A Hilbertian Projection Approach with Dictionary Dividing Strategy: Accelerating Nonlinear Estimation Algorithm with Multiscale Gaussians

Masaaki Takizawa* and Masahiro Yukawa*
* Dept. Electronics and Electrical Engineering, Keio University, Japan
E-mail: takizawa@nc-toyama.ac.jp, yukawa@elec.keio.ac.jp

***Abstract*—A novel dictionary dividing scheme for online nonlinear estimation algorithms with multiscale Gaussians is proposed to perform a projection in an appropriate reproducing kernel Hilbert space. The proposed dictionary dividing strategy mitigates the inequivalence of the norm of multiscale Gaussians, which leads to degradations of adaptation speed for certain Gaussians. Based on a Hilbertian projection with the dictionary dividing strategy, a fast nonlinear estimation algorithm, which adapts scales and centers of Gaussians as well as its heights, is presented. The numerical example shows that using the Hilbertian projection with the proposed dictionary dividing scheme ameliorates the adaptation speed of Gaussian heights.**

I. Introduction

The problem of adaptive estimation of nonlinear functions appears in various fields of engineering. The accuracy of estimates depends on the choice of a nonlinear model. Thus, selecting an appropriate model is an important issue that has been actively studied in statistical inference. The Gaussian model is a commonly used model for nonlinear estimation tasks due to its generalization capabilities. Although Gaussian model has been succeeded in Gaussian processes [1], radial basis function (RBF) networks [2], kernel adaptive filtering [3], and multikernel adaptive filtering [4], their estimation performance depends heavily on the choice of parameters of the Gaussian function, such as scales and centers. One way to obtain such parameters is using batch methods [5–10] for training data. Unfortunately, this approach is inefficient since training data may have a different statistical property from test data, such as covariate shift and/or colored signals. It is therefore of great importance to develop methods to find appropriate Gaussian parameters adaptively.

Some related works have been studied in [11–13] to obtain appropriate parameters (scales and centers) in the Gaussian model. The methods in [14–16] adapt both the scales and centers to minimize the instantaneous squared error by iterative algorithms. However, the nonconvexity of the instantaneous squared error cost function implies that the solutions derived by iterative algorithms depend on the initial values of the parameters. Remarkably, the adaptation speed of the scales could be unacceptably slow when the initial scale is far from optimal. To alleviate the sensitivity to the initial conditions, a reasonable selection strategy for initial Gaussian scales is proposed in [16], employing multiple initial values for the Gaussian scales. In [15], a steepening scheme for the instantaneous squared error function is proposed to enhance the learning speed.

Let us turn our attention to the update of the Gaussian heights. In the context of kernel adaptive filtering with Gaussian kernel, updating algorithms for the heights of the Gaussians have been studied in [4, 17–27]. In [28], kernel adaptive filtering algorithms are classified into two approaches: (i) Euclidean-space approach and (ii) a reproducing kernel Hilbert space (RKHS) approach. It has been shown that the algorithms classified into the RKHS approach tend to enjoy a decorrelation property for error surface and thus yield faster convergence. Convergence speed is quite essential in adaptive/online learning, especially in the fields dealing with time-variant systems such as acoustic signal processing. Although the methods [15, 16] successfully obtained the appropriate scales and centers, the update of the Gaussian heights is based on the Euclidean-space approach, which means that there is room for improvement in its convergence speed.

To make a RKHS-approach-based algorithm for updating the estimate with multiscale Gaussians practical, it is key to select an appropriate RKHS where the projection is performed. The selection is nontrivial since the Gaussian model may include multiscale Gaussians with a wide range of scales due to the adaptations of the Gaussian parameters. In this paper, a novel dictionary dividing strategy to adapt the heights of multiscale Gaussians, each in an appropriate RKHS, is proposed. The idea of the proposed dictionary dividing strategy is to consider multiple RKHSs and allocate Gaussian functions to each RKHS so that some requirements obtained from the norm between two Gaussians with different scales are satisfied. Based on the RKHS projection with the dictionary dividing strategy, a fast nonlinear estimation algorithm with the adaptations of Gaussian parameters (scales and centers) is presented. As revealed by computer experiments, the algorithm with the proposed dictionary dividing strategy enjoys fast adaptation speed as well as reasonable adaptation of Gaussian parameters.

II. Preliminary

1) Problem Settings: Let $\mathbb{R}^L$ and $\mathbb{N}$ be the L-dimensional Euclidean space and the set of nonnegative integers, respectively. We estimate a nonlinear function $\psi : \mathbb{R}^L \to \mathbb{R}$ with sequentially arriving input signals $\boldsymbol{u}_n \in \mathcal{U} \subset \mathbb{R}^L$, and its

noisy output $d_n := \psi(\boldsymbol{u}_n) + \nu_n \in \mathbb{R}$, where $n \in \mathbb{N}$ is the time instant, $\boldsymbol{u}_n$ is assumed an i.i.d. random vector, and ν_n is an additive noise. Gaussian function is defined as

$$g(\boldsymbol{u}; \xi, \boldsymbol{c}) := \exp\left(-\frac{\|\boldsymbol{u}-\boldsymbol{c}\|^2}{2\xi}\right), \tag{1}$$

where $\xi > 0$ is the scale and $\boldsymbol{c} \in \mathbb{R}^L$ is the center of the Gaussian. Here, $\|\cdot\|$ denotes the standard Euclidean norm.

The nonlinear function ψ is estimated by a weighted sum of the r Gaussian functions:

$$g(\cdot; \xi^{(j)}, \boldsymbol{c}^{(j)}),\ j = 1, \cdots, r, \tag{2}$$

where, $\xi^{(j)} > 0$ and $\boldsymbol{c}^{(j)} \in \mathbb{R}^L$. For the visibility of the paper, we use the shorthand notation for the Gaussian function:

$$g^{(j)}(\cdot) := g(\cdot; \xi^{(j)}, \boldsymbol{c}^{(j)}). \tag{3}$$

A. Estimation Model

In this work, we assume any prior knowledge of the target function ψ is unavailable, and thus appropriate numbers and the parameters (scales and centers) of Gaussians for reasonable estimation are also unavailable. To perform estimation with appropriate number of Gaussians which have appropriate parameters, the numbers and parameters of Gaussians are changed in the process of estimation as well as the heights (coefficients) of Gaussians. Our time-varying model is thus given by

$$\varphi_n(\boldsymbol{u}) := \sum_{j=1}^{r_n} h_n^{(j)} g_n^{(j)}(\boldsymbol{u}), \tag{4}$$

with the shorthand notation for the Gaussian functions

$$g_n^{(j)}(\cdot) := g(\cdot; \xi_n^{(j)}, \boldsymbol{c}_n^{(j)}). \tag{5}$$

Here, $\xi_n^{(j)} > 0$, $\boldsymbol{c}_n^{(j)} \in \mathbb{R}^L$, and $h_n^{(j)} \in \mathbb{R}$ are the scale, center, and height, respectively, for the j-th Gaussian at time instant n. The dictionary $\mathcal{D}_n$ is defined as the set of r_n Gaussian functions:

$$\mathcal{D}_n := \{g_n^{(j)}\}_{j \in \{1,2,\cdots,r_n\}}. \tag{6}$$

B. Update of Gaussian Heights in RKHS

Update schemes for the Gaussian heights $h_n^{(j)}$ in the model (4) have been studied in the field of kernel adaptive filtering under the use of the Gaussian kernel

$$\kappa_{\xi_\kappa}(\boldsymbol{u}, \boldsymbol{v}) := \exp\left(-\frac{\|\boldsymbol{u}-\boldsymbol{v}\|^2}{2\xi_\kappa}\right),\ \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^L, \tag{7}$$

where $\xi_\kappa > 0$ is the scale parameter. In [28], kernel adaptive filtering algorithms are classified into two classes according to space where the algorithms are formulated: (i) the space of coefficient vectors and (ii) a reproducing kernel Hilbert space (RKHS). It has been shown by simulations and a theoretical aspect that the RKHS-type algorithms tend to enjoy the decorrelation property of error surfaces and thus yield faster convergence than the algorithms formulated in the space of coefficient vectors [28, 29]. Motivated by this study, the updating algorithm for the coefficients $h_n^{(j)}$ in (4) is formulated in an RKHS.

Let $\mathcal{H}_{\xi_\kappa}$ be the RKHS induced by the Gaussian kernel (7). For the RKHS $\mathcal{H}_{\xi_\kappa}$, the following fact is known.

Fact 1 ([30]): The Gaussian function $g(\cdot; \xi, \boldsymbol{u}),\ \boldsymbol{u} \in \mathbb{R}^L$ with the scale $\frac{\xi_\kappa}{2} < \xi$ is contained in the RKHS $\mathcal{H}_{\xi_\kappa}$.

Under the assumption $\frac{\xi_\kappa}{2} < \xi^{(j)},\ \forall j = 1, \cdots, r_n$, the set $\{g_n^{(j)}\}_{j \in \{1,\cdots,r_n\}}$ spans the dictionary subspace

$$\mathcal{M}_n := \operatorname{span}\{g_n^{(j)}\}_{j \in \{1,\cdots,r_n\}} \tag{8}$$

of the RKHS $\mathcal{H}_{\xi_\kappa}$.

We define the metric projection onto the nonempty closed convex set of a real Hilbert space.

Definition 1: Let $\mathcal{X}$ be a real Hilbert space equipped with a norm $\|\cdot\|_{\mathcal{X}}$. Then, the metric projection of a point $x \in \mathcal{X}$ onto a nonempty closed convex set $\mathcal{K} \subset \mathcal{X}$ is defined as

$$P_{\mathcal{K}}(x) := \operatorname*{argmin}_{y \in \mathcal{K}} \|x - y\|_{\mathcal{X}}. \tag{9}$$

The proposed updating scheme for the heights $h_n^{(j)}$ is based on the projection onto the dictionary subspace $\mathcal{M}_n$. Specifically, the Gaussian heights are updated in the direction of the projection $P_{\mathcal{M}_n}(\kappa_{\xi_\kappa}(\cdot, \boldsymbol{u}_n))$ in the RKHS $\mathcal{H}_{\xi_\kappa}$. Note that $\kappa_{\xi_\kappa}(\cdot, \boldsymbol{u}_n)$ is the normal vector of the zero-instantaneous-error hyperplane $\Pi_n := \{f \in \mathcal{H}_{\xi_\kappa} : d_n - f(\boldsymbol{u}_n) = 0\}$. For more details about the projection onto the dictionary subspace, see [22].

III. Proposed Algorithm

A. Inner Product of Two Gaussians

To compute the projection onto the dictionary subspace $\mathcal{M}_n$, there are two problems to be addressed:

1) How to compute the inner product between Gaussians in the RKHS $\mathcal{H}_{\xi_\kappa}$ (answered in Theorem 1).
2) How to select the RKHS $\mathcal{H}_{\xi_\kappa}$ where the algorithm is formulated; i.e., how to select the scale parameter ξ_κ of the Gaussian kernel (7) (answered in Scheme 1).

To address the problems, let us start with the following lemma:

Lemma 1 ([31]): Let $C(\mathbb{R}^L)$ be the set of differentiable functions, $L_p(\mathbb{R}^L),\ p > 0$ be the set of p-th power integrable functions. Suppose $f_\kappa \in C(\mathbb{R}^L) \cap L_1(\mathbb{R}^L) : \mathbb{R}^L \to \mathbb{R}$ is a real-valued positive definite function. Define

$$\mathcal{H} := \left\{ f \in L_2(\mathbb{R}^L) \cap C(\mathbb{R}^L) : \frac{\hat{f}}{\sqrt{\hat{f}_\kappa}} \in L_2(\mathbb{R}^L) \right\}, \tag{10}$$

and the inner product can be defined by

$$\langle f_1, f_2 \rangle_{\mathcal{H}} := (2\pi)^{-L/2} \int_{\mathbb{R}^L} \frac{\hat{f}_1(\boldsymbol{t})\overline{\hat{f}_2}(\boldsymbol{t})}{\hat{f}_\kappa(\boldsymbol{t})} d\boldsymbol{t},\ \boldsymbol{t} \in \mathbb{R}^L, \tag{11}$$

where $\hat{f}$ denotes the Fourier transform of the function f. Then $\mathcal{H}$ is a real Hilbert space with the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and a reproducing kernel $\kappa(\cdot, \cdot) := f_\kappa(\cdot - \cdot)$.

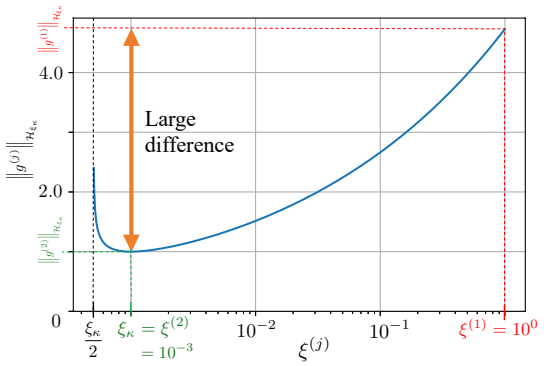


Fig. 1. The norm $\left\|g^{(j)}\right\|_{\mathcal{H}_{\xi_\kappa}}$ with $\xi^{(1)} = 10^0$ and $\xi_\kappa = \xi^{(2)} = 10^{-3}$.

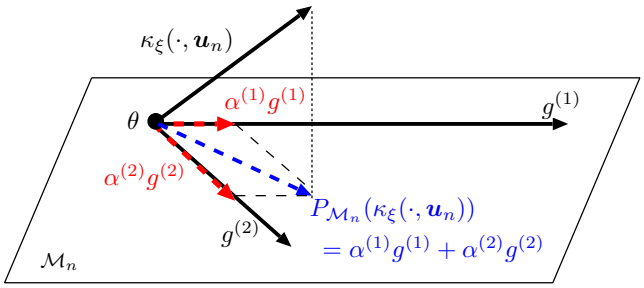


Fig. 2. Projection onto the two-dimensional dictionary subspace $\mathcal{M}_n$ with $\xi^{(1)} \gg \xi^{(2)} > \xi_\kappa$. Due to the large norm $\left\|g^{(1)}\right\|_{\mathcal{H}_{\xi_\kappa}}$, the coefficient $\alpha^{(1)} \in \mathbb{R}$ of the projection has a smaller value than $\alpha^{(2)} \in \mathbb{R}$; i.e, $\alpha^{(1)} \ll \alpha^{(2)}$.

The Fourier transform of the Gaussian function is given in the following lemma.

Lemma 2: The Fourier transform $\hat{g}$ of the Gaussian function g is given by

$$\hat{g}(\boldsymbol{t}; \xi, \boldsymbol{c}) = \xi^{L/2} \exp\left(-\frac{\xi \|\boldsymbol{t}\|^2}{2}\right) \exp\left(-i\boldsymbol{c}^\mathsf{T}\boldsymbol{t}\right), \ \boldsymbol{t} \in \mathbb{R}^L, \tag{12}$$

where $i := \sqrt{-1}$.

Proof: See Appendix A. ■

The Gaussian kernel (7) can be defined by the Gaussian function: $\kappa_{\xi_\kappa}(\cdot, \cdot) := g(\cdot - \cdot; \xi_\kappa, \boldsymbol{0})$. For the function $\frac{\hat{g}(\cdot; \xi, \boldsymbol{c})}{\sqrt{\hat{g}(\cdot; \xi_\kappa, \boldsymbol{0})}}$, the following Lemma can be verified.

Lemma 3: Under the assumption $\xi > \frac{\xi_\kappa}{2}$, it holds that

$$\frac{\hat{g}(\cdot; \xi, \boldsymbol{c})}{\sqrt{\hat{g}(\cdot; \xi_\kappa, \boldsymbol{0})}} \in L_2(\mathbb{R}^L). \tag{13}$$

Proof: See Appendix B. ■

The following theorem then can be verified by Lemma. 1, 2, and 3.

Theorem 1: In the RKHS $\mathcal{H}_{\xi_\kappa}$, the inner product of two Gaussians $g^{(1)}$ and $g^{(2)}$ with the scales $\xi^{(1)}, \xi^{(2)} > \frac{\xi_\kappa}{2}$ is given by

$$\left\langle g^{(1)}, g^{(2)} \right\rangle_{\mathcal{H}_{\xi_\kappa}} = \sqrt{\frac{\xi^{(1)}\xi^{(2)}}{\xi_\kappa(\xi^{(1)} + \xi^{(2)} - \xi_\kappa)}} \exp\left(-\frac{\left\|\boldsymbol{c}^{(1)} - \boldsymbol{c}^{(2)}\right\|^2}{2(\xi^{(1)} + \xi^{(2)} - \xi_\kappa)}\right). \tag{14}$$

Proof: See Appendix C. ■

B. Motivations for Dictionary Dividing

1) Norm of Gaussian Functions: The following result is immediately available from (14) in Theorem 1.

Corollary 1: For the Gaussian function with the scale $\xi > \frac{\xi_\kappa}{2}$, its norm

$$\|g(\cdot; \xi, \boldsymbol{u})\|_{\mathcal{H}_{\xi_\kappa}} = \frac{\xi^{\frac{1}{2}}}{\{\xi_\kappa(2\xi - \xi_\kappa)\}^{\frac{1}{4}}} \tag{15}$$

has the minimum value 1 at $\xi = \xi_\kappa$ and is a monotonically increasing for $|\xi - \xi_\kappa|$.

Figure 1 shows the norm $\left\|g^{(j)}\right\|_{\mathcal{H}_{\xi_\kappa}}$ with $\xi^{(1)} = 10^0$ and $\xi_\kappa = \xi^{(2)} = 10^{-3}$. Due to the monotonically increasing property presented in Corollary 1, the norm $\left\|g^{(1)}\right\|_{\mathcal{H}_{\xi_\kappa}}$ has a larger value than $\left\|g^{(2)}\right\|_{\mathcal{H}_{\xi_\kappa}}$ which has the minimum value 1.

2) Projection onto Dictionary Subspace: Figure 2 illustrates the projection onto the two-dimensional dictionary subspace $\mathcal{M}_n$ with the Gaussian scales $\xi^{(1)} \gg \xi^{(2)} > \xi_\kappa$. Since $\xi^{(1)} \gg \xi^{(2)}$, the norm $\left\|g^{(1)}\right\|_{\mathcal{H}_{\xi_\kappa}}$ is larger than $\left\|g^{(2)}\right\|_{\mathcal{H}_{\xi_\kappa}}$ (see Corollary 1). The large norm $\left\|g^{(1)}\right\|_{\mathcal{H}_{\xi_\kappa}}$ reduces the rate of the update in the direction of $g^{(1)}$, and a large step size is therefore required to provide sufficient update rate for $g^{(1)}$. However, such a large step size causes instability of the algorithm since the rate of the update in the direction of $g^{(2)}$ is significantly larger than the rate of $g^{(1)}$. On the other hand, a small step size that tends to stabilize the algorithm slows down the algorithm due to the small update rate in the direction of $g^{(1)}$. The slow growth of Gaussians with large scales, moreover, causes the large dictionary size since many Gaussians with small scales are required to reduce the error.

3) Selection of the RKHS $\mathcal{H}_{\xi_\kappa}$: As discussed in the previous subsection, differences of the norms have a negative impact in terms of learning speed. From the above discussion and Fact 1, the following desirable properties for the RKHS $\mathcal{H}_{\xi_\kappa}$ are obtained:

(a) $\frac{\xi_\kappa}{2} < \xi^{(j)}, \ \forall j \in \{1, \cdots, r_n\}$ (see Fact 1).
(b) $\xi^{(j)}, \ \forall j \in \{1, \cdots, r_n\}$ should not be extremely larger than ξ_κ.

However, it is unrealistic to select an RKHS $\mathcal{H}_{\xi_\kappa}$ so that satisfying the above properties in practical situations, since the scales may be widely distributed due to the adaptations for the Gaussian parameters. In this study, we address the above problem with a novel design scheme for the dictionary, which will be presented in the next subsection.

C. Dictionary Dividing Strategy

The idea of the proposed design scheme is to consider multiple RKHSs $\mathcal{H}_{\xi_\kappa^{(1)}}, \mathcal{H}_{\xi_\kappa^{(2)}}, \cdots, \mathcal{H}_{\xi_\kappa^{(Q)}}$ and allocate Gaussian

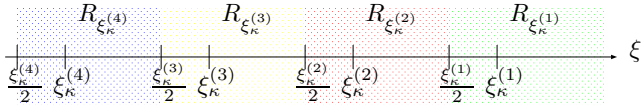


Fig. 3. The segmentation of the Gaussian scales with $Q = 4$. The axis is in logarithmic scale.

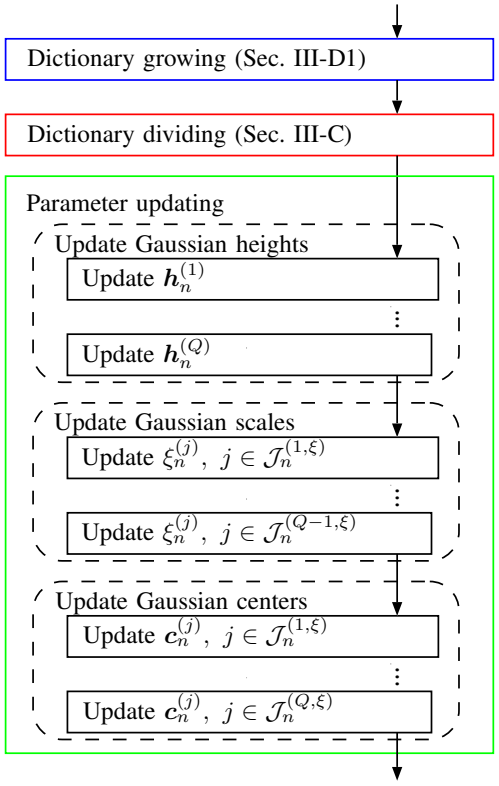


Fig. 4. A block diagram of the proposed algorithm.

functions to each RKHS so that the properties presented in Section III-B3 hold. To realize this idea, the dictionary $\mathcal{D}_n$ is divided into Q subsets $\mathcal{D}^{(1)}, \mathcal{D}^{(2)}, \cdots, \mathcal{D}^{(Q)}$ on the scales basis, and the coefficients of Gaussians which are in the q-th dictionary $\mathcal{D}^{(q)}$ are updated in the RKHS $\mathcal{H}_{\xi_\kappa^{(q)}}$. The proposed dictionary dividing strategy is given below.

Scheme 1 (Dictionary Dividing Strategy): Set Q scales $\xi_\kappa^{(1)} > \xi_\kappa^{(2)} > \cdots > \xi_\kappa^{(Q)} > 0$ s.t. $\frac{\xi_\kappa^{(q)}}{2} > \xi_\kappa^{(q+1)}$, $q = 1, \cdots, Q-1$. Then the Q ranges $R^{(1)}, \cdots, R^{(Q)}$ are defined as $R^{(1)} := (\frac{\xi_\kappa^{(1)}}{2}, +\infty)$ and $R^{(q)} := (\frac{\xi_\kappa^{(q)}}{2}, \frac{\xi_\kappa^{(q-1)}}{2}]$, $q = 2, \cdots, Q$. The divided dictionaries are defined as $\mathcal{D}_n^{(q)} := \{g_n^{(j)}\}_{j \in \mathcal{J}_n^{(q)}}$, where $\mathcal{J}_n^{(q)} := \{j_n^{(q,1)}, j_n^{(q,2)}, \cdots, j_n^{(q,r_n^{(q)})}\}$, s.t. $\forall \xi_n^{(j)} \in R^{(q)},\ j \in \mathcal{J}_n^{(q)}$.

Figure 3 shows the segmentation of the Gaussian scales with $Q = 4$.

D. The Proposed Algorithm

Figure 4 shows a block diagram of the proposed algorithm. The proposed algorithm consists of three blocks: (i) the dictionary growing block, (ii) the dictionary dividing block, and (ii) the parameter updating block. In the first block, the dictionary is initialized to an empty set ($r_0 := 0$), and it grows under a selection strategy based on the coherence criterion using the sequentially coming data. In the second block, dictionary dividing is performed according to Scheme. 1. In the third block, the parameters (heights, scales, and centers) are updated in this order. The parameters of the Gaussians in $\mathcal{D}_n^{(1)}$ are updated first. Then, those of the Gaussians in $\mathcal{D}_n^{(2)}$ are updated, and so on. This order comes from the idea of updating Gaussian functions with large scales preferentially to reduce the error over a wide range rapidly. For more details about the updating order of the Gaussian parameters, see [16]. To prevent the scales from becoming smaller than $\frac{\xi^{(q)}}{2}$, the scales of Gaussians in $\mathcal{D}_n^{(Q)}$ are not updated.

Each step will be described below.

1) Dictionary Growing: Under the online setting considered in this paper, an adequate number of Gaussians and the range of Gaussian centers are unknown prior to estimation. The proposed algorithm thus starts the estimation with the empty dictionary $\mathcal{D}_0 := \emptyset$ and a new Gaussian with the predefined initial Gaussian scale $\xi_{\text{init}} > 0$ enters to dictionary based on the coherence criterion [32] to keep the dictionary within a reasonable size.

2) Parameters Updating:

Selective updating strategy: When $g_n^{(j)}(\boldsymbol{u}_n)$ is nearly zero, the updates of the associated Gaussian parameters may not largely affect the estimation. To reduce the computational costs for updating the heights, $s_n^{(q,h)}$ Gaussians out of $r_n^{(q)}$ Gaussians are selected for each dictionary $\mathcal{D}_n^{(q)}$, and then associated heights are updated. The idea of the selection strategy is the following: select a few Gaussians $\{g_n^{(j)}\}_{j \in \mathcal{J}_n^{(q)}}$ that are most relevant to the estimate at the current input $\boldsymbol{u}_n$, where $\mathcal{J}_n^{(q,h)} := \{j_n^{(1,q)}, j_n^{(2,q)}, \cdots, j_n^{(s_n^{(q,h)},q)}\} \subset \mathcal{J}_n^{(q)}$ for the number $s_n^{(q,h)} (\leq r_n^{(q)})$ of selected elements at time n. The same applies to the scales and centers, for which the number of selected Gaussians is denoted by $s_n^{(q,\xi)}$ and $s_n^{(q,c)}$, respectively.

Update of Gaussian parameters: The Gaussian heights are updated in the direction of the projection $P_{\mathcal{M}_n^{(q)}}(\kappa_{\xi^{(q)}}(\cdot, \boldsymbol{u}_n))$ in the RKHS $\mathcal{H}_{\xi^{(q)}}$. Here, $\mathcal{M}_n^{(q)}$ is the subspace defined by

$$\mathcal{M}_n^{(q)} := \text{span}\{g_n^{(j)}(\cdot)\}_{j \in \mathcal{J}_n^{(q)}}. \tag{16}$$

To exclude from the dictionary adaptively redundant/obsolete Gaussians that make no contribution to the estimation without causing serious performance degradations, soft thresholding and dictionary pruning are performed after the update (see the operator T_λ in Table I).

For the Gaussian scales and centers, gradient update is performed for the squared error function. At time instant n, the parameters (scales, and centers of r_n Gaussians) can be expressed by vectors and a matrix, respectively, as $\boldsymbol{\xi} := [\xi^{(1)}, \xi^{(2)}, \cdots, \xi^{(r_n)}]^\mathsf{T} \in \mathbb{R}_{++}^{r_n}$ and $\boldsymbol{C} :=$

TABLE I
THE PROPOSED ALGORITHM.

Initialization: $\mathcal{D}_n := \emptyset$.
Start nth iteration: Observe $\boldsymbol{u}_n$ and d_n.
1) Dictionary Growing :
If $\max_{j\in\{1,\cdots,r_n\}} c(\xi_n^{(j)}, \boldsymbol{c}_n^{(j)}, \xi_{\text{init}}, \boldsymbol{u}_n) \leq \delta \in [0,1]$, $g_n^{(r_n+1)}(\cdot)$ enters $\mathcal{D}_n$ with $(h_n^{(r_n+1)}, \xi_n^{(r_n+1)}, \boldsymbol{c}_n^{(r_n+1)}) := (0, \xi_{\text{init}}, \boldsymbol{u}_n)$.
Here, $c(\xi_u, \boldsymbol{u}, \xi_v, \boldsymbol{v}) := \left| \frac{\langle g(\cdot;\xi_u,\boldsymbol{u}), g(\cdot;\xi_v,\boldsymbol{v})\rangle_{\mathcal{H}_{\xi_{\text{init}}}}}{\|g(\cdot;\xi_u,\boldsymbol{u})\|_{\mathcal{H}_{\xi_{\text{init}}}} \|g(\cdot;\xi_v,\boldsymbol{v})\|_{\mathcal{H}_{\xi_{\text{init}}}}} \right|$ is the coherence.
2) Gaussian heights update :
for $q = 1 : Q$
(a) Construct $\mathcal{J}_n^{(q,h)} = \{j_n^{(q,1)}, \cdots, j_n^{(q,s_n^{(q,h)})}\} \in \mathcal{J}_n^{(q)}$ s.t. $g_n^{(k)}(\boldsymbol{u}_n) \geq g_n^{(j)}(\boldsymbol{u}_n)$, $\forall k \in \mathcal{J}_n^{(q,h)}$ and $j \in \mathcal{J}_n^{(q)} \setminus \mathcal{J}_n^{(q,h)}$.
(b) Update the coefficients by $\tilde{\boldsymbol{h}}_n^{(q)} \leftarrow \tilde{\boldsymbol{h}}_n^{(q)} - \mu_h \boldsymbol{\alpha}_n^{(q)}$ and $\boldsymbol{h}_n^{(q)} \leftarrow T_\lambda\left(\hat{\boldsymbol{h}}_n^{(q)}\right)$, where $\boldsymbol{\alpha}_n^{(q)} = {\boldsymbol{G}_n^{(q)}}^{-1} \boldsymbol{g}_n^{(q)}$
with $\boldsymbol{g}_n^{(q)} := [g_n^{(j_n^{(q,1)})}(\boldsymbol{u}_n), \cdots, g_n^{(j_n^{(q,s_n^{(q,h)})})}(\boldsymbol{u}_n)]^\mathsf{T} \in \mathbb{R}^{s_n^{(q,h)}}$ and $[G_n^{(q)}]_{k,l} := \left\langle g_n^{(j_n^{(q,k)})}, g_n^{(j_n^{(q,l)})} \right\rangle_{\mathcal{H}_{\xi_\kappa^{(q)}}}$, $k,l \in \{1,\cdots,s_n^{(q,h)}\}$.
Here, $\mu_h > 0$, $\tilde{\boldsymbol{h}}_n^{(q)} := [h_n^{(j_n^{(q,1)})}, \cdots, h_n^{(j^{(q,s_n^{(q,h)})})}]^\mathsf{T}$, $\hat{\boldsymbol{h}}_n^{(q)} := \begin{cases} \tilde{h}^{(q,j)}, & j \in \mathcal{J}_n^{(q)} \\ h^{(q,j)}, & \text{otherwise.} \end{cases}$
and T_λ with $\lambda > 0$ is the soft thresholding operator
which is given by $[T_\lambda(\boldsymbol{h})]_j := \begin{cases} \text{sgn}(h^{(j)})(|h^{(j)}| - \lambda), & |h^{(j)}| \geq \lambda \\ 0 & |h^{(j)}| < \lambda. \end{cases}$
Coefficients which are zero and their corresponding Gaussians are eliminated from the dictionary.
end
3) Gaussian scales update :
for $q = 1 : Q-1$
(a) Construct $\mathcal{J}_n^{(q,\xi)} = \{j_n^{(q,1)}, \cdots, j_n^{(q,s_n^{(q,\xi)})}\} \in \mathcal{J}_n^{(q)}$ s.t. $g_n^{(j)}(\boldsymbol{u}_n) \geq g_n^{(k)}(\boldsymbol{u}_n)$, $\forall k \in \mathcal{J}_n^{(q,\xi)}$ and $j \in \mathcal{J}_n^{(q)} \setminus \mathcal{J}_n^{(q,\xi)}$.
(b) Select the index $j \in \mathcal{J}_n^{(q,\xi)}$ s.t. $g_n^{(j)}(\boldsymbol{u}_n) \geq g_n^{(k)}(\boldsymbol{u}_n)$, $\forall k \in \mathcal{J}_n^{(q,\xi)}$.
(c) Update $\xi_n^{(j)}$ by $\xi_n^{(j)} \leftarrow \xi_n^{(j)} \exp\left(-\mu_\xi \frac{\partial F_n}{\partial \xi^{(j)}}(\boldsymbol{\xi}_n, \boldsymbol{C}_n)\right)$ with $\mu_\xi > 0$ and remove the index j from $\mathcal{J}_n^{(q,\xi)}$.
(d) Repeat (b) and (c) until $\mathcal{J}_n^{(q,\xi)}$ becomes empty set.
end
4) Gaussian centers update :
for $q = 1 : Q$
(a) Construct $\mathcal{J}_n^{(q,c)} = \{j_n^{(q,1)}, \cdots, j_n^{(q,s_n^{(q,c)})}\} \in \mathcal{J}_n^{(q)}$ s.t. $g_n^{(k)}(\boldsymbol{u}_n) \geq g_n^{(j)}(\boldsymbol{u}_n)$, $\forall k \in \mathcal{J}_n^{(q,c)}$ and $j \in \mathcal{J}_n^{(q)} \setminus \mathcal{J}_n^{(q,c)}$.
(b) Select the index $j \in \mathcal{J}_n^{(q,c)}$ s.t. $g_n^{(j)}(\boldsymbol{u}_n) \geq g_n^{(k)}(\boldsymbol{u}_n)$, $\forall k \in \mathcal{J}_n^{(q,c)}$.
(c) Update $\boldsymbol{c}_n^{(j)}$ by $\boldsymbol{c}_n^{(j)} \leftarrow \boldsymbol{c}_n^{(j)} - \mu_c \frac{\partial F_n}{\partial \boldsymbol{c}^{(j)}}(\boldsymbol{\xi}_n, \boldsymbol{C}_n)$ with $\mu_c > 0$ and remove the index j from $\mathcal{J}_n^{(q,c)}$.
(d) Repeat (b) and (c) until $\mathcal{J}_n^{(q,c)}$ becomes empty set.
end
5) Dictionary Dividing : See Section III-C.
end

$[\boldsymbol{c}^{(1)}\ \boldsymbol{c}^{(2)}\ \cdots\ \boldsymbol{c}^{(r_n)}] \in \mathbb{R}^{L\times r_n}$. The instantaneous squared error function is then given by

$$F_n(\boldsymbol{\xi}, \boldsymbol{C}) := \frac{1}{2}(d_n - \varphi_n(\boldsymbol{u}_n))^2. \tag{17}$$

To keep the scale parameters positive, the multiplicative gradient update for the squared error function is employed. The standard gradient update is performed for the centers.

Tables I summarizes the proposed nonlinear estimation algorithm.

The computational complexity of the proposed algorithms at each time instant $n \in \mathbb{N}$ is generally given in terms of the dictionary size r_n as well as the dimension L of the input space $\mathcal{U}$. The computational complexity of the proposed algorithm depends also on the number Q of the divided dictionaries and the cardinalities $\left|\mathcal{J}_n^{(q,h)}\right|$, $\left|\mathcal{J}_n^{(q,\xi)}\right|$, and $\left|\mathcal{J}_n^{(q,c)}\right|$. Typically, $\left|\mathcal{J}_n^{(q,h)}\right|$, $\left|\mathcal{J}_n^{(q,\xi)}\right|$, and $\left|\mathcal{J}_n^{(q,c)}\right|$ are constructed so that their cardinalities are less than 3, and thus the computational complexity of the proposed algorithm can be kept reasonably low even though the dictionary size r_n becomes significantly large. In the next section, we show that the proposed algorithm yields reasonable estimation performances even though only a few Gaussian parameters are updated for each divided dictionaries.

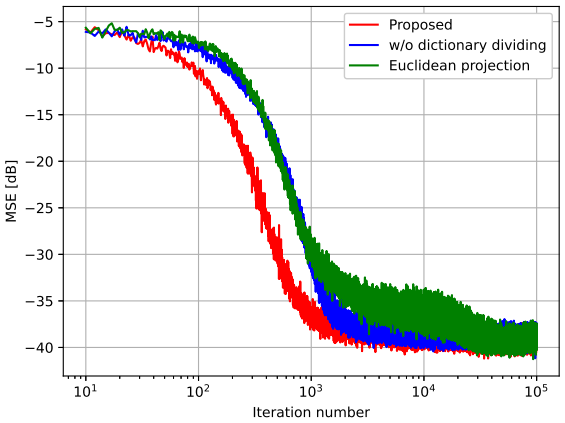
IV. SIMULATION RESULTS

We show the efficacy of the proposed nonlinear estimation algorithm for system identification problems of a toy example. We consider the nonlinear function $\psi(u) = \sum_{i=1}^3 h_i^* \exp\left(-\frac{|u-c_i^*|^2}{2\xi_i^*}\right)$, which is the weighted sum of three Gaussian functions with $h_1^* = -1$, $h_2^* = -1$, $h_3^* = 1$, $\xi_1^* = 10^{-2}$, $\xi_2^* = 5$, $\xi_3^* = 50$, $c_1^* = 15$, $c_2^* = 8$, and $c_3^* = 10$. The observed signal is generated as $d_n := \psi(u_n) + \nu_n$, $n \in \mathbb{N}$, where u_n is the input data of which each element is randomly generated from a uniform distribution within the region $[0, 20]$ and $\nu_n \sim \mathcal{N}(0, 5.0 \times 10^{-2})$ is the additive white Gaussian noise.

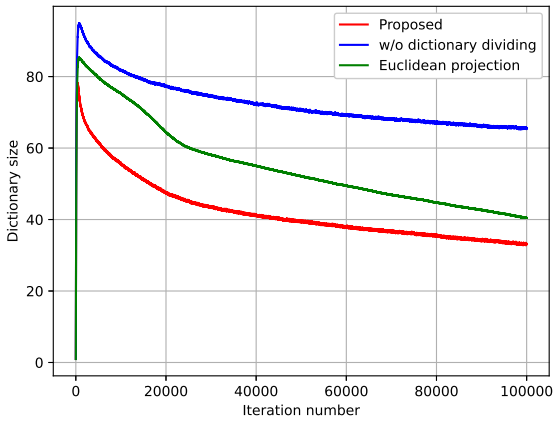
To show that the proposed algorithm improves the learning speed of the algorithm, the performance of the algorithm is

evaluated and compared with the performance of the algorithm with the Euclidean projection: i,e, $\boldsymbol{G}_n^{(q)} = \boldsymbol{I},\ q = 1, \cdots, Q$ is employed for the update of the Gaussian heights, where $\boldsymbol{I}$ is the identity matrix. Moreover, to show the efficacy of the dictionary dividing strategy, the proposed algorithm is compared with the proposed algorithm without the dictionary dividing block. Note that, for the algorithm without the dictionary dividing, the learning of the scale $\xi^{(j)}$ stops when $\xi_n^{(j)} < \xi_{\text{init}}$. The scales of the Gaussian kernels are $\xi_\kappa^{(1)} = 10^1$, $\xi_\kappa^{(2)} = 10^0$, $\xi_\kappa^{(3)} = 10^{-1}$, and $\xi_\kappa^{(4)} = 10^{-2}$. Initial Gaussian scales is $\xi_{\text{init}} = \xi^{(3)}$. Every time instant n, the proposed algorithm updates three Gaussian heights, one Gaussian scale, and one Gaussian center for each divided dictionary $\mathcal{D}_n^{(q)}$; i.e., $s_n^{(q,h)} = 3$, $s_n^{(q,\xi)} = 1$, and $s_n^{(q,c)} = 1,\ \forall q \in \{1, \cdots, Q\}$. For the algorithm with the Euclidean projection, the numbers of the updated parameters are selected so that the computational complexity averaged over the simulation is close to the complexity of the proposed algorithm; $s_n^{(q,h)} = 3$, $s_n^{(q,\xi)} = 3$, and $s_n^{(q,c)} = 3,\ \forall q \in \{1, \cdots, Q\}$. For the algorithm without the dictionary dividing, those parameters are selected so that the numbers of the updated parameters at every iteration are equal to the proposed algorithm; $s_n^{(h)} = 12$, $s_n^{(\xi)} = 4$, and $s_n^{(c)} = 4$. The step sizes are selected so that the steady-state errors are close to each other for all algorithms. The other parameters are chosen so that the errors are nearly identical to the proposed algorithm with the dictionary dividing at the steady state. The results are averaged over 200 runs.

Fig. 5 gives the (a) MSEs and (b) dictionary sizes. The results show that the proposed algorithm with the dictionary dividing strategy is superior to the algorithm without the dictionary dividing and the algorithm with the Euclidean projection in the sense of both the MSE and the dictionary size. Fig. 5(a) shows that the learning speed of the algorithm with the Euclidean projection slows down compared with the proposed algorithm due to the ellipsoidal error surface (for more details, see [28]). Moreover, the convergence speed of the algorithm without dictionary dividing is slower than the proposed algorithm due to the small update rate of large-scale Gaussians. The proposed algorithm enjoys faster convergence thanks to the decorrelation property of the RKHS approach and the proposed dictionary dividing strategy. Fig. 5(b) shows the dictionary sizes of the algorithm without dictionary dividing is larger than other algorithms, since a lot of Gaussians with small scales are required to reduce the error.



(a)



(b)

Fig. 5. Experimental results: (a) MSE and (b) dictionary size

V. Conclusions

In this paper, we proposed a fast nonlinear estimation algorithm based on the RKHS projection and the adaptations of Gaussian parameters. A novel dictionary dividing strategy was proposed by focusing on the property of the norm of the Gaussian functions in an RKHS. Thanks to the novel dictionary dividing strategy, the Gaussian heights are updated in appropriate RKHSs. As revealed by computer experiments, the proposed algorithm enjoys fast learning speed thanks to the RKHS-projection and the dictionary dividing strategy.

Appendix

A. Proof of Lemma 2

The Fourier transform $\hat{g}(\cdot; \xi, \boldsymbol{c})$ of the Gaussian function $g(\cdot; \xi, \boldsymbol{c})$ is given by

$$\begin{aligned}\hat{g}(\boldsymbol{t}) &= \frac{1}{(2\pi)^{L/2}} \int \exp\left(-i\boldsymbol{x}^{\mathsf{T}}\boldsymbol{t}\right) \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}\|^2}{2\xi}\right) d\boldsymbol{x} \\ &= \frac{1}{(2\pi)^{L/2}} \prod_{l=1}^{L} \exp\left(-\frac{{c^{(l)}}^2}{2\xi}\right) \exp\left(\frac{(i\xi t^{(l)} - c^{(l)})^2}{2\xi}\right) \\ &\quad \times \int \exp\left(-\frac{(x^{(l)} + (i\xi t^{(l)} - c^{(l)}))^2}{2\xi}\right) dx^{(l)}. \quad (18)\end{aligned}$$

By changing the variable of the integral in (18) to $z^{(l)} := \frac{x^{(l)}+i\xi t^{(l)}-c^{(l)}}{\sqrt{2\xi}}$, we obtain

$$\begin{aligned}\hat{g}(\boldsymbol{t}) &= \frac{1}{(2\pi)^{L/2}}\prod_{l=1}^{L}\exp\left(-\frac{{c^{(l)}}^2}{2\xi}\right)\exp\left(\frac{(i\xi t^{(l)}-c^{(l)})^2}{2\xi}\right)\\ &\quad\times\int\exp\left(-{z^{(l)}}^2\right)\sqrt{2\xi}dz^{(l)}\\ &= \xi^{L/2}\exp\left(-\frac{\xi}{2}\|\boldsymbol{t}\|^2\right)\exp\left(-i\boldsymbol{c}^{\mathsf{T}}\boldsymbol{t}\right). \end{aligned} \tag{19}$$

B. Proof of Lemma 3

The squared absolute value of $\frac{\hat{g}(\boldsymbol{t};\xi,\boldsymbol{c})}{\sqrt{\hat{g}(\boldsymbol{t};\xi_\kappa,\boldsymbol{0})}}$ is

$$\begin{aligned}\left|\frac{\hat{g}(\boldsymbol{t};\xi,\boldsymbol{c})}{\sqrt{\hat{g}(\boldsymbol{t};\xi_\kappa,\boldsymbol{0})}}\right|^2 &= \left(\frac{\xi^{L/2}\exp\left(-\frac{\xi\|\boldsymbol{t}\|^2}{2}\right)}{\sqrt{\xi_\kappa^{L/2}\exp\left(-\frac{\xi_\kappa\|\boldsymbol{t}\|^2}{2}\right)}}\right)^2\\ &= \frac{\xi^L}{\xi_\kappa^{L/2}}\exp\left(\|\boldsymbol{t}\|^2\left(\frac{\xi_\kappa}{2}-\xi\right)\right). \end{aligned} \tag{20}$$

Since $2\xi > \xi_\kappa$ and $\xi < \infty$, the integral of (20) is

$$\int\left|\frac{\hat{g}(\boldsymbol{t};\xi,\boldsymbol{c})}{\sqrt{\hat{g}(\boldsymbol{t};\xi_\kappa,\boldsymbol{0})}}\right|^2 d\boldsymbol{t} = \frac{(2\pi)^{L/2}\xi^L}{\left(\xi_\kappa(2\xi-\xi_\kappa)\right)^{L/2}}, \tag{21}$$

and it holds that $\int\left|\frac{\hat{g}(\boldsymbol{t};\xi,\boldsymbol{c})}{\sqrt{\hat{g}(\boldsymbol{t};\xi_\kappa,\boldsymbol{0})}}\right|^2 d\boldsymbol{t} < \infty$.

C. Proof of Theorem 1

By Lemma 1 with Lemma 3, the inner product $\left\langle g(\cdot;\xi^{(1)},\boldsymbol{c}^{(1)}), g(\cdot;\xi^{(2)},\boldsymbol{c}^{(2)})\right\rangle_{\mathcal{H}_{\xi_\kappa}}$ is given by

$$\begin{aligned}&\left\langle g(\cdot;\xi^{(1)},\boldsymbol{c}^{(1)}), g(\cdot;\xi^{(2)},\boldsymbol{c}^{(2)})\right\rangle_{\mathcal{H}_{\xi_\kappa}}\\ &= \frac{1}{(2\pi)^{L/2}}\int\hat{g}(\boldsymbol{t};\xi^{(1)},\boldsymbol{c}^{(1)})\overline{\hat{g}(\boldsymbol{t};\xi^{(2)},\boldsymbol{c}^{(2)})}/\hat{g}(\boldsymbol{t};\xi_\kappa,\boldsymbol{0})d\boldsymbol{t}. \end{aligned} \tag{22}$$

From (19) and (22), we obtain

$$\begin{aligned}&\left\langle g(\cdot;\xi^{(1)},\boldsymbol{c}^{(1)}), g(\cdot;\xi^{(2)},\boldsymbol{c}^{(2)})\right\rangle_{\mathcal{H}_{\xi_\kappa}}\\ &= \left(\frac{\xi^{(1)}\xi^{(2)}}{\xi_\kappa(\xi^{(1)}+\xi^{(2)}-\xi_\kappa)}\right)^{L/2}\exp\left(\frac{-\left\|\boldsymbol{c}^{(1)}-\boldsymbol{c}^{(2)}\right\|^2}{2(\xi^{(1)}+\xi^{(2)}-\xi_\kappa)}\right). \end{aligned} \tag{23}$$

REFERENCES

[1] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[2] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, no. 2, pp. 246–257, June 1991.

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

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

[5] J. Racine, "An efficient cross-validation algorithm for window width selection for nonparametric kernel regression," *Communications in Statistics - Simulation and Computation*, vol. 22, no. 4, pp. 1107–1114, 1993.

[6] G. C. Cawley and N. L. C. Talbot, "Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers," *Pattern Recognition*, vol. 36, no. 11, pp. 2585 – 2592, 2003.

[7] S. An, W. Liu, and S. Venkatesh, "Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression," *Pattern Recognition*, vol. 40, no. 8, pp. 2154 – 2162, 2007, part Special Issue on Visual Information Processing.

[8] E. Herrmann, "Local bandwidth choice in kernel regression estimation," *Journal of Computational and Graphical Statistics*, vol. 6, no. 1, pp. 35–54, 1997.

[9] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. New York: Routledge: Taylor and Francis, 1998.

[10] M. C. Jones, J. S. Marron, and S. J. Sheather, "A brief survey of bandwidth selection for density estimation," *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 401–407, 1996.

[11] 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.

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

[13] 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.

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

[15] M. Takizawa and M. Yukawa, "Steepening squared error function facilitates online adaptation of gaussian scales," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 5450–5454.

[16] ——, "Joint learning of model parameters and coefficients for online nonlinear estimation," *IEEE Access*, vol. 9, pp. 24 026–24 040, 2021.

[17] 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.

[18] 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.

[19] 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.

[20] B. Chen, S. Zhao, P. Zhu, S. Seth, and J. C. Príncipe, "Online efficient learning with quantized KLMS and l_1 regularization," in *Proc. Int. Joint Conf. Neural Networks*, 2012.

[21] 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.

[22] 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.

[23] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proc. Int. Conf. Machine Learning*. ACM, 2004.

[24] M. Yukawa, "Adaptive learning in Cartesian product of reproducing kernel Hilbert spaces," *IEEE Trans. Signal Processing*, vol. 63, no. 22, pp. 6037–6048, Nov. 2015.

[25] M. Kasparick, R. L. G. Cavalcante, S. Valentin, S. Stańczak, and M. Yukawa, "Kernel-based adaptive online reconstruction of coverage maps with side information," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 7, pp. 5461–5473, July 2016.

[26] D. A. Awan, R. L. G. Cavalcante, M. Yukawa, and S. Stanczak, "Detection for 5G-NOMA: An online adaptive machine learning approach," in *2018 IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6.

[27] B. Shin, M. Yukawa, R. L. G. Cavalcante, and A. Dekorsy, "Distributed adaptive learning with multiple kernels in diffusion networks," *IEEE Transactions on Signal Processing*, vol. 66, no. 21, pp. 5505–5519, Nov 2018.

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

[29] M. Yukawa and K.-R. Müller, "Why does a hilbertian metric work efficiently in online learning with kernels?" *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1424–1428, 2016.
[30] H. Q. Minh, "Some properties of gaussian reproducing kernel hilbert spaces and their implications for function approximation and learning theory," *Constructive Approximation*, vol. 32, pp. 307–338, 2010.
[31] H. Wendland, *Scattered Data Approximation*, ser. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2004.
[32] 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.