Projected Kernel Least Mean p-power Algorithm: Convergence Analyses and Modifications

Ji Zhao, Hongbin Zhang, *Senior Member, IEEE*, Gang Wang, Jian Andrew Zhang, *Senior Member, IEEE*

***Abstract*—Sparsified kernel adaptive filters (SKAFs) is an attractive filtering solution with low memory and computational complexity. Most of existing SKAFs are based on the mean square error (MSE) criterion under Gaussian noise assumption for its simplicity and convenience. When the assumption deviates largely from the underlying truth, the performance of these methods could degrade significantly. In this paper, we propose a novel SKAF, named as projected kernel least mean p-power algorithm (PKLMP), based on the mean p-power error (MPE) criterion and vector projection (VP) method. We provide convergence analyses in terms of the stead-state MSE, based on a Taylor expansion method, and derive the lower and upper bounds for the steady-state excess MSE. We also conduct mean convergence analysis for PKLMP, and derive convergence conditions. To exploit the information in the desired outputs, we further derive a modified PKLMP by smoothing the desired signal. Finally, a simple and effective online variable kernel centers strategy is proposed to improve the filtering performance of the proposed KAFs. Simulation results under a static function estimation, a chaotic time-series prediction, and two real-world time-series predictions are conducted and validate the effectiveness of the proposed PKLMP algorithms.**

***Index Terms*—Sparsification, Smoothed output, Variable center, Kernel adaptive filter, Least mean p-power, Convergence analysis.**

I. Introduction

THE kernel method has been widely used to solve nonlinear problems, e.g., time series prediction, nonlinear regression, patter classification, and image processing [1]. However, most kernel methods are batch algorithms having computational complexity and memory costing in the order of $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, respectively [2]. This limits kernel methods to real-time applications. As an online version of kernel method, the kernel adaptive filter (KAF) is receiving increasing attention in the signal processing community.

Among the best known kernel adaptive filtering algorithms are the kernel recursive least-squares algorithm (KRLS) [3], the kernel least-mean-square algorithm (KLMS) [2], the kernel affine projection algorithm [4], and the kernel normalized least-mean-square algorithm [5]. Compared with their corresponding linear counterparts, these KAFs can achieve faster convergence rate and superior filtering accuracy in nonlinear system identification problems.

The main drawback of KAFs is that the functional representation of classical kernel-based algorithms grows linearly with the number of processed data, which results in increased memory and computational complexity. To overcome this problem, a variety of sparsification methods have been proposed, e.g., approximate linear dependency [3], coherence criterion (CC) [5], novelty criterion (NC) [6], surprise criterion [7], vector quantization (VQ) method [8], sparsity-promoting regularization [9], fixed memory budget model [10], and sliding window method [11]. From these methods, sparsified KAFs (SKAFs) are derived and shown to be capable of effectively suppressing the growth of the structure. Recently, from the perspective of a feature space, we proposed a new and sample online sparsification method, called as vector projection (VP) method [12], [13]. VP can project the transformed input data to its most relevant center (MRC) in a dictionary. Similar to VQ, the VP method utilizes the discarded redundant data to update the coefficients of the MRC in the dictionary. Furthermore, VP can also exploit the hidden information from the input data to refine the corresponding coefficients for filtering accuracy improvement.

Note that almost all the aforementioned KAFs or sparsified ones are developed from the mean square error (MSE) criterion under the additive Gaussian noise assumption for mathematical simplicity and convenience. However, in practical circumstances, the signals are often contaminated by non-Gaussian and impulsive noises, and the KAFs can suffer from severe performance degradation owing to merely considering the second-order statistics contained in the MSE. Hence, it is necessary to find robust models to deal with non-Gaussian noises.

Recently, the mixed-norm criterion switching between the mean fourth error and MSE [14], and a kernel affine projection algorithm with sign error [15] were proposed to deal with impulsive noises modelled by Bernoulli-Gaussian (BG) distribution. However, the BG distribution is not suitable for modelling the fat-tailed noise process, which widely exists in physics and in various fields of engineering applications. The α-stable distribution with low probability but large amplitude is regarded as a better model. For this model, various adaptive algorithms based upon the least mean p-power error (MPE) criterion have been developed, because MPE is an appropriate measure of optimality for minimizing the fractional lower-order statistics of the prediction error [16]. The celebrated adaptive filters based on MPE include the least mean p-power (LMP) [16], the diffusion LMP [17], the smoothed LMP [18],

This work was supported in part by the National Natural Science Foundation of China (Grant nos. 61971100), the China Schoparship Council (Grant no. 201806070013).

J. Zhao, H. Zhang and G. Wang are with the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, PR China. (e-mail: zhaoji@std.uestc.edu.cn, zhanghb@uestc.edu.cn, wanggang_hld@uestc.edu.cn).

J. Andrew.Zhang is with the Global Big Data Technologies Centre, University of Technology Sydney, Sydney, NSW 2007 Australia. (e-mail: Andrew.Zhang@uts.edu.au).

the recursive least p-power (RLP) [19] and its combination [20]. Furthermore, MPE was applied into reproducing kernel Hibert space (RKHS) to develop the kernelized LMP (KLMP) [21] and kernelized RLP (KRLP) filters [22], which can be viewed as generalizations of KLMS and KRLS, respectively. However, both KLMP and KRLP also have the limitation of linearly increasing network structure.

In this paper, by introducing the MPE criterion and VP method, we propose a novel SKAF, named as projected kernel least mean p-power algorithm (PKLMP), which can overcome the aforementioned problems. The main contributions of this paper are as follows:

- We incorporate the VP method into KLMP to develop the PKLMP algorithm for overcoming the problem of growing kernel network;
- We provide an analytical sufficient condition for the convergence of PKLMP, in terms of the MSE. Based on a Taylor expansion method, we derive theoretical lower and upper bounds for the steady-state excess MSE (EMSE), which are applicable to numerous types of noises, including the α-stable noises (α-SN). For $p \in [1, 2)$, we also prove that PKLMP can achieve better filtering accuracy than quantized KLMS (QKLMS) [8] in the presence of impulsive noise;
- We provide mean convergence analysis for PKLMP, and derive conditions for convergence;
- We propose a modified PKLMP by smoothing the desired signal, called as PKLMP-SD, which can effectively deal with the situation when the desired outputs of the projected data are different to those of MRCs. This can happen when, e.g., the underlying systems are contaminated by α-SN;
- We also introduce a simple strategy to change online kernel centers, which can further improve the filtering performance of the proposed PKLMP algorithm.

The rest of this paper is organized as follows. In Section II, we first introduce KLMP, and then describe the concept of VP and derive PKLMP. In Section III, we conduct convergence analyses for PKLMP with different values of p. In Section IV, further modifications are proposed for PKLMP. Simulation results are presented in Section V to verify the effectiveness of PKLMP and the modified methods. Finally, concluding remarks are given in Section VI.

II. Projected kernel least mean p-power algorithm

Consider the learning of a continuous nonlinear input-output mapping $f : \mathbb{U} \to \mathbb{R}$ based on the sequence of input-output pairs $\{\boldsymbol{u}(n), d(n)\}_{n=1}^{+\infty}$, where $\boldsymbol{u}(n) \in \mathbb{U} \subset \mathbb{R}^l$ is the input data vector, and $d(n) \in \mathbb{R}$ is the desired output at discrete time n. KAF is a kernel-based sequential estimator of f such that f_n is refined on the basis of the last estimate f_{n-1} and current sample $\{\boldsymbol{u}(n), d(n)\}$. We start with revisiting the KLMP algorithm and the VP method, and then introduce our proposed PKLMP algorithm.

A. Kernel Least Mean p-power Algorithm

A Mercer kernel is a continuous, symmetric and positive-definite function $k(\cdot, \cdot) : \mathbb{U} \times \mathbb{U} \to \mathbb{R}$. The most important property of a Mercer kernel is the *kernel trick*, which allows inner-product based algorithms to be performed in a relatively high dimensional feature space $\mathbb{F}$ [24], i.e.,

$$k(\boldsymbol{u}, \boldsymbol{u}') = \langle \varphi(\boldsymbol{u}), \varphi(\boldsymbol{u}') \rangle_{\mathbb{F}}, \tag{1}$$

where $\boldsymbol{u} \in \mathbb{U}$, $\langle \cdot, \cdot \rangle_{\mathbb{F}}$ denotes the inner product in $\mathbb{F}$, $\varphi(\cdot)$ is a nonlinear mapping function induced by $\kappa(\cdot, \cdot)$, and $\varphi(\boldsymbol{u}) \in \mathbb{F}$. In addition, the $\mathbb{F}$ space is essentially the same as a RKHS induced by the kernel if we identify $\varphi(\boldsymbol{u}) = \kappa(\boldsymbol{u}, \cdot)$ [25]. In this work, we do not distinguish between these two spaces if no confusion arises.

KLMP is actually the linear LMP algorithm in $\mathbb{F}$ [21], [22]. First of all, $\varphi(\cdot)$ is applied to transform the input $\boldsymbol{u}(n)$ into $\mathbb{F}$. Then, applying the LMP algorithm to a new sample pair $\{\varphi(\boldsymbol{u}(n)), d(n)\}$, we obtain

$$\begin{cases} \boldsymbol{\Omega}(0) = 0 \\ e(n) = d(n) - \boldsymbol{\Omega}(n-1)^T \varphi(n) \\ \boldsymbol{\Omega}(n) = \boldsymbol{\Omega}(n-1) + \eta |e(n)|^{p-2} e(n) \varphi(n), \end{cases} \tag{2}$$

where $e(n)$ is the estimation error at iteration n, $\eta > 0$ is the step size, $p \geq 1$ ensures the convexity of MPE [31], [33], $\varphi(n) = \varphi(\boldsymbol{u}(n))$, $\boldsymbol{\Omega}(n)$ denotes the corresponding weight vector in $\mathbb{F}$, and T stands for the transposition of a matrix or vector. Actually, f_n is the linear composition of $\boldsymbol{\Omega}(n)$ and $\varphi(\cdot)$ [24], i.e., $f_n = \boldsymbol{\Omega}(n)^T \varphi(\cdot)$.

Hence, we define the learning rule in the input space as follows

$$\begin{cases} f_0 = 0 \\ e(n) = d(n) - f_{n-1}(\boldsymbol{u}(n)) \\ f_n = f_{n-1} + \eta |e(n)|^{p-2} e(n) \kappa(\boldsymbol{u}(n), \cdot). \end{cases} \tag{3}$$

Let $\boldsymbol{\omega}(n) = [\omega_1(n), \omega_2(n), \ldots, \omega_n(n)]^T$ be a kernel-weight vector (KWV) at instant n, and $\boldsymbol{\mathcal{D}}(n) = \{\mathcal{D}_1(n), \mathcal{D}_2(n), \ldots, \mathcal{D}_n(n)\}$ be a dictionary. Then f_n in (3) can be represented as

$$f_n = \sum_{l=1}^{n} \omega_l(n) \kappa(\mathcal{D}_l(n), \cdot), \tag{4}$$

where $\omega_l(n) = \eta |e(l)|^{p-2} e(l)$, $\mathcal{D}_l(n) = \boldsymbol{u}(l)$, $|\boldsymbol{\mathcal{D}}(n)|_c = n$, and $|\cdot|_c$ denotes the cardinality of a set.

Remark 1. Compared with KLMS [2], we can observe that KLMP is actually a variable-step-size KLMS with $\eta(n) = \eta |e(n)|^{p-2}$. Hence, they have almost the same computational complexity. When the α-SN is present, $e(n)$ will become larger, which results in a smaller $\eta(n)$ due to $1 \leq p < 2$. In this case, $\eta(n)$ is robust to larger outliers. However, in online learning, KLMP yields a growing kernel network by allocating a new kernel unit to every new sample.

B. Projected Kernel Least Mean p-power Algorithm

Recently, to suppress the growing structure of a KAF, we proposed a simple and efficient online vector projection (VP) method in $\mathbb{F}$ [12], [13]. In this paper, applying the VP method

to KLMP, we develop a projected kernel least mean p-power (PKLMP) algorithm.

Let

$$\begin{cases} j^\star = \arg\max\limits_{1\le j\le M} \cos(\varphi(n), \varphi(\mathcal{D}_j(n-1))) \\ \Theta = \cos\big(\varphi(n), \varphi(\mathcal{D}_{j^\star}(n-1))\big), \end{cases} \tag{5}$$

where the cosine-relation function $\cos(\mathbf{x},\mathbf{y}) = \frac{\langle \mathbf{x},\mathbf{y}\rangle_{\mathbb{F}}}{\|\mathbf{x}\|_{\mathbb{F}}\cdot\|\mathbf{y}\|_{\mathbb{F}}}$, with $\|\cdot\|_{\mathbb{F}}$ denoting the norm in the space $\mathbb{F}$. Using the idea of VP, PKLMP can be obtained by projecting $\varphi(n)$ on the weight-update equation $\boldsymbol{\Omega}(n) = \boldsymbol{\Omega}(n-1) + \eta|e(n)|^{p-2}e(n)\varphi(n)$ in (2), namely

$$\begin{cases} \boldsymbol{\Omega}(0) = 0 \\ e(n) = d(n) - \boldsymbol{\Omega}(n-1)^T\varphi(n) \\ \boldsymbol{\Omega}(n) = \boldsymbol{\Omega}(n-1) + \eta|e(n)|^{p-2}e(n)P(\varphi(n)), \end{cases} \tag{6}$$

where $P(\cdot)$ is the vector projection operator, and $P(\varphi(n)) = a(n)\varphi(\mathcal{D}_{j^\star}(n-1))$. Here, $a(n)$ is an approximation factor in VP and can be estimated as

$$a(n) = \begin{cases} \dfrac{\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))}{\|\varphi(\mathcal{D}_{j^\star}(n-1))\|_{\mathbb{F}}^2}, & \text{if} \quad \Theta \ge \varepsilon_c \\ 1, & \text{otherwise}, \end{cases} \tag{7}$$

where $\mathcal{D}_{j^\star}(n-1)$ is the $j^\star$th entry of a dictionary $\boldsymbol{\mathcal{D}}(n-1)$ with M codewords at iteration $n-1$, and ε_c is a pre-selected coherence threshold, which provides a tradeoff between the filtering accuracy and computation complexity.

The learning rule for PKLMP in the original input space $\mathbb{U}$ can then be represented as

$$\begin{cases} f_0 = 0, \quad e(n) = d(n) - f_{n-1}(\boldsymbol{u}(n)) \\ f_n = h_{-j^\star}^n + h_{j^\star}^n \\ h_{-j^\star}^n = \sum\limits_{l=1, l\ne j^\star}^{M} \omega_l(n-1)\kappa(\mathcal{D}_l(n-1), \cdot) \\ h_{j^\star}^n = (\omega_{j^\star}(n-1) + \eta|e(n)|^{p-2}a(n)e(n)) \\ \qquad \times \kappa(\mathcal{D}_{j^\star}(n-1), \cdot), \end{cases} \tag{8}$$

where $h_{-j^\star}^n$ contains all entries of $\boldsymbol{\omega}(n)$ except for the $j^\star$th, which is contained in $h_{j^\star}^n$. Therefore, we can obtain the PKLMP algorithm as summarized in Algorithm 1.

The main steps of Algorithm 1 are 3), 4) and 5). Based on the cosine-relation between $\varphi(n)$ and $\varphi(\boldsymbol{\mathcal{D}}(n-1))$, PKLMP decides whether to change the dictionary or not. In step 4), PKLMP keeps the dictionary unchanged, and uses $\eta|e(n)|^{p-2}a(n)e(n)$ to update the $j^\star$th entry of KWV at instant $n-1$. In step 5), the dictionary absorbs $\boldsymbol{u}(n)$ as its new element, and KWV uses $\eta|e(n)|^{p-2}e(n)$ to expand itself.

Remark 2. From Algorithm 1, one can see that PKLMP only absorbs the data which satisfies the significant criterion (i.e., $\Theta < \varepsilon_c$). With the evolution of adaption, PKLMP achieves a SKAF with $|\boldsymbol{\mathcal{D}}(n)|_c = M \ll n$. Although, we operate the vector projection in $\mathbb{F}$, as shown in (8), PKLMP can be effectively executed in $\mathbb{U}$.

Moreover, when $p = 2$, PKLMP becomes a PKLMS algorithm, and there are some close relationships between the PKLMS algorithm and several existing ones such as QKLMS and the modified QKLMS (MQKLMS) algorithm [23]. When

Algorithm 1: Projected Kernel Least Mean p-power (PKLMP) Algorithm

Initialization:
step size $\eta > 0$, Mercer kernel $\kappa(\cdot,\cdot)$
coherence threshold $0 < \varepsilon_c < 1$
dictionary $\boldsymbol{\mathcal{D}}(1) = \{\boldsymbol{u}(1)\}$ with $M = 1$
KWV $\boldsymbol{\omega}(1) = [\eta|d(1)|^{p-2}d(1)]^T$

Computation:
while $\{\boldsymbol{u}(n),\ d(n)\}$ $(n \ge 2)$ available **do**
1) the output:
$f_{n-1}(\boldsymbol{u}(n)) = \sum_{l=1}^{M} \omega_l(n-1)\kappa(\mathcal{D}_l(n-1), \boldsymbol{u}(n))$
2) the estimation error:
$e(n) = d(n) - f_{n-1}(\boldsymbol{u}(n))$
3) the cosine-relation between $\varphi(n)$ and $\varphi(\boldsymbol{\mathcal{D}}(n-1))$:
$\Theta = \max_{1\le j\le M} \frac{\langle \varphi(n), \varphi(\mathcal{D}_j(n-1))\rangle_{\mathbb{F}}}{\|\varphi(n)\|_{\mathbb{F}}\cdot\|\varphi(\mathcal{D}_j(n-1))\|_{\mathbb{F}}}$
4) **if** $\Theta \ge \varepsilon_c$
keep the dictionary unchanged:
$\boldsymbol{\mathcal{D}}(n) = \boldsymbol{\mathcal{D}}(n-1)$, $M \Leftarrow M$
update the coefficient of the MRC:
$\omega_{j^\star}(n) = \omega_{j^\star}(n-1) + \eta|e(n)|^{p-2}a(n)e(n)$
with $a(n) = \frac{\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))}{\|\varphi(\mathcal{D}_{j^\star}(n-1))\|_{\mathbb{F}}^2}$
5) **otherwise**
change the dictionary:
$\boldsymbol{\mathcal{D}}(n) = \{\boldsymbol{\mathcal{D}}(n-1), \boldsymbol{u}(n)\}$, $M \Leftarrow M+1$
update the KWV:
$\boldsymbol{\omega}(n) = [\boldsymbol{\omega}(n-1)^T, \eta|e(n)|^{p-2}e(n)]^T$
end while

a Gaussian kernel (with a kernel size h) is applied to PKLMS, we have

$$\begin{cases} a(n) = \kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1)) \\ f_n = h_{-Gj^\star}^n + h_{Gj^\star}^n \\ h_{Gj^\star}^n = \big(\omega_{j^\star}(n-1) + \eta e(n)\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))\big) \\ \qquad \times \kappa(\mathcal{D}_{j^\star}(n-1), \cdot), \end{cases} \tag{9}$$

where $h_{-Gj^\star}^n$ and $h_{Gj^\star}^n$ are similar to $h_{-j^\star}^n$ and $h_{j^\star}^n$, respectively, except that a Gaussian kernel is used in the former. In this case, PKLMS has the same update formations as MQKLMS. However, there exist significant differences between the two. MQKLMS uses the VQ method in the original space to control the network size, and it considers the term $\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))$ (in (9) of [23]) from the perspective of gradient method. Comparatively, PKLMS applies the VP method in a feature space to reduce the computational complexity, and the term $\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))$ in the approximation factor $a(n)$ is an integrated part of the VP method in PKLMS. Furthermore, when $\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1)) \approx 1$, we can get

$$\begin{cases} f_n = h_{-Gj^\star}^n + h_{Gj^\star}^{n1} \\ h_{Gj^\star}^{n1} = (\omega_{j^\star}(n-1) + \eta e(n))\kappa(\mathcal{D}_{j^\star}(n-1), \cdot), \end{cases} \tag{10}$$

and PKLMS becomes equivalent to QKLMS. Hence, the proposed PKLMP can be viewed as a generalization of MQKLMS and QKLMS.

TABLE I: Differences between PKLMS and other related algorithms with $h_{Gj^\star}^n = (j_a^\star + \eta j_b^\star) j_c^\star : j_a^\star = \omega_{j^\star}(n-1), j_c^\star = \kappa(\mathcal{D}_{j^\star}(n-1), \cdot)$.

Algorithm	Expressions of $j_b^\star$
PKLMS	$e(n)\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))$
QKLMS	$e(n)$
MQKLMS	$e(n)\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))$
HYPASS Q = 1	$\frac{e(n)}{\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))}$

In addition, from the perspective of projection, PKLMS is also related to the hyperplane projection along affine subspace (HYPASS) algorithm with $Q = 1$ [26], which is based on the projection-onto-convex-sets theory [27]. And, the difference between PKLMS and HYPASS is that $\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1))$ in PKLMS becomes a denominator in HYPASS. Table I summarizes the differences among PKLMS, QKLMS, MQKLMS and HYPASS.

III. Convergence Analyses

In this section, we first provide a stability analysis, and then conduct *mean-square convergence* analyses for PKLMP with different values of p, i.e., converging in terms of the steady-state MSE. We also analyse the mean convergence behaviour of PKLMP and derive its convergence conditions.

Let f^o denote the unknown nonlinear model that needs to be evaluated. According to the universal approximation property [28], there is an optimal vector $\boldsymbol{\Omega}^o \in \mathbb{F}$ such that $f^o = \langle \boldsymbol{\Omega}^o, \varphi(\cdot) \rangle_{\mathbb{F}}$. Thus, we can represent the desired signal $d(n)$ as

$$d(n) = f^o(\boldsymbol{u}(n)) + v(n) = \langle \boldsymbol{\Omega}^o, \varphi(n) \rangle_{\mathbb{F}} + v(n), \tag{11}$$

where $v(n)$ is the disturbance noise.

A. Stability Analysis

According to (11), the estimation error $e(n)$ can be expressed as

$$e(n) = d(n) - \boldsymbol{\Omega}(n-1)^T \varphi(n) = e_a(n) + v(n), \tag{12}$$

where $e_a(n) = \tilde{\boldsymbol{\Omega}}(n-1)^T \varphi(n)$ denotes the *a-priori error*, and $\tilde{\boldsymbol{\Omega}}(n-1) = \boldsymbol{\Omega}^o - \boldsymbol{\Omega}(n-1)$ is the weight error.

Subtracting $\boldsymbol{\Omega}^o$ from both sides of the last line of (6), we have

$$\begin{aligned} \underbrace{\boldsymbol{\Omega}^o - \boldsymbol{\Omega}(n)}_{\widetilde{\boldsymbol{\Omega}}(n)} &= \underbrace{\boldsymbol{\Omega}^o - \boldsymbol{\Omega}(n-1)}_{\widetilde{\boldsymbol{\Omega}}(n-1)} - \eta \underbrace{|e(n)|^{p-2} e(n)}_{F(e(n))} \underbrace{P(\varphi(n))}_{a(n)\varphi_p(n)} \\ \Rightarrow \widetilde{\boldsymbol{\Omega}}(n) &= \widetilde{\boldsymbol{\Omega}}(n-1) - \eta F(e(n)) a(n) \varphi_p(n). \end{aligned} \tag{13}$$

Squaring both sides of last line of (13), and then taking the expectations, we have

$$\begin{aligned} &E\left[\|\tilde{\boldsymbol{\Omega}}(n)\|_{\mathbb{F}}^2\right] - E\left[\|\tilde{\boldsymbol{\Omega}}(n-1)\|_{\mathbb{F}}^2\right] \\ &= E\left[(\eta a(n) F(e(n)))^2 \kappa_p(n)\right] \\ &\quad - 2E\left[\eta a(n) F(e(n)) \tilde{\boldsymbol{\Omega}}(n-1)^T \varphi_p(n)\right], \end{aligned} \tag{14}$$

where $\kappa_p(n) = \langle \varphi_p(n), \varphi_p(n) \rangle_{\mathbb{F}}$. To guarantee a converging solution, $E[\|\tilde{\boldsymbol{\Omega}}(n)\|_{\mathbb{F}}^2] - E[\|\tilde{\boldsymbol{\Omega}}(n-1)\|_{\mathbb{F}}^2] \leq 0$ should be satisfied, which means the value of η in PKLMP at $\forall n$ should satisfy

$$\begin{cases} E\left[\|\tilde{\boldsymbol{\Omega}}(n)\|_{\mathbb{F}}^2\right] - E\left[\|\tilde{\boldsymbol{\Omega}}(n-1)\|_{\mathbb{F}}^2\right] \leq 0 \\ \qquad\qquad \Downarrow \\ \begin{cases} E\left[a(n) F(e(n)) \tilde{\boldsymbol{\Omega}}(n-1)^T \varphi_p(n)\right] > 0 \\ 0 < \eta \leq \dfrac{2E\left[a(n) F(e(n)) \tilde{\boldsymbol{\Omega}}(n-1)^T \varphi_p(n)\right]}{E\left[a(n)^2 \kappa_p(n) F(e(n))^2\right]}. \end{cases} \end{cases} \tag{15}$$

Remark 3. The last inequality in (15) presents a sufficient condition for mean-square convergence of PKLMP. However, in practice, it is difficult to know the upper bound exactly. Hence, it merely indicates that η needs to be small enough to make the algorithm converge, and cannot be directly applied as a strict convergence condition for KAFs.

B. Convergence in terms of Steady-state MSE

We define the EMSE as $\xi = \lim_{n\to\infty} E[e_a(n)^2]$, and use it to study the steady-state MSE of PKLMP. Before proceeding, we list the following assumptions that are used in the rest of this section.

- A1. The noise signal $v(n)$ follow multiple independent and identical distributions with zero-mean and the finite variance σ_v^2, and are independent of the input sequence $\{\boldsymbol{u}(n)\}$.
- A2. The a-priori error $e_a(n)$ with zero-mean is independent of $v(n)$.

Assumptions A1 and A2 are commonly used in steady-state performance analysis for adaptive filters, e.g., in [8], [23], and [29]. Taking the limits of both sides of (14), we get

$$\begin{aligned} &\lim_{n\to\infty} E\left[\|\tilde{\boldsymbol{\Omega}}(n)\|_{\mathbb{F}}^2\right] - \lim_{n\to\infty} E\left[\|\tilde{\boldsymbol{\Omega}}(n-1)\|_{\mathbb{F}}^2\right] \\ &= \lim_{n\to\infty} E\left[(\eta a(n) F(e(n)))^2 \kappa_p(n)\right] \\ &\quad - 2 \lim_{n\to\infty} E\left[\eta a(n) F(e(n)) \tilde{\boldsymbol{\Omega}}(n-1)^T \varphi_p(n)\right]. \end{aligned} \tag{16}$$

When the algorithm reaches the steady-state, we have $\lim_{n\to\infty} E[\|\tilde{\boldsymbol{\Omega}}(n)\|_{\mathbb{F}}^2] = \lim_{n\to\infty} E[\|\tilde{\boldsymbol{\Omega}}(n-1)\|_{\mathbb{F}}^2]$, and then

$$\begin{aligned} &\eta \lim_{n\to\infty} E\left[a(n)^2 \kappa_p(n) F(e(n))^2\right] \\ &= 2 \lim_{n\to\infty} E\left[a(n) F(e(n)) \tilde{\boldsymbol{\Omega}}(n-1)^T \varphi_p(n)\right]. \end{aligned} \tag{17}$$

We further assume that $a^2(n)\kappa_p(n)$ and $F(e(n))^2$ are asymptotically uncorrelated (the rationality of this assumption was discussed in [29]). Since $a(n)$ is only an approximation factor in VP, we can assume that $a(n)$ is asymptotically uncorrelated with $F(e(n))\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi_p(n)$. In particular, if we set $a(n) = 1$, this assumption becomes a truth. Then (17) becomes

$$\begin{aligned} &\frac{\eta}{2} \lim_{n\to\infty} E\left[a(n)^2 \kappa_p(n)\right] \lim_{n\to\infty} E\left[F(e(n))^2\right] \\ &= \lim_{n\to\infty} E\left[a(n)\right] \lim_{n\to\infty} E\left[F(e(n)) \tilde{\boldsymbol{\Omega}}(n-1)^T \varphi_p(n)\right]. \end{aligned} \tag{18}$$

It is well known that adaptive filters based on LMP achieve different performance for different values of p. Hence, we derive the steady-state performance of PKLMP using a Taylor

expansion method, which can decouple the correlation between $e_a(n)$ and $v(n)$ [31].

Since $F(e) = |e|^{p-2}e$ and $e = e_a + v$, taking the Taylor expansion of $F(e)$ with respect to e_a around v, we get

$$\begin{cases} F(e) = F(v) + F'(v)e_a + \dfrac{1}{2}F''(v)e_a^2 + O(e_a^2) \\ F'(v) = (p-1)|v|^{p-2} \\ F''(v) = (p-1)(p-2)|v|^{p-4}v, \end{cases} \tag{19}$$

where $F'(v)$ and $F''(v)$ denote the first and second derivatives, and $O(e_a^2)$ contains the third and higher-order terms. Hence, based on (19), we can have

$$\begin{aligned} &F(e(n))^2 = F(v(n))^2 + 2F(v(n))F'(v(n))e_a(n) \\ &+ F(v(n))F''(v(n))e_a^2(n) + F'(v(n))^2e_a^2(n) + O(e_a^2(n)) \\ &= |v(n)|^{2p-2} + 2(p-1)|v(n)|^{2p-4}e_a(n) \\ &+ (p-1)(2p-3)e_a^2(n) + O(e_a^2(n)), \end{aligned} \tag{20}$$

and, based on A1 and A2, we have

$$\begin{aligned} &\lim_{n\to\infty} E[F(e(n))^2] = E\left[|v(n)|^{2p-2}\right] + E\left[O(e_a^2(n))\right] \\ &+ (p-1)(2p-3)E\left[|v(n)|^{2p-4}\right] \lim_{n\to\infty} E\left[e_a^2(n)\right] \\ &= \xi_v^{2p-2} + (p-1)(2p-3)\xi_v^{2p-4}\xi, \end{aligned} \tag{21}$$

where $\xi_v^p = E[|v|^p]$ and $E\left[O(e_a^2(n))\right]$ is ignored, and

$$\begin{aligned} &\lim_{n\to\infty} E\left[F(e(n))\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi_p(n)\right] \\ &= (p-1)\xi_v^{p-2}\left(\xi + \lim_{n\to\infty} E\left[e_a(n)g_p(n)\right]\right), \end{aligned} \tag{22}$$

where $g_p(n) = \tilde{\boldsymbol{\Omega}}(n-1)^T(\varphi_p(n) - \varphi(n))$. Combining (18), (21) and (22), and after some calculations, we obtain

$$\begin{cases} \xi = \dfrac{\eta a_\infty^\kappa \xi_v^{2p-2} - 2a_\infty \lim_{n\to\infty} E[e_a(n)g_p(n))]}{2a_\infty(p-1)\xi_v^{p-2} - \eta a_\infty^\kappa(p-1)(2p-3)\xi_v^{2p-4}} \\ a_\infty = \lim\limits_{n\to\infty} E[a(n)] \\ a_\infty^\kappa = \lim\limits_{n\to\infty} E[a(n)^2\kappa_p(n)]. \end{cases} \tag{23}$$

When a Gaussian kernel is used in PKLMP, we get $\kappa_p(n) = 1$, and thus $a_\infty^\kappa \approx (a_\infty)^2$. In the steady-state, a_∞ can be approximated by ε_c. Hence, (23) can be rewritten as

$$\xi = \frac{\eta\varepsilon_c\xi_v^{2p-2} - 2\lim_{n\to\infty} E[e_a(n)g_p(n))]}{2(p-1)\xi_v^{p-2} - \eta\varepsilon_c(p-1)(2p-3)\xi_v^{2p-4}}. \tag{24}$$

We can see that $\kappa(\mathcal{D}_{j^\star}(n-1), \boldsymbol{u}(n)) \geq \varepsilon_c$ in the steady-state, and we also have

$$\begin{aligned} &\left|E\left[\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi(n)g_p(n)\right]\right| \\ &\leq E\left[\left\|\tilde{\boldsymbol{\Omega}}(n-1)\right\|_{\mathbb{F}}^2\right] \underbrace{\left\|\varphi_p(n) - \varphi(n)\right\|_{\mathbb{F}}}_{\sqrt{2-2\kappa(\varphi_p(n),\varphi(n))}} \\ &\overset{(a)}{\leq} E\left[\|\boldsymbol{\Omega}^o\|_{\mathbb{F}}^2\right]\sqrt{2-2\varepsilon_c}, \end{aligned} \tag{25}$$

where the inequality (a) is obtained based on the fact $\|\tilde{\boldsymbol{\Omega}}(n)\|_{\mathbb{F}}^2 \leq \|\tilde{\boldsymbol{\Omega}}(n-1)\|_{\mathbb{F}}^2\| \leq \ldots \leq \|\boldsymbol{\Omega}^o\|_{\mathbb{F}}^2$. Therefore, injecting (25) into (24), we get

$$\begin{cases} \text{Low}(p,\varepsilon_c) \leq \xi \leq \text{Up}(p,\varepsilon_c) \\ \text{Low}(p,\varepsilon_c) = \max\left\{\dfrac{\eta\varepsilon_c\xi_v^{2p-2} - 2C_{\varepsilon_c}^{\Omega^o}}{C_{\varepsilon_c}^{p\xi_v}}, 0\right\} \\ \text{Up}(p,\varepsilon_c) = \dfrac{\eta\varepsilon_c\xi_v^{2p-2} + 2C_{\varepsilon_c}^{\Omega^o}}{C_{\varepsilon_c}^{p\xi_v}} \\ C_{\varepsilon_c}^{\Omega^o} = E\left[\|\boldsymbol{\Omega}^o\|_{\mathbb{F}}^2\right]\sqrt{2-2\varepsilon_c} \\ C_{\varepsilon_c}^{p\xi_v} = 2(p-1)\xi_v^{p-2} - \eta\varepsilon_c(p-1)(2p-3)\xi_v^{2p-4}, \end{cases} \tag{26}$$

where $\text{Low}(p,\varepsilon_c)$ and $\text{Up}(p,\varepsilon_c)$ denote the lower and upper bounds of ξ, respectively.

Remark 4. The EMSE ξ is derived in (24), however, it is hard to estimate the value of $\varphi_p(n)$ and then evaluate the EMSE exactly. Therefore, we provide the bounds of ξ in (26) to characterize the EMSE. In addition, for any given $p \geq 1$, it is not difficult to find ξ's bounds numerically from (26). And, from this formulation, we can obtain the following important observations:

- When $p = 2$, (26) reduces to

$$\begin{cases} \text{Low}(2,\varepsilon_c) \leq \xi \leq \text{Up}(2,\varepsilon_c) \\ \text{Low}(2,\varepsilon_c) = \max\left\{\dfrac{\eta\varepsilon_c\sigma_v^2 - 2C_{\varepsilon_c}^{\Omega^o}}{2-\eta\varepsilon_c}, 0\right\} \\ \text{Up}(2,\varepsilon_c) = \dfrac{\eta\varepsilon_c\sigma_v^2 + 2C_{\varepsilon_c}^{\Omega^o}}{2-\eta\varepsilon_c}. \end{cases} \tag{27}$$

  From (27) we can see that, when $\varepsilon_c = 1$, the EMSE of PKLMP with $p = 2$ is $\xi = \frac{\eta\sigma_\nu^2}{2-\eta}$, which is actually the EMSE for KLMS [8];
- When $1 \leq p < 2$, we consider the fractional lower order moment (FLOM) of $|e(n)|$ and the noise $v(n)$ modeled by the α-SN. The α-SN has infinite variance, with finite pth-order moments for $p < \alpha \in (0, 2]$, where α denotes the characteristic factor that measures the tail heaviness of the distribution. Other three parameters related to the α-SN are: $\beta \in (-\infty, +\infty)$ denotes the location parameter; $\gamma > 0$ is the dispersion parameter; and $\delta \in [-1, 1]$ is the symmetry parameter [32]. Hence, based on the FLOM property of α-SN, (26) becomes

$$\begin{cases} \text{Low}(p_<,\varepsilon_c) \leq \xi \leq \text{Up}(p_<,\varepsilon_c) \\ \text{Low}(p_<,\varepsilon_c) = \max\left\{\dfrac{\eta\varepsilon_c C_{\alpha 1}^{\gamma p_<} - 2C_{\varepsilon_c}^{\Omega^o}}{C_{\varepsilon_c}^{p\alpha}}, 0\right\} \\ \text{Up}(p_<,\varepsilon_c) = \dfrac{\eta\varepsilon_c C_{\alpha 1}^{\gamma p_<} + 2C_{\varepsilon_c}^{\Omega^o}}{C_{\varepsilon_c}^{p\alpha}} \\ C_{\varepsilon_c}^{p\alpha} = 2(p-1)(C_{\alpha 2}^{\gamma p_<} - \eta\varepsilon_c(2p-3)C_{\alpha 3}^{\gamma p_<}) \\ C_{\alpha 1}^{\gamma p_<} = C(2p-2,\alpha)\gamma^{\frac{2p-2}{\alpha}} \\ C_{\alpha 2}^{\gamma p_<} = C(p-2,\alpha)\gamma^{\frac{p-2}{\alpha}} \\ C_{\alpha 3}^{\gamma p_<} = C(2p-4,\alpha)\gamma^{\frac{2p-4}{\alpha}}, \end{cases} \tag{28}$$

  where "$p_<$" denotes that p takes values over $[1, 2)$, and

$$C(p,\alpha) = \frac{2^{p+1}\Gamma(\frac{p+1}{2})\Gamma(\frac{-p}{\alpha})}{\alpha\sqrt{\pi}\Gamma(\frac{-p}{2})}, \tag{29}$$

with $\Gamma(\cdot)$ denoting the Gamma function [32].

In addition, for $1 \leq p < 2$, We also provide a qualitative analysis to demonstrate that the steady-state EMSE of PKLMP is lower than that of QKLMS. As mentioned in Remark 1, when $1 \leq p < 2$, the KLMP is robust to the impulsive noise. Hence, when larger outliers are present, we assume $|e(n)| > 1$. Under a Gaussian kernel, based on (6), we can get

$$\eta(n) = \eta a(n)|e(n)|^{p-2} \leq \eta\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}(n-1)) < \eta. \quad (30)$$

To this end, we can approximate $\eta(n)$ with a fixed step size η', which is not related to $e(n)$ and fulfills $\eta' < \eta$. Replacing $\eta(n)$ with η' in (17) yields

$$\frac{\eta'}{2} \lim_{n\to\infty} E\left[e(n)^2\right] = \lim_{n\to\infty} E\left[e(n)\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi_p(n)\right], \quad (31)$$

where $\kappa_p(n) = 1$ for a Gaussian kernel.

Applying $e(n) = e_a(n) + v(n)$ and A1 into (31), we get

$$\begin{aligned}
\xi &= \lim_{n\to\infty} E[e_a(n)^2] \\
&= \frac{\eta'\sigma_v^2 - 2\lim_{n\to\infty} E[\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi(n)\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi_p(n)]}{2-\eta'} \\
&< \frac{\eta\sigma_v^2 - 2\lim_{n\to\infty} E[\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi(n)\tilde{\boldsymbol{\Omega}}(n-1)^T\varphi_p(n)]}{2-\eta} \\
&= \xi_{QKLMS}. \quad (32)
\end{aligned}$$

Remark 5. Based on (32), the EMSE of PKLMP is always smaller than that of QKLMS when a Gaussian kernel is used, $p \in [1, 2)$, ε_c is set to $\exp(-\|\boldsymbol{u}(n) - \boldsymbol{\mathcal{D}}(n-1)\|^2/2h^2)$, and the initial step size of PKLMP is less than that of QKLMS. This means that, in the impulsive noise environments, PKLMP can achieve better filtering accuracy than QKLMS, as will be validated by simulation results in Section V.

C. Mean Convergence Analysis

In this part, with some approximations, we analyze the mean convergence behavior of the proposed PKLMP algorithm. Taking the expectation operation on both sides of last row in (6), we obtain

$$\begin{aligned}
&E\left[\boldsymbol{\Omega}(n)\right] = E\left[\boldsymbol{\Omega}(n-1)\right] + \eta\times \\
&E\left[|e(n)|^{p-2}\left(d(n) - \boldsymbol{\Omega}(n-1)^T\varphi(n)\right)a(n)\varphi_p(n)\right]. \quad (33)
\end{aligned}$$

Assume that $|e(n)|^{p-2}$ is uncorrelated with $(d(n) - \boldsymbol{\Omega}(n-1)^T\varphi(n))a(n)\varphi_p(n)$. This is a strong assumption, but it facilitates the analysis [33]. Together with the assumption of independence between the input sequence and weight vector, (33) can be represented as

$$\begin{aligned}
E[\boldsymbol{\Omega}(n)] = E[\boldsymbol{\Omega}(n-1)] + \eta E[|e(n)|^{p-2}]\times \\
a(n)\left(\boldsymbol{r}_{dp} - \boldsymbol{R}_{np}E[\boldsymbol{\Omega}(n-1)]\right), \quad (34)
\end{aligned}$$

where $\boldsymbol{r}_{dp} = E[d(n)\varphi_p(n)]$, and $\boldsymbol{R}_{np} = E[\varphi_p(n)\varphi(n)^T]$ (assuming $\boldsymbol{R}_{np}$ is positive definite).

Since $e(n) = e_a(n) + v(n)$, when the variance of $v(n)$ is larger than that of $e_a(n)$, we can assume that $e(n)$ is dominated by $v(n)$ in the transient state. Therefore, to make (34) numerically assessable, we can replace $|e(n)|^{p-2}$ by $|v(n)|^{p-2}$ in (34). Subtracting $E[\boldsymbol{\Omega}^o]$ on both sides of (34), we obtain the following stochastic difference formulation

$$\begin{aligned}
\zeta(n) = (\boldsymbol{I} - \eta E[|v(n)|^{p-2}]a(n)\boldsymbol{R}_{np})\zeta(n-1) \\
+ \eta E[|v(n)|^{p-2}]a(n)(\boldsymbol{r}_{dp} - \boldsymbol{R}_{np}E[\boldsymbol{\Omega}^o]), \quad (35)
\end{aligned}$$

where $\zeta(n) = E[\boldsymbol{\Omega}^o - \boldsymbol{\Omega}(n)]$, and $\boldsymbol{I}$ is an identity matrix. To guarantee the convergence of (35), the following condition needs to be satisfied $|1 - \eta E[|v(n)|^{p-2}]a(n)\lambda_{max}| < 1$, i.e.,

$$0 < \eta < \frac{2}{E[|v(n)|^{p-2}]a(n)\lambda_{max}}, \quad (36)$$

where λ_{max} means the maximal eigenvalue of $\boldsymbol{R}_{np}$. A more conservative condition can be obtained as

$$0 < \eta < \frac{2}{E[|v(n)|^{p-2}]a(n)\mathrm{Tr}(\boldsymbol{R}_{np})}, \quad (37)$$

where $\mathrm{Tr}(\boldsymbol{R}_{np})$ denotes the trace of $\boldsymbol{R}_{np}$. Moreover, when no sparsification method is applied and a Gaussian kernel is used, (37) reduces to

$$0 < \eta < \frac{2}{E[|v(n)|^{p-2}]}. \quad (38)$$

Remark 6. For the following two types of noise $v(n)$, (38) can be further elaborated.

- The noise $v(n)$ has finite variance, i.e., $\sigma_v^2 < \infty$. In this case, we can approximate $E[|v(n)|^{p-2}]$ as $E[|v(n)|^2]^{\frac{p-2}{2}}$, hence (38) becomes
$$0 < \eta < \frac{2}{E[|v(n)|^2]^{\frac{p-2}{2}}}, \quad (39)$$
which is a convergence condition for KLMP with $p \geq 2$.
- The noise $v(n)$ has infinite variance, and more specifically we consider the α-SN. In this case, $E[|v(n)|^{p-2}] = C(p-2,\alpha)\gamma^{\frac{p-2}{\alpha}}$, where $C(p-2,\alpha)$ is defined in (29). Thus, (38) becomes
$$0 < \eta < \frac{2}{C(p-2,\alpha)\gamma^{\frac{p-2}{\alpha}}}, \quad (40)$$
which is a convergence condition for KLMP with $p \in [1, 2)$ under the α-SN environment.

IV. Modifications to PKLMP

In this section, firstly, we discuss how to improve the filtering accuracy of PKLMP by exploiting the information embedded in the output data. Then we propose a simple and efficient method for updating the centers of a dictionary $\boldsymbol{\mathcal{D}}(n-1)$. In general, the centers of a dictionary are fixed in most sparsification methods, and dated centers may result in performance degradation of KAFs.

A. PKLMP with Smoothed Desired Outputs

As we can see, PKLMP only considers the feature space (equivalent with the input space) compression, and assumes that the corresponding outputs of the projected data are equal to those of MRC in the dictionary. In the impulsive noise cases, however, the outputs in a neighborhood may have large

variation. Such disturbance in desired outputs may dramatically deteriorate the filtering accuracy. Hence, it is necessary to smooth the outputs to improve the performance.

In the PKLMP algorithm, we can minimize the following instantaneous cost function

$$J(n) = |e(n)|^p, \tag{41}$$

where $e(n) = d(n) - f_{n-1}(\boldsymbol{u}(n))$ denotes the estimation error, and $p > 0$. Applying the gradient descent method to minimize $J(n)$ in the output space, we obtain

$$\begin{aligned} d_s(n) &= d(n) - \eta_1 \nabla_{d(n)} J(n) \\ &= d(n) - \eta_1 |e(n)|^{p-2} e(n), \end{aligned} \tag{42}$$

where $d_s(n)$ denotes the smoothed output, $\nabla_{d(n)} J(n)$ is the gradient of the loss function $J(n)$ with respect to $d(n)$, η_1 denotes the step size for refining the current output. In this way, $d(n)$ can be adjusted to constrain large disturbances induced by large outliers. Then, the prediction error is also updated by $e_s(n) = d_s(n) - f_{n-1}(\boldsymbol{u}(n))$, and the loss function therefore becomes

$$J_s(n) = |e_s(n)|^p. \tag{43}$$

To this end, the methods similar to those in PKLMP can be applied to obtain

$$\begin{cases} f_0 = 0, \quad e(n) = d(n) - f_{n-1}(\boldsymbol{u}(n)) \\ d_s(n) = d(n) - \eta_1 |e(n)|^{p-2} e(n) \\ e_s(n) = d_s(n) - f_{n-1}(\boldsymbol{u}(n)) \\ f_n = h^n_{-sj^\star} + h^n_{sj^\star} \\ h^n_{-sj^\star} = \displaystyle\sum_{l=1, l\neq j^\star}^{M} \omega_l(n-1)\kappa(\mathcal{D}_l(n-1), \cdot) \\ h^n_{sj^\star} = (\omega_{j^\star}(n-1) + \eta |e_s(n)|^{p-2} a(n) e_s(n)) \\ \qquad \times \kappa(\mathcal{D}_{j^\star}(n-1), \cdot), \end{cases} \tag{44}$$

where $h^n_{-sj^\star}$ and $h^n_{sj^\star}$ are similar to $h^n_{-j^\star}$ and $h^n_{j^\star}$, respectively, except that $e_s(n)$ is used in the former. We call this modified PKLMP method as *projected kernel least p-power with smoothed desired output* (PKLMP-SD), and summarize it in Algorithm 2.

Remark 7. We can see that, in (42), the gradient-based method only considers $d(n)$ and $e(n)$, which does not cause large increase in the computational complexity and storage cost. Hence, PKLMP-SD has similar complexity to PKLMP. Furthermore, PKLMP-SD considers the case that, for two MRCs in a projection region, their corresponding desired outputs may be significantly different. Compared with PKLMP, the PKLMP-SD algorithm not only exploits the information hidden in the output space, but also smoothes the $e_s(n)$ of $h^n_{sj^\star}$ in (44), thereby leading to filtering accuracy improvement.

B. KAF with Variable Centers

In most sparsification methods, the centers of a dictionary are fixed. However, variable centers (VC) may lead to a better filtering performance for KAFs. From the idea of VP, we know that, when $\Theta \geq \varepsilon_c$, the dictionary remains unchanged, and $\boldsymbol{u}(n)$ is just replaced by $\mathcal{D}_{j^\star}(n-1)$, as shown in the step 4) in Algorithm1. Actually, we can use the information contained in $\boldsymbol{u}(n)$ to update $\mathcal{D}_{j^\star}(n-1)$ and the learning system. Hence, we propose a simple method to vary the centers of a dictionary as follows:

$$\begin{cases} \text{if} \quad \Theta \geq \varepsilon_c \\ \text{then} \quad \mathcal{D}_{j^\star}(n) = \mathcal{D}_{j^\star}(n-1) - \eta_{\mathcal{D}} \nabla_{\mathcal{D}^\star_j} J(n), \end{cases} \tag{45}$$

where Θ and $j^\star$ are given in Algorithm 1, $\eta_{\mathcal{D}} > 0$ denotes the step size, and $J(n)$ is defined in (41). When a Gaussian kernel is used, $\nabla_{\mathcal{D}_{j^\star}} J(n)$ can be represented as

$$\nabla_{\mathcal{D}_{j^\star}} J(n) = -\frac{p\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star})\omega_{j^\star}}{h^2 |e(n)|^{1-p}} (\boldsymbol{u}(n) - \mathcal{D}_{j^\star}), \tag{46}$$

where $\kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}) = \exp\left(-\frac{\|\boldsymbol{u}(n) - \mathcal{D}_{j^\star}(n-1)\|^2}{2h^2}\right)$.

Injecting (46) into (45) and letting $\eta_2 = \eta_{\mathcal{D}} p / h^2$, we get

$$\begin{aligned} \mathcal{D}_{j^\star}(n) &= \mathcal{D}_{j^\star}(n-1) + \eta_2 \kappa(\boldsymbol{u}(n), \mathcal{D}_{j^\star}) \omega_{j^\star} \\ &\quad \times |e(n)|^{p-2} e(n) (\boldsymbol{u}(n) - \mathcal{D}_{j^\star}). \end{aligned} \tag{47}$$

Remark 8. The proposed VC method is simple but can improve the filtering performances of KAFs in most situations. It is worth noting that, when one uses the VC method, $\mathcal{D}_{j^\star}(n)$ should be updated after $\boldsymbol{w}_{j^\star}(n) = \boldsymbol{w}_{j^\star}(n-1) + \eta |e(n)|^{p-2} a(n) e(n)$. The variable centers method is integrated in Algorithm 2.

Remark 9. We briefly analyze the computational complexity of the proposed algorithms. For PKLMP, the complexity for computing $f_{n-1}(\boldsymbol{u}(n)) = \sum_{l=1}^{M} \omega_l(n-1)\kappa(\mathcal{D}_l(n-1), \boldsymbol{u}(n))$ scales linearly with the dictionary size M. With the Gaussian kernel, no computation is required for the cosine-relation $\Theta = \max_{1\leq j\leq M} \frac{\langle \varphi(n), \varphi(\mathcal{D}_j(n-1)) \rangle_{\mathbb{F}}}{\|\varphi(n)\|_{\mathbb{F}} \cdot \|\varphi(\mathcal{D}_j(n-1))\|_{\mathbb{F}}}$, since $\|\varphi(n)\|_{\mathbb{F}} = \|\varphi(\mathcal{D}_j(n-1))\|_{\mathbb{F}} = 1$ and $\langle \varphi(n), \varphi(\mathcal{D}_j(n-1)) \rangle_{\mathbb{F}}$ have been calculated in $f_{n-1}(\boldsymbol{u}(n))$. Hence, the complexity of PKLMP is $\mathcal{O}(M)$; For PKLMP-SD, the extra complexity is for computing $d_s(n)$ and $e_s(n)$. Therefore, the complexity of PKLMP-SD is still $\mathcal{O}(M)$; For the VC method, as an extra operation, the $\mathcal{D}_{j^\star}(n) = \mathcal{D}_{j^\star}(n-1) - \eta_{\mathcal{D}} \nabla_{\mathcal{D}^\star_j} J(n)$ has computational complexity of $\mathcal{O}(1)$. Therefore, the cost of PKLMP-SD-VC is also $\mathcal{O}(M)$.

V. Simulation and Experimental Results

In this section, we conduct simulations to demonstrate the effectiveness of the proposed PKLMP methods. The tested examples include estimation for an artificial static function, prediction for the short-term Chua's chaotic time series, and prediction for the real-world Internet traffic and the real-world Sunspot number. Unless stated otherwise, we apply a Gaussian kernel with a kernel size $h = 1$ in the experiments. To evaluate the filtering performance, we use the metric *testing MSE* (TMSE), which is defined as [8], [24]

$$\text{TMSE} = \frac{1}{N} \sum_{n=1}^{N} \left(d(n) - f_{n-1}(\boldsymbol{u}(n))\right)^2, \tag{48}$$

where N is the number samples in a testing set.

We define some subclasses of PKLMP schemes based on the value of the approximation factor $a(n)$, which has a key role

Algorithm 2: Modifications of PKLMP

Initialization:
step size $\eta > 0$, $\eta_1 > 0$, $\eta_2 > 0$, Mercer kernel $\kappa(\cdot,\cdot)$
coherence threshold $0 < \varepsilon_c < 1$,
dictionary $\boldsymbol{\mathcal{D}}(1) = \{\boldsymbol{u}(1)\}$ with $M = 1$,
KWV $\boldsymbol{w}(1) = [\eta|d(1)|^{p-2}d(1)]$.

Computation:
while $\{\boldsymbol{u}(n)$, d(n)$\}$ $(n \geq 2)$ available **do**
1) the output:
$f_{n-1}(\boldsymbol{u}(n)) = \eta \sum_{l=1}^{M} \boldsymbol{w}_l(n-1)\kappa(\mathcal{D}_l(n-1), \boldsymbol{u}(n))$
2) the estimation error: $e(n) = d(n) - f_{n-1}(\boldsymbol{u}(n))$
3) the cosine-relation between $\varphi(n)$ and $\varphi(\boldsymbol{\mathcal{D}}(n-1))$:
$\Theta = \max_{1\leq j\leq M} \frac{\langle\varphi(n),\varphi(\mathcal{D}_j(n-1))\rangle_{\mathbb{F}}}{\|\varphi(n)\|_{\mathbb{F}}\|\varphi(\mathcal{D}_j(n-1))\|_{\mathbb{F}}}$
4) **if** $\Theta \geq \varepsilon_c$
smooth the corresponding output:
$d_s(n) = d(n) - \eta_1|e(n)|^{p-2}e(n)$
update the estimation error:
$e_s(n) = d_s(n) - f_{n-1}(\boldsymbol{u}(n))$
keep the dictionary: $\boldsymbol{\mathcal{D}}(n) = \boldsymbol{\mathcal{D}}(n-1)$, $M \Leftarrow M$
update the coefficient of the MRC:
$\boldsymbol{w}_{j^\star}(n) = \boldsymbol{w}_{j^\star}(n-1) + \eta|e_s(n)|^{p-2}a(n)e_s(n)$
with $a(n) = \frac{\kappa(\boldsymbol{u}(n),\mathcal{D}_{j^\star}(n-1))}{\|\varphi(\mathcal{D}_{j^\star}(n-1))\|_{\mathbb{F}}^2}$
if apply VC
then $\mathcal{D}_{j^\star}(n) = \mathcal{D}_{j^\star}(n-1) - \eta_{\mathcal{D}}\nabla_{\mathcal{D}_j^\star}J(n)$
else $\mathcal{D}_{j^\star}(n) = \mathcal{D}_{j^\star}(n)$
5) **Otherwise**
smooth the corresponding output: $d_s(n) = d(n)$
update the estimation error: $e_s(n) = e(n)$
change the dictionary:
$\boldsymbol{\mathcal{D}}(n) = \{\boldsymbol{\mathcal{D}}(n-1), \boldsymbol{u}(n)\}$, $M \Leftarrow M+1$
update the KWV:
$\boldsymbol{w}(n) = \left[\boldsymbol{w}(n-1)^T, \eta|e_s(n)|^{p-2}e_s(n)\right]$
end while

in PKLMP. As we explained in Remark 2, when a Gaussian kernel is used, $a(n)$ is smaller than and approaching to 1. So we set $a(n) = 1$, and call the PKLMP scheme in this case as *hard PKLMP* (HP-KLMP). Accordingly, we call the original PKLMP with the exact $a(n)$ as *soft PKLMP* (SP-KLMP). Similarly, we get *SP-KLMS* and *HP-KLMS* for KLMS, and *SP-KLMP-SD* and *HP-KLMP-SD* for KLMP-SD.

A. Static Function Estimation (SFE)

We consider the following artificial function [8],

$$d(n) = \frac{\exp\left(\frac{(u(n)+1)^2}{-2}\right) + \exp\left(\frac{(u(n)-1)^2}{-2}\right)}{5} + v(n), \quad (49)$$

where the input samples $\{u(n)\} \in \mathbb{R}$ are generated following a white Gaussian process with zero-mean and variance $\sigma_u^2 = 1$, and the noise $\{v(n)\}$ is independent of $\{u(n)\}$. In this section, the numbers of training and testing data are 500 and 100, respectively. And, all simulation results are averaged over 500 independent runs.

1) Lower and Upper Bounds: We first verify the accuracy of the lower and upper bounds (27) for SP-KLMP with $p = 2$, i.e., SP-KLMS. For comparison, we also consider the corresponding lower and upper bounds for QKLMS in (36) of [8]. In this trial, $\{v(n)\}$ is white Gaussian noise with mean 0 and variance $\sigma_v^2 = 0.04$. The step size is $\eta = 0.6$ for both algorithms[1]. For a fair comparison, the coherence threshold ε_c is estimated via $\varepsilon_c = \exp(-0.5\varepsilon_U^2)$, where ε_U is the distance threshold of QKLMS. According to (27), we need to estimate $E[\|\boldsymbol{\Omega}^o\|_{\mathbb{F}}^2]$. Generally, it is hard to get the optimal solution. Fortunately, (49) can be model as

$$\begin{cases} d(n) = f^o + v(n) \\ f^o = 0.2(\kappa(\cdot,1) + \kappa(\cdot,-1)), \end{cases} \quad (50)$$

where $\kappa(\cdot,1) = \exp\left(-0.5(\cdot-1)^2\right)$ and $\kappa(\cdot,-1) = \exp\left(-0.5(\cdot+1)^2\right)$. Hence, $E[\|\boldsymbol{\Omega}^o\|_{\mathbb{F}}^2] = \|f^o\|^2 = 0.0908$.

Fig. 1 plots the EMSE for these two algorithms. From this figure, we can have the following observations:

- The EMSE indeed lies between the derived lower and upper bounds for SP-KLMS;
- The lower bound of SP-KLMS has very similar behavior to that of QKLMS;
- When ε_U increases from 0.35 to 10 (namely, ε_U decreases from 0.9406 to 0), SP-KLMS has smaller upper bound than that of QKLMS, which means the upper bound $\mathrm{Up}(\varepsilon_c)$ in (27) is more accurate;
- When $\varepsilon_U \approx 0$ (namely, $\varepsilon_c \approx 1$), the EMSE of SP-KLMS is close to that of KLMS, which can be theoretically estimated as $\xi = \frac{\eta\sigma_\nu^2}{2-\eta} = 0.0171$;
- When $\varepsilon_c \in (0.00073, 0.9406)$, SP-KLMS can realize smaller EMSE than KLMS. Furthermore, when $\varepsilon_c \in (0.1103, 0.9406)$, SP-KLMS outperforms QKLMS in terms of EMSE.

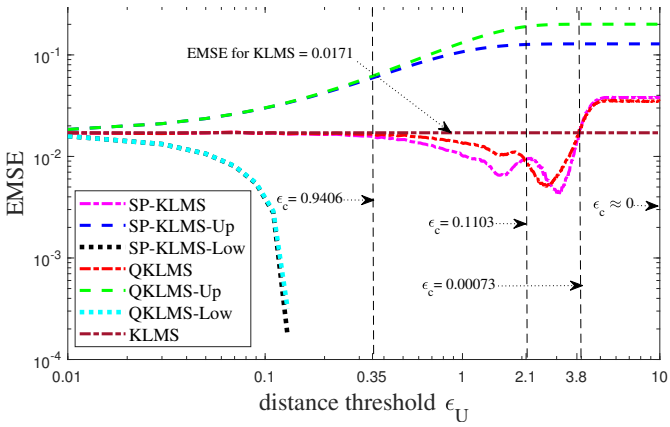


Fig. 1: The EMSE for SP-KLMS and QKLMS versus the distance threshold $\varepsilon_U \in [0.01, 10]$ in SFE.

Fig. 2 shows the TMSE learning curves for SP-KLMS with $\varepsilon_c = 0.3247$, QKLMS with $\varepsilon_U = 1.5$, HYPASS with $Q = 1$ and step size 0.45, and KLMS. Their final network sizes are 3.617, 3.617, 3.617 and 500. The proposed SP-KLMS is shown to outperform the other three algorithms.

[1]Note that $\eta = 0.6 < 2$ in (39) guarantees the convergence of KLMS, and it also fits for PKLMS to achieve convergence as shown in Fig. 2.

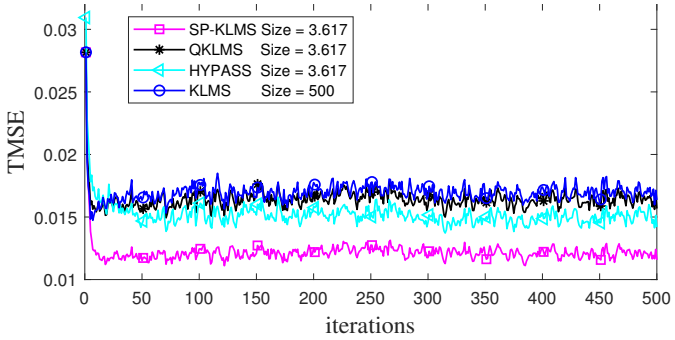


Fig. 2: The TMSE learning curves for SP-KLMS, QKLMS, HYPASS and KLMS, when $\varepsilon_U = 1.5$ or $\varepsilon_c = 0.3247$ in SFE.

2) Selection of the Step Size: A proper value of the step size η can ensure the convergence of PKLMP. We validate the bound of η in (40) instead of (37), because it is generally hard to obtain $\mathrm{Tr}(\boldsymbol{R}_{np})$ in (37). In other words, we study the influence of η on the convergence of KLMP with $p \in [1, 2)$. Here, the noise $\{v(n)\}$ is modeled by the α-SN with a parameter vector $\boldsymbol{P}_\alpha = [\alpha, \beta, \gamma, \delta]$. In this trial, we set $\eta \in \{0.05, 0.15, 0.3, 0.69\}$, $p = 1.5$, $\boldsymbol{P}_\alpha = [1.6, 0, 0.1, 0]$, and thus $2/\big(C(p-2,\alpha)\gamma^{\frac{p-2}{\alpha}}\big) = 0.6814$. Fig. 3 plots the corresponding TMSE learning curves. From this figure, we obtain the following observations:

- The value $\eta = 0.69 > 0.6814$ diverges the TMSE learning cure, which demonstrates the tightness of the theoretic bound for η in (40).
- A smaller η value, such as 0.05, 0.15 and 0.3, leads to convergence of the algorithm. This result is consistent with the usual rule for step size in adaptive filtering algorithms.

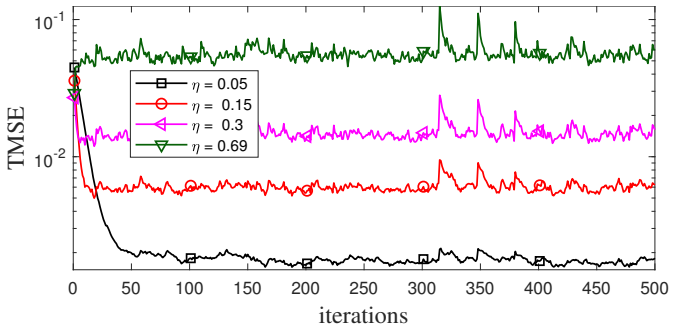


Fig. 3: The TMSE learning curves for KLMP with $p = 1.5$ and $\eta \in \{0.05, 0.15, 0.3, 0.69\}$ in SFE.

3) Coherence Threshold: We then investigate how the coherence threshold ε_c affects the performance of SP-KLMP in the presence of α-SN, where $\boldsymbol{P}_\alpha = [1.6, 0, 0.1, 0]$. In this trial, based on the results in Fig. 3, we set $\eta = 0.05$ for SP-KLMP. Fig. 4 shows the steady-state TMSE estimated by the last 100 TMSE for different SP-KLMP algorithms with respect to various values of $\varepsilon_c \in [0.02, 1]$. From this figure, we can get the following observations:

- The two steady-state TMSE curves for $p \in \{1, 1.3\}$ are smoother than the other two, and SP-KLMP with $p = 1.3$ outperforms that with $p = 1$;
- For a majority values of ε_c, SP-KLMP with $p = 1.6$ achieves the lowest final TMSE;
- The final TMSE curve for $p = 1.9$ fluctuates widely, since $p = 1.9 > \alpha = 1.6$, resulting in larger misalignment.

Therefore, we can conclude that, in impulsive noise environments modelled by α-SN, ε_c has less negative influence on the filtering accuracy of SP-KLMP when p is smaller than but close to the characteristic factor α. Table II summarizes some averaged dictionary sizes for these SP-KLMP algorithms with respect to $\varepsilon_c \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$.

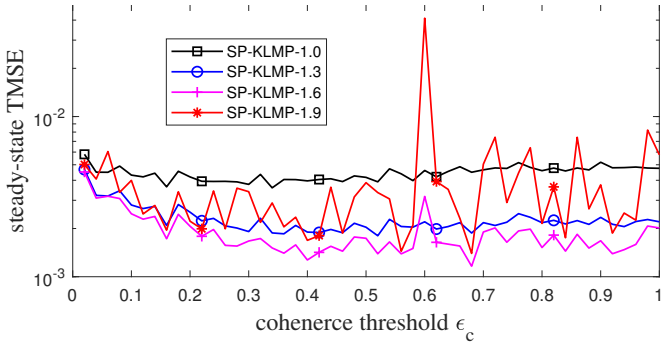


Fig. 4: The steady-state TMSE curves for SP-KLMP with different values of p versus $\varepsilon_c \in [0.02, 1]$ in SFE.

4) Filtering Performance: Using the example of the impulsive noise above, we test the filtering performance for SP-KLMP, HP-KLMP, SP-KLMP-SD and HP-KLMP-SD. For comparisons, the NC and CC sparsification methods are applied to KLMP, and are denoted as NC-KLMP and CC-KLMP, respectively. In this experiment, we set $p = 1.5$ for all KLMP algorithms. Other parameters of the algorithms are chosen such that they have almost the same initial convergence rate. Hence, $\eta = 0.05$ and $\varepsilon_c = 0.4$ for all PKLMP algorithms, $\eta_1 = 0.1$ for two smoothed out algorithms, the step sizes for CC-KLMP and NC-KLMP are set as 0.1 and 0.08, respectively, $\varepsilon_c = 0.97$ for CC-KLMP, and $\delta_1 = 0.2$ and $\delta_2 = 0.01$ for NC-KLMP. Fig. 5 plots the TMSE learning curves for these six KLMP algorithms, and delivers the following messages:

- Under similar initial convergence rates, the KLMP algorithms using VP outperform NC-KLMP and CC-KLMP in terms of filtering accuracy and the dictionary size;
- SP-KLMP-SD and HP-KLMP-SD achieve better filtering accuracy than SP-KLMP and HP-KLMP, respectively. This indicates that the smooth-desired output method does improve the filtering performance for KAF;
- Compared to the two HP-KLMPs[2], the two SP-KLMPs achieve smaller TMSE.

Hence we can conclude that the VP method can efficiently improve the filtering performance for KLMP.

5) VC Method: We demonstrate the effectiveness of the proposed VC method. From Fig. 5, we know that HP-KLMP and SP-KLMP-SD achieve the worst and best filtering accuracy, respectively, among four PKLMPs. Therefore, we apply

[2]As mentioned in (10), HP-KLMP can be treated as KLMP sparsified by the VQ method, since the approximation factor $a(n) = 1$ in HP-KLMP.

VC to HP-KLMP and SP-KLMP-SD, and denote them as HP-KLMP-VC and SP-KLMP-SD-VC, respectively. Except for $\eta_2 = 0.5$, other parameters are the same as those in the fourth experiment. In addition, we also use the random feature method [35] and the Nyström method [36] to KLMP, and they are denoted as RF-KLMP and Nys-KLMP, respectively. For RF-KLMP, the $D = 20$, $\sigma = 5$, and step-size is 0.002; For Nys-KLMP, the $m = 10$, $\sigma = 1$, and step-size is 0.05. Fig. 6 shows the TMSE curves for the six algorithms. From this figure, we have following observations:

- HP-KLMP-VC and SP-KLMP-SD-VC achieve better filtering accuracy than HP-KLMP and SP-KLMP-SD, respectively, which indicates that the VC method is efficient.
- RF-KLMP and Nys-KLMP can realize slightly better filtering accuracy than HP-KLMP and HP-KLMP-VC. However, two SP-KLMP-SD algorithms outperform RF-KLMP and Nys-KLMP in terms of TMSE accuracy.

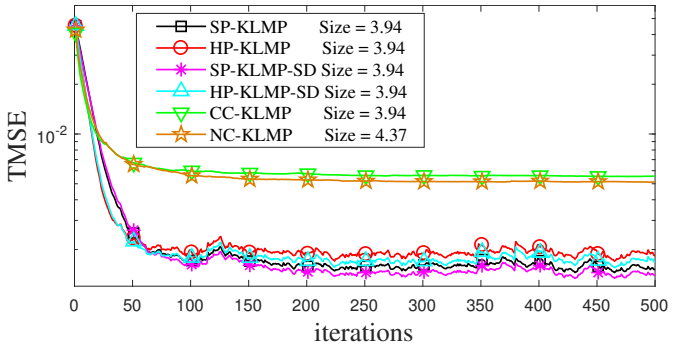


Fig. 5: The TMSE learning curves for the six KLMPs in SFE.

In Fig. 7, we also validate that the proposed PKLMP algorithms can work effectively in the case of $p \in (0, 1)$. The proposed algorithms are shown to have good convergence behaviour when the value of p approaches 1.

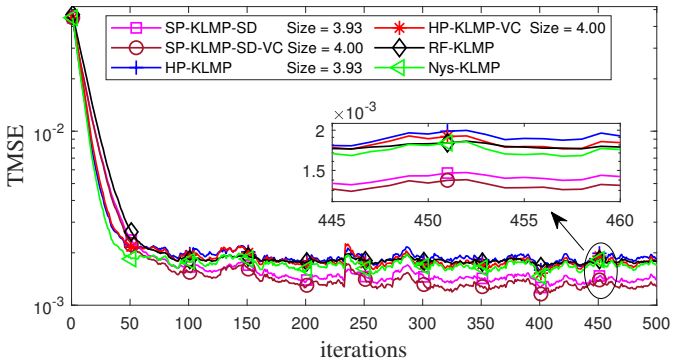


Fig. 6: The TMSE learning curves for these KLMPs in SFE.

6) Comparison with other error criteria: To test the efficient of VP method, we apply it to other KAFs based on some error criteria, such as logarithmic error loss (LEL) [37], kernel risk-sensitive loss (KRSL) [30], kernel mean p-power error (KMPE) [38], and maximum correntropy (MC) [34]. And correspond algorithms are named as SP-LEL, SP-KRSL, SP-KMPE and SP-MC, respectively. For comparison, the VQ

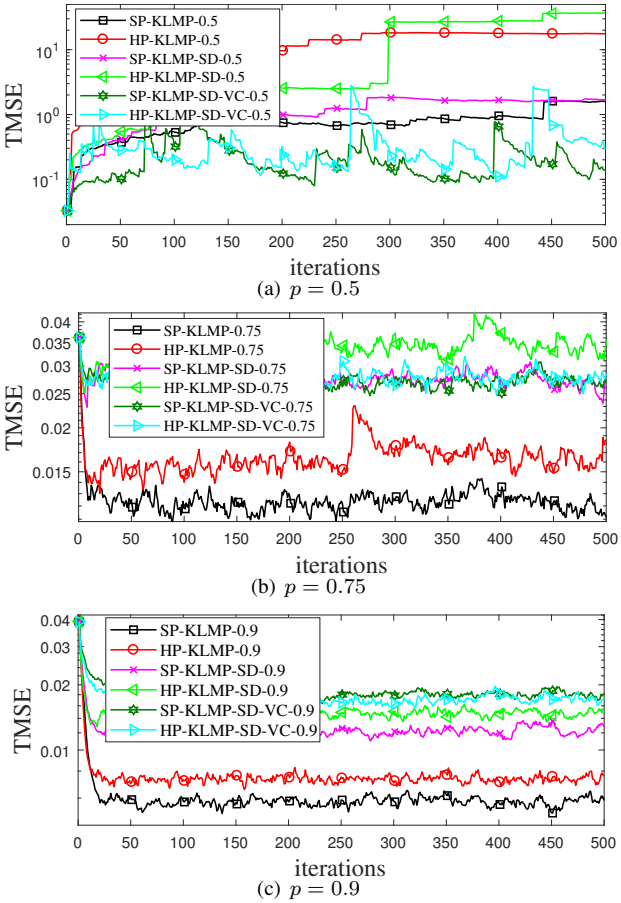


Fig. 7: The averaged TMSE learning curves for proposed KLMPs with $p \in \{0.5, 0.75, 0.9\}$ in SFE.

method is also applied to these criteria resulting in Q-LEL, Q-KRSL, Q-KMPE and Q-MC. For all algorithms, the step-size is 0.05; for LEL, the $p = 2$ and $\alpha = 1$; for KRSL, the $\lambda = 2$ and $\sigma_1 = 1$; for KMPE, the $p = 1$ and $\sigma = 2$; for MC, the $\sigma = 2$; for all SP algorithms, the $\varepsilon_c = 0.4$; for all quantized ones, the $\varepsilon_U = \sqrt{-2\ln(\varepsilon_c)}$. Fig. 8 plots the corresponding TMSE learning curves including the one of SP-KLMP-SD-VC. From this figure, we can observe that

- All SP algorithms achieve bettering filtering accuracy than their corresponding quantized versions.
- KAFs based on these error criteria cannot always realize better TMSE result than the KAF based on LMP, i.e., SP-KLMP-SD-VC.

TABLE II: Some averaged dictionary sizes for SP-KLMP algorithms in SFE.

ε_c	Size	ε_c	Size	ε_c	Size
0.1	2.66	0.2	3.08	0.3	3.46
0.4	3.94	0.5	4.44	0.6	5.04
0.7	5.86	0.8	7.02	0.9	10.42

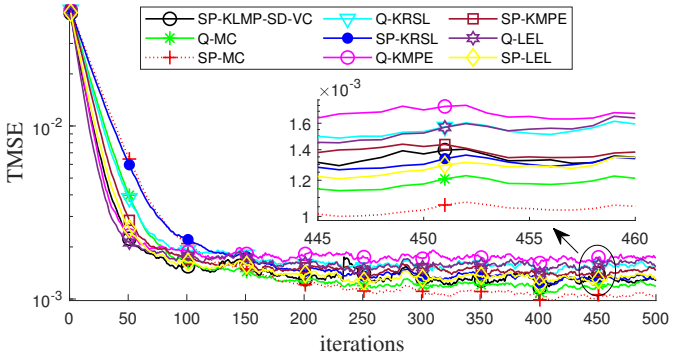


Fig. 8: The TMSE learning curves of SP and quantized KAFs based on various error criteria in static function estimation.

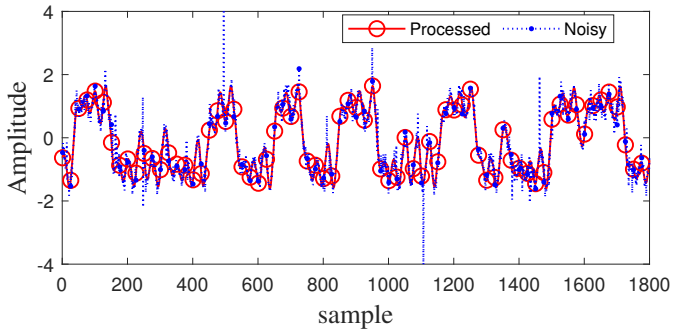


Fig. 9: A segment of the processed chaotic time series and its noisy version.

B. Prediction for Chua's Chaotic Time Series

Chua's circuit in a dimensionless form [23] is represented as

$$\begin{cases} x' = a(y - x - g(x)) \\ y' = x - y + z \\ z' = -by, \end{cases} \tag{51}$$

where $a > 0$, $b > 0$, and $g(x)$ is a piecewise-linear function given by

$$g(x) = \begin{cases} dx & -1 < x < 1 \\ cx + (d-c)\text{sgn}(x) & |x| \geq 1, \end{cases} \tag{52}$$

where $d < c < 0$. If we set $a = 15.6$, $b = 28$, $c = -1.143$, $d = -0.714$, and the initial conditions are $x(0) = 0.7$, $y(0) = z(0) = 0$, the system (51) generates chaotic time series. The time series is obtained using the function $ode45$ in Matlab. The first component (i.e., x) is used in the following learning tasks. Before the training, the signal is preprocessed to have zero mean and unit variance.

In this part, we apply the latest five samples $\boldsymbol{u}(n) = [x(n-5), \ldots, x(n-1)]^T$ to predict the current point $d(n) = x(n)$. We separate the whole sampled data into 40 segments with 1800×200 data points in total for performance evaluation, and in each simulation, the training size is 1800 and the testing size is 200. The performance is evaluated over 40 independent simulations. A segment of the processed Chua's chaotic time series and its noisy version contaminated by α-SN with $\boldsymbol{P}_\alpha = [1.6, 0, 0.1, 0]$ are shown in Fig. 9. In the simulation, the Gaussian kernel with $h = 0.75$ is used for all KLMPs with $p = 1.5$, and other parameters are listed in Table III. Fig. 10 plots the TMSE curves for various KLMP algorithms, and Fig. 11 shows the network size evolution curves for these KLMP algorithms. We can have the following observations from both figures:

- At almost the same initial convergence rate, SP-KLMP-SD-VC and HP-KLMP-SD-VC achieve better filtering accuracy compared with other algorithms;
- All PKLMP algorithms achieve smaller TMSE than NC-KLMP, CC-KLMP, RF-KLMP and Nys-KLMP;
- The VC method can increase convergence rate and improve filtering accuracy for SP-KLMP-SD and HP-KLMP-SD;
- Three SP-KLMPs (i.e., SP-KLMP, SP-KLMP-SD, and SP-KLMP-SD-VC) achieve slightly better filtering accuracy than the corresponding three HP-KLMPs (i.e., HP-KLMP, HP-KLMP-SD, and HP-KLMP-SD-VC);
- All PKLMP algorithms show almost the same network size evolution behavior, and they achieve much smaller network size than NC-KLMP and CC-KLMP.

Furthermore, we also compare SP-KLMP-SD-VC with other SP and quantized algorithms mentioned in Fig. 8. Table III lists the parameters and steady-state TMSE reults. The corresponding averaged TMSE learning curves are shown in Fig. 12. From this figure, we can see that

- Among all KAFs, SP-KLMP-SD-VC achieves the best TMSE result;
- In comparison with VQ, the VP method can slightly improve the filtering accuracy of KAFs.

TABLE III: Parameters setting and filtering results for Chua's chaotic time series.

Algorithm	Parameters	TMSE	Size
NC-KLMP	$\eta = 0.115$ $\delta_1 = 0.09$ $\delta_2 = 0.002$	0.0130	364.8
CC-KLMP	$\eta = 0.115$ $\varepsilon_c = 0.991$	0.0153	306.3
SP-KLMP	$\eta = 0.05$	0.0069	40.5
HP-KLMP	$\varepsilon_c = 0.9$	0.0070	40.5
SP-KLMP-SD	$\eta = 0.05$ $\eta_1 = 0.1$	0.0067	40.5
HP-KLMP-SD	$\varepsilon_c = 0.9$	0.0068	40.5
SP-KLMP-SD-VC	$\eta = 0.05$ $\eta_1 = 0.1$	**0.0055**	39.8
HP-KLMP-SD-VC	$\eta_2 = 0.5$ $\varepsilon_c = 0.9$	0.0057	39.9
RF-KLMP	$\sigma = 1$ $D = 80$ $\eta = 0.005$	0.0157	–
Nys-KLMP	$\eta = 0.1$ $\sigma = 1$ $m = 200$	0.0094	–
Q-KRSL	$\varepsilon_U = 0.3443$ $\sigma_1 = 1$	0.0073	40.5
SP-KRSL	$\lambda = 2$ $\eta = 0.1$ $\varepsilon_c = 0.9$	0.0071	40.5
Q-KMPE	$\varepsilon_U = 0.3443$ $\sigma = 2$	0.0084	40.5
SP-KMPE	$p = 1$ $\eta = 0.1$ $\varepsilon_c = 0.9$	0.0084	40.5
Q-LEL	$\varepsilon_U = 0.3443$ $\alpha = 1$	0.0077	40.5
SP-LEL	$p = 2$ $\eta = 0.1$ $\varepsilon_c = 0.9$	0.0075	40.5
Q-MC	$\varepsilon_U = 0.3443$ $\sigma = 2$	0.0067	40.5
SP-MC	$\eta = 0.1$ $\varepsilon_c = 0.9$	0.0066	40.5

C. Prediction for Real-World Data

In this experiment, two types of real-world data is adopted to test the performance of SP-KLMP-SD-VC and HP-KLMP-SD-VC, which achieve better filtering performance than other PKLMPs as demonstrated in Fig. 10. The first dataset is the

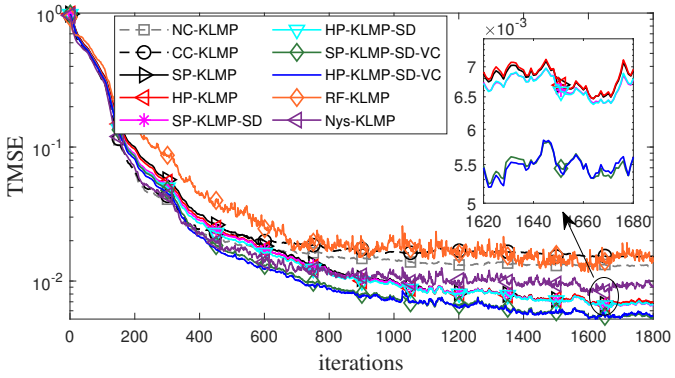


Fig. 10: The TMSE learning curves for different KLMP algorithms in Chua's chaotic time series.

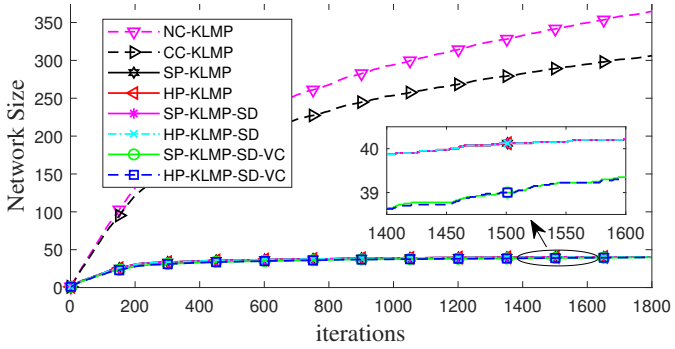


Fig. 11: The network size evolution curves for different KLMP algorithms Chua's chaotic time series.

Internet traffic named A5M from the homepage of Prof. Paulo Cortez[3]. There are 14772 samples in the dataset. In this first trial, the task is to predict the current point using the previous ten consecutive points, and we separate the whole data into 7 segments with 1800×200 samples. The second dataset is the Sunspot number recorded from January 1749 to September 2018[4], and the total number of the dataset is 3235. In the second trial, we use the 4 past points to predict the current one, and apply the first 3000 samples to train KAFs and the rest 235 points for testing.

For convenience of computation, the A5M dataset and Sunspot dataset are normalized to the range $[0, 1]$. For the two trials, we set $h = 1$ and $p = 3$ for A5M, and set $h = 0.5$ and $p = 2$ for Sunspot. In addition, Table IV and Table V list other parameters for the two datasets, respectively, which are chosen such that all algorithms produce almost the same final network size.

Fig. 13 and Fig. 14 plot the TMSE learning curves different sparsified KLMP algorithms and KAFs based on various error criteria for the Internet traffic dataset, respectively. From these two figures, we can see:

- HP-KLMP-SD-VC and SP-KLMP-SD-VC can realize almost the same filtering performance as that of Nys-KLMP. And these three algorithms outperform CC-KLMP, NC-KLMP and RF-KLMP;

[3]http://www3.dsi.uminho.pt/pcortez/series/
[4]http://www.sidc.be/silso/datafiles

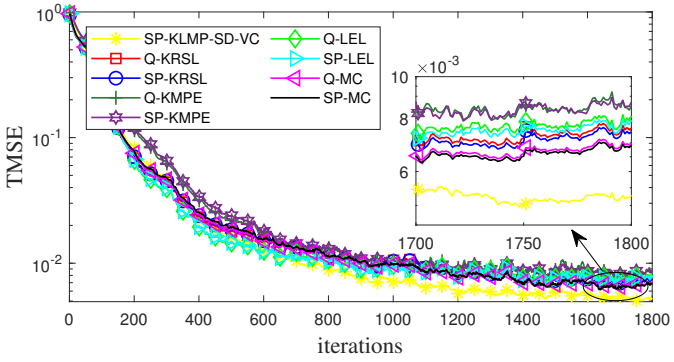


Fig. 12: The TMSE learning curves of SP and quantized KAFs based on various error criteria in Chua's chaotic time series.

- SP-KLMP-SD-VC achieves the best filtering accuracy among all KAFs. In addition, various SP algorithms realize slightly better TMSE results than their corresponding quantized algorithms as shown in Table IV.

Fig. 15 and Fig. 16 plot the TMSE learning curves different sparsified KLMP algorithms and KAFs based on various error criteria for the Internet traffic dataset, respectively. From these two figures, we can see:

- HP-KLMP-SD-VC and SP-KLMP-SD-VC can achieve faster convergence rate than RF-KLMP and Nys-KLMP. All of the four schemes can realize similar steady-state misalignment and they outperform CC-KLMP and NC-KLMP;
- Among all KAFs, SP-KLMP-SD-VC realizes the fastest convergence rate. KAFs sparsified by VP can realize almost the same filtering accuracy as the quantized algorithms as shown in Table V.

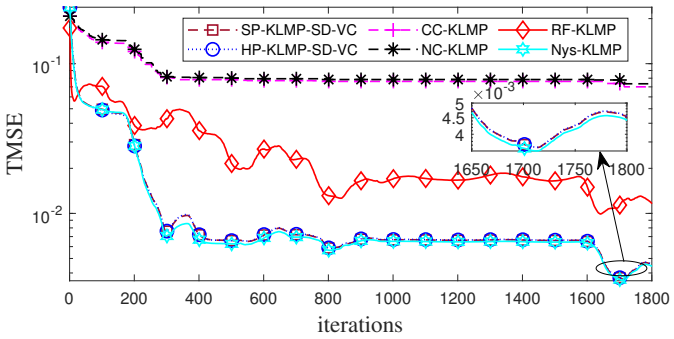


Fig. 13: The TMSE learning curves for different KLMP algorithms in Internet traffic dataset.

VI. Conclusion

In this paper, we applied vector projection to suppress the growth of network size for the KLMP algorithm, and obtained a new class of SKAF, named projected KLMP (PKLMP). Theoretical convergence analyses were provided and validated by simulation results. Based on the stability analysis in RKHS, we derived a sufficient condition for the mean-square convergence for PKLMP. Based on a Taylor expansion method, we derived the general EMSE expression for PKLMP, and obtained its

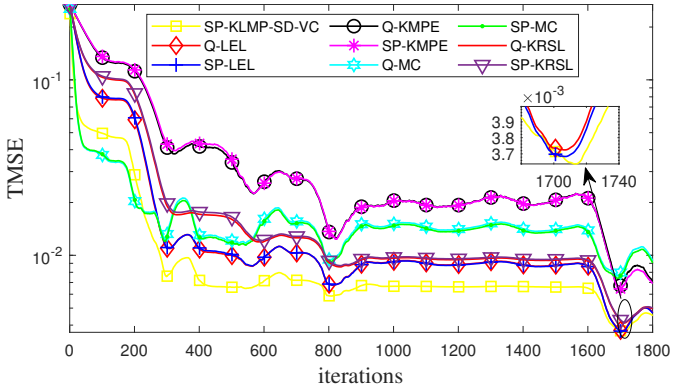


Fig. 14: The TMSE learning curves of SP and quantized KAFs based on various error criteria in Internet traffic dataset.

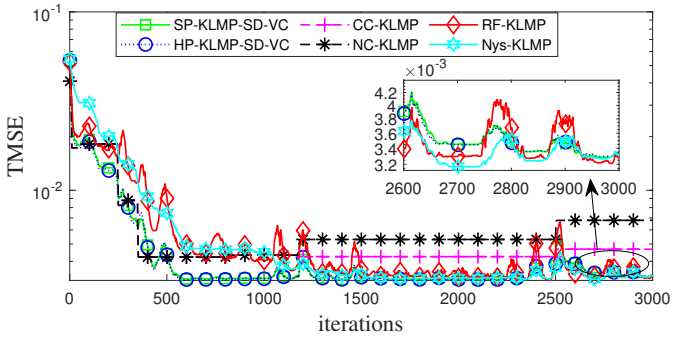


Fig. 15: The TMSE learning curves for different KLMP algorithms for Sunspot dataset.

lower and upper bounds when a Gaussian kernel is used. For $p \in [1, 2)$ we show analytically that PKLMP can achieve better filtering accuracy than QKLMS under impulsive noises. In addition, with some approximations, we conducted the mean convergence analysis for PKLMP. We applied the gradient method to update the output of MRC, which is shown to be able to better exploit the information contained in a desired output. Finally we presented a simple yet efficient variable centers method, which can improve the filtering accuracy and convergence rate for PKLMP. A number of simulations validate the efficiency of proposed algorithms.

TABLE IV: Parameters setting and filtering results for internet traffic

Algorithm	Parameters	TMSE	Size
NC-KLMP	$\eta = 0.2\ \delta_1 = 0.3\ \delta_2 = 0.01$	0.0741	12.57
CC-KLMP	$\eta = 0.2\ \varepsilon_c = 0.96$	0.0706	13.14
SP-KLMP-SD-VC	$\eta = 0.05\ \eta_1 = 0.1$	0.0042	12.86
HP-KLMP-SD-VC	$\eta_2 = 0.5\ \varepsilon_c = 0.96$	0.0042	12.86
RF-KLMP	$\sigma = 2\ D = 50\ \eta = 0.01$	0.0124	–
Nys-KLMP	$\eta = 0.1\ \sigma = 1\ m = 100$	**0.0041**	–
Q-KRSL	$\varepsilon_U = 0.2857\ \sigma_1 = 1$	0.0046	13.14
SP-KRSL	$\lambda = 2\ \eta = 0.1\ \varepsilon_c = 0.96$	0.0047	13.14
Q-KMPE	$\varepsilon_U = 0.2857\ \sigma = 2$	0.0080	13.14
SP-KMPE	$p = 1\ \eta = 0.1\ \varepsilon_c = 0.96$	0.0076	13.14
Q-LEL	$\varepsilon_U = 0.2857\ \alpha = 1$	0.0045	13.14
SP-LEL	$p = 2\ \eta = 0.01\ \varepsilon_c = 0.96$	0.0045	13.14
Q-MC	$\varepsilon_U = 0.2857\ \sigma = 1$	0.0101	13.14
SP-MC	$\eta = 0.05\ \varepsilon_c = 0.96$	0.0097	13.14

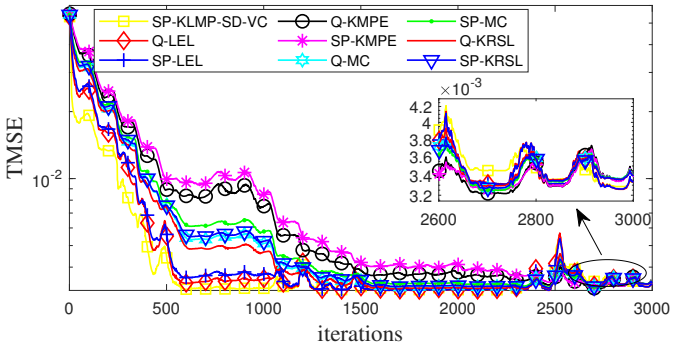


Fig. 16: The TMSE learning curves of SP and quantized KAFs based on various error criteria for Sunspot dataset.

TABLE V: Parameters setting and filtering results for sunspot

Algorithm	Parameters	TMSE	Size
NC-KLMP	$\eta = 0.15\ \delta_1 = 0.3\ \delta_2 = 0.025$	0.0068	18
CC-KLMP	$\eta = 0.15\ \varepsilon_c = 0.83$	0.0047	17
SP-KLMP-SD-VC	$\eta = 0.02\ \eta_1 = 0.1$	**0.0033**	17
HP-KLMP-SD-VC	$\eta_2 = 0.1\ \varepsilon_c = 0.83$	**0.0033**	17
RF-KLMP	$\sigma = 0.5\ D = 50\ \eta = 0.001$	0.0034	–
Nys-KLMP	$\eta = 0.01\ \sigma = 0.5\ m = 200$	**0.0033**	–
Q-KRSL	$\varepsilon_U = 0.3052\ \sigma_1 = 1$	0.0034	17
SP-KRSL	$\lambda = 2\ \eta = 0.01\ \varepsilon_c = 0.83$	0.0034	17
Q-KMPE	$\varepsilon_U = 0.3052\ \sigma = 2$	0.0035	17
SP-KMPE	$p = 1.5\ \eta = 0.01\ \varepsilon_c = 0.83$	0.0034	17
Q-LEL	$\varepsilon_U = 0.3052\ \alpha = 1$	0.0034	17
SP-LEL	$p = 2\ \eta = 0.01\ \varepsilon_c = 0.83$	0.0034	17
Q-MC	$\varepsilon_U = 0.3052\ \sigma = 1$	0.0034	17
SP-MC	$\eta = 0.01\ \varepsilon_c = 0.83$	0.0034	17

The methods provided in this paper can be extended to other KAFs based on some error criteria, i.e., LEL, KRSL, KMPE and MC. Particularly, when SD and VC methods are applied to these criteria, one can change the cost function (41) to one of them, which may improve the performance of KAFs. Furthermore, the VP method only considers simplified cosine-relation of two transformed data, some complicated relationships, such as transformed data distribution, may be applied to vector projection to achieve better performance.

REFERENCES

[1] H. Fan and Q. Song, "A linear recurrent kernel online learning algorithm with sparse updates," *Neural Netw.*, vol. 50, pp. 142–153, 2014.

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

[3] 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.

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

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

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

[7] W. Liu, I. Park, and J. C. Príncipe, "An information theoretic approach of designing sparse kernel adaptive filters," *IEEE Trans. Neural Netw.*, vol. 20, no. 12, pp. 1950–1961, Dec. 2009.

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

[9] B. Chen, N. Zheng, and J. C. Príncipe, "Sparse kernel recursive least squares using L1 regularization and a fixed-point sub-iteration," in *IEEE Int. Conf., Acoust. Speech Signal Process. (ICASSP)*, 2014, pp. 5257–5261.

[10] S. Zhao, B. Chen, P. Zhu, and J. C. Príncipe, "Fixed budget quantized kernel least-mean-square algorithm," *Signal Process.*, vol. 93, no. 9, pp. 2759–2770, 2013.

[11] V. S. Vaerenbergh, J. Via, and I. Santamaria, "A sliding-window kernel RLS algorithm algorithm and its application to nonliear channel identification," in *IEEE Int. Conf., Acoust. Speech Signal Process. (ICASSP)*, 2006, pp. 789–792.

[12] J. Zhao, H. Zhang, and G. Wang, "Projected kernel recursive maximum correntropy,", *IEEE Trans. Circuits Syst. II Express Briefs,* vol. 65, no. 7, pp. 963–967, Jul. 2018.

[13] J. Zhao and H. Zhang, "Projected kernel recursive least squares algorithm," in *Int. Conf., Neural Inform., Process. (ICONIP)*, 2017, pp. 356–365.

[14] Q. Y. Miao and C. G. Li, "Kernel least-mean mixed-norm algorithm," in *Int. Conf., Automat. Contor. Artif. Intell. (ACAI)*, 2012, vol.2 , no.6 pp. 1285–1288.

[15] S. Wang, J. Feng, and C. K. Tse, "Kernel affine projection sign algorithms for combating impulse interference," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 60, no. 11, pp. 811–815, Nov. 2013.

[16] S. Pei and C. Tseng, "Least mean p-power error criterion for adaptive FIR filter," *IEEE J. Sel. areas Commun.*, vol. 12, no. 9, pp. 154–1547, Dec. 1994.

[17] F. Wen, "Diffusion least-mean p-power algorithms for distributed estimation in alpha-stable noise environments," *Electron. Lett.*, vol. 49, no. 21, pp. 1355–1356, 2013.

[18] B. Chen, L. Xing, Z. Wu, J. Liang, and J. C. Prncipe, "Smoothed least mean p-power error criterion for adaptive filtering," *Digit. Signal Process.*, vol.40, pp. 154–163, 2015.

[19] M. Belge and E. L. Miller, "A sliding window RLS-like adaptive algorithm for filtering alpha-stable noise," *IEEE Signal Process. Lett.*, vol. 7, no. 4, pp. 86–89, Apr. 2000.

[20] A. N. Vzquez and J. A. Garca, "Combination of recursive least p-norm algorithms for filtering in alpha-stable noise," *IEEE Trans. Signal Process.*, vol. 60, no. 3, pp. 1478–1482, Mar. 2012.

[21] W. Gao and J. Chen, "Kernel least mean p-power algorithm," *IEEE Signal Process. Lett.*, vol. 24, no. 7, pp. 996–1000, Jul. 2017.

[22] W. Ma, J. Duan, W. Man, H. Zhao, and B. Chen, "Robust kernel adaptive filters based on mean p-power error for noisy chaotic time series prediction," *Eng. Appl. Artif. Intell.*, vol. 58, pp. 101–110, 2017.

[23] Y. Zheng, S. Wang, J. Feng, and C. K. Tse, "A modified quantized kernel least mean square algorithm for prediction of chaotic time series," *Digit. Signal Process.,* vol. 48, pp. 130–136, 2016.

[24] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*, New York: Wiley, 2010.

[25] C. A. Micchelli, X. Yuesheng, and Z. Haizhang, "Universal Kernels," *J. Mach. Learn. Res.*, vol. 7, pp. 2651–2667, 2006.

[26] M. Yukawa and R. Ishii, "An efficient kernel adaptive filtering algorithm using hyperplane projection along affine subspace," *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, Bucharest, pp. 2183–2187, 2012.

[27] S. Theodoridis, K. Slavakis and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 97–123, Jan. 2011.

[28] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 121–167, 1998.

[29] T. Y. A. Naffouri and A. H. Sayed, "Adaptive filters with error nonlinearities: Mean-square analysis and optimum design," *EURASIP J. Appl. Signal Processing*, vol. 2001, no. 4, pp. 192–205, 2001.

[30] X. Luo, J. Deng, W. Wang, J. Wang and W. Zhao, "A quantized kernel learning algorithm using a minimum kernel risk-sensitive loss criterion and bilateral gradient technique," *Entropy*, vol. 19, no. 7, p. 365, 2017.

[31] B. Lin, R. He, X. Wang and B. Wang, "The steady-state mean-square error analysis for least mean p-order algorithm," *IEEE Signal Process. Lett.*, vol. 16, no. 3, pp. 176–179, Mar. 2009.

[32] M. Shao and C.L. Nikias, "Signal processing with fractional lower order moments: stable processes and their applications," *Proceed. IEEE*, vol. 81, no. 7, pp. 986–1010, Jul. 1993.

[33] B. Weng and K. E. Barner, "Nonlinear system identification in impulsive environments," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2588–2594, Jul. 2005.

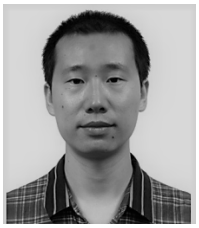
[34] S. Wang, Y. Zheng, S. Duan, L. Wang and H. Tao, "Quantized kernel maximum correntropy and its mean square convergence analysis," *Digit. Signal Process.*, vol. 63, pp. 164–176, Apr. 2017.

[35] A. Sign, N. Ahuja, and P. Mouline, "Online learning with kernels: Overcoming the growing sum problem," in: *Proc. IEEE Int. Workshop Mach. Learn. Signal Process.*, Sep. 2012, pp. 1–6.

[36] S. Wang, W. Wang, L. Dang, and Y. Jiang, "Kernel least mean square based on the Nyström method," *Circuits Syst. Signal Process.*, vol.38, pp. 3133–3151, 2019.

[37] K. Xiong and S. Wang, "Robust least mean logarithmic square adaptive filtering algorithms," *J. Franklin I.*, vol.356, pp. 654–674, 2019.

[38] B. Chen, L. Xing, X. Wang, J. Qin, and N. Zheng, "Robust learning with kernel mean p-power error loss," *IEEE Trans. Cybern.*, vol.48, no.7, pp. 2101–2113, Jul. 2018.



Ji Zhao received the BEng degree in Communication Engineering from Southwest University of Science and Technology, Mianyang, China, in 2013, and the MEng degrees in Signal and Information Processing from Southwest University, Chongqing, China, in 2016. Now, he is a PhD. candidate of Circuit and System in University of Electronic Science and Technology of China, Chengdu. From October 2018 to October 2019, he was a joint PhD. Student in Global Big Data Technologies Centre, University of Technology Sydney, Sydney, NSW 2007 Australia. His current research interests include adaptive kernel filtering, signal processing, and machine learning.

Hongbin Zhang (M'06-SM'12) received the BEng degree in Aerocraft Design from Northwestern Polytechnical University, Xian, China, in 1999, and the MEng and PhD. degrees in Circuits and Systems from the University of Electronic Science and Technology of China, Chengdu, in 2002 and 2006, respectively. From 2002 to 2017, He was a Professor with the School of Electrical Engineering, University of Electronic Science and Technology of China. From 2018, he has been a Professor with School of Information and Communication Engineering, University of Electronic Science and Technology of China. From December 2011 to December 2014, he was a Post-Doctor with School of Automation,, Nanjing University of Science and Technology, Nanjing, Jiangsu. His current research interests include fuzzy control, stochastic control and time-delay control systems, and non-linear signal processing.

Gang Wang received the B.E. degree in communication engineering and the Ph.D. degree in biomedical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 1999 and 2008, respectively, where he is currently an Associate Professor with the School of Information and Communication Engineering.His current research interests include distributed signal processing and intelligent systems.

J. Andrew Zhang (M'04-SM'11) received the B.Sc. degree from Xi'an JiaoTong University, China, in 1996, the M.Sc. degree from Nanjing University of Posts and Telecommunications, China, in 1999, and the Ph.D. degree from the Australian National University, in 2004. Currently, Dr. Zhang is an Associate Professor in the School of Electrical and Data Engineering, University of Technology Sydney, Australia. He was a researcher with Data61, CSIRO, Australia from 2010 to 2016, the Networked Systems, NICTA, Australia from 2004 to 2010, and ZTE Corp., Nanjing, China from 1999 to 2001. Dr. Zhang's research interests are in the area of signal processing for wireless communications and sensing. He has published more than 170 papers in leading international Journals and conference proceedings, and has won 5 best paper awards. He is a recipient of CSIRO Chairman's Medal and the Australian Engineering Innovation Award in 2012 for exceptional research achievements in multi-gigabit wireless communications.