{c}_{j,n}; \xi_{j,n})$, if $h_{n+1,j} = 0$.

3.4. Adaptation of Kernel Scales

The policy of updating the scale parameter $\xi_{j,n}$ is to suppress the cost J_n on the space $\mathbb{R}_+$ of positive real numbers. To restrict the scale parameter $\xi_{j,n}$ to $\mathbb{R}_+$, we employ the mirror descent method [19] with the negative entropy for the squared error cost J_n to update $\xi_{j,n}$. The mirror descent method updates $\xi_{j,n}$ as

$$\xi_{j,n+1} = \operatorname*{argmin}_{\xi \in \mathbb{R}_+} \left\{ \left\langle \xi, \frac{\partial J_n(\alpha_n)}{\partial \xi_j} \right\rangle + \frac{B_\phi(\xi || \xi_{j,n})}{\mu_{j,n}^{(\xi)}} \right\}, \quad (6)$$

where $\alpha_n := \{(h_{j,n}, \xi_{j,n}, \boldsymbol{c}_{j,n})\}_{j=1,\cdots,r_n}$, $B_\phi(\xi || \xi_{j,n}) := \phi(\xi) - \phi(\xi_{j,n}) - \langle \nabla\phi(\xi_{j,n}), \xi - \xi_{j,n} \rangle$ is a Bregman-divergence with the continuous convex function $\phi(x) := x \log x - x$, $x > 0$, and $\mu_{j,n}^{(\xi)} = \xi_{j,n} \mu^{(\eta)}$ for some small constant $\mu^{(\eta)} > 0$ is the stepsize parameter (see Remark below). The partial differential in (6) is given by

$$\frac{\partial J_n(\alpha_n)}{\partial \xi_j} = -\frac{2 e_n h_{j,n} \|\boldsymbol{u}_n - \boldsymbol{c}_{j,n}\|^2 \kappa(\boldsymbol{u}_n, \boldsymbol{c}_{j,n}; \xi_{j,n})}{\xi_{j,n}^2}. \quad (7)$$

Differentiating the right side of (6), substituting $\nabla\phi(x) = \log x$ and letting the derivative be zero, we obtain the following update equation:

$$\xi_{j,n+1} = \exp\left(\log(\xi_{j,n}) - \mu_{j,n}^{(\xi)} \frac{\partial J_n(\alpha_n)}{\partial \xi_j} \right). \quad (8)$$

Remark on the stepsize $\mu_{j,n}^{(\xi)}$: The update (8) can also be attained through the update for the dual variable $\eta := \nabla\phi(\xi) := \log \xi$ under the Legendre transform $\nabla\phi : \mathbb{R}_+ \to \mathbb{R}$, for which the inverse transform is given by $(\nabla\phi)^{-1}(\eta) := \nabla\phi^*(\eta) = \mathrm{e}^\eta$, where ϕ^* is the Fenchel-Legendre conjugate of ϕ [21]. If we regard J_n as a function of the dual variables $\eta_{j,n} := \log \xi_{j,n}$ with the other parameters fixed to $\boldsymbol{h}_n$ and $\boldsymbol{c}_{j,n}$s, the steepest descent update is given by

$$\eta_{j,n+1} = \eta_{j,n} - \mu^{(\eta)} \frac{\partial J_n(\alpha_n)}{\partial \eta_j} = \eta_{j,n} - \mu^{(\eta)} \xi_{j,n} \frac{\partial J_n(\alpha_n)}{\partial \xi_j}, \quad (9)$$

where the second equality is due to $\frac{\partial J_n(\alpha_n)}{\partial \eta_j} = \frac{\partial J_n(\alpha_n)}{\partial \xi_j} \frac{\partial \xi_j}{\partial \eta_j} = \xi_j \frac{\partial J_n(\alpha_n)}{\partial \xi_j}$. By transforming the update equation (9) back to the ξ domain by the inverse mapping $(\nabla\phi)^{-1}$, we obtain (8) with the stepsize $\mu_{j,n}^{(\xi)} = \xi_{j,n} \mu^{(\eta)}$.

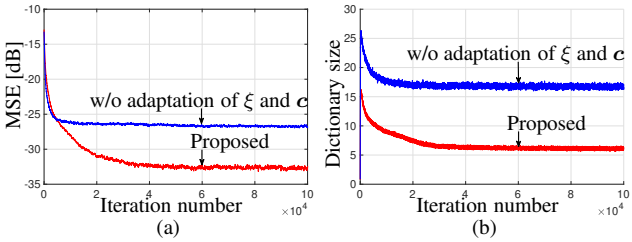


Fig. 3. Results of Experiment 1.

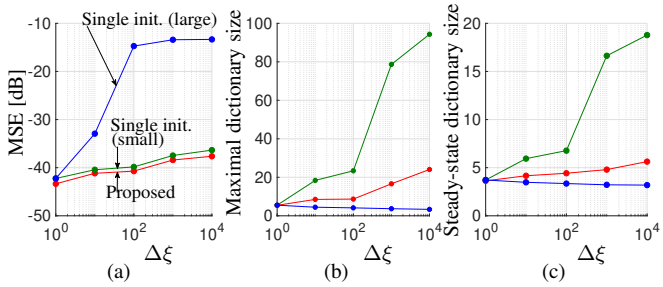


Fig. 4. Results of Experiment 2.

3.5. Adaptation of Kernel Centers

The update of kernel centers $\boldsymbol{c}_{j,n}$ is derived by using the stochastic gradient descent method for J_n. The update is given as

$$\boldsymbol{c}_{j,n+1} = \boldsymbol{c}_{j,n} - \mu^{(c)} \frac{\partial J_n(\alpha_n)}{\partial \boldsymbol{c}_j}, \quad (10)$$

where $\mu^{(c)} > 0$ is the stepsize parameter and

$$\frac{\partial J_n(\alpha_n)}{\partial \boldsymbol{c}_j} = -\frac{4 e_n h_{j,n} \kappa(\boldsymbol{u}_n, \boldsymbol{c}_{j,n}; \xi_{j,n})(\boldsymbol{u}_n - \boldsymbol{c}_{j,n})}{\xi_{j,n}}. \quad (11)$$

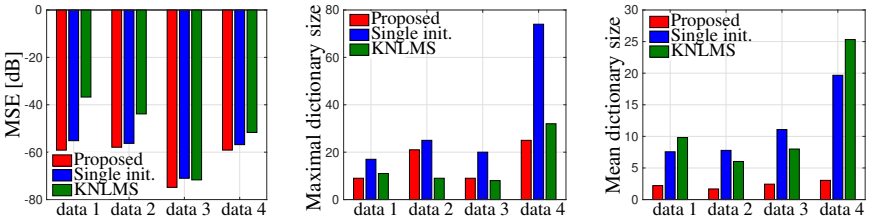
We finally remark that the computational complexity of the proposed algorithm is linear with respect to the dictionary size r_n.

3.6. Relations to Prior Works

In the kernel adaptive filtering context, some related works have been proposed to adapt the kernel scales [22–24] and centers [25, 26] in the dictionary to minimize the squared error. The method in [22] updates the scales only when each kernel enters the dictionary and keeps those scales unchanged after that. Its performance is therefore rather limited. The method proposed in [23] uses a common scale parameter for all kernel functions. The method in [24] updates both scales and centers individually, as in the way of the proposed approach. The key difference is however that the method in [24] does not explicitly care the nonconvexity of the problem and initializes the scales of kernels to a single value. As will be shown in Section 4, this type of simple initialization strategy causes a serious tradeoff between the accuracy of estimation and the computational complexity. In contrast, the proposed approach yields a reasonably high estimation accuracy with a reasonable complexity for a wide range of initial conditions.

4. SIMULATION RESULTS

We show the efficacy of the proposed algorithm for system identification problems of two toy examples and a time series prediction problem of four real data. For the proposed algorithm, the dictionaries are constructed by the multiscale screening method presented



(a) (b) (c)

Fig. 5. Results of Experiment 3. Data 1: temperature, Data 2: humidity, Data 3: pressure, and Data 4: visibility.

in Sec. 3.2, and $\omega_{j,n} := (|h_{j,n}| + \beta)^{-1}$ [27] with $\beta = 10^{-4}$ is employed to the weighted ℓ_1 norm.

Experiment 1: A Basic Performance of the Proposed Algorithm

We consider the following nonlinear function $\psi(u) = 3\mathrm{e}^{-30(u-2.5)^2} - 5\mathrm{e}^{-40(u-5)^2} + 2\mathrm{e}^{-50(u-7.5)^2} + 2\mathrm{e}^{-0.25(u-4)^2}$, $u \in \mathbb{R}$, which is the sum of four Gaussian functions. The observed signal is generated as $d_n := \psi(u_n) + v_n$, $n \in \mathbb{N}$, where u_n is the input data randomly generated from a uniform distribution within the region $[0, 10]$ and $v_n \sim \mathcal{N}(0, 1.0 \times 10^{-2})$ is the additive white Gaussian noise.

To verify that the proposed algorithm adapts the kernel scale and center efficiently, the performance of the proposed algorithm is evaluated, and compared with the performance of the proposed algorithm without the adaptation of the kernel scales and centers. For the proposed algorithm with the adaptation of the kernel parameters, the three initial kernel scales $\xi_{\mathrm{init}}^{(1)} = 10$, $\xi_{\mathrm{init}}^{(2)} = 1/5$, and $\xi_{\mathrm{init}}^{(3)} = 1/100$ are roughly chosen, so that the range $[\xi_{\mathrm{init}}^{(3)}, \xi_{\mathrm{init}}^{(1)}]$ includes all the Gaussian scales of the target function, and $\xi_{\mathrm{init}}^{(2)}$ is the middle value of $[\xi_{\mathrm{init}}^{(3)}, \xi_{\mathrm{init}}^{(1)}]$. For the algorithm without the adaptation, $\xi_{\mathrm{init}}^{(1)} = 1$, $\xi_{\mathrm{init}}^{(2)} = 1/10$, and $\xi_{\mathrm{init}}^{(3)} = 1/50$ are chosen so that the algorithm achieves the best performance. The stepsizes and the regularization parameters of the proposed algorithm with the adaptation are chosen so that the dictionary size is close to the number of the Gaussians of the target and the steady-state MSE becomes as low as possible. The parameters for the algorithm without the adaptation are chosen so that the convergence rate of the MSE is almost same as the algorithm with the adaptation.

Fig. 3 depicts (a) the MSE and (b) the maximal dictionary size, and (c) the steady-state dictionary size averaged over 100 runs. The algorithm with the adaptation of the kernel scales and centers (Proposed) is superior to the algorithms without the adaptation (w/o adaptation of ξ and $\boldsymbol{c}$) in the sense of both the MSE and the dictionary size. This result shows the effectiveness of adapting the kernel parameters.

Experiment 2: Insensitivity to the Choice of Initial Kernel Scales

We consider the nonlinear function $\psi(u) = \mathrm{e}^{-\frac{(u-4)^2}{\xi^*}} - 0.5\mathrm{e}^{-\frac{(u-6)^2}{\xi^*}}$. with $\xi^* = 1$. The observed signal is generated as $d_n := \psi(u_n) + v_n$, $n \in \mathbb{N}$, where u_n is the input data randomly generated from a uniform distribution within the region $[0, 10]$ and $v_n \sim \mathcal{N}(0, 1.0 \times 10^{-2})$ is the additive white Gaussian noise.

To verify the proposed algorithm is insensitive to the selection of initial kernel scales, we demonstrate estimation performances of the proposed algorithm while varying the initial kernel scales, and compare the performance with a state-of-the-art algorithm that adapts the kernel scales and centers with single initial values [24] (see Sec. 3.6). For the single initialization algorithm, the following two settings of the kernel scales are considered: (small) $\xi_{\mathrm{init}} = \xi^*/\Delta\xi$ and (large) $\xi_{\mathrm{init}} = \xi^*\Delta\xi$ with $\Delta\xi = 10^0,\ 10^1,\ 10^2,\ 10^3$, and 10^4. For the proposed algorithms, the performance under the following two initial scales are tested: $\xi_{\mathrm{init}}^{(1)} = \xi^*\Delta\xi$ and $\xi_{\mathrm{init}}^{(2)} = \xi^*/\Delta\xi$ with $\Delta\xi$ defined above.

Fig. 4 illustrates the results of the experiment 2: (a) the MSEs, (b) the maximal and (c) the steady-state dictionary sizes averaged over 100 runs. From Fig. 4(a), one can see that the single initialization algorithm (large) attains the poor performance when ξ_{init} is large. Although the single initialization algorithm (small) is insensitive to the choice of ξ_{init}, the dictionary size greatly increases when ξ_{init} is large, as illustrated in Fig. 4(b). On the other hand, the proposed algorithm is insensitive to the choice of $[\xi_{\mathrm{init}}^{(1)}, \xi_{\mathrm{init}}^{(2)}]$ in the sense of both the MSE and the dictionary size.

Experiment 3: Real Data

We demonstrate the performance of the proposed algorithm in an application to online prediction of the following four data from the appliances energy prediction dataset which are available in *UCI Machine Learning Repository* [28]:

data 1: Temperature in kitchen area [Celsius]
data 2: Humidity in kitchen area [Celsius]
data 3: Pressure (from Chievres weather station) [mm Hg]
data 4: Visibility (from Chievres weather station) [km]

Each datum d_n is predicted with $\boldsymbol{u}_n := [d_{n-1}, d_{n-2}, \cdots, d_{n-L}]^\mathsf{T} \in \mathcal{U} \subset \mathbb{R}^L$ for $L = 8$. The proposed algorithm is compared with the kernel normalized least mean square (KNLMS) algorithm with ℓ_1 regularization [16], which is a benchmark algorithm in the field of kernel adaptive filtering, and the single initialization algorithm which have appeared in Experiment 2. For the proposed algorithm, the following three initial kernel scales are employed: $\xi_{\mathrm{init}}^{(1)} = 1$, $\xi_{\mathrm{init}}^{(2)} = 0.1$, and $\xi_{\mathrm{init}}^{(3)} = 0.01$. For the single initialization algorithm, the following three settings are tested: (i) $\xi_{\mathrm{init}} = 1$, (ii) $\xi_{\mathrm{init}} = 0.1$, and (iii) $\xi_{\mathrm{init}} = 0.01$, and then the value which attains the lowest MSE is chosen. For KNLMS, the kernel scale is chosen as $\xi = 1/50,\ 1/10,\ 1/6,\ 1/6$ for the data 1, 2, 3, 4, respectively, so that the algorithm achieves the best performance. Each algorithm operates a single run over the data. Fig. 5 summarizes the result: (a) the MSEs, (b) the maximal dictionary sizes, and (c) the mean dictionary sizes. The MSEs and the mean dictionary sizes are computed by averaging over all the iterations. We note that the dictionary sizes of all algorithms change dynamically.

It can be seen from Fig. 5 that the MSEs of the proposed algorithm are smaller than those of KNLMS and the single initialization algorithm for all data. Although the maximal dictionary sizes of the proposed algorithm is larger than KNLMS for data 2 and 3, the mean and steady-state dictionary sizes are smaller than KNLMS and the single initialization algorithm for all data, thanks to the adaptation of kernel parameters and the multiscale screening method.

5. CONCLUSION

This paper proposed an efficient kernel adaptive filtering algorithm. The proposed algorithm adapted the kernel parameters to construct an efficient filter. To avoid to fall into local minima, we presented a novel dictionary growing scheme, named the multiscale screening method. The multiscale screening method reduced the sensitivity for the initial kernel scales while maintaining small dictionary size. Numerical examples showed efficacy of the proposed algorithm with the insensitivity to the initial kernel parameters.

6. REFERENCES

[1] T. J. Dodd, V. Kadirkamanathan, and R. F. Harrison, "Function estimation in Hilbert space using sequential projections," in *IFAC Conf. Intell. Control Syst. Signal Process.*, 2003, pp. 113–118.

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

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

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

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

[6] K. Slavakis, S. Theodoridis, and I. Yamada, "Online kernel-based classification using adaptive projection algorithms," *IEEE Trans. Signal Processing*, vol. 56, no. 7, pp. 2781–2796, July 2008.

[7] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering*. New Jersey: Wiley, 2010.

[8] B. Chen, S. Zhao, P. Zhu, and J. C. Príncipe, "Quantized kernel least mean square algorithm," *IEEE Trans. Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 22–32, Dec. 2012.

[9] S. V. Vaerenbergh, M. Lazaro-Gradilla, and I. Santamaria, "Kernel recursive least-squares tracker for time-varying regression," *IEEE Trans. Neural Network and Learning Systems*, vol. 23, no. 8, pp. 1313–1326, Aug 2012.

[10] M. Takizawa and M. Yukawa, "Adaptive nonlinear estimation based on parallel projection along affine subspaces in reproducing kernel Hilbert space," *IEEE Trans. Signal Processing*, vol. 63, no. 16, pp. 4257–4269, Aug. 2015.

[11] ——, "Efficient dictionary-refining kernel adaptive filter with fundamental insights," *IEEE Trans. Signal Processing*, vol. 64, no. 16, pp. 4337–4350, Aug. 2016.

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

[13] ——, "Adaptive learning in cartesian product of reproducing kernel hilbert spaces," *IEEE Trans. Signal Processing*, vol. 63, no. 22, pp. 6037–6048, Nov. 2015.

[14] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, no. 3, pp. 337–404, May 1950.

[15] B. Schölkopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA: MIT Press, 2001.

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

[17] M. Yukawa and R. Ishii, "Online model selection and learning by multikernel adaptive filtering," in *Proc. EUSIPCO*, 2013, pp. 1–5.

[18] O. Toda and M. Yukawa, "Online model-selection and learning for nonlinear estimation based on multikernel adaptive filtering," *IEICE Trans. Fundamentals*, vol. 1, no. E100-A, pp. 236–250, Jan. 2017.

[19] A. Nemirovski and D. Yudin, *Problem complexity and Method Efficiency in Optimization*. Wiley, 1983.

[20] Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada, "A sparse adaptive filtering using time-varying soft-thresholding techniques," in *Proc. IEEE ICASSP*, 2010, pp. 3734–3737.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[22] B. Chen, J. Liang, N. Zheng, and J. C. Principe, "Kernel least mean square with adaptive kernel size," *Neurocomputing*, vol. 191, pp. 95–105, 2013.

[23] T. Wada and T. Tanaka, "Doubly adaptive kernel adaptive filtering," in *Proc. APSIPA*, 2017, tA-P3.6.

[24] T. Wada, K. Fukumori, and T. Tanaka, "Dictionary learning for gaussian kernel adaptive filtering with variablekernel center and width," in *Proc. IEEE ICASSP*, April 2018.

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

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

[27] M. Yukawa, Y. Tawara, M. Yamagishi, and I. Yamada, "Sparsity-aware adaptive filters based on lp-norm inspired soft-thresholding technique," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2012, pp. 2749–2752.

[28] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml