AN EFFICIENT SPARSE KERNEL ADAPTIVE FILTERING ALGORITHM BASED ON ISOMORPHISM BETWEEN FUNCTIONAL SUBSPACE AND EUCLIDEAN SPACE

Masa-aki Takizawa and Masahiro Yukawa

Department of Electronics and Electrical Engineering, Keio University, Japan

ABSTRACT

The existing kernel filtering algorithms are classified into two categories depending on what space the optimization is formulated in. This paper bridges the two different approaches by focusing on the isomorphism between the dictionary subspace and a Euclidean space with the inner product defined by the kernel matrix. Based on the isomorphism, we propose a novel kernel adaptive filtering algorithm which adaptively refines the dictionary and thereby achieves excellent performance with a small dictionary size. Numerical examples show the efficacy of the proposed algorithm.

1. INTRODUCTION

We address an adaptive estimation problem of a nonlinear system $\psi : \mathcal{U} \to \mathbb{R}$ with sequentially arriving input-output pairs $(\boldsymbol{u}_n, d_n)_{n\in\mathbb{N}} \subset \mathcal{U} \times \mathbb{R}$. Here, the input space $\mathcal{U}$ is a compact subset of the L dimensional Euclidean space $\mathbb{R}^L$. Kernel adaptive filtering is an attractive approach for this task [1–11]. In kernel adaptive filtering, ψ is estimated by an element of a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ associated with a prespecified positive definite kernel [12] $\kappa : \mathcal{U} \times \mathcal{U} \to \mathbb{R},\ (\boldsymbol{x}, \boldsymbol{y}) \mapsto \kappa(\boldsymbol{x}, \boldsymbol{y})$. A kernel adaptive filter $\varphi_n : \mathcal{U} \to \mathbb{R}$ at time $n \in \mathbb{N}$ is given by

$$\varphi_n(\cdot) = \sum_{j\in\mathcal{J}_n} h_{j,n}\kappa(\cdot, \boldsymbol{u}_j),\ n \in \mathbb{N}, \tag{1}$$

where $h_{j,n} \in \mathbb{R}$ are the filter coefficients and $\mathcal{J}_n := \{j_1^{(n)}, j_2^{(n)}, \cdots, j_{r_n}^{(n)}\} \subset \{0, 1, \cdots, n\}$ indicates the dictionary $\{\kappa(\cdot, \boldsymbol{u}_j)\}_{j\in\mathcal{J}_n}$ which is assumed linearly independent.

The kernel least mean square (KLMS) algorithm [3] updates the filter only when the current input datum $\boldsymbol{u}_n$ is added into the dictionary. The quantized KLMS (QKLMS) algorithm [8] eliminates such limitation by updating the coefficient of a dictionary element which is maximally coherent to $\kappa(\cdot, \boldsymbol{u}_n)$. A more systematic scheme has been proposed in [9] under the name of *hyperplane projection along affine subspace (HYPASS)*, using the projection of $\kappa(\cdot, \boldsymbol{u}_n)$ onto the dictionary subspace $\mathcal{M}_n := \mathrm{span}\{\kappa(\cdot, \boldsymbol{u}_j)\}_{j\in\mathcal{J}_n} \subset \mathcal{H}$. Specifically, it is based on the following optimization problem:

$$\min_{\varphi\in\Pi_n} \|\varphi - \varphi_n\|_{\mathcal{H}},\ \ n \in \mathbb{N}, \tag{2}$$

where $\Pi_n := \{\varphi \in \mathcal{M}_n : \varphi(\boldsymbol{u}_n) = \langle\varphi, \kappa(\cdot, \boldsymbol{u}_n)\rangle_{\mathcal{H}} = d_n\}$. Here, $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ and $\|\cdot\|_{\mathcal{H}}$ denote the inner product and the norm defined in $\mathcal{H}$, respectively. All those algorithms formulate the optimization problem in the RKHS $\mathcal{H}$, and thus we classify them into the RKHS approach (cf. [7]). The algorithms presented in [1, 5, 8, 10] also share the same spirit.

In contrast, the kernel normalized least mean square (KNLMS) algorithm [4] is based on the following optimization problem:

$$\min_{\boldsymbol{h}\in H_n} \|\boldsymbol{h} - \boldsymbol{h}_n\|,\ \ n \in \mathbb{N}, \tag{3}$$

where $\boldsymbol{h}_n := \left[h_{j_1^{(n)},n}, h_{j_2^{(n)},n}, \cdots, h_{j_{r_n}^{(n)},n}\right]^{\mathsf{T}}$ and $H_n := \{\boldsymbol{h} \in \mathbb{R}^{r_n} : \langle\boldsymbol{\kappa_n}, \boldsymbol{h}\rangle = d_n\}$ is a zero-instantaneous-error hyperplane with the kernelized input vector $\boldsymbol{\kappa}_n := [\kappa(\boldsymbol{u}_n, \boldsymbol{u}_{j_1^{(n)}}), \kappa(\boldsymbol{u}_n, \boldsymbol{u}_{j_2^{(n)}}), \cdots, \kappa(\boldsymbol{u}_n, \boldsymbol{u}_{j_{r_n}^{(n)}})]^{\mathsf{T}}$. Here, $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ denote the canonical inner product and the Euclidean norm defined in $\mathbb{R}^L$, respectively. This algorithm formulates the optimization problem in the parameter space $\mathbb{R}^L$, and thus we classify it into the parameter-space approach (cf. [7]). The algorithms presented in [2, 7] also share the same spirit. To the best of the authors' knowledge, there has been no literature that studies the relation between the two distinct approaches.

The first contribution of this paper is to provide a basis to clarify the relationship between the two approaches. We show that the dictionary subspace $\mathcal{M}_n$ and an r_n-dimensional Euclidean space with an inner product defined with the kernel matrix, say $\boldsymbol{G}_n$, are isomorphic. This means that the learning in $\mathcal{M}_n$ can be regarded as the learning in $\mathbb{R}^{r_n}$ with the particular $\boldsymbol{G}_n$ inner product. Based on the isomorphism between $\mathcal{M}_n$ and $\mathbb{R}^{r_n}$, we define the *restricted gradient*, which is the gradient of the cost functional under the restriction to $\mathcal{M}_n$. The restricted gradient, together with the isomorphism, provides a way to view the behaviors of the two approaches in a common space, either in $\mathcal{M}_n$ or in $\mathbb{R}^{r_n}$. It turns out that one cannot generally say that one of the two approaches is better than the other.

The second contribution is to derive a promising RKHS-type algorithm that suppresses the weighted squared-distance functions penalized by the popular ℓ_1 norm; the penalty term is for the sake of adaptive refinements of the dictionary. A straightforward approach is to apply the adaptive proximal forward-backward splitting (APFBS) algorithm [13] to the cost function (which is the sum of smooth and nonsmooth functions) under the $\boldsymbol{G}_n$ inner product. However, the proximity operator defined with the $\boldsymbol{G}_n$ inner product does not work well when $\boldsymbol{G}_n$ has a large eigenvalue spread. We therefore propose a heuristic, but efficient, algorithm that employs the proximity operator defined with the standard inner product. Although the proposed algorithm uses different inner products between the forward and backward steps, we show that it still enjoys a monotone approximation property regarding a cost function with a certain modified weighted ℓ_1 norm under some conditions. The proposed algorithm also enjoys fast convergence due to the use of parallel projection (data reusing). The numerical examples show that the proposed algorithm enjoys a high adaptation-capability while maintaining a small dictionary size and low computational complexity.

Relation to prior work: An adaptive dictionary-refinement technique based on the proximity operator of a weighted (block) ℓ_1 norm for kernel adaptive filtering has first been proposed by Yukawa in 2011 [7, 14] for the parameter-space approach in the multikernel adaptive filtering context. A sim-

This work was supported by KDDI Foundation.

ilar algorithm (for the monokernel case) has been proposed and analyzed by Gao *et al.* in 2013 [15]. The sparse QKLMS algorithm has been proposed by Chen *et al.* in 2012 [16]; this algorithm is based on the subgradient method and has no guarantee of monotone approximation.

2. ISOMORPHISMS OF A FUNCTIONAL SUBSPACE AND A EUCLIDEAN SPACE

2.1. Viewing RKHS approach in parameter-space

Define the $r_n \times r_n$ kernel matrix $\boldsymbol{G}_n$ whose (s,t) entry is given by $[\boldsymbol{G}_n]_{s,t} := \kappa(\boldsymbol{u}_{j_s^{(n)}}, \boldsymbol{u}_{j_t^{(n)}})$, where $1 \le s,t \le r_n$ (r_n is the dictionary size). The matrix $\boldsymbol{G}_n$ is ensured to be positive definite due to the assumption that the dictionary is linearly independent.[1] We can therefore define an inner product by $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\boldsymbol{G}_n} := \boldsymbol{x}^{\mathsf{T}} \boldsymbol{G}_n \boldsymbol{y},\ \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^{r_n}$.

Lemma 1 *A pair of real Hilbert spaces* $(\mathcal{M}_n, \langle \cdot,\cdot \rangle_{\mathcal{H}})$ *and* $\left(\mathbb{R}^{r_n}, \langle \cdot,\cdot \rangle_{\boldsymbol{G}_n}\right)$ *are isomorphic under the correspondence*

$$\mathcal{M}_n \ni \varphi := \sum_{j \in \mathcal{J}_n} h_j \kappa(\cdot, \boldsymbol{u}_j) \longleftrightarrow [h_{j_1^{(n)}}, h_{j_2^{(n)}}, \cdots, h_{j_{r_n}^{(n)}}]^{\mathsf{T}} =: \boldsymbol{h} \in \mathbb{R}^{r_n}. \quad (4)$$

Proof: Because the dictionary is linearly independent, the correspondence is clearly a bijective mapping. The inner product of φ and $\hat{\varphi} := \sum_{j \in \mathcal{J}_n} \hat{h}_j \kappa(\cdot, \boldsymbol{u}_j)$ is $\langle \varphi, \hat{\varphi} \rangle_{\mathcal{H}} = \sum_{j \in \mathcal{J}_n} \sum_{j \in \mathcal{J}_n} h_i \hat{h}_j \kappa(\boldsymbol{u}_i, \boldsymbol{u}_j) = \boldsymbol{h}^{\mathsf{T}} \boldsymbol{G}_n \hat{\boldsymbol{h}} = \left\langle \boldsymbol{h}, \hat{\boldsymbol{h}} \right\rangle_{\boldsymbol{G}_n}$. This verifies that the bijective mapping is inner product preserving. □

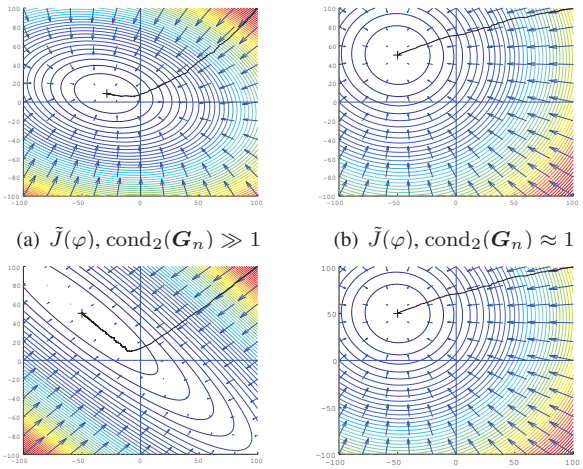
Lemma 1 states that the learning in $\mathcal{M}_n$ can be regarded as the learning in $\mathbb{R}^{r_n}$ with the inner product $\langle \cdot,\cdot \rangle_{\boldsymbol{G}_n}$. This reveals that the KNLMS and (fully-updating version of) HYPASS algorithms can be regarded as operating the projection onto the same hyperplane $H_n \subset \mathbb{R}^{r_n}$ with the canonical and $\boldsymbol{G}_n$ inner products, respectively. Note here that Π_n and H_n in (2) and (3) can be regarded to be the same under the correspondence in (4).

2.2. Restricted gradient and error surface consideration

We reconsider the two approaches from a stochastic-gradient viewpoint. It is straightforward to derive a stochastic-gradient method for the mean squared error (MSE) cost function $J(\boldsymbol{h}) := E[\{d_n - \langle \boldsymbol{h}, \boldsymbol{\kappa}_n \rangle\}^2]$. On the other hand, it is not straightforward to derive a stochastic-gradient method for $\tilde{J}(\varphi) := E[\{d_n - \langle \varphi, \kappa(\cdot, \boldsymbol{u}_n) \rangle_{\mathcal{H}}\}^2]$ in such a way that the learning is done within the dictionary subspace $\mathcal{M}_n$. We therefore define the gradient of $\tilde{J}(\varphi)$ at $\varphi \in \mathcal{M}_n$ under the restriction to the dictionary subspace $\mathcal{M}_n$; the *restricted gradient* is denoted by $\nabla_{|\mathcal{M}_n} \tilde{J}(\varphi)$. The direction $\Delta\varphi^*$ of the restricted gradient $\nabla_{|\mathcal{M}_n} \tilde{J}(\varphi)$ is given by

$$\Delta\varphi^* = \underset{\Delta\varphi \in \mathcal{M}_n,\ \|\Delta\varphi\|_{\mathcal{H}}=1}{\arg\max} \left\langle \nabla \tilde{J}(\varphi), \Delta\varphi \right\rangle_{\mathcal{H}}. \quad (5)$$

[1] The positive definiteness can be verified by noting that (i) the matrix $\boldsymbol{G}_n$ is automatically positive semidefinite by the definition of positive definite kernels and that (ii) the dictionary is linearly independent if and only if $\boldsymbol{h}^{\mathsf{T}} \boldsymbol{G}_n \boldsymbol{h} = 0 \Leftrightarrow \boldsymbol{h} = \boldsymbol{0},\ \boldsymbol{h} \in \mathbb{R}^{r_n}$.



(a) $\tilde{J}(\varphi), \mathrm{cond}_2(\boldsymbol{G}_n) \gg 1$ (b) $\tilde{J}(\varphi), \mathrm{cond}_2(\boldsymbol{G}_n) \approx 1$

(c) $J(\boldsymbol{h}), \mathrm{cond}_2(\boldsymbol{G}_n) \gg 1$ (d) $J(\boldsymbol{h}), \mathrm{cond}_2(\boldsymbol{G}_n) \approx 1$

Fig. 1. Equal error contours of $\tilde{\mathcal{J}}(\varphi)$ and $\mathcal{J}(\boldsymbol{h})$ for $r_n = 2$.

See [1] for the computation of $\nabla \tilde{J}(\varphi)$. The following proposition can easily be verified by Lemma 1.

Proposition 1 *The direction* $\Delta\varphi^*$ *of the restricted gradient* $\nabla_{|\mathcal{M}_n} \tilde{J}(\varphi)$ *given in (5) at* $\varphi (\longleftrightarrow \boldsymbol{h} \in \mathbb{R}^{r_n})$ *can be represented as follows* ($\alpha := [\nabla J(\boldsymbol{h})^{\mathsf{T}} \boldsymbol{G}_n^{-1} \nabla J(\boldsymbol{h})]^{-1/2} > 0$)*:*

$$\Delta\varphi^* \longleftrightarrow \Delta\boldsymbol{h}^* = \underset{\|\Delta\boldsymbol{h}\|_{\boldsymbol{G}_n}=1}{\arg\max} \left\langle \boldsymbol{G}_n^{-1} \nabla J(\boldsymbol{h}), \Delta\boldsymbol{h} \right\rangle_{\boldsymbol{G}_n} = \alpha \boldsymbol{G}_n^{-1} \nabla J(\boldsymbol{h}). \quad (6)$$

Definition 1 *The restricted gradient* $\nabla_{|M} \tilde{J}(\varphi)$ *is defined by*

$$\nabla_{|\mathcal{M}_n} \tilde{J}(\varphi) \longleftrightarrow \nabla_{\boldsymbol{G}_n} J(\boldsymbol{h}) := \boldsymbol{G}_n^{-1} \nabla J(\boldsymbol{h}). \quad (7)$$

While the error contours of the parameter-space approach is governed by $\boldsymbol{R} := E[\boldsymbol{\kappa}_n \boldsymbol{\kappa}_n^{\mathsf{T}}]$, those of the RKHS approach is governed by the modified autocorrelation matrix $\boldsymbol{G}_n^{-\frac{1}{2}} \boldsymbol{R} \boldsymbol{G}_n^{-\frac{1}{2}}$ as can be seen from the above arguments. Therefore, the error-contours are close to each other when the eigenvalue spread of $\boldsymbol{G}_n$ is close to the unity, while quite different when the eigenvalue spread is large. This is illustrated in Fig. 1 which depicts the equal-error contours of $\tilde{J}(\varphi)$ and $J(\boldsymbol{h})$ together with the behaviors of the associated approaches. It is seen that one cannot tell in general which of $\boldsymbol{R}$ and $\boldsymbol{G}_n^{-\frac{1}{2}} \boldsymbol{R} \boldsymbol{G}_n^{-\frac{1}{2}}$ is better conditioned, implying that one cannot tell in general which of the two approaches perform better.[2]

[2] Some may immediately think that the dictionary could be designed so that $\boldsymbol{G}_n^{-\frac{1}{2}} \boldsymbol{R} \boldsymbol{G}_n^{-\frac{1}{2}}$ is well conditioned, provided that an estimate of $\boldsymbol{R}$ is available. However, this straightforward intuition stems only from the aspect of the convergence speed. A more critical aspect to be considered in designing the dictionary is the representation ability which should be discussed apart from the convergence speed.

3. PROPOSED SPARSE ALGORITHM

3.1. Cost function and a straightforward idea

Define a sequence of convex functions $(\Theta_n)_{n\in\mathbb{N}}$ as follows:

$$\Theta_n(\boldsymbol{h}) := \Phi_n(\boldsymbol{h}) + \lambda\Omega_n(\boldsymbol{h}),\ \boldsymbol{h}\in\mathbb{R}^{r_n}, \tag{8}$$

where $\lambda > 0$ is the regularization parameter and

$$\Phi_n(\boldsymbol{h}) := \frac{1}{2}\sum_{\iota\in\mathcal{I}_n}\nu_\iota^{(n)} d^2_{\boldsymbol{G}_n}(\boldsymbol{h}, C_\iota^{(n)}) \quad \text{(smooth)}, \tag{9}$$

$$\Omega_n(\boldsymbol{h}) := \|\boldsymbol{w}_n\circ\boldsymbol{h}\|_1 \quad \text{(nonsmooth)}. \tag{10}$$

Here, $\Phi_n(\boldsymbol{h})$ is a weighted squared-distance function with $\nu_\iota^{(n)} > 0$ satisfying $\sum_{\iota\in\mathcal{I}_n}\nu_\iota^{(n)} = 1$, $\iota\in\mathcal{I}_n := \{n, n-1, \cdots, n-p+1\}$, and $d_{\boldsymbol{G}_n}(\boldsymbol{h}, C_\iota^{(n)}) := \min_{\hat{\boldsymbol{h}}\in C_\iota^{(n)}}\left\|\boldsymbol{h}-\hat{\boldsymbol{h}}\right\|_{\boldsymbol{G}_n}$ denotes the metric distance to the closed convex sets:

$$C_\iota^{(n)} := \left\{\boldsymbol{h}\in\mathbb{R}^{r_n} : \left(\left\langle\boldsymbol{h}, \boldsymbol{G}_n^{-1}\boldsymbol{\kappa}_n\right\rangle_{\boldsymbol{G}_n} - d_\iota\right)^2 \le \rho\right\},\ \iota\in\mathcal{I}_n,$$

where $\rho\ge 0$. Note that $C_\iota^{(n)}$s accommodate the p most recent data so that the algorithm attains fast convergence. The second term $\Omega_n(\boldsymbol{h})$ is the weighted l_1 norm, for dictionary sparsification (refinement), with the weights $\boldsymbol{w}_n := [w^{(n)}_{j_1^{(n)}}, w^{(n)}_{j_2^{(n)}}, \cdots, w^{(n)}_{j_{r_n}^{(n)}}]^\mathsf{T}$, $w_j^{(n)} > 0$, $\forall j\in\mathcal{J}_n$; $\boldsymbol{w}_n\circ\boldsymbol{h}$ denotes the Hadamard product of $\boldsymbol{w}_n$ and $\boldsymbol{h}$.

A natural idea in the light of Section 2 would be to apply APFBS to the function sequence $(\Theta_n)_{n\in\mathbb{N}}$ with the inner product $\langle\cdot,\cdot\rangle_{\boldsymbol{G}_n}$. This straightforward approach, however, does not work well due to the following two reasons. First, the proximity operator of Ω_n in the Hilbert space $(\mathbb{R}^{r_n}, \langle\cdot,\cdot\rangle_{\boldsymbol{G}_n})$ has no closed form expression. Second, even if we compute it by an iterative algorithm, e.g. the proximal forward-backward splitting method, efficient dictionary-refinements are not achieved when the eigenvalues spread of $\boldsymbol{G}_n$ is large. (This happens when coherent data exist in the dictionary.) This motivates us to propose a modified algorithm presented in the following subsection.

3.2. Proposed sparse algorithm

The proposed algorithm employs the canonical inner product $\langle\cdot,\cdot\rangle_{\boldsymbol{I}}$ for the proximity operator (backward step) while employing the different inner product $\langle\cdot,\cdot\rangle_{\boldsymbol{G}_n}$ for the gradient (forward step). This allows a closed-form expression of the proximity operator and also brings efficient dictionary-refinements.

Algorithm 1 (Φ-PASS II) *For the initial estimate $\boldsymbol{h}_0 := 0$, generate the sequence $(\boldsymbol{h}_n)_{n\in\mathbb{N}}$ by*

$$\boldsymbol{h}_{n+1} := T\left[\mathrm{prox}^{\boldsymbol{I}}_{\mu_n\lambda\Omega_n}\left(\boldsymbol{h}_n - \mu_n\nabla_{\boldsymbol{G}_n}\Phi_n(\boldsymbol{h}_n)\right)\right] \tag{11}$$

where $\mu_n\in[0,2]$ is the step size, the proximity operator is defined as

$$\mathrm{prox}^{\boldsymbol{I}}_{\mu_n\lambda\Omega_n}(\boldsymbol{x}) := \underset{\boldsymbol{y}\in\mathbb{R}^{r_n}}{\arg\min}\left(f(\boldsymbol{y}) + \frac{1}{2\mu_n\lambda}\|\boldsymbol{x}-\boldsymbol{y}\|^2_{\boldsymbol{I}}\right), \tag{12}$$

Table 1. Summary of the proposed algorithm.

The Φ-PASS II algorithm

Requirement : step size $\mu_n\in[0,2]$
Initialization : $\mathcal{J}_{-1} := \emptyset$
Filter output : $\varphi_n(\boldsymbol{u}_n) := \sum_{j\in\mathcal{J}_n} h_{j,n}\kappa(\boldsymbol{u}_n, \boldsymbol{u}_j)$
Filter update :
1. Define $\mathcal{J}_n$ based on the coherence criterion [9]
$$\mathcal{J}_n := \begin{cases}\{j\in\mathcal{J}_{n-1} : h_{j,n-1}\neq 0\}\cup\{n\},\\ \quad\text{if } \max_{j\in\mathcal{J}_n}\frac{|\kappa(\boldsymbol{u}_n,\boldsymbol{u}_j)|}{\sqrt{\kappa(\boldsymbol{u}_n,\boldsymbol{u}_n)}\sqrt{\kappa(\boldsymbol{u}_j,\boldsymbol{u}_j)}}\le\sigma,\\ \{j\in\mathcal{J}_{n-1} : h_{j,n-1}\neq 0\}, \quad\text{otherwise,}\end{cases}$$
where $\sigma > 0$.
2. If $n\in\mathcal{J}_n$, let $h_{n,n} := 0$.
3. $P^{\boldsymbol{G}_n}_{C_\iota^{(n)}}(\boldsymbol{h}_n) = \boldsymbol{h}_n + \varsigma_\iota^{(n)}\frac{|d_\iota - \boldsymbol{h}_n^\mathsf{T}\boldsymbol{\kappa}_\iota| - \sqrt{\rho}}{\boldsymbol{\kappa}_\iota^\mathsf{T}\boldsymbol{G}_n^{-1}\boldsymbol{\kappa}_\iota}\boldsymbol{G}_n^{-1}\boldsymbol{\kappa}_\iota$, $\iota\in\mathcal{I}_n$,
where $\varsigma_\iota^{(n)} := 0$, if $\left|d_\iota - \left\langle\boldsymbol{h}_n, \boldsymbol{G}_n^{-1}\boldsymbol{\kappa}_\iota\right\rangle_{\boldsymbol{G}_n}\right| \le \sqrt{\rho}$,
and $\varsigma_\iota^{(n)} := \mathrm{sgn}(d_\iota - \left\langle\boldsymbol{h}_n, \boldsymbol{G}_n^{-1}\boldsymbol{\kappa}_\iota\right\rangle_{\boldsymbol{G}_n})$, otherwise.
4. $\hat{\boldsymbol{h}}_n = \boldsymbol{h}_n + \mu_n\left(\sum_{\iota\in\mathcal{I}_n}\nu_\iota^{(n)} P^{\boldsymbol{G}_n}_{C_\iota^{(n)}}(\boldsymbol{h}_n) - \boldsymbol{h}_n\right)$
5. $h_{j,n+1}$
$= \mathrm{sgn}(\hat{h}_{j,n})\max\{|\hat{h}_{j,n}| - \mu_n\lambda w_j^{(n)}, 0\},\ j\in\mathcal{J}_n$

and $T : \mathbb{R}^{r_n}\to\mathbb{R}^{r_{n+1}}$ is the operator (i) that removes the zero components and (ii) that adds zero as a new entry at the bottom of the vector if the current datum has significant novelty for the current dictionary.

The summary of the Φ-PASS II algorithm is presented in Table 1, in which $\mathrm{sgn}(\cdot)$ denotes the signum function defined as $\mathrm{sgn}(x) = 1$, if $x\ge 0$, $\mathrm{sgn}(x) = -1$, if $x < 0$.

Although Φ-PASS II uses different inner products between the forward and backward steps, a monotone approximation property still holds for a modified cost function with a certain modified weighted ℓ_1 norm under some conditions, as shown in the following proposition.

Proposition 2 (Monotone approximation) *Assume that (A1) $\mathrm{sgn}(\boldsymbol{G}_n\boldsymbol{W}_n\boldsymbol{a}) = \mathrm{sgn}(\boldsymbol{a})$, $\forall\boldsymbol{a}\in\{1,-1\}^{r_n}$, and (A2) $\hat{\boldsymbol{h}}_n := \boldsymbol{h}_n - \mu_n\nabla_{\boldsymbol{G}_n}\Phi_n(\boldsymbol{h}_n)\in\mathcal{D}_n := \{\boldsymbol{h}\in\mathbb{R}^{r_n} : |h_i| > \mu_n\lambda w^{(n)}_{j_i^{(n)}}, i = 1,2,\cdots,r_n\}$. Then, Algorithm 1 satisfies the monotone approximation property:*

$$\left\|\tilde{\boldsymbol{h}}_{n+1} - \boldsymbol{h}^*\right\|_{\boldsymbol{G}_n} < \|\boldsymbol{h}_n - \boldsymbol{h}^*\|_{\boldsymbol{G}_n} \tag{13}$$

for any $\boldsymbol{h}^*\in\mathcal{S}_n := \underset{\boldsymbol{h}\in\mathbb{R}^{r_n}}{\arg\min}\tilde{\Theta}_n(\boldsymbol{h})$, if $\boldsymbol{h}_n\notin\mathcal{S}_n\neq\emptyset$, where $\tilde{\boldsymbol{h}}_{n+1} = \mathrm{prox}^{\boldsymbol{I}}_{\mu_n\lambda\Omega_n}(\hat{\boldsymbol{h}}_n)$ and

$$\tilde{\Theta}_n(\boldsymbol{h}) := \Phi_n(\boldsymbol{h}) + \lambda\tilde{\Omega}_n(\boldsymbol{h}),\ \boldsymbol{h}\in\mathbb{R}^{r_n} \tag{14}$$

with a modified weighted l_1 norm $\tilde{\Omega}_n(\boldsymbol{h}) = \|\tilde{\boldsymbol{w}}_n\circ\boldsymbol{h}\|_1$, $\boldsymbol{h}\in\mathbb{R}^{r_n}$. Here, $\tilde{\boldsymbol{w}}_n := \boldsymbol{G}_n\boldsymbol{W}_n\,\mathrm{sgn}(\hat{\boldsymbol{h}}_n)$ with $\boldsymbol{W}_n := \mathrm{diag}(\boldsymbol{w}_n)$.

Sketch of proof: By the assumptions (A1) and (A2), we have $\mathrm{sgn}(\tilde{\boldsymbol{w}}_n) = \mathrm{sgn}(\hat{\boldsymbol{h}}_n) = \mathrm{sgn}(\tilde{\boldsymbol{h}}_{n+1})$. Hence, it follows that $\partial_{\boldsymbol{I}}\Omega_n(\tilde{\boldsymbol{h}}_{n+1}) = \{\boldsymbol{W}_n\,\mathrm{sgn}(\tilde{\boldsymbol{h}}_{n+1})\} = \boldsymbol{G}_n^{-1}\{\tilde{\boldsymbol{w}}_n\} = \boldsymbol{G}_n^{-1}\partial_{\boldsymbol{I}}\tilde{\Omega}_n(\tilde{\boldsymbol{h}}_{n+1}) = \partial_{\boldsymbol{G}_n}\tilde{\Omega}_n(\tilde{\boldsymbol{h}}_{n+1})$. Here, for a continuous

Table 2. Computational complexity of the proposed and conventional algorithms.

Proposed	$O(r^3)+(r^2+r)L/2$ $+p(r^2+3r)+3r$
Proposed (low-complexity)	$p[O(s^3)+(s^2-s)L/2$ $+s^2+2s+2r]+3r+rL$
Sparse QKLMS [16]	$O((r-1)^2)+rL+r^2+2r$
FOBOS-KLMS [15]	$5r+rL$

convex function $f:\mathbb{R}^{r_n}\to\mathbb{R}$ and a positive definite matrix $\boldsymbol{A}\in\mathbb{R}^{r_n\times r_n}$, $\partial_{\boldsymbol{A}}f(\boldsymbol{x}):=\{\tilde{\boldsymbol{x}}\in\mathbb{R}^{r_n}:\langle\boldsymbol{y}-\boldsymbol{x},\tilde{\boldsymbol{x}}\rangle_{\boldsymbol{A}}+f(\boldsymbol{x})\le f(\boldsymbol{y}),\ \forall\boldsymbol{y}\in\mathbb{R}^{r_n}\}\neq\emptyset$ denotes the subdifferential of f at $\boldsymbol{x}\in\mathbb{R}^{r_n}$. Since $\mathrm{prox}^{\boldsymbol{I}}_{\mu_n\lambda\Omega_n}=(I+\mu_n\lambda\partial_{\boldsymbol{I}}\Omega_n)^{-1}$, it holds that $\hat{\boldsymbol{h}}_n-\tilde{\boldsymbol{h}}_{n+1}\in\mu_n\lambda\partial_{\boldsymbol{I}}\Omega_n(\tilde{\boldsymbol{h}}_{n+1})=\mu_n\lambda\partial_{\boldsymbol{G}_n}\tilde{\Omega}_n(\tilde{\boldsymbol{h}}_{n+1})$, implying that $\tilde{\boldsymbol{h}}_{n+1}=\mathrm{prox}^{\boldsymbol{G}_n}_{\mu_n\lambda\tilde{\Omega}_n}(\hat{\boldsymbol{h}}_n)$. This verifies the claim (cf. [13]). □

The assumption (A1) holds, for instance, if $\boldsymbol{W}_n=\boldsymbol{I}$ and $\boldsymbol{G}_n$ is diagonally dominant. The assumption (A2) is violated if $\hat{\boldsymbol{h}}_n$ contains some nearly zero components. In such a case, however, those minor components are discarded and it does not seriously affect the overall performance, as will be shown in Section 4.

The computational complexity of the Φ-PASS II algorithm can be reduced by selecting and updating only a few, say $s\le r_n$, coefficients of $\kappa(\cdot,\boldsymbol{u}_j)$ that are maximally coherent to $\kappa(\cdot,\boldsymbol{u}_\iota)$, $\iota\in\mathcal{I}_n$. See [10] for this low-complexity strategy. The computational complexity of the proposed algorithm and the related algorithms is presented in Table 2. The low complexity version of the proposed algorithm is quite efficient since the number of selected coefficients to be updated is typically $s=1$ or $s=2$.

4. NUMERICAL EXAMPLES

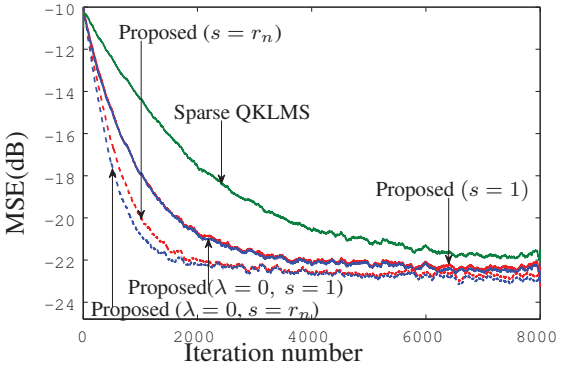
We compare the performance of the Φ-PASS II algorithm with its non-sparse counterpart (i.e., $\lambda=0$) and the sparse QKLMS algorithm [16] in an application to noise cancellation.[3] The noise signal x_n is assumed white and uniformly distributed within the range of $[-0.5,0.5]$, and the distorted noise signal is given by

$$d_n=x_n-0.3d_{n-1}-0.8d_{n-1}x_{n-1}+0.2x_{n-1}+0.4d_{n-2}.$$

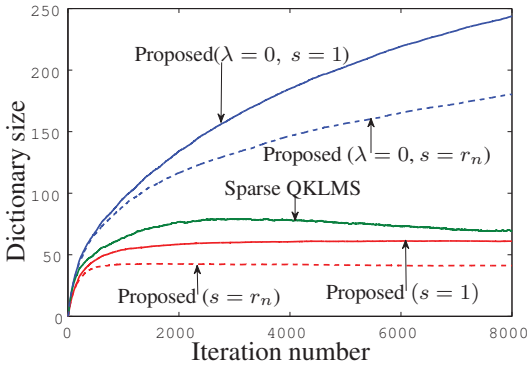
The original noise x_n is predicted as a function of $\boldsymbol{u}_n:=[d_n,d_{n-1},\cdots,d_{n-L+2},\hat{x}_{n-1}]^T\in\mathcal{U}\subset\mathbb{R}^L$ $(L=12)$, where $\hat{x}_{n-1}:=\varphi_{n-1}(\boldsymbol{u}_{n-1})$ is a replica of x_{n-1}. We employ the Gaussian kernel $\kappa(\boldsymbol{x},\boldsymbol{y}):=\exp(-\zeta\|\boldsymbol{x}-\boldsymbol{y}\|^2)$ for $\zeta=6$.

For the proposed algorithm, the full version $s=r_n$ and the low-complexity version $s=1$ are tested and the data-reusing factor is set to $p=8$. The step size is set to $\mu_n=0.7$ for the proposed algorithms and $\eta=0.3$ for Sparse QKLMS. (The step size is chosen so that each algorithm attains the best performance.) The regularization parameter is set to $\lambda=3\times10^{-5}$ for the proposed algorithms, and $\gamma=3\times10^{-6}$ for Sparse QKLMS. The weight of the l_1 norm is set to $w_j^{(n)}:=1/(|h_j^{(n)}|+\epsilon)$, $j\in\mathcal{J}_n$, for $\epsilon:=1\times10^{-4}$. For Sparse QKLMS algorithm, the regularization parameter for the kernel matrix $\boldsymbol{K}_n$ is set to $\lambda=1\times10^{-4}$. Uniform weights are used; i.e., $\nu_\iota^{(n)}=(\min\{p,n+1\})^{-1}$ for all $\iota\in\mathcal{I}_n$, and the error bound is set to $\rho=0$. The coherence threshold σ is set to $\sigma=0.75$ for all algorithms. For Sparse QKLMS, those dictionary elements whose coefficients have their absolute values smaller than 0.01 are discarded at each iteration.

[3] FOBOS-KLMS did not perform well in this experiment. This is because the off-diagonal entries of $\boldsymbol{G}_n$ are non-negligibly large and the error surface for FOBOS-KLMS is unfavorable such as the one depicted in Fig. 1(c).



(a) MSE learning curves.



(b) Dictionary size growing curves.

Fig. 2. Simulation results.

Fig. 2(a) depicts the MSE learning curves and Fig. 2(b) the time evolution of the dictionary size. It can be seen that the performance of Proposed ($s=1$) is almost identical to that of Proposed ($\lambda=0$, $s=1$) while it maintains a significantly small dictionary size. Moreover, the average complexities of Proposed ($s=1$) and sparse QKLMS are 1820 and 10959, respectively. Proposed ($s=1$) outperforms Sparse QKLMS despite its lower complexity as well as its smaller dictionary size.

5. CONCLUSION

We proposed the Φ-PASS II algorithm which adaptively refines the dictionary by a shrinkage operator and suppresses the estimation errors by parallel projections with past data reused. We showed a monotone approximation property of the proposed algorithm under conditions. The algorithm was derived based on the isomorphism between the dictionary subspace and a Euclidean space, which, together with the restricted gradient, provides a basis to clarify the relation of the RKHS and parameter-space approaches. The numerical examples showed the efficacy of the proposed algorithm.

6. REFERENCES

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

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

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

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

[5] K. Slavakis, S. Theodoridis, and I. Yamada, "Adaptive constrained learning in reproducing kernel Hilbert spaces: the robust beamforming case," *IEEE Trans. Signal Process.*, vol. 57, no. 12, pp. 4744–4764, Dec. 2009.

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

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

[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, 2012.

[9] M. Yukawa and R. Ishii, "An efficient kernel adaptive filtering algorithm using hyperplane projection along affine subspace," in *Proc. EUSIPCO*, 2012, pp. 2183–2187.

[10] T. Masa-aki and M. Yukawa, "An efficient data-reusing kernel adaptive filtering algorithm based on parallel hyperslab projecton along affine subspace," in *IEEE ICASSP*, 2013, pp. 3557–3561.

[11] 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.

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

[13] 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.

[14] M. Yukawa, "Nonlinear adaptive filtering techniques with multiple kernels," in *Proc. EUSIPCO*, 2011, pp. 136–140.

[15] W. Gao, J. Chen, C. Richard, and J. Huang, "Online dictionary learning for kernel LMS analysis and forward-backward splitting algorithm," in *IEEE Trans. Signal Process.*, 2013, submitted.

[16] 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.