A Kernel Affine Projection-Like Algorithm in Reproducing Kernel Hilbert Space

Qishuai Wu, Yingsong Li, *Senior Member, IEEE,* Yuriy V. Zakharov, *Senior Member, IEEE,* Wei Xue, Wanlu Shi

***Abstract*—A kernel affine projection-like algorithm (KAPLA) is proposed in reproducing kernel Hilbert space in non-Gaussian environments. The cost function for the developed algorithm is constructed by using the correntropy approach and Gaussian kernel to deal with nonlinear channel estimation. The devised algorithm can efficiently operate in the impulse noise. As a consequence, the proposed KAPLA algorithm provides good performance for nonlinear channel equalization in implulse-noise environments. Simulations results in different mixed noise environments verify the superior behavior of KAPLA compared to known algorithms.**

***Index Terms*—correntropy; reproducing kernel Hilbert space; kernel affine projection-like algorithm; non-Gaussian environments**

I. Introduction

Kernel method is powerful non-parametric modeling technique, which is popular in nonlinear adaptive filtering (AF) [1]. The kernel AF (KAF) algorithm employs the kernel learning method implemented in reproducing kernel Hilbert spaces (RKHS). The kernel least-mean-square (KLMS) algorithm [2] and its multiple variants were developed for non-linear signal processing [3]–[8]. Furthermore, the kernel affine projection (AP) algorithm (KAPA) has been proposed [8], which can reduce the gradient noise. However, the KLMS algorithm and KAPA are constructed by minimizing the squared error, which might result in performance degradation in scenarios with non-Gaussian noise [9], [10], which might result in performance degradation.

To improve the estimation performance in non-Gaussian environments, the AP sign (APS) algorithm, the maximum correntropy criterion (MCC) algorithm and their variants have been proposed and investigated in recent works [11]–[18]. The APS and MCC-based algorithms converge fast and achieve low mean square error (MSE) in non-Gaussian noisy environments. The kernel MCC (KMCC) algorithm [19] is developed for identifying non-linear systems under non-Gaussian interference by introducing the kernel method into the MCC algorithm. Exploiting the RKHS, the kernel APS (KAPS) algorithm has been developed [20]. One can notice that from the KLMS algorithm to the KAPA and KMCC algorithms, the kernel theory was successfully employed to enhance the algorithm behavior in noisy non-Gaussian environments.

In this paper, the nonlinear channel equalization (NCE) problem in non-Gaussian noise is considered. Herein, the AP approach and MCC are combined to devise a cost function for development of a novel robust algorithm in non-Gaussian environments. The gradient descent principle and the Lagrange multiplier method are used to derive the update recursion for an affine projection-like algorithm (APLA). Then, the kernel method in RKHS is incorporated into the recursionfor the NCE problem. Finally, the update recursion for the new algorithm, namely the kernel APLA (KAPLA), is proposed. Simulated results show that the KAPLA outperforms the KAPA, KLMS, KAPS and KMCC algorithms in the convergence speed and steady state estimation error.

The structure of the manuscript is presented below. Section II introduces the KAPA algorithm. Section III presents the derivation of the proposed KAPLA. Section IV verifies the KAPLA's behavior via computer simulations. Finally, in Section V, the conclusion is given.

II. Review of the KAPA algorithm

In the classical AP adaptive filter, the input matrix $\mathbf{U}(n) = [\mathbf{u}(n), \mathbf{u}(n-1), ..., \mathbf{u}(n-M+1)]$ groups M most recent signal vectors $\mathbf{u}(n) = [u(n), u(n-1), u(n-2), ..., u(n-L+1)]^T$, where n indicates the time slot, and L represents the filter length. In the APA, the output $M \times 1$ vector $\mathbf{y}(n)$ is

$$\mathbf{y}(n) = \mathbf{U}^T(n)\,\mathbf{w}(n-1), \tag{1}$$

and the a priori error vector $\mathbf{e}(n)$ is given by

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n), \tag{2}$$

where the desired signal vector is $\mathbf{d}(n) = [d(n), d(n-1), d(n-2), ..., d(n-M+1)]^T$, $d(n)$ denotes the desired signal, and $\mathbf{w}(n-1) \in R^{L\times 1}$ denotes the AF weighting vector at instant $n-1$. The APA update recursion is expressed as [21]

$$\begin{aligned}\mathbf{w}(n) = \mathbf{w}(n-1) \\ + \xi\mathbf{U}(n)\left[\mathbf{U}^T(n)\,\mathbf{U}(n) + \varepsilon\mathbf{I}_M\right]^{-1}\mathbf{e}(n),\end{aligned} \tag{3}$$

Manuscript received on June 12, 2019; revised on Aug. 30, 2019, and revised on .

This work was supported in part by the Fundamental Research Funds for the Central Universities under Grants HEUCFG201829 and 3072019CFG0801 and the National Key Research and Development Program of China under Grant 2016YFE0111100, and the China Postdoctoral Science Foundation under Grants 2017M620918 and 2019T120134.

Qishuai Wu, Yingsong Li, Wei Xue, Wanlu Shi are with the College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China. (e-mail: liyingsong@ieee.org).

Yingsong Li is also with the Key Laboratory of Microwave Remote Sensing, National Space Science Center, Chinese Academy of Sciences, Beijing 100190, China

Yuriy V. Zakharov is with the Department of Electronic Engineering, University of York, York YO10 5DD, U.K.

where $\varepsilon > 0$ is a regularization factor, $\mathbf{I}_M$ is an M-order identity matrix and ξ is the step-size.

The idea of the kernel method is to map the input signal vector space $\mathbf{U}$ into a high-dimensional featured space $\mathbf{F}$, where the mapping $\boldsymbol{\varphi}$ is constructed as $\boldsymbol{\varphi} : \mathbf{U} \to \mathbf{F}$. Then, the kernel method is integrated into linear AF algorithms with the help of the Mercer's theorem [1]

$$\kappa(\mathbf{u}, \mathbf{u}') = \boldsymbol{\varphi}^T(\mathbf{u})\boldsymbol{\varphi}(\mathbf{u}'), \tag{4}$$

which defines the relationship between the kernel $\kappa(\mathbf{u}, \mathbf{u}')$ and the mapping $\boldsymbol{\varphi}$, and where (4) is also regarded as the kernel trick. Usually, the Gaussian kernel defined as [1]

$$\kappa(\mathbf{u}, \mathbf{u}') = \exp\left(-\frac{\|\mathbf{u} - \mathbf{u}'\|}{\sigma^2}\right) \tag{5}$$

is considered. Herein, σ denotes the kernel width. According to the kernel method, $\mathbf{u}(n)$ is mapped into a featured space $\mathbf{F}$ as $\boldsymbol{\varphi}(\mathbf{u}(n))$; below, we denote $\boldsymbol{\varphi}(n) = \boldsymbol{\varphi}(\mathbf{u}(n))$. Straightforward calculations transform the recursion (3) into

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \xi\boldsymbol{\Phi}(n)[\mathbf{G}(n) + \varepsilon\mathbf{I}_M]^{-1}\mathbf{e}(n), \tag{6}$$

where $\boldsymbol{\Phi}(n) = [\boldsymbol{\varphi}(n), \boldsymbol{\varphi}(n-1), ..., \boldsymbol{\varphi}(n-M+1)]$, and $\mathbf{e}(n)$ represents a priori-error in RKHS given by $\mathbf{e}(n) = \mathbf{d}(n) - \boldsymbol{\Phi}^T(n)\mathbf{w}(n-1)$, and $\mathbf{G}(n) = \boldsymbol{\Phi}^T(n)\boldsymbol{\Phi}(n)$.

Based on (6), the recursion can be factorized and described as follows.

$$\begin{cases} \mathbf{w}(0) = 0, \\ \mathbf{w}(1) = \xi d(1)\boldsymbol{\varphi}(1) = \mathbf{a}_1(1)\boldsymbol{\varphi}(1), \\ \quad\vdots \\ \mathbf{w}(n-1) = \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\boldsymbol{\varphi}(m), \\ \mathbf{w}(n) = \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\boldsymbol{\varphi}(m) \\ \quad + \xi\boldsymbol{\Phi}(n)[\mathbf{G}(n) + \varepsilon\mathbf{I}_M]^{-1}\mathbf{e}(n), \end{cases} \tag{7}$$

where $\mathbf{a}_m(n-1)$ is the mth element in $\mathbf{a}(n-1)$, which is the expansion coefficient vector. According to (7), the weighting vector in the featured space is modified to be

$$\mathbf{w}(n) = \sum_{m=1}^{n}\mathbf{a}_m(n)\boldsymbol{\varphi}(m). \tag{8}$$

From the above derivation, elements of the expansion coefficient vector can be obtained as:

$$\mathbf{a}_j(n) = \begin{cases} \xi e_{n+1-j}(n)[\mathbf{G}(n) + \varepsilon\mathbf{I}_M]^{-1}, \text{ if } j = n, \\ \mathbf{a}_j(n-1) + \xi e_{n+1-j}(n)[\mathbf{G}(j) + \varepsilon\mathbf{I}_M]^{-1}, \\ \text{ if } n - M + 1 \le j \le n-1, \\ \mathbf{a}_j(n-1), \text{ if } 1 \le j < n - M + 1, \end{cases} \tag{9}$$

where $e_{n+1-j}(n)$ is obtained as

$$e_{n+1-j}(n) = d(j) - \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\kappa_{n,m}, \tag{10}$$

where $\kappa_{n,m} = \kappa(\mathbf{u}(n), \mathbf{u}(m))$ and $e_{n+1-j}(n)$ is the a priori error of $\{\mathbf{u}(j), d(j)\}$ using $\mathbf{w}(n-1)$. Equation (9) is interpreted as follows. Firstly, a new element $\mathbf{a}_n(n)$ is set to $\xi e_1(n)[\mathbf{G}(n) + \varepsilon\mathbf{I}_M]^{-1}$. Then, the coefficients are updated for $(M-1)$ most recent elements by adding $\xi e_{n+1-j}(n)[\mathbf{G}(j) + \varepsilon\mathbf{I}_M]^{-1}$ for $(n - M + 1) \le j \le (n-1)$, and the other coefficients are unchanged.

III. KERNEL AFFINE PROJECTION-LIKE ALGORITHM

In this section, the KAPLA is derived. The algorithm is devised via finding the solution of the following minimization problem [22], [23]

$$\begin{array}{ll} \min\limits_{\mathbf{w}(n)} & \|\mathbf{w}(n) - \mathbf{w}(n-1)\|^2 \\ s.t. & \tilde{\mathbf{e}}(n) = [\mathbf{1}_M - \xi\mathbf{b}(n)] \odot \mathbf{e}(n), \end{array} \tag{11}$$

where $\mathbf{1}_M = [1, 1, ..., 1]^T$ represents a unity $M \times 1$ vector, $\mathbf{b}(n) = \exp\left(-\frac{\mathbf{e}(n)\odot\mathbf{e}(n)}{2\delta^2}\right)$, δ represents a specified kernel width, $\odot$ denotes the Hadamard product, $\tilde{\mathbf{e}}(n) = \mathbf{d}(n) - \mathbf{U}^T(n)\mathbf{w}(n)$ are posteriori errors and $\|\cdot\|^2$ represents the Euclidean vector norm. According to the Lagrange multiplier method [21], the created new cost function is obtained as

$$\begin{aligned} J(n) = &\|\mathbf{w}(n) - \mathbf{w}(n-1)\|^2 \\ &+ \boldsymbol{\lambda}\{\tilde{\mathbf{e}}(n) - [\mathbf{1}_M - \xi\mathbf{b}(n)] \odot \mathbf{e}(n)\}, \end{aligned} \tag{12}$$

where $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, ..., \lambda_M]$ acts as the Lagrange multiplier vector. Taking the gradients

$$\begin{aligned} \frac{\partial J(n)}{\partial \mathbf{w}(n)} &= 2[\mathbf{w}(n) - \mathbf{w}(n-1)] - \mathbf{U}(n)\boldsymbol{\lambda}^T, \\ \frac{\partial J(n)}{\partial \boldsymbol{\lambda}} &= \tilde{\mathbf{e}}(n) - [\mathbf{1}_M - \xi\mathbf{b}(n)], \end{aligned} \tag{13}$$

and setting them to zero results in

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \frac{1}{2}\mathbf{U}(n)\boldsymbol{\lambda}^T, \tag{14}$$

$$\mathbf{d}(n) = \mathbf{U}^T(n)\mathbf{w}(n) + [\mathbf{1}_M - \xi\mathbf{b}(n)] \odot \mathbf{e}(n). \tag{15}$$

The vector $\boldsymbol{\lambda}^T$ is then given by

$$\boldsymbol{\lambda}^T = 2\xi\left[\mathbf{U}^T(n)\mathbf{U}(n)\right]^{-1}\mathbf{b}(n) \odot \mathbf{e}(n). \tag{16}$$

Substituting $\boldsymbol{\lambda}^T$ from (16) into (14), one can obtain the updating recursion for the APLA:

$$\begin{aligned} \mathbf{w}(n) = &\mathbf{w}(n-1) \\ &+ \xi\mathbf{U}(n)\left[\mathbf{U}^T(n)\mathbf{U}(n)\right]^{-1}\mathbf{b}(n) \odot \mathbf{e}(n). \end{aligned} \tag{17}$$

More generally, the updating recursion is generalized as:

$$\begin{aligned} &\mathbf{w}(n) = \mathbf{w}(n-1) \\ &+\xi\mathbf{U}(n)\left[\mathbf{U}^T(n)\mathbf{U}(n) + \varepsilon\mathbf{I}_M\right]^{-1}\mathbf{b}(n) \odot \mathbf{e}(n). \end{aligned} \tag{18}$$

Taking the kernel method into account, (18) becomes

$$\begin{aligned} &\mathbf{w}(n) = \mathbf{w}(n-1) \\ &+\xi\boldsymbol{\Phi}(n)[\mathbf{G}(n) + \varepsilon\mathbf{I}_M]^{-1}\mathbf{b}(n) \odot \mathbf{e}(n). \end{aligned} \tag{19}$$

According to (7) and (19), $\mathbf{w}(n)$ is given by

$$\begin{aligned} \mathbf{w}(n) = &\sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\boldsymbol{\varphi}(m) \\ &+\xi\boldsymbol{\Phi}(n)[\mathbf{G}(n) + \varepsilon\mathbf{I}_M]^{-1}\mathbf{b}(n) \odot \mathbf{e}(n). \end{aligned} \tag{20}$$

The estimate of $\mathbf{d}(n)$ is given by

$$\hat{\mathbf{d}}(n) = \mathbf{\Phi}^T(n)\,\mathbf{w}(n-1). \tag{21}$$

Noticing the kernel trick in (4), $\hat{\mathbf{d}}(n)$ can be transformed into

$$\hat{\mathbf{d}}(n) = \left[\sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\,\kappa_{n,m}, \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\kappa_{n-1,m}, \ldots, \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\kappa_{n-M+1,m}\right]^T. \tag{22}$$

The a priori error vector is then obtained as

$$\mathbf{e}(n) = \mathbf{d}(n) - \hat{\mathbf{d}}(n). \tag{23}$$

Substituting (22) into (23) yields

$$\mathbf{e}(n) = \mathbf{d}(n) - \left[\sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\,\kappa_{n,m}, \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\kappa_{n-1,m}, \ldots, \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\kappa_{n-M+1,m}\right]^T. \tag{24}$$

Comparing (8) and (21), the expansion coefficient vector is

$$\mathbf{a}_j(n) = \begin{cases} \xi p_{n+1-j}(n)\left[\mathbf{G}(n) + \varepsilon\mathbf{I}_M\right]^{-1}, \text{ if } j = n, \\ \mathbf{a}_j(n-1) + \xi p_{n+1-j}(n)\left[\mathbf{G}(j) + \varepsilon\mathbf{I}_M\right]^{-1}, \\ \quad \text{if } n-M+1 \le j \le n-1, \\ \mathbf{a}_j(n-1), \text{ if } 1 \le j < n-M+1, \end{cases} \tag{25}$$

where $p_{n+1-j}(n)$ is found from

$$p_{n+1-j}(n) = e_{n+1-j}(n)\exp\left(-\frac{(e_{n+1-j}(n))^2}{2\delta^2}\right). \tag{26}$$

The equation (25) can be interpreted as follows. Firstly, a new element $\mathbf{a}_n(n)$ is set to $\xi p_1(n)\left[\mathbf{G}(n) + \varepsilon\mathbf{I}_M\right]^{-1}$. Then, the $M-1$ most recent elements are updated by adding terms $\xi p_{n+1-j}(n)\left[\mathbf{G}(j) + \varepsilon\mathbf{I}_M\right]^{-1}$ for $(n-M+1) \le j \le (n-1)$, and the remaining coefficients are kept unchanged. The KAPLA is summarized in Algorithm 1.

TABLE I: Parameters of algorithms

Algorithm	ξ	M	δ	σ	ε
KLMS	0.02	-	-	1	-
KAPA	0.02	10	-	1	0.1
KMCC	0.02	-	0.55	1	-
KAPS	0.01	10	-	1	0.1
KAPLA	0.02	10	0.55	1	0.1
KAPLA	0.02	10	0.4	1	0.1

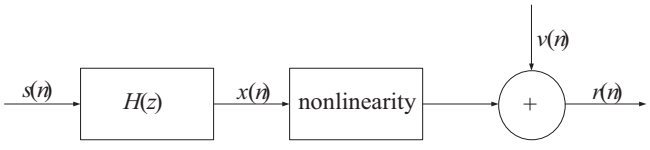

Fig. 1: Classic nonlinear channel structure.

Algorithm 1 KAPLA

Input: ξ, σ, δ, ε
Output: $\mathbf{a}$

Initialisation :
$\mathbf{a}_1(1) = \xi d(1)$
while $\{\mathbf{u}(n), d(n)\}$ are available **do**
  $\mathbf{a}_n(n-1) = 0$
  for $j = \max(1, n-M+1) : n$ **do**
    compute elements of the vector $\hat{\mathbf{d}}(j)$
    $\hat{d}(j) = \sum_{m=1}^{n-1}\mathbf{a}_m(n-1)\,\kappa_{j,m}$
    compute elements of the vector $\mathbf{e}(n)$:
    $e_{n+1-j}(n) = d(j) - \hat{d}(j)$
    update $\min\{n, M\}$ most recent elements $\mathbf{a}_j(n)$ as in (25)
    $\mathbf{a}_j(n) = \mathbf{a}_j(n-1) + \xi p_{n+1-j}(n)\left[\mathbf{G}(j) + \varepsilon\mathbf{I}_M\right]^{-1}$
  end for
  if $n > M$ **then**
    for $j = 1 : n-M$ **do**
      $\mathbf{a}_j(n) = \mathbf{a}_j(n-1)$
    end for
  end if
end while

IV. Simulation Results and Discussions

A NCE problem is now considered in different noise environments to analyze the robustness and estimation performance of the proposed KAPLA. In Fig. 1, a non-linear channel model consisting of a memoryless nonlinearity and a linear filter is presented. This channel should be equalized. A binary signal $s(n)$ is transmitted through the channel. The received signal $r(n)$ is observed in the presence of additive noise $v(n)$. The NCE is regarded as a regression problem with input-output data $\{[r(n), r(n+1), r(n+2), \ldots, r(n+l)], s(n-D)\}$. Herein, D and l are equalization lag time and the time embedding length [1], respectively; $l = 3$ and $D = 2$ are selected in the simulation. The linear filter has a transfer-function $H(z) = 1 - 0.5z^{-1}$. The received $x(n)$ is obtained as $x(n) = 0.5s(n-1) + s(n)$. The output signal $r(n)$ is given by $r(n) = -0.9x^2(n) + x(n) + v(n)$. The noise $v(n)$ is generated by mixing two noise signals, namely $v_1(n)$ and $v_2(n)$ [24]. Four noise models with zero-mean are considered as follows.

1) Bernoulli-distributed noise $v_1(n)$ that has a power of 0.45 mixed with a Gaussian-distributed noise $v_2(n)$ with power of 0.08 are employed in Simulation-1.

2) Laplace-distributed noise $v_1(n)$ that has a power of 0.45 mixed with Gaussian-distributed noise $v_2(n)$ that has a power of 0.08 are employed in Simulation-2.

3) Bernoulli-distributed noise $v_1(n)$ that has a power of 0.45 mixed with uniformly distributed noise $v_2(n)$ that has a power of 1 are employed in Simulation-3.

4) Bernoulli-distributed noise $v_1(n)$ that has a power of 0.45 is used in Simulation-4.

In these simulation experiments, the total noise power is 0.1 and other parameters are given in Table I. The simulation results are obtained by computing the average error from 50

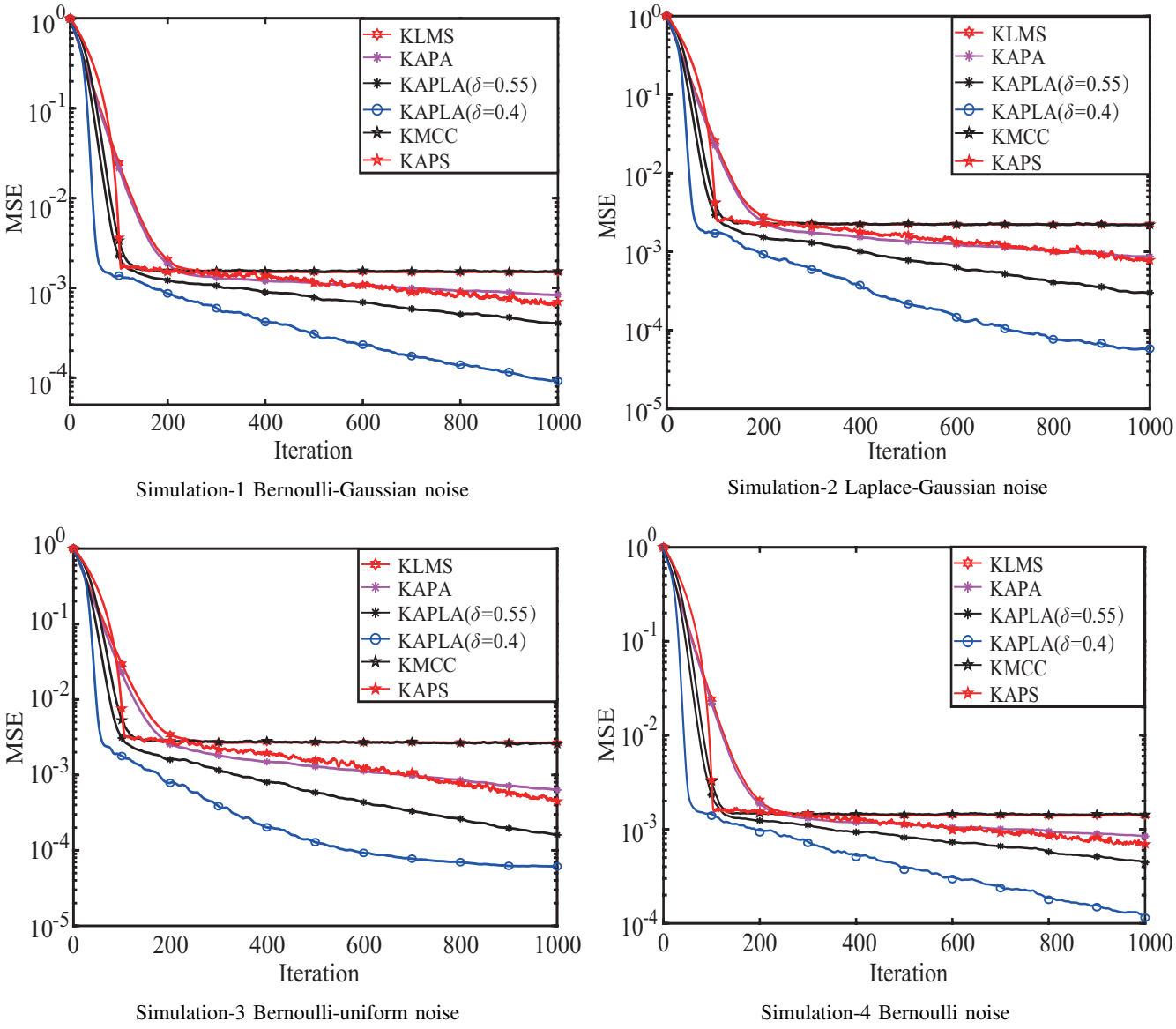

Fig. 2: Performance of the KAPLA under the mixed noise.

independent Monte-Carlo runs. The size of training data and testing data are set to 1000 and 100, respectively. The behaviors of the devised KAPLA compared with KLMS, KAPA, KMCC and KAPS algorithms in the four noise environments are presented in Fig. 2. It can be seen that the KAPLA provides the fastest convergence in all the scenarios. Moreover, the steady-state MSE of the KAPLA are lower than that of the other mentioned algorithms.

Next, the KAPLA is investigated to establish how the kernel-width δ influences its behavior. The kernel bandwidth σ is set to 1, and other KAPLA parameters are the same as those in TABLE I. The noise model in this experiment is the same as in Simulation-1, and the MSEs are obtained from the last 100 iterations, which are supposed to operate in the steady state. The obtained results are shown in Fig. 3, where it is seen that the MSE behavior of KAPLA is improved with the reduction of the kernel width δ within a certain range, and the value of δ has significant effect on the steady state MSE of the KAPLA.

Finally, the tracking ability of KAPLA is tested via introducing an abrupt change of the channel in the training [8]. Simulation results in the experiment are calaulated by averaging over 200 independent Monte-Carlo runs, and the size of the training data is 1500. The KAPLA parameters are the same as those in TABLE I, but $\delta = 0.4$. At the initial stage, the channel model is described as $r(n) = -0.9x^2(n) + x(n) + v(n)$, while after 500 iterations, the channel is switched to $r(n) = 0.9x^2(n) - x(n) + v(n)$. The results are presented in Fig. 4. It is seen that the KAPLA provides the best performance compared with the other mentioned algorithms.

V. Conclusions

A kernel affine projection-like algorithm in reproducing kernel Hilbert space has been proposed and investigated for

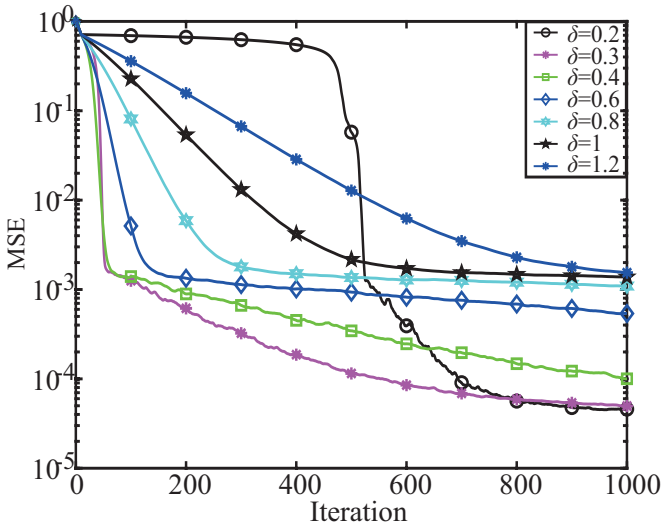

Fig. 3: MSE performance of KAPLA against δ.

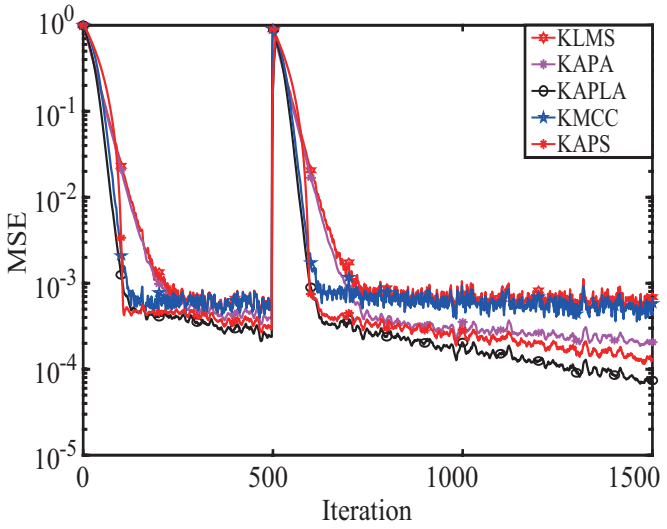

Fig. 4: Learning performance for non-linear equalization with an abrupt change of the channel at iteration 500.

nonlinear channel equalization in scenarios with non-Gaussian noises. The proposed algorithm (KAPLA) is implemented via the correntropy scheme to construct a novel affine projection-like algorithm, and then, the kernel method is incorporated into the algorithm for dealing with the non-linear channel. The results of simulations have demonstrated that KAPLA achieves the best MSE behavior compared with popular kernel algorithms.

REFERENCES

[1] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction.* Hoboken, NJ, USA: Wiley, 2010.

[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, Mar. 2008.

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

[4] 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, Nov. 2009.

[5] S. Wang, Y. Zheng, and C. Ling, "Regularized kernel least mean square algorithm with multiple-delay feedback," *IEEE Signal Process. Lett.*, vol.23, no.1, pp.98-101, Jan. 2016.

[6] J. Zhao, X. Liao, S. Wang, and C. K. Tse, "Kernel least mean square with single feedback," *IEEE Signal Process. Lett.*, vol.22, no.7, pp.953-957, Jul. 2015.

[7] B. Chen, J. Liang, N. Zheng, and J. C. Príncipe, "Kernel least mean square with adaptive kernel size," *Neurocomputing*, vol.191, pp.95-106, Feb. 2016.

[8] W. Liu, J. C. Príncipe, "Kernel affine projection algorithms," *EURASIP J. Adv. Signal Process.*, vol.2008, no.1, pp.1-13, Mar. 2008.

[9] L. Shi, Y. Lin, and X. Xie, "Combination of affine projection sign algorithms for robust adaptive filtering in non-gaussian impulsive interference," *Electron. Lett.*, vol.50, no.6, pp.466-467, Mar. 2014.

[10] K. Pelekanakis, and M. Chitre, "Adaptive sparse channel estimation under symmetric α-stable noise," *IEEE Trans. Wireless Commun.*, vol.13, no.6, pp.3183-3195, Jun. 2014.

[11] T. Shao, Y. R. Zheng, and J. Benesty, "An affine projection sign algorithm robust against impulsive interferences," *IEEE Signal Process. Lett.*, vol.17, no.4, pp.327-330, Jan. 2010.

[12] A. Singh, J. C. Príncipe, "Using correntropy as a cost function in linear adaptive filters," *In Proceedings of the 2009 International Joint Conference on Neural Networks, (IJCNN)*, Atlanta, GA, USA, pp.2950-2955, Jun. 2009.

[13] W. Shi, Y. Li, and Y. Wang, "Noise-free maximum correntropy criterion algorithm in non-Gaussian environment," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, doi: 10.1109/TCSII.2019.2914511.

[14] Y. Li, Z. Jiang, W. Shi, et al, "Blocked maximum correntropy criterion algorithm for cluster-sparse system identifications," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, doi: 10.1109/TCSII.2019.2891654.

[15] R. He, W. S. Zheng, and B. G. Hu, "Maximum correntropy criterion for robust face recognition," *IEEE Trans. Pattern Analysis. Machine Intelligence*, vol.33, no.8, pp.1561-1576, Dec. 2010.

[16] B. Chen, L. Xing, H. Zhao, N. Zheng, and J. C. Príncipe, "Generalized correntropy for robust adaptive filtering," *IEEE Trans. Signal Process.*, vol.64, no.13, pp.3376-3387, Mar. 2016.

[17] W. Ma, J. Duan, W. Man, J. Liang, and B. Chen, "General mixed-norm-based diffusion adaptive filtering algorithm for distributed estimation over network," *IEEE Access*, vol.5, pp.3376-3387, Jan. 2017.

[18] S. Yu, X. You, X. Jiang, W. Ou, Z. Zhu, and Y. Zhao, *Generalized Kernel Normalized Mixed-norm Algorithm: Analysis And Simulations.* Istanbul, Turkey: Springer, 2015.

[19] S. Zhao, B. Chen, and J. C. Príncipe, "Kernel adaptive filtering with maximum correntropy criterion," *The 2011 International Joint Conference on Neural Networks*, San Jose, CA, USA, Aug. 2011.

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

[21] A. Sayed, *Fundamentals of Adaptive Filtering.* New York, NY, USA: Wiley, 2003.

[22] D. B. Haddad, M. R. Petraglia, and A. Petraglia, "A unified approach for sparsity-aware and maximum correntropy adaptive filters," *in 24th European Signal Processing Conference (EUSIPCO)*, Budapest, Hungary, Sep. 2016, pp.170-174.

[23] Y. Li, Y. Wang, R. Yang, F. Albu, "A soft parameter function penalized normalized maximum correntropy criterion algorithm for sparse system identification," *Entropy*, vol.19, no.1, pp.1-15, 2017.

[24] V. Z. Filipovic, "Consistency of the robust recursive Hammerstein model identification algorithm," *J. Frankl. Inst.*, vol.352, no.5, pp.1932-1945, May 2015.