Nonlinear Online Learning – A Kernel SMF Approach

Kewei Chen*, Stefan Werner†, Anthony Kuh‡ and Yih-Fang Huang*
* University of Notre Dame, Notre Dame, USA
Email:{kchen6, huang}@nd.edu
† Norwegian University of Science and Technology, Trondheim, Norway
Email:stefan.werner@ntnu.no
‡ University of Hawaii, Honolulu, USA
Email:kuh@hawaii.edu

***Abstract*—Principles of adaptive filtering and signal processing are useful tools in machine learning. Nonlinear adaptive filtering techniques, though often are analytically intractable, are more suitable for dealing with complex practical problems. This paper develops a nonlinear online learning algorithm with a kernel set-membership filtering (SMF) approach. One of the main features in the SMF framework is its data-dependent selective update of parameter estimates. Accordingly, the kernel SMF algorithm can not only selectively update its parameter estimates by making discerning use of the input data, but also selectively increase the dimension of the kernel expansions with a model sparsification criterion. This results in more sparse kernel expansions and less computation in the update of parameter estimates, making the proposed online learning algorithm more effective. Both analytical and numerical results are presented in this paper to corroborate the above statements.**

I. Introduction

Learning algorithms have received much attention, especially in the last decade or so, due to much increased interest in artificial intelligence and machine learning. This increased interest is attributable, in part, to the greatly increased data processing power and, in part, to the improved modeling and algorithmic development in recent years. Most of the learning algorithms, however, are developed with linear models, see, e.g., [1]. Perceptron networks [2], such as those used in deep learning, are perhaps a good example of nonlinear learning mechanism. Since most data processors have much more powerful computational power these days, time may be ripe that nonlinear learning schemes can be employed to solve more complex and realistic problems. Over the years, many nonlinear learning algorithms have been developed, see, e.g., [3], [4], [5], [6], [7], [8]. This paper presents a nonlinear online learning algorithm based on set-membership filtering (SMF).

The SMF framework is an adaptive filtering paradigm that features data-dependent selective update of the estimates for the filter coefficients [9], [10]. A key assumption in the SMF framework is that the filtering error is bounded in magnitude. Accordingly, if the filtering error is less than the presumed magnitude bound, no update of the parameter estimates is needed. Checking on whether or not the filtering error exceeds the presumed magnitude bound is sometimes termed *innovation check* in the SMF literature. Since its inception, the SMF algorithms have been shown to be viable alternatives to the traditional adaptive filtering algorithms such as recursive least squares (RLS) and least mean squares (LMS). Instead of updating the parameter estimates at every iteration, i.e., at every new data point, the SMF algorithms update only when there is sufficient *innovation*, which is measured by the filtering/prediction error. Though using only a fraction of the data to update the parameter estimates, the SMF algorithms perform (at least) comparably to their counterparts of traditional algorithms, namely, RLS and LMS. This feature opens up many avenues for further exploration, see, e.g., [11]. To date, however, most of the SMF algorithms have been developed using linear models, see, e.g., [9], [10], [11], [12].

This paper derives a kernel-SMF algorithm, namely, the K-DH-OBE algorithm, for nonlinear online learning based on one of the SMF algorithms, i.e., DH-OBE algorithm [10]. The DH-OBE algorithm can be viewed as a weighted RLS algorithm with a data-dependent forgetting factor. Our results show that the K-DH-OBE algorithm has good performance in terms of steady-state mean squared error (MSE) and convergence rate, as compared to other kernel algorithms [13], [14]. It is shown in this paper that a major advantage of the K-DH-OBE algorithm is that it results in more sparse kernel expansions and less frequent update of parameter estimates, thus less computation.

This paper is organized as follows: Section II provides the background of kernel based online learning. Section III shows the derivations of the K-DH-OBE algorithm, and Section IV presents some simulation results. Conclusion and a short discussion on future work are given in Section V.

II. Background

This section presents the basics of kernel based online learning. The *kernel trick* is a commonly used approach to extend linear algorithms to nonlinear ones [15], [16], [17]. Assume $\mathbf{x}_i$ and $\mathbf{x}_j$ are in space $\mathcal{X}$, a reproducing kernel [18] of a Hilbert space $\mathcal{H}$ is a function

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \boldsymbol{\phi}(\mathbf{x}_i), \boldsymbol{\phi}(\mathbf{x}_j) \rangle_{\mathcal{H}}$$

that maps from $\mathcal{X} \times \mathcal{X}$ to $\mathbb{R}$. So for any pair of points $\mathbf{x}_i$ and $\mathbf{x}_j$ in $\mathcal{X}$ that are mapped to $\mathcal{H}$ by $\boldsymbol{\phi}(\cdot)$, the kernel can

evaluate the inner product of $\boldsymbol{\phi}(\mathbf{x}_i)$ and $\boldsymbol{\phi}(\mathbf{x}_j)$ without explicit knowledge of either $\boldsymbol{\phi}(\cdot)$ or $\mathcal{H}$.

One example of using the kernel trick to solve nonlinear least squares problems is to find a function $f(\cdot)$ of $\mathcal{H}$ to minimize the sum of n squared errors, given a set of data pairs $\mathcal{D}^n = \{\mathbf{x}_i, y_i\}_{i=1}^n$. Specifically

$$\min_{f\in\mathcal{H}} \sum_{i=1}^{n} |y_i - f(\mathbf{x}_i)|^2, \tag{1}$$

where y_i is the desired output and $f(\mathbf{x}_i)$ is the model output. The representer theorem [19], [20] shows that the solution $f(\cdot)$ of (1) can be expressed as

$$f(\cdot) = \sum_{j=1}^{n} w_j \kappa(\cdot, \mathbf{x}_j). \tag{2}$$

Then (1) can be rewritten as $\min_{\mathbf{w}} ||\mathbf{y} - \mathbf{K}\mathbf{w}||^2$, where $\mathbf{K}$ is the Gram matrix with $[\mathbf{K}_{ij}] = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. The weights $\mathbf{w}$ can be further found by solving $\mathbf{K}\mathbf{w} = \mathbf{y}$.

In the online learning scenario where data become available sequentially, at time instant k, the prediction of y_k given $\mathcal{D}^{k-1} \bigcup \{\mathbf{x}_k\}$ can be expressed as

$$f_k(\mathbf{x}_k) = \sum_{j=1}^{k-1} w_j \kappa(\mathbf{x}_k, \mathbf{x}_j). \tag{3}$$

However, the complexity of the problem grows every time when a new data point arrives, which poses a challenge for the *real-time* operation for online algorithms. Therefore, one needs to control the dimension of the model so that the computation complexity would not keep increasing as the number of data points increases. Specifically, in the following form

$$f_k(\mathbf{x}_k) = \sum_{j=1}^{m} w_j \kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_j}), \tag{4}$$

where $\mathcal{J}_k = \{\alpha_j\}_{j=1}^m$ is a subset of $\{i\}_{i=1}^k$, the dimension m of the dictionary $\{\kappa(\cdot, \mathbf{x}_{\alpha_j})\}_{j=1}^m$ should stop increasing at some point in time.

In the literature, Engel et al. proposed the kernel recursive least squares (KRLS) algorithm [13] that adopted an approximate linear dependence (ALD) criterion as a sparsification rule to control the model dimension and used RLS algorithm to update the weights. The ALD-based sparsification rule suggests inserting the new data point into the dictionary only if it is not approximately linearly dependent on the dictionary vectors. The major criticism is that it leads to costly computations. To reduce the computational complexity at each iteration, Richard et al. proposed the kernel normalized least mean squares (KNLMS) algorithm [14] that adopted the coherence-based sparsification rule to control the model dimension. Specifically, at time instant k, the coherence-based sparsification rule suggests inserting $\kappa(\cdot, \mathbf{x}_k)$ into the dictionary only if the coherence is smaller than a pre-specified threshold μ, i.e.,

$$\max_{\alpha_j \in \mathcal{J}_{k-1}} |\kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_j})| \leq \mu. \tag{5}$$

It was shown that the model dimension under the coherence-based sparsification rule remains finite as k goes to infinity.

III. K-DH-OBE Algorithm

In this section, we employ the kernel trick with the SMF principles to solve nonlinear online learning problems. Set-membership filtering algorithms aim to update the unknown parameters such that the output estimation error is upper bounded in magnitude over a model space, see, e.g., [10], [12]. To illustrate this, consider the following nonlinear model that characterizes the input-output relationship

$$y_k = f_k(\mathbf{x}_k) + n_k, \tag{6}$$

where $y_k \in \mathbb{R}$, the set of real numbers, and $\mathbf{x}_k \in \mathbb{R}^{N\times 1}$, the set of N-dimensional vectors, denote, respectively, the output signal and input signal vector. Also, $n_k \in \mathbb{R}$ denotes the model uncertainty (or noise) and $f_k(\mathbf{x}_k)$ is as shown in (4).

At time instant $k-1$, denote the weights as $\mathbf{w}_{k-1}$ and the dictionary as $\{\kappa(\cdot, \mathbf{x}_{\alpha_j})\}_{j=1}^m$. According to (4), the model can be expressed as $f_{k-1}(\mathbf{x}_{k-1}) = \mathbf{w}_{k-1}^T \mathbf{u}_{k-1}$ with

$$\mathbf{u}_{k-1} = [\kappa(\mathbf{x}_{k-1}, \mathbf{x}_{\alpha_1}) \ \cdots \ \kappa(\mathbf{x}_{k-1}, \mathbf{x}_{\alpha_m})]^{\mathrm{T}}.$$

At the next time instant k, the K-DH-OBE algorithm updates the estimates of the weights to satisfy that

$$\mathcal{C}_k = \{\mathbf{w} \in \mathbb{R}^N : |y_k - \mathbf{w}^{\mathrm{T}}\mathbf{u}_k|^2 \leq \gamma^2\}, \tag{7}$$

where $\mathcal{C}_k$ is called the *constraint set* and it is a degenerate ellipsoid in the parameter space; while γ^2 is a prescribed estimation error bound.

Given a sequence of data pairs $\mathcal{D}^k = \{\mathbf{x}_i, y_i\}_{i=1}^k$, if the parameter vector to be estimated remains constant, it must lie inside the intersection of all the constraint sets, namely,

$$\mathbf{w} \in \Omega_k \triangleq \cap_{i=1}^k \mathcal{C}_k = \Omega_{k-1} \cap \mathcal{C}_k. \tag{8}$$

The set Ω_k in the above equation is termed the *exact membership set*. Clearly, every point in the exact membership set is a legitimate estimate for $\mathbf{w}$ as it is consistent with the presumed model and the received data. We note that Ω_k is a sequence of monotone non-increasing sets, i.e., $\Omega_k \subseteq \Omega_{k-i}$ for any $i \geq 1$.

Intuitively, if the data pairs $\{\mathbf{x}_i, y_i\}_{i=1}^k$ are rich enough, Ω_k will be smaller when k grows larger. It is thus likely that, at some point in time k, $\Omega_k = \Omega_{k-1}$. The key point is that whenever a constraint set is not useful in reducing the size of the exact membership set, it can be discarded and its associated data pair is not used for updating the parameter estimates. This results in the so-called *data-dependent selective update* for the parameter estimates. In essence, the SMF algorithms *adapt only when necessary*.

For the purpose of analysis, it is desirable to obtain an effective analytical description of the exact membership set Ω_k. In practice, however, it is usually more convenient to find some analytically tractable outer bounding sets for Ω_k. One good candidate for such bounding sets is ellipsoid.

Let $\mathcal{E}_{k-1}$ be an ellipsoid that, at time instant $k-1$, outer bounds the exact membership set Ω_{k-1}, i.e., $\mathcal{E}_{k-1} \supset \Omega_{k-1}$. This bounding ellipsoid can be formulated as

$$\mathcal{E}_{k-1} = \{\mathbf{w} \in \mathbb{R}^N : [\mathbf{w} - \mathbf{w}_{k-1}]^{\mathrm{T}} \mathbf{P}_{k-1}^{-1} [\mathbf{w} - \mathbf{w}_{k-1}] \leq \sigma_{k-1}^2\} \tag{9}$$

where $\mathbf{w}_{k-1}$ is the center of the ellipsoid and $\mathbf{P}_{k-1}$ is a positive semi-definite matrix that characterizes the size (namely, the lengths of the semi-axes) of the ellipsoid. Then, given $\mathcal{E}_{k-1}$, and the constraint set $\mathcal{C}_k$ obtained at time k, we shall find an ellipsoid $\mathcal{E}_k$ that outer bounds the following intersection

$$\mathcal{E}_k \supset \mathcal{E}_{k-1} \cap \mathcal{C}_k. \tag{10}$$

The family of optimal bounding ellipsoid (OBE) algorithms [10], [12], [21] have been proposed to approximate the exact membership set. DH-OBE algorithm [10] recursively computes $\mathbf{w}_k$, $\mathbf{P}_k$, and σ_k^2 defining $\mathcal{E}_k \supset \mathcal{E}_{k-1} \cap \mathcal{C}_k$ through a linear combination of (9) and (7), specifically, $\mathcal{E}_k = (1 - \lambda_k)\mathcal{E}_{k-1} + \lambda_k \mathcal{C}_k$ with $\lambda_k \in [0, 1)$. At each time instant k, each value of λ_k yields a corresponding bounding ellipsoid and λ_k is chosen such that σ_k^2 is minimized.

In this paper, we shall use the kernel trick to derive the K-DH-OBE algorithm to recursively update the parameters of $\mathcal{E}_k$ for our nonlinear algorithm. Essentially, the K-DH-OBE algorithm needs to address two issues:

- whether or not the new data pair $(\mathbf{x}_k, y_k)$ is innovative enough to necessitate an update for the parameter estimates, which can be determined by the innovation check, i.e., check if $e_k^2 + \sigma_{k-1}^2 > \gamma^2$ holds as shown in (15);
- whether or not $\kappa(\cdot, \mathbf{x}_k)$ should be inserted into the existing dictionary, which can be determined by the coherence-based sparsification rule as shown in (5).

In our proposed K-DH-OBE algorithm, we shall first conduct the innovation check. We adopt the coherence criterion to check if it should be inserted into the dictionary only when the new data pair necessitates an update for the parameter estimates. Depending on whether or not the dimension of the dictionary increases, the algorithm should have two cases for the weights update, which are discussed separately as follows.

A. Case 1: Dimension Remains The Same

When the innovation check indicates that the new data pair $(\mathbf{x}_k, y_k)$ necessitates an update for the parameter estimates but the coherence-based sparsification rule does not suggest inserting $\kappa(\cdot, \mathbf{x}_k)$ into the existing dictionary, the dimension remains the same. Thus the model is $f_k(\mathbf{x}_k) = \mathbf{w}_k^T \mathbf{u}_k$ with

$$\mathbf{u}_k = [\kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_1}) \ \cdots \ \kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_m})]^{\mathrm{T}}. \tag{11}$$

Similarly to the linear DH-OBE algorithm, the recursive updating formulas can be obtained by finding an ellipsoid $\mathcal{E}_k$ that outer bounds $\mathcal{E}_{k-1} \cap \mathcal{C}_k$. The following theorem provides the recursive updating formulas for the parameters of $\mathcal{E}_k$.

Theorem 1: Let ellipsoid $\mathcal{E}_{k-1}$ be the optimal bounding ellipsoid which is characterized by $\mathbf{w}_{k-1}$, $\mathbf{P}_{k-1}$ and σ_{k-1}^2 as shown in (9). At time instant k, the following recursive expressions for $\mathbf{w}_k$, $\mathbf{P}_k$, and σ_k^2 defining $\mathcal{E}_k \supset \mathcal{E}_{k-1} \cap \mathcal{C}_k$ are obtained through a linear combination of (9) and (7), specifically, $\mathcal{E}_k = (1 - \lambda_k^*)\mathcal{E}_{k-1} + \lambda_k^* \mathcal{C}_k$ with $\lambda_k^* \in [0, 1)$ being defined in [10] such that σ_k^2 is minimized:

$$\mathbf{w}_k = \mathbf{w}_{k-1} + \lambda_k^* \mathbf{P}_k \mathbf{u}_k e_k \tag{12a}$$

$$\mathbf{P}_k^{-1} = (1 - \lambda_k^*)\mathbf{P}_{k-1}^{-1} + \lambda_k^* \mathbf{u}_k \mathbf{u}_k^{\mathrm{T}} \tag{12b}$$

$$e_k = y_k - \mathbf{w}_{k-1}^{\mathrm{T}} \mathbf{u}_k \tag{12c}$$

$$\sigma_k^2 = (1 - \lambda_k^*)\sigma_{k-1}^2 + \lambda_k^* \gamma^2 - \frac{\lambda_k^*(1 - \lambda_k^*)e_k^2}{1 - \lambda_k^* + \lambda_k^* G_k} \tag{12d}$$

$$G_k = \mathbf{u}_k^{\mathrm{T}} \mathbf{P}_{k-1} \mathbf{u}_k, \tag{12e}$$

In addition, by employing the matrix inversion lemma, the computation of $\mathbf{P}_k$ can be simplified as

$$\mathbf{P}_k = \frac{1}{1 - \lambda_k^*} \left[\mathbf{P}_{k-1} - \frac{\lambda_k^* \mathbf{P}_{k-1} \mathbf{u}_k \mathbf{u}_k^{\mathrm{T}} \mathbf{P}_{k-1}}{1 - \lambda_k^* + \lambda_k^* G_k} \right] \tag{13}$$

and the computation of $\mathbf{w}_k$ can be simplified as

$$\mathbf{w}_k = \mathbf{w}_{k-1} + \frac{\lambda_k^* \mathbf{P}_{k-1} \mathbf{u}_k e_k}{1 - \lambda_k^* + \lambda_k^* G_k}. \tag{14}$$

Proof: The proof is provided in the Appendix. ■

With $\mathbf{P}_k$ positive semi-definite and define $\beta_k \triangleq (\gamma^2 - \sigma_{k-1}^2)/e_k^2$, the optimal λ_k^* is computed by

$$\lambda_k^* = \begin{cases} \min(\xi, \nu_k), & \text{if } e_k^2 + \sigma_{k-1}^2 > \gamma^2 \\ 0, & \text{otherwise,} \end{cases} \tag{15}$$

where

$$\nu_k = \begin{cases} \xi, & \text{if } e_k^2 = 0 \\ \frac{1-\beta_k}{2}, & \text{if } G_k = 1 \\ \frac{1}{1-G_k}\left[1 - \sqrt{\frac{G_k}{1+\beta_k(G_k-1)}}\right] & \text{if } \beta_k(G_k - 1) + 1 > 0 \\ \xi & \text{if } \beta_k(G_k - 1) + 1 \leq 0 \end{cases}$$

and $\xi \in (0, 1)$ is a design parameter.

Notice that if, at any iteration k, $e_k^2 + \sigma_{k-1}^2 \leq \gamma^2$, then the optimal $\lambda_k^* = 0$, which results in $\mathbf{w}_k = \mathbf{w}_{k-1}, \mathbf{P}_k = \mathbf{P}_{k-1}$, and $\sigma_k^2 = \sigma_{k-1}^2$ according to (12). In other words, if $e_k^2 + \sigma_{k-1}^2 \leq \gamma^2$ holds, then the computation of (12) is not required. In essence, the data-dependent selective update feature of SMF algorithms still holds, which reduces computation cost by skipping the parameters update at such iterations.

B. Case 2: Dimension Increases

When the innovation check indicates that the new data pair $(\mathbf{x}_k, y_k)$ necessitates an update for the parameter estimates and the coherence based sparsification rule suggests inserting $\kappa(\cdot, \mathbf{x}_k)$ into the existing dictionary, then the dimension increases and the model is $f_k(\mathbf{x}_k) = \mathbf{w}_k^T \tilde{\mathbf{u}}_k$ with

$$\begin{aligned} \tilde{\mathbf{u}}_k &= [\kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_1}) \ \cdots \ \kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_m}) \ \kappa(\mathbf{x}_k, \mathbf{x}_k)]^{\mathrm{T}} \\ &= \left[\begin{array}{c|c} \mathbf{u}_k^{\mathrm{T}} & u_{ak} \end{array} \right]^{\mathrm{T}}. \end{aligned} \tag{16}$$

The dimensions for the weight $\mathbf{w}_{k-1}$ and the corresponding shaping matrix $\mathbf{P}_{k-1}$ before the update should also be increased by one to match the dimensions of $\tilde{\mathbf{u}}_k$, i.e.,

$$\tilde{\mathbf{w}}_{k-1} = \left[\begin{array}{c|c} \mathbf{w}_{k-1}^{\mathrm{T}} & 0 \end{array} \right]^{\mathrm{T}}, \tag{17}$$

$$\tilde{\mathbf{P}}_{k-1} = \left[\begin{array}{c|c} \mathbf{P}_{k-1} & \mathbf{0}_{m+1} \\ \hline \mathbf{0}_{m+1}^{\mathrm{T}} & \delta^{-2} \end{array}\right]. \tag{18}$$

In essence, the above equations are implemented to add a new semi-axis to the previous optimal bounding ellipsoid, resulting in a new ellipsoid in the higher dimension. Accordingly, the parameter δ in (18) must be chosen properly to ensure that the higher dimension bounding ellipsoid will outer bound the exact membership set in the next iteration. The recursive updating formulas can then be obtained by finding an ellipsoid $\mathcal{E}_k$ that outer bounds the intersection between the enlarged $\mathcal{E}_{k-1}$ and $\mathcal{C}_k$. Specifically, the weights update is given by

$$\mathbf{w}_k = \tilde{\mathbf{w}}_{k-1} + \lambda_k^* \mathbf{P}_k \tilde{\mathbf{u}}_k e_k \tag{19a}$$

$$\mathbf{P}_k^{-1} = (1-\lambda_k^*)\tilde{\mathbf{P}}_{k-1}^{-1} + \lambda_k^* \tilde{\mathbf{u}}_k \tilde{\mathbf{u}}_k^{\mathrm{T}} \tag{19b}$$

$$e_k = y_k - \mathbf{w}_{k-1}^{\mathrm{T}} \mathbf{u}_k \tag{19c}$$

$$\sigma_k^2 = (1-\lambda_k^*)\sigma_{k-1}^2 + \lambda_k^* \gamma^2 - \frac{\lambda_k^*(1-\lambda_k^*)e_k^2}{1-\lambda_k^*+\lambda_k^*\tilde{G}_k} \tag{19d}$$

$$\tilde{G}_k = \tilde{\mathbf{u}}_k^T \tilde{\mathbf{P}}_{k-1} \tilde{\mathbf{u}}_k = \mathbf{u}_k^{\mathrm{T}} \mathbf{P}_{k-1} \mathbf{u}_k + u_{ak}^2/\delta^2, \tag{19e}$$

where λ_k^* is computed according to (15) with G_k replaced by $\tilde{G}_k$.

In addition, the computation of the shaping matrix $\mathbf{P}_k$ can be simplified. Substituting (16) and (18) to (19b) yields

$$\mathbf{P}_k^{-1} = (1-\lambda_k^*)\left[\begin{array}{c|c} \mathbf{P}_{k-1}^{-1} & \mathbf{0}_{m+1} \\ \hline \mathbf{0}_{m+1}^{\mathrm{T}} & \delta^2 \end{array}\right] + \lambda_k^* \left[\begin{array}{c|c} \mathbf{u}_k \mathbf{u}_k^{\mathrm{T}} & \mathbf{u}_k u_{ak} \\ \hline u_{ak}\mathbf{u}_k^{\mathrm{T}} & u_{ak}^2 \end{array}\right]. \tag{20}$$

Applying the matrix inversion lemma to (20) gives

$$\mathbf{P}_k = \left[\begin{array}{c|c} \mathbf{P}_{11} & \mathbf{p}_1 \\ \hline \mathbf{p}_1^{\mathrm{T}} & r \end{array}\right], \tag{21}$$

where

$$\mathbf{P}_{11} = \frac{1}{1-\lambda_k^*}\left[\mathbf{P}_{k-1} - \frac{\lambda_k^* \mathbf{P}_{k-1}\mathbf{u}_k\mathbf{u}_k^{\mathrm{T}}\mathbf{P}_{k-1}}{1-\lambda_k^*+\lambda_k^*(u_{ak}^2/\delta^2+G_k)}\right],$$

$$\mathbf{p}_1 = -\frac{\lambda_k^*}{1-\lambda_k^*}\frac{1}{\delta^2}\frac{u_{ak}\mathbf{P}_{k-1}\mathbf{u}_k}{1-\lambda_k^*+\lambda_k^*(u_{ak}^2/\delta^2+G_k)},$$

$$r = \frac{1}{\delta^2}\frac{1-\lambda_k^*+\lambda_k^* G_k}{1-\lambda_k^*+\lambda_k^*(u_{ak}^2/\delta^2+G_k)}.$$

Furthermore, the weights updated (19a) can be computed by

$$\mathbf{w}_k = \left[\begin{array}{c} \mathbf{w}_{k-1} \\ 0 \end{array}\right] + \left[\begin{array}{c} \frac{\lambda_k^* e_k \mathbf{P}_{k-1}\mathbf{u}_k}{1-\lambda_k^*+\lambda_k^*(G_k+u_{ak}^2/\delta^2)} \\ \frac{1}{\delta^2}\frac{\lambda_k^* e_k u_{ak}}{1-\lambda_k^*+\lambda_k^*(G_k+u_{ak}^2/\delta^2)} \end{array}\right].$$

The pseudocode that summarizes the K-DH-OBE algorithm is as shown in Algorithm 1.

Algorithm 1 K-DH-OBE algorithm

Initialization
  Initiate $\sigma_1^2, \gamma, \xi, \delta, \mu$
  Insert $\kappa(\cdot, \mathbf{x}_1)$ into the dictionary, set $m = 1$
  Denote the dictionary as $[\kappa(\cdot, \mathbf{x}_{\alpha_1})]$
  Compute $\mathbf{u}_1 = [\kappa(\mathbf{x}_1, \mathbf{x}_{\alpha_1})]$, set $\mathbf{w}_1 = [1]$
For $k > 1$, repeat
  Get $(\mathbf{x}_k, y_k)$
  Compute $\mathbf{u}_k = [\kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_1}) \cdots \kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_m})]$
  Compute $e_k = y_k - \mathbf{w}_{k-1}^T \mathbf{u}_k$
  if $e_k^2 + \sigma_{k-1}^2 > \gamma^2$ ▷ Innovation check
    if $\max_{j=1,\cdots,m} |\kappa(\mathbf{x}_k, \mathbf{x}_{\alpha_j})| \leq \mu$ ▷ Case 2
      $m \leftarrow m + 1$
      Insert $\kappa(\cdot, \mathbf{x}_k)$ into the dictionary
      Denote $\kappa(\cdot, \mathbf{x}_k)$ as $\kappa(\cdot, \mathbf{x}_{\alpha_m})$
      Update $\mathbf{w}_k$ using (19)
    else ▷ Case 1
      Update $\mathbf{w}_k$ using (12)
    end
  end

IV. Simulations

This section presents a simulation study that compares the proposed K-DH-OBE algorithm with the KNLMS algorithm [14] and the KRLS algorithm [13] using the same nonlinear model

$$y_k = [0.8 - 0.5\exp(-y_{k-1}^2)]y_{k-1} - [0.3 + 0.9\exp(-y_{k-1}^2)]y_{k-2} + 0.1\sin(y_{k-1}\pi) + n_k,$$

where y_k is the desired output corrupted by Gaussian measurement noise $n_k \sim \mathcal{N}(0, 0.01)$. The initial condition is $y_1 = y_2 = 0.1$. Each online algorithm is used to learn a nonlinear model $\hat{y}_k = f_k(\mathbf{x}_k)$ at every time instant k, where $\mathbf{x}_k = [y_{k-1}\ y_{k-2}]^T$. In this study, we use the same Gaussian kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-3.73||\mathbf{x}_i - \mathbf{x}_j||^2)$ as that used in [14], and the threshold μ in the coherence criterion is set to be 0.5. Other details of the experiment setup for the KNLMS algorithm and the KRLS algorithm can be found in [14].

For the K-DH-OBE algorithm, we set the initial value $\sigma_1^2 = 100$ and $\xi = 0.005$, while other design parameters, namely, γ and δ, are determined by a grid search based on the MSE averaged over 10 independent experiments. Each experiment includes a sequence of 5000 samples, and the MSE is evaluated on the last 3000 iterations (after the algorithm converged), i.e. $\sum_{k=2001}^{5000}(y_k - \mathbf{w}_{k-1}^{\mathrm{T}}\mathbf{u}_k))^2/3000$. The parameters that yield smallest MSE are $\gamma = 5$ and $\delta = 0.2$. Fig. 1 shows the learning curves of K-DH-OBE, KNLMS and KRLS, which are obtained by averaging over 200 independent experiments for each time instant and smoothing with a moving average window that averages over 20 consecutive samples. A plot of σ_k^2 in the K-DH-OBE algorithm obtained from one of the experiments is shown in Fig. 2.

Another two hundred 10,000-sample independent experiments are conducted to test the performance of the model. As reported in Table I, the MSE for K-DH-OBE is 0.0103, which is achieved with only 1,659 iterations of weights update by virtue of the inherent selective update property of SMF algorithms. In addition, K-DH-OBE algorithm yields more sparse dictionary. An example of increase of the dimensions for K-DH-OBE, KNLMS and KRLS obtained from one experiment is shown in Fig. 3.

V. Conclusion

This paper has derived a nonlinear online learning algorithm based on the SMF framework and kernel method. In addi-

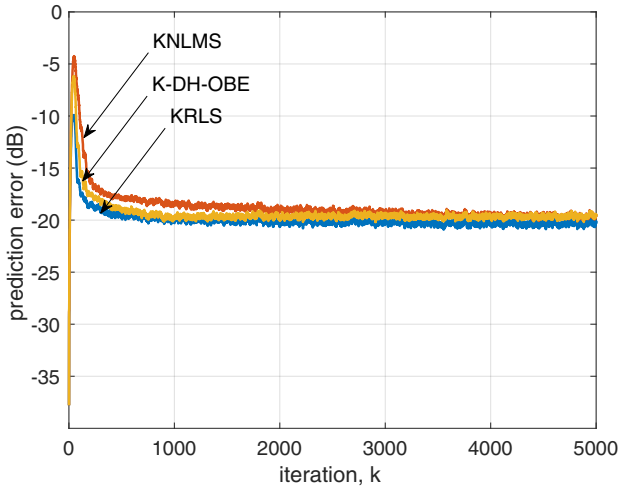

Fig. 1. Learning curves of K-DH-OBE, KNLMS and KRLS obtained by averaging two hundred 5,000-sample independent experiments.

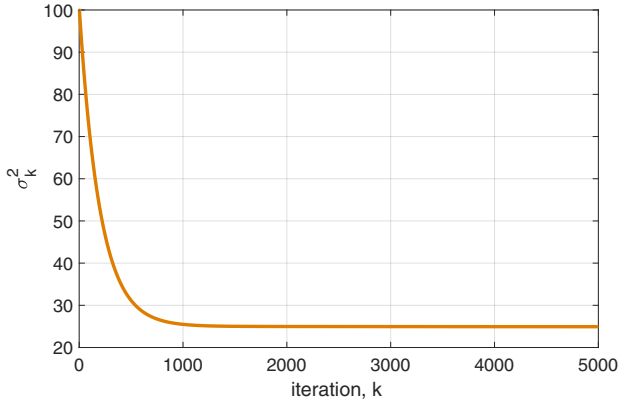

Fig. 2. σ_k^2 versus iterations in K-DH-OBE obtained from one experiment.

TABLE I
PERFORMANCE TESTED ON TWO HUNDRED 10,000-SAMPLE INDEPENDENT EXPERIMENTS

Algorithm	MSE*	Dimension	Update Times
K-DH-OBE	0.0103	20.11	1,659
KNLMS	0.0105	22.16	10,000
KRLS	0.0091	22.91	10,000

*MSE is averaged over the last 5000 iterations.

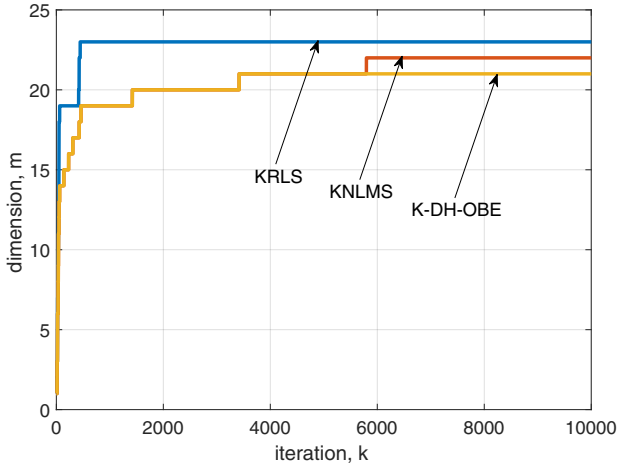

Fig. 3. Increase of the dimension for K-DH-OBE, KNLMS and KRLS obtained from one experiment.

tion to the coherence-based sparsification rule that controls the size of kernel expansions, the data-dependent selective update property embedded in the SMF framework results in a more sparse model and, more importantly, much less frequent updates of the parameter estimates. Simulation experiments showed that K-DH-OBE algorithm achieved smaller MSE and faster convergence than KNLMS algorithm, and had many fewer iterations that update the parameter estimates. Future work will include more theoretical analysis on the convergence properties.

VI. ACKNOWLEDGMENT

This work was supported, in part, by the Research Council of Norway; and, in part, by the University of Notre Dame.

APPENDIX

Proof of Theorem 1: Linear combination of (9) and (7) yields the bounding ellipsoid at time instant k, i.e.,

$$\begin{aligned}\mathcal{E}_k &= \{\mathbf{w}\in\mathbb{R}^N : [\mathbf{w}-\mathbf{w}_k]^{\mathrm{T}}\mathbf{P}_k^{-1}[\mathbf{w}-\mathbf{w}_k]\le\sigma_k^2\}\\ &= \{\mathbf{w}\in\mathbb{R}^N : (1-\lambda_k)[\mathbf{w}-\mathbf{w}_{k-1}]^{\mathrm{T}}\mathbf{P}_{k-1}^{-1}[\mathbf{w}-\mathbf{w}_{k-1}]\\ &\quad +\lambda_k|y_k-\mathbf{w}^{\mathrm{T}}\mathbf{u}_k|^2\le(1-\lambda_k)\sigma_{k-1}^2+\lambda_k\gamma^2\}.\end{aligned} \tag{22}$$

The following relations are directly obtained via identification-by-terms of the expanded expressions in (22)

$$\mathbf{P}_k^{-1} = (1-\lambda_k)\mathbf{P}_{k-1}^{-1}+\lambda_k\mathbf{u}_k\mathbf{u}_k^{\mathrm{T}} \tag{23}$$

$$\mathbf{w}_k = \mathbf{P}_k\left[(1-\lambda_k)\mathbf{P}_{k-1}^{-1}\mathbf{w}_{k-1}+\lambda_k y_k\mathbf{u}_k\right] \tag{24}$$

$$\begin{aligned}\sigma_k^2 &= (1-\lambda_k)\sigma_{k-1}^2+\lambda_k\gamma_k^2-\lambda_k y_k^2\\ &\quad -(1-\lambda_k)\mathbf{w}_{k-1}^{\mathrm{T}}\mathbf{P}_{k-1}^{-1}\mathbf{w}_{k-1}+\mathbf{w}_k^{\mathrm{T}}\mathbf{P}_k^{-1}\mathbf{w}_k.\end{aligned} \tag{25}$$

After premultiplying (23) with $\mathbf{P}_k$, we obtain, after rearranging the terms, the relation $(1-\lambda_k)\mathbf{P}_k\mathbf{P}_{k-1}^{-1}=\mathbf{I}-\lambda_k\mathbf{P}_k\mathbf{u}_k\mathbf{u}_k^{\mathrm{T}}$, which after substitution in (24) together with (12c) gives (12a).

Relation (14) is obtained by rewriting the correction terms for the parameter update in (12a), i.e., $\lambda_k \mathbf{P}_k \mathbf{u}_k e_k$, using the matrix inversion lemma as follows

$$\begin{aligned}\lambda_k \mathbf{P}_k \mathbf{u}_k e_k &= \lambda_k \left[(1-\lambda_k)\mathbf{P}_{k-1}^{-1} + \lambda_k \mathbf{u}_k \mathbf{u}_k^{\mathrm{T}}\right]^{-1} \mathbf{u}_k e_k \\ &= \frac{\lambda_k}{1-\lambda_k}\left[\mathbf{P}_{k-1} - \frac{\mathbf{P}_{k-1}\mathbf{u}_k\mathbf{u}_k^{\mathrm{T}}\mathbf{P}_{k-1}}{\frac{1-\lambda_k}{\lambda_k} + \mathbf{u}_k^{\mathrm{T}}\mathbf{P}_{k-1}\mathbf{u}_k}\right]\mathbf{u}_k e_k \\ &= \frac{\lambda_k \mathbf{P}_{k-1}\mathbf{u}_k e_k}{1-\lambda_k+\lambda_k \mathbf{u}_k^{\mathrm{T}}\mathbf{P}_{k-1}\mathbf{u}_k}\end{aligned}$$

Finally, relation (12d) is obtained by substituting (23) and (14) into (25).

REFERENCES

[1] J. Neter, M. H. Kutner, C. J. Nachtsheim, and W. Wasserman, *Applied linear statistical models*. Irwin Chicago, 1996, vol. 4.

[2] J. Tang, C. Deng, and G.-B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 4, pp. 809–821, 2016.

[3] M. O. Franz and B. Schölkopf, "A unifying view of Wiener and Volterra theory and polynomial kernel regression," *Neural computation*, vol. 18, no. 12, pp. 3097–3118, 2006.

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

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

[6] M. Jiu and H. Sahbi, "Nonlinear deep kernel learning for image annotation," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1820–1832, 2017.

[7] N. D. Vanli, M. O. Sayin, I. Delibalta, and S. S. Kozat, "Sequential nonlinear learning for distributed multiagent systems via extreme learning machines," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 3, pp. 546–558, 2017.

[8] S. Yin, C. Yang, J. Zhang, and Y. Jiang, "A data-driven learning approach for nonlinear process monitoring based on available sensing measurements," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 643–653, 2017.

[9] E. Fogel and Y.-F. Huang, "On the value of information in system identification-bounded noise case," *Automatica*, vol. 18, no. 2, pp. 229–238, 1982.

[10] S. Dasgupta and Y.-F. Huang, "Asymptotically convergent modified recursive least-squares with data-dependent updating and forgetting factor for systems with bounded noise," *IEEE Transactions on information theory*, vol. 33, no. 3, pp. 383–392, 1987.

[11] S. Gollamudi, S. Kapoor, S. Nagaraj, and Y.-F. Huang, "Set-membership adaptive equalization and an updator-shared implementation for multiple channel communications systems," *IEEE Transactions on Signal Processing*, vol. 46, no. 9, pp. 2372–2385, 1998.

[12] S. Nagaraj, S. Gollamudi, S. Kapoor, and Y.-F. Huang, "BEACON: An adaptive set-membership filtering technique with sparse updates," *IEEE Transactions on Signal Processing*, vol. 47, no. 11, pp. 2928–2941, 1999.

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

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

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

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

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

[18] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.

[19] G. Kimeldorf and G. Wahba, "Some results on tchebycheffian spline functions," *Journal of mathematical analysis and applications*, vol. 33, no. 1, pp. 82–95, 1971.

[20] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *International conference on computational learning theory*. Springer, 2001, pp. 416–426.

[21] J. R. Deller, M. Nayeri, and S. Odeh, "Least-square identification with error bounds for real-time signal processing and control," *Proceedings of the IEEE*, vol. 81, no. 6, pp. 815–849, 1993.