CONVERGENCE ANALYSIS OF THE AUGMENTED COMPLEX KLMS ALGORITHM WITH PRE-TUNED DICTIONARY

Wei Gao [†‡] *Jie Chen* [⋆] *Cédric Richard* [†] *Jose-Carlos M. Bermudez* [*] *Jianguo Huang* [‡]

[†]Université de Nice Sophia-Antipolis, France
[⋆]University of Michigan, Ann Arbor, USA
[*]Federal University of Santa Catarina, Florianópolis, Brazil
[‡]Northwestern Polytechnical University, Xi'an, China

gao_wei@mail.nwpu.edu.cn dr.jie.chen@ieee.org cedric.richard@unice.fr j.bermudez@ieee.org jghuang@nwpu.edu.cn

ABSTRACT

Complex kernel-based adaptive algorithms have been recently introduced for complex-valued nonlinear system identification. These algorithms are built upon the same framework as complex linear adaptive filtering techniques and Wirtinger's calculus in complex reproducing kernel Hilbert spaces. In this paper, we study the convergence behavior of the augmented complex Gaussian KLMS algorithm. Simulation results illustrate the accuracy of the analysis.

Index Terms— Kernel adaptive filtering, complex RKHS, complex Gaussian kernel, non-circular data

1. INTRODUCTION

Single-kernel adaptive filters have been extensively studied over the last decade, and their performance have been investigated experimentally and theoretically on a variety of real-valued nonlinear system identification problems. Typical filtering algorithms in reproducing kernel Hilbert spaces (RKHS) are the KRLS algorithm [1], the sliding-window KRLS algorithm [2], and the quantized KRLS algorithm [3]. The KNLMS algorithm was independently introduced in [4–7]. The KLMS algorithm, proposed in [8, 9], has attracted much attention in recent years because of its simplicity and robustness. An analysis of its convergence behavior with Gaussian kernel is reported in [10], and a closed-form condition for convergence is introduced in [11]. The stability of this algorithm with ℓ_1-norm regularization is studied in [12, 13].

Kernel-based adaptive filtering algorithms for complex data have recently attracted attention since they ensure phase processing. This is of importance for applications in communication, radar and sonar. A complexified kernel LMS algorithm and pure complex kernel LMS algorithm are introduced in [14]. A direct extension of the derivations in [10] is proposed in [15] to analyze the convergence behavior of complex KLMS algorithm (CKLMS). The augmented CKLMS algorithm (ACKLMS) is presented in [16, 17], and its normalized counterpart is described in [18, 19]. These works show that augmented complex-valued algorithms provide significantly improved performance compared with complex-valued algorithms. Finally, the quaternion KLMS algorithm has been recently introduced in [20] as an extension of complex-valued KLMS algorithms.

The aim of this paper is to analyze the convergence behavior of the ACKLMS algorithm. First, we introduce some definitions and a general framework for pure complex multikernel adaptive filtering algorithms. This framework relies on multikernel adaptive filters that has previously been derived for use with real-valued data in [21–24]. Then, we derive models for the convergence behavior in the mean and mean-square sense of the ACKLMS algorithm with Gaussian kernels. Finally, the accuracy of these models is checked with simulation results.

2. COMPLEX MULTI-KERNEL LMS

2.1. Preliminaries

Consider the complex input/output sequence $\{(\boldsymbol{u}(n), d(n))\}_{n=1}^{N}$ with $\boldsymbol{u}(n) \in \mathbb{U}$ and $d(n) \in \mathbb{C}$, where $\mathbb{U}$ is a compact of $\mathbb{C}^L$. The complex input vector can be expressed in the form

$$\begin{aligned}\boldsymbol{u}(n) &= \sqrt{1-\rho^2}\,\boldsymbol{u}_{\text{re}}(n) + \mathrm{i}\rho\,\boldsymbol{u}_{\text{im}}(n)\\ &= \boldsymbol{u}_I(n) + \mathrm{i}\,\boldsymbol{u}_Q(n)\end{aligned} \tag{1}$$

where the subscripts I and Q denote "in-phase" and "quadrature" components, and $\mathrm{i} = \sqrt{-1}$. The sequence $\boldsymbol{u}_{\text{re}}(n)$ (resp., $\boldsymbol{u}_{\text{im}}(n)$) is supposed to be zero-mean, independent, and identically distributed according to a real-valued Gaussian distribution. The entries of each input vector $\boldsymbol{u}_{\text{re}}(n)$ (resp., $\boldsymbol{u}_{\text{im}}(n)$) can, however, be correlated. In addition, the sequences $\boldsymbol{u}_{\text{re}}(n)$ and $\boldsymbol{u}_{\text{im}}(n)$ are assumed to be independent. This implies that $E\{\boldsymbol{u}(n-i)\boldsymbol{u}^H(n-j)\} = \boldsymbol{0}$ for $i \neq j$, where the operator $(\cdot)^H$ denotes Hermitian transpose. The circularity of input data is controlled by parameter ρ. Setting $\rho = \sqrt{2}/2$ results in a circular input, while ρ approaching to 0 or 1 leads to a highly non-circular input.

Let $\kappa_{\mathbb{C}} : \mathbb{U} \times \mathbb{U} \to \mathbb{C}$ be a complex reproducing kernel. We denote by $(\mathbb{H}, \langle\cdot,\cdot\rangle_{\mathbb{H}})$ the induced complex RKHS with its inner product. Complex reproducing kernels include the Szego kernel, the Bergman kernel, and the so-called pure complex Gaussian kernel. The latter is the extension of the Gaussian kernel for complex arguments. The pure complex Gaussian kernel is defined as follows [25]

$$\kappa_{\mathbb{C}}(\boldsymbol{u}, \boldsymbol{v}) = \exp\left(-\sum_{\ell=1}^{L} (u_\ell - v_\ell^*)^2 / 2\xi^2\right) \tag{2}$$

with u_ℓ and v_ℓ the ℓ-th entries of $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^L$. The parameter $\xi > 0$ denotes the kernel bandwidth and $(\cdot)^*$ denotes the conjugate operator. The conjugate of kernel $\kappa_{\mathbb{C}}(\boldsymbol{u}, \boldsymbol{v})$ is defined by

$$\kappa_{\mathbb{C}}^{\star}(\boldsymbol{u}, \boldsymbol{v}) = \exp\left(-\sum_{\ell=1}^{L} (v_\ell - u_\ell^*)^2 / 2\xi^2\right). \tag{3}$$

Note that $(\cdot)^\star$ is defined on kernels and should not be confounded with the complex conjugate $(\cdot)^*$. We shall focus on the above complex Gaussian kernel in the sequel.

This work was partially supported by the National Natural Science Foundation of China (61271415, 61401499).

2.2. A framework for complex multi-kernel algorithms

Let $\{\kappa_{\mathbb{C},k}\}_{k=1}^{K}$ be the family of candidate complex kernels, and $\mathbb{H}_k$ the RKHS defined by each $\kappa_{\mathbb{C},k}$. Consider the space $\mathbb{H}$ of multidimensional mappings

$$\begin{aligned}\Phi:\ \ &\mathbb{C} \to \mathbb{C}^K\\ &\boldsymbol{u} \mapsto \Phi(\boldsymbol{u}) = \text{col}\{\varphi_1(\boldsymbol{u}), \dots, \varphi_K(\boldsymbol{u})\}\end{aligned} \tag{4}$$

with $\varphi_k \in \mathbb{H}_k$ and col$\{\cdot\}$ the operator that stacks its arguments on top of each other. Let $\langle\cdot,\cdot\rangle_{\mathbb{H}}$ be the inner product in $\mathbb{H}$ defined as

$$\langle\Phi, \Phi'\rangle_{\mathbb{H}} = \textstyle\sum_{k=1}^{K} \langle\varphi_k, \varphi'_k\rangle_{\mathbb{H}_k}. \tag{5}$$

The space $\mathbb{H}$ equipped with the inner product $\langle\cdot,\cdot\rangle_{\mathbb{H}}$ is a Hilbert space as $(\mathbb{H}_k, \langle\cdot,\cdot\rangle_{\mathbb{H}_k})$ is a complex Hilbert space for all k. We can then define the vector-valued representer of evaluation $\boldsymbol{\kappa}_{\mathbb{H}}(\cdot, \boldsymbol{u})$ such that

$$\Phi(\boldsymbol{u}) = [\Phi, \boldsymbol{\kappa}_{\mathbb{H}}(\cdot, \boldsymbol{u})] \tag{6}$$

with $\boldsymbol{\kappa}_{\mathbb{H}}(\cdot, \boldsymbol{u}) = \text{col}\{\kappa_{\mathbb{C},1}(\cdot, \boldsymbol{u}), \dots, \kappa_{\mathbb{C},K}(\cdot, \boldsymbol{u})\}$ and $[\cdot,\cdot]$ the entry-wise inner product. This yields the following reproducing property

$$\boldsymbol{\kappa}_{\mathbb{H}}(\boldsymbol{u}, \boldsymbol{v}) = [\boldsymbol{\kappa}_{\mathbb{H}}(\cdot, \boldsymbol{u}), \boldsymbol{\kappa}_{\mathbb{H}}(\cdot, \boldsymbol{v})]. \tag{7}$$

Let $\Psi = \text{col}\{\psi_1, \dots, \psi_K\}$ be a vector-valued function in space $\mathbb{H}$, and let $\psi = \sum_{k=1}^{K} \psi_k$ with $\psi_k \in \mathbb{H}_k$ be the scalar-valued function that sums the entries of Ψ, namely, $\psi = \mathbf{1}_K^\top \Psi$ with $\mathbf{1}_K$ the all-one column vector of length K.

Given a valued input-output sequence $\{(d(n), \boldsymbol{u}(n))\}_{n=1}^{N}$, we aim at estimating a multidimensional function Ψ in $\mathbb{H}$ that minimizes the regularized least-square error

$$\min_{\Psi\in\mathbb{H}} J(\Psi) = \sum_{n=1}^{N} \left|d(n) - \mathbf{1}_K^\top \Psi(\boldsymbol{u}(n))\right|^2 + \lambda \|\mathbf{1}_K^\top \Psi\|_{\mathbb{H}}^2 \tag{8}$$

with $\lambda \geq 0$ a regularization constant. By virtue of the generalized multidimensional representer theorem, not presented in this paper due to lack of space, the optimum function Ψ can be written as

$$\Psi(\cdot) = \text{col}\Big\{\sum_{n=1}^{N} \alpha_{n,k}^{*}\, \kappa_{\mathbb{C},k}(\cdot, \boldsymbol{u}(n))\Big\}_{k=1}^{K}. \tag{9}$$

For simplicity, without loss of generality, we shall omit the regularization term in problem (8), which can be reformulated as

$$\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \left|d(n) - \sum_{k=1}^{K} \boldsymbol{\alpha}_k^H \boldsymbol{\kappa}_{\mathbb{C},k}(n)\right|^2 \tag{10}$$

where $\boldsymbol{\alpha} = \text{col}\{\boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_K\}$ with $\boldsymbol{\alpha}_k = (\alpha_{1,k}, \dots, \alpha_{N,k})^\top$ is the unknown weight vector, and $\boldsymbol{\kappa}_{\mathbb{C},k}(n)$ is the $N \times 1$ kernelized input vector with j-th entry $\kappa_{\mathbb{C},k}(\boldsymbol{u}(j), \boldsymbol{u}(n))$. Calculating the directional derivative of $J(\boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$ by Wirtinger's calculus yields

$$\partial_{\boldsymbol{\alpha}_k} J(\boldsymbol{\alpha}) = -2 \sum_{n=1}^{N} e^{*}(n)\, \boldsymbol{\kappa}_{\mathbb{C},k}(\cdot, \boldsymbol{u}(n)). \tag{11}$$

where $e(n) = d(n) - \sum_{k=1}^{K} \boldsymbol{\alpha}_k^H \boldsymbol{\kappa}_{\mathbb{C},k}(n)$. Approximating (11) by its instantaneous estimate $\partial_{\boldsymbol{\alpha}_k} J(\boldsymbol{\alpha}) \approx -2\, e^{*}(n)\, \kappa_{\mathbb{C},k}(\cdot, \boldsymbol{u}(n))$, we obtain the stochastic gradient descent algorithm:

$$\boldsymbol{\alpha}(n+1) = \boldsymbol{\alpha}(n) + \eta\, e^{*}(n)\, \boldsymbol{\kappa}_{\mathbb{H}}(n) = \sum_{i=1}^{n} \eta\, e^{*}(i)\, \boldsymbol{\kappa}_{\mathbb{H}}(i) \tag{12}$$

with η a positive step-size, $\boldsymbol{\kappa}_{\mathbb{H}}(n) = \text{col}\{\boldsymbol{\kappa}_{\mathbb{C},k}(n)\}_{k=1}^{K}$ the complex kernelized input vector, and $e(n) = d(n) - \boldsymbol{\alpha}^H(n)\, \boldsymbol{\kappa}_{\mathbb{H}}(n)$ the estimation error. Finally, the optimal function is of the form

$$\psi(\cdot) = \sum_{n=1}^{N} \sum_{k=1}^{K} \alpha_{n,k}^{*}\, \kappa_{\mathbb{C},k}(\cdot, \boldsymbol{u}(n)). \tag{13}$$

2.3. Augmented complex kernel LMS (ACKLMS)

In order to overcome the problem of the increasing amount n of observations in an online context, a fixed-size model is usually adopted:

$$\psi(\cdot) = \sum_{m=1}^{M} \sum_{k=1}^{K} \alpha_{m,k}^{*}\, \kappa_{\mathbb{C},k}(\cdot, \boldsymbol{u}(\omega_m)) \tag{14}$$

where $\boldsymbol{\omega} \triangleq \{\kappa_{\mathbb{H}}(\cdot, \boldsymbol{u}(\omega_m))\}_{m=1}^{M}$ is the so-called dictionary of the filter ψ, and M its length. Limiting the number of single-kernel filters to $K = 2$, and setting the two kernels to (2)-(3), the ACKLMS algorithm based on model (14) is given by (See [18] for an introduction to ACKLMS):

$$\begin{aligned}\hat{d}(n) = \sum_{m=1}^{M} \big[&\alpha_{1,m}^{*}(n)\, \kappa_{\mathbb{C}}(\boldsymbol{u}(n), \boldsymbol{u}(\omega_m))\\ &+ \alpha_{2,m}^{*}(n)\, \kappa_{\mathbb{C}}^{\star}(\boldsymbol{u}(n), \boldsymbol{u}(\omega_m))\big]\\ = \boldsymbol{\alpha}^H&(n)\, \boldsymbol{\kappa}_{\mathbb{H},\omega}(n).\end{aligned} \tag{15}$$

The ACKLMS algorithm can be viewed as a complex Gaussian bi-kernel case of the complex multi-kernel algorithm [18, 19]. It can be expected that ACKLMS algorithm outperforms the existing CKLMS algorithms due to the flexibility of complex multi-kernels.

3. ACKLMS PERFORMANCE ANALYSIS

We shall now study the transient and steady-state of the mean-square error conditionally to dictionary $\boldsymbol{\omega}$ of the complex Gaussian bi-kernel LMS algorithm, that is,

$$E\left\{|e(n)|^2 \,|\, \boldsymbol{\omega}\right\} = \int_{\mathbb{U}\times\mathbb{C}} |e(n)|^2\, d\rho(\boldsymbol{u}(n), d(n) \,|\, \boldsymbol{\omega}), \tag{16}$$

with $e(n) = d(n) - \hat{d}(n)$ and ρ a Borel probability measure. We shall use the subscript ω for quantities conditioned on dictionary $\boldsymbol{\omega}$. Given $\boldsymbol{\omega}$, the estimation error at time instant n is given by

$$e_\omega(n) = d(n) - \hat{d}_{\boldsymbol{\omega}}(n) \tag{17}$$

with $\hat{d}_{\boldsymbol{\omega}}(n) = \hat{d}(n)|\boldsymbol{\omega}$. Multiplying $e_\omega(n)$ by its conjugate and taking the expected value yields the mean-square-error (MSE)

$$\begin{aligned}J_{\text{MSE},\omega} = &\,E\{|d(n)|^2\}\\ &- 2\,\text{Re}\big(\boldsymbol{p}_{\kappa d,\omega}^H \boldsymbol{\alpha}_\omega(n)\big) + \boldsymbol{\alpha}_\omega^H(n) \boldsymbol{R}_{\kappa,\omega} \boldsymbol{\alpha}_\omega(n)\end{aligned} \tag{18}$$

with $\boldsymbol{R}_{\kappa,\omega} = E\left\{\boldsymbol{\kappa}_{\mathbb{H},\omega}(n) \boldsymbol{\kappa}_{\mathbb{H},\omega}^H(n)|\boldsymbol{\omega}\right\}$ the correlation matrix of input data, and $\boldsymbol{p}_{\kappa d,\omega} = E\left\{\boldsymbol{\kappa}_{\mathbb{H},\omega}(n) d^{*}(n)|\boldsymbol{\omega}\right\}$ the cross-correlation vector between $\boldsymbol{\kappa}_{\mathbb{H},\omega}(n)$ and $d(n)$. As $\boldsymbol{R}_{\kappa,\omega}$ is positive definite, the optimum weight vector is given by

$$\boldsymbol{\alpha}_{\text{opt},\omega} = \arg\min_{\boldsymbol{\alpha}_\omega} J_{\text{MSE},\omega}(\boldsymbol{\alpha}_\omega) = \boldsymbol{R}_{\kappa,\omega}^{-1}\, \boldsymbol{p}_{\kappa d,\omega} \tag{19}$$

and the minimum MSE is

$$J_{\text{min},\omega} = E\left\{|d(n)|^2\right\} - \boldsymbol{p}_{\kappa d,\omega}^H\, \boldsymbol{R}_{\kappa,\omega}^{-1}\, \boldsymbol{p}_{\kappa d,\omega}. \tag{20}$$

3.1. Mean weight error analysis

The weight update of the ACKLMS algorithm is given by

$$\boldsymbol{\alpha}_\omega(n+1) = \boldsymbol{\alpha}_\omega(n) + \eta\, e^*_\omega(n)\, \boldsymbol{\kappa}_{\mathbb{H},\omega}(n). \tag{21}$$

Let $\boldsymbol{v}_\omega(n)$ be the weight error vector defined as

$$\boldsymbol{v}_\omega(n) = \boldsymbol{\alpha}_\omega(n) - \boldsymbol{\alpha}_{\text{opt},\omega}. \tag{22}$$

The weight error vector update equation is then given by

$$\boldsymbol{v}_\omega(n+1) = \boldsymbol{v}_\omega(n) + \eta\, e^*_\omega(n)\, \boldsymbol{\kappa}_{\mathbb{H},\omega}(n). \tag{23}$$

The error (17) is consequently rewritten as

$$e_\omega(n) = d(n) - \boldsymbol{\kappa}^H_{\mathbb{H},\omega}(n)\boldsymbol{v}_\omega(n) - \boldsymbol{\kappa}^H_{\mathbb{H},\omega}(n)\boldsymbol{\alpha}_{\text{opt},\omega}. \tag{24}$$

Substituting (24) into (23) yields

$$\begin{aligned}\boldsymbol{v}_\omega(n+1) = \boldsymbol{v}_\omega(n) &+ \eta\big(d^*(n)\boldsymbol{\kappa}_{\mathbb{H},\omega}(n) \qquad\qquad (25)\\ &- \boldsymbol{\kappa}^H_{\mathbb{H},\omega}(n)\boldsymbol{v}_\omega(n)\boldsymbol{\kappa}_{\mathbb{H},\omega}(n) - \boldsymbol{\kappa}^H_{\mathbb{H},\omega}(n)\boldsymbol{\alpha}_{\text{opt},\omega}\boldsymbol{\kappa}_{\mathbb{H},\omega}(n)\big).\end{aligned}$$

Taking expected value of (25), using the CMIA hypothesis introduced in [26], and (19), we get the mean weight error model:

$$E\{\boldsymbol{v}_\omega(n+1)\} = (\boldsymbol{I} - \eta\, \boldsymbol{R}_{\kappa,\omega})E\{\boldsymbol{v}_\omega(n)\}. \tag{26}$$

The (i,j)-th entry of matrix $\boldsymbol{R}_{\kappa,\omega}$ is given by

$$[\boldsymbol{R}_{\kappa,\omega}]_{i,j} = E\{\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_i))\,[\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_j))]^*\} \tag{27}$$

with the complex Gaussian bi-kernel $\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_m))$ given by

$$\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_m)) = \begin{cases}\kappa_{\mathbb{C}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_m)), & 1 \le m \le M\\ \kappa^\star_{\mathbb{C}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_m)), & M+1 \le m \le 2M\end{cases}$$

Let us define a new vector that separates the real and imaginary parts of $\boldsymbol{u}(n)$ such that $\tilde{\boldsymbol{u}}(n) = \text{col}\{\boldsymbol{u}_I(n), \boldsymbol{u}_Q(n)\} \in \mathbb{R}^{2L}$. With the Gaussian kernels (2)-(3), the expected value of (27) can be obtained by making use of the moment generating function in [26]. We get (28) where δ_m is the indicator function

$$\delta_m = \begin{cases}1, & 1 \le m \le M\\ -1, & M+1 \le m \le 2M\end{cases} \tag{29}$$

and $\boldsymbol{R}_{\tilde{u}} = E\{\tilde{\boldsymbol{u}}(n)\,\tilde{\boldsymbol{u}}^\top(n)\}$. The definition of $\boldsymbol{H}(i,j)$ in (28) depends on i and j as follows:

$$\boldsymbol{H}(i,j) = \begin{pmatrix}\boldsymbol{I} & \boldsymbol{O}\\ \boldsymbol{O} & -\boldsymbol{I}\end{pmatrix}, \; 1 \le i,j \le M \text{ and } M+1 \le i,j \le 2M$$

$$\boldsymbol{H}(i,j) = \begin{pmatrix}\boldsymbol{I} & 1\mathrm{i}\,\boldsymbol{I}\\ 1\mathrm{i}\,\boldsymbol{I} & -\boldsymbol{I}\end{pmatrix}, \; 1 \le i \le M \text{ and } M+1 \le j \le 2M$$

$$\boldsymbol{H}(i,j) = \begin{pmatrix}\boldsymbol{I} & -1\mathrm{i}\,\boldsymbol{I}\\ -1\mathrm{i}\,\boldsymbol{I} & -\boldsymbol{I}\end{pmatrix}, \; 1 \le j \le M \text{ and } M+1 \le i \le 2M$$

Vector $\boldsymbol{b}$ in (28) is given by

$$\boldsymbol{b} = \begin{pmatrix} -\displaystyle\sum_{s=\{i,j\}} \boldsymbol{u}_I(\omega_s) + 1\mathrm{i}[\delta_i\, \boldsymbol{u}_Q(\omega_i) - \delta_j\, \boldsymbol{u}_Q(\omega_j)]\\ -\displaystyle\sum_{s=\{i,j\}} \boldsymbol{u}_Q(\omega_s) + 1\mathrm{i}[-\delta_i\, \boldsymbol{u}_I(\omega_i) + \delta_j\, \boldsymbol{u}_I(\omega_j)]\end{pmatrix}. \tag{30}$$

Equation (26) leads to the following theorem (without proof due to lack of space):

Theorem 3.1 *(Stability in the mean) Assume CMIA introduced in [26] holds. Then, for any initial condition, given a dictionary $\boldsymbol{\omega}$, the Gaussian ACKLMS algorithm* (21) *asymptotically converges in mean if the step size is chosen to satisfy*

$$0 < \eta < 2/\lambda_{\max}(\boldsymbol{R}_{\kappa,\omega}) \tag{31}$$

where $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue of its matrix argument. The entries of $\boldsymbol{R}_{\kappa,\omega}$ are given by (28).

3.2. Mean-square error analysis

Using (24) and CMIA, MSE is related to the second-order moment of the weight vector by [10]

$$J_{\text{MSE},\omega}(n) = J_{\min,\omega} + \text{trace}\,\{\boldsymbol{R}_{\kappa,\omega}\boldsymbol{C}_{v,\omega}(n)\} \tag{32}$$

where $\boldsymbol{C}_{v,\omega}(n) = E\left\{\boldsymbol{v}_\omega(n)\boldsymbol{v}^H_\omega(n)\right\}$ is the autocorrelation matrix of the weight error vector $\boldsymbol{v}_\omega(n)$, and $J_{\min,\omega}$ is the minimum MSE given by (20). The analysis of the MSE behavior (32) requires a recursive model for $\boldsymbol{C}_{v,\omega}(n)$. Post-multiplying (25) by its Hermitian conjugate, taking the expected value, and using CMIA, we get the following recursion for sufficiently small step sizes

$$\begin{aligned}\boldsymbol{C}_{v,\omega}(n+1) \approx\; &\boldsymbol{C}_{v,\omega}(n)\\ &- \eta\,[\boldsymbol{R}_{\kappa,\omega}\boldsymbol{C}_{v,\omega}(n) + \boldsymbol{C}_{v,\omega}(n)\boldsymbol{R}_{\kappa,\omega}]\\ &+ \eta^2\,\boldsymbol{T}_\omega(n) + \eta^2\,\boldsymbol{R}_{\kappa,\omega}J_{\min,\omega}\end{aligned} \tag{33}$$

with

$$\begin{aligned}&\boldsymbol{T}_\omega(n)\\ &= E\left\{\boldsymbol{\kappa}_{\mathbb{H},\omega}(n)\boldsymbol{\kappa}^H_{\mathbb{H},\omega}(n)\boldsymbol{v}_\omega(n)\boldsymbol{v}^H_\omega(n)\boldsymbol{\kappa}_{\mathbb{H},\omega}(n)\boldsymbol{\kappa}^H_{\mathbb{H},\omega}(n)\right\}.\end{aligned} \tag{34}$$

Evaluating (34) is a significant step in the analysis since $\boldsymbol{\kappa}_{\mathbb{H},\omega}(n)$ is a nonlinear transformation of a quadratic form of $\boldsymbol{u}(n)$. Using CMIA to determine the (i,j)-th element of $\boldsymbol{T}_\omega(n)$ in (34) yields

$$\begin{aligned}[\boldsymbol{T}_\omega(n)]_{i,j} \approx \sum_{\ell=1}^{M}\sum_{p=1}^{M} &E\{\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_i))\,[\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_j))]^*\\ &\times \kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_\ell))\,[\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_p))]^*\}\cdot[\boldsymbol{C}_{v,\omega}(n)]_{\ell,p}.\end{aligned} \tag{35}$$

This expression can be written as

$$[\boldsymbol{T}_\omega(n)]_{i,j} \approx \text{trace}\,\{\boldsymbol{K}_\omega(i,j)\,\boldsymbol{C}_{v,\omega}(n)\} \tag{36}$$

where the (ℓ,p)-th entry of the matrix $\boldsymbol{K}_\omega(i,j)$ is given by

$$\begin{aligned}[\boldsymbol{K}_\omega(i,j)]_{\ell,p} = E\{&\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_i))\,[\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_j))]^*\\ &\times \kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_\ell))\,[\kappa_{\mathbb{H}}(\boldsymbol{u}(n),\boldsymbol{u}(\omega_p))]^*\}.\end{aligned} \tag{37}$$

Similarly, we also rewrite (37) in terms of vector $\tilde{\boldsymbol{u}}(n)$ and use the moment generating function [26]. This leads to (38)-(39). The definition of $\boldsymbol{L}(i,j)$ in (38) depends on i and j as follows:

$$\boldsymbol{L}(i,j) = \begin{pmatrix}2\boldsymbol{I} & \boldsymbol{O}\\ \boldsymbol{O} & -2\boldsymbol{I}\end{pmatrix} \begin{cases}1 \le i,j,\ell,p \le M\\ 1 \le i,j \le M; M+1 \le \ell,p \le 2M\\ 1 \le \ell,p \le M; M+1 \le i,j \le 2M\\ 1 \le i,\ell \le M; M+1 \le j,p \le 2M\\ 1 \le j,p \le M; M+1 \le i,\ell \le 2M\\ M+1 \le i,j,\ell,p \le 2M\end{cases}$$

$$\boldsymbol{L}(i,j) = \begin{pmatrix}2\boldsymbol{I} & 1\mathrm{i}\,\boldsymbol{I}\\ 1\mathrm{i}\,\boldsymbol{I} & -2\boldsymbol{I}\end{pmatrix} \begin{cases}1 \le j \le M; M+1 \le i,\ell,p \le 2M\\ 1 \le \ell \le M; M+1 \le i,j,p \le 2M\\ 1 \le j,\ell,p \le M; M+1 \le i \le 2M\\ 1 \le i,j,\ell \le M; M+1 \le p \le 2M\end{cases}$$

$$
\begin{aligned}
[\boldsymbol{R}_{\kappa,\omega}]_{i,j} =& \left|\boldsymbol{I} + \frac{2}{\xi^2}\boldsymbol{H}(i,j)\boldsymbol{R}_{\tilde{u}}\right|^{-\frac{1}{2}} \cdot \exp\left(-\frac{1}{2\xi^2}[\textstyle\sum_{s=\{i,j\}}\|\boldsymbol{u}_I(\omega_s)\|^2 - \sum_{s=\{i,j\}}\|\boldsymbol{u}_Q(\omega_s)\|^2]\right) \\
&\times \exp\left(\frac{1\mathrm{i}}{\xi^2}[\delta_i\,\boldsymbol{u}_I^\top(\omega_i)\boldsymbol{u}_Q(\omega_i) - \delta_j\,\boldsymbol{u}_I^\top(\omega_j)\boldsymbol{u}_Q(\omega_j)]\right) \cdot \exp\left(\frac{1}{2\xi^4}\,\boldsymbol{b}^\top\boldsymbol{R}_{\tilde{u}}(\boldsymbol{I} + \frac{2}{\xi^2}\boldsymbol{H}(i,j)\boldsymbol{R}_{\tilde{u}})^{-1}\,\boldsymbol{b}\right)
\end{aligned}
\tag{28}
$$

$$
\begin{aligned}
[\boldsymbol{K}_\omega(i,j)]_{\ell,p} =& \left|\boldsymbol{I} + \frac{2}{\xi^2}\boldsymbol{L}(i,j)\boldsymbol{R}_{\tilde{u}}\right|^{-\frac{1}{2}} \exp\left(\frac{1\mathrm{i}}{\xi^2}[\delta_i\,\boldsymbol{u}_I^\top(\omega_i)\boldsymbol{u}_Q(\omega_i) - \delta_j\,\boldsymbol{u}_I^\top(\omega_j)\boldsymbol{u}_Q(\omega_j) + \delta_\ell\,\boldsymbol{u}_I^\top(\omega_\ell)\boldsymbol{u}_Q(\omega_\ell) - \delta_p\,\boldsymbol{u}_I^\top(\omega_p)\boldsymbol{u}_Q(\omega_p)]\right) \\
&\times \exp\left(-\frac{1}{2\xi^2}(\textstyle\sum_{s=\{i,j,\ell,p\}}\|\boldsymbol{u}_I(\omega_s)\|^2 - \sum_{s=\{i,j,\ell,p\}}\|\boldsymbol{u}_Q(\omega_s)\|^2)\right) \cdot \exp\left(\frac{1}{2\xi^4}\,\boldsymbol{f}^\top\boldsymbol{R}_{\tilde{u}}(\boldsymbol{I} + \frac{2}{\xi^2}\boldsymbol{L}(i,j)\boldsymbol{R}_{\tilde{u}})^{-1}\,\boldsymbol{f}\right)
\end{aligned}
\tag{38}
$$

$$
\boldsymbol{f} = \begin{pmatrix} -\sum_{s=\{i,j,\ell,p\}}\boldsymbol{u}_I(\omega_s) + 1\mathrm{i}\,[\delta_i\,\boldsymbol{u}_Q(\omega_i) - \delta_j\,\boldsymbol{u}_Q(\omega_j) + \delta_\ell\,\boldsymbol{u}_Q(\omega_\ell) - \delta_p\,\boldsymbol{u}_Q(\omega_p)] \\ -\sum_{s=\{i,j,\ell,p\}}\boldsymbol{u}_Q(\omega_s) + 1\mathrm{i}\,[-\delta_i\,\boldsymbol{u}_I(\omega_i) + \delta_j\,\boldsymbol{u}_I(\omega_j) - \delta_\ell\,\boldsymbol{u}_I(\omega_\ell) + \delta_p\,\boldsymbol{u}_I(\omega_p)] \end{pmatrix}
\tag{39}
$$

$$
\boldsymbol{L}(i,j) = \begin{pmatrix} 2\boldsymbol{I} & -1\mathrm{i}\,\boldsymbol{I} \\ -1\mathrm{i}\,\boldsymbol{I} & -2\boldsymbol{I} \end{pmatrix} \begin{cases} 1 \le i \le M; M+1 \le j,\ell,p \le 2M \\ 1 \le p \le M; M+1 \le i,j,\ell \le 2M \\ 1 \le i,\ell,p \le M; M+1 \le j \le 2M \\ 1 \le i,j,p \le M; M+1 \le \ell \le 2M \end{cases}
$$

$$
\boldsymbol{L}(i,j) = \begin{pmatrix} 2\boldsymbol{I} & 2\mathrm{i}\,\boldsymbol{I} \\ 2\mathrm{i}\,\boldsymbol{I} & -2\boldsymbol{I} \end{pmatrix} \; 1 \le j,\ell \le M; M+1 \le i,p \le 2M
$$

$$
\boldsymbol{L}(i,j) = \begin{pmatrix} 2\boldsymbol{I} & -2\mathrm{i}\,\boldsymbol{I} \\ -2\mathrm{i}\,\boldsymbol{I} & -2\boldsymbol{I} \end{pmatrix} \; 1 \le i,p \le M; M+1 \le j,\ell \le 2M
$$

3.3. Steady-State behavior

In order to determine the steady-state of recursion (33), we rewrite it in a lexicographic form. Let vec $\{\cdot\}$ denote the operator that stacks the columns of a matrix on top of each other. Vectorizing $\boldsymbol{C}_{v,\omega}(n)$ and $\boldsymbol{R}_{\kappa,\omega}$ by $\boldsymbol{c}_{v,\omega}(n) = \text{vec}\,\{\boldsymbol{C}_{v,\omega}(n)\}$ and $\boldsymbol{r}_{\kappa,\omega} = \text{vec}\,\{\boldsymbol{R}_{\kappa,\omega}\}$, we can rewrite (33) as follows

$$
\boldsymbol{c}_{v,\omega}(n) = \boldsymbol{G}_\omega\,\boldsymbol{c}_{v,\omega}(n) + \eta^2\,J_{\min,\omega}\,\boldsymbol{r}_{\kappa,\omega} \tag{41}
$$

with $\boldsymbol{G}_\omega = \boldsymbol{I} - \eta(\boldsymbol{G}_{\omega,1} + \boldsymbol{G}_{\omega,2}) + \eta^2\boldsymbol{G}_{\omega,3}$. Matrix $\boldsymbol{G}_\omega$ is found by the use of the following definitions:

- $\boldsymbol{I}$ is the identity matrix of dimension $4M^2 \times 4M^2$;
- $\boldsymbol{G}_{\omega,1} = \boldsymbol{I} \otimes \boldsymbol{R}_{\kappa,\omega}$, where $\otimes$ denotes the Kronecker product;
- $\boldsymbol{G}_{\omega,2} = \boldsymbol{R}_{\kappa,\omega} \otimes \boldsymbol{I}$;
- $\boldsymbol{G}_{\omega,3}$ is given by $[\boldsymbol{G}_{\omega,3}]_{i+2(j-1)M,\ell+2(p-1)M} = [\boldsymbol{K}_\omega(i,j)]_{\ell,p}$ with $1 \le i,j,\ell,p \le 2M$.

Assuming convergence, the closed-formed solution of the recursion (41) in steady-state is given by

$$
\boldsymbol{c}_{v,\omega}(\infty) = \eta^2\,J_{\min,\omega}(\boldsymbol{I} - \boldsymbol{G}_\omega)^{-1}\boldsymbol{r}_{\kappa,\omega}. \tag{42}
$$

From equation (32), the steady-state MSE is finally given by

$$
J_{\text{MSE},\omega}(\infty) = J_{\min,\omega} + \text{trace}\,\{\boldsymbol{R}_{\kappa,\omega}\boldsymbol{C}_{v,\omega}(\infty)\} \tag{43}
$$

where the second term on the right side is the steady-state EMSE.

4. EXPERIMENT

This section provides an example of nonlinear system identification to check the accuracy of the convergence models. We considered the complex valued input sequence

$$
u(n) = \rho_0\,u(n-1) + \sigma_u\,\sqrt{1-\rho_0^2}\,w(n) \tag{44}
$$

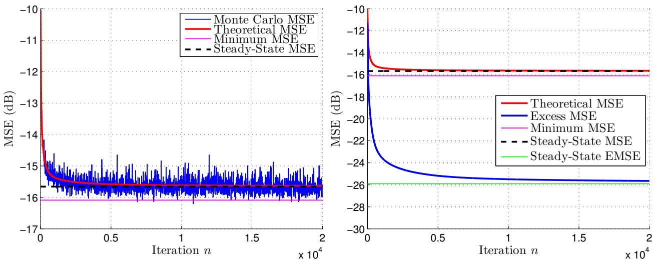
with $w(n) = \sqrt{1-\rho^2}\,w_{\text{re}}(n) + \mathrm{i}\,\rho\,w_{\text{im}}(n)$. Parameter ρ was set to 0.1 corresponding to highly non-circular, and the random variables $w_{\text{re}}(n)$ and $w_{\text{im}}(n)$ were distributed according zero-mean i.i.d. Gaussian distributions with standard deviation $\sigma_w = 1$. Both parameters ρ_0 and σ_u were set to 0.5. The system to be identified was

$$
\begin{cases} y(n) = (0.5 - 0.1\mathrm{i})\,u(n) - (0.3 - 0.2\mathrm{i})\,u(n-1) \\ d(n) = y(n) + (1.25 - 1\mathrm{i})\,y^2(n) + (0.35 - 0.2\mathrm{i})\,y^3(n) + z(n) \end{cases}
$$

where $z(n)$ is a complex additive zero-mean Gaussian noise with standard deviation $\sigma_z = 0.1$. At each time n, ACKLMS algorithm was updated with input vector $\boldsymbol{u}(n) = [u(n), u(n-1)]^\top$ and the reference signal $d(n)$. The correlation matrix $\boldsymbol{R}_{\tilde{u}}$ is thus given by

$$
\boldsymbol{R}_{\tilde{u}} = \sigma_u^2 \begin{pmatrix} (1-\rho^2) & (1-\rho^2)\rho_0 & 0 & 0 \\ (1-\rho^2)\rho_0 & (1-\rho^2) & 0 & 0 \\ 0 & 0 & \rho^2 & \rho^2\rho_0 \\ 0 & 0 & \rho^2\rho_0 & \rho^2 \end{pmatrix}. \tag{45}
$$

The pure complex Gaussian bandwidth ξ and the step-size η were set to 0.55 and 0.1, respectively. We used the coherence sparsification criterion proposed in [5] with threshold $\mu_0 = 0.3$ to construct a fixed dictionary of length $M = 12$. All simulation curves were obtained by averaging over 200 Monte Carlo runs. It is shown in Figure 1 that the theoretical curves consistently agree with the Monte Carlo simulations in both transient and steady-state.



(a) Theory vs. Monte Carlo.

(b) Steady-state results.

Fig. 1. Simulation results of ACKLMS algorithm.

5. CONCLUSION

In this paper, we presented the ACKLMS algorithm based on the framework of complex multi-kernel. Then we derived a theoretical model of convergence for ACKLMS with pre-tuned dictionary. In future works, we will study how using this model to design dictionaries, and set the step-size and the kernel bandwidth, that allow to reach specified MSE or convergence speed.

6. REFERENCES

[1] Y. Engel, S. Mannor, and R. Meir, "Kernel recursive least squares," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, 2004.

[2] S. Van Vaerenbergh, J. Via, and I. Santamaria, "A sliding-window kernel RLS algorithm and its application to nonlinear channel identification," in *Proc. IEEE ICASSP*, Toulouse, France, 2006, pp. 789–792.

[3] B. Chen, S. Zhao, P. Zhu, and J. C. Principe, "Quantized kernel recursive least squares algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 9, pp. 1484–1491, 2013.

[4] P. Honeine, C. Richard, and J.-C. M. Bermudez, "On-line nonlinear sparse approximation of functions," in *Proc. IEEE ISIT'07*, Nice, France, 2007, pp. 956–960.

[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, 2009.

[6] S. Slavakis and S. Theodoridis, "Sliding window generalized kernel affine projection algorithm using projection mappings," *EURASIP J. Adv. Signal Process.*, vol. 2008:735351, 2008.

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

[8] C. Richard, "Filtrage adaptatif non-linéaire par méthodes de gradient stochastique court-terme à noyau," in *Proc. GRETSI*, Louvain-la-Neuve, Belgium, 2005, pp. 1–4.

[9] W. Liu and J. C. Principe, "Kernel affine projection algorithms," *EURASIP J. Adv. Signal Process.*, vol. 2008:784292, 2008.

[10] W. D. Parreira, J.-C. M. Bermudez, C. Richard, and J.-Y. Tourneret, "Stochastic behavior analysis of the Gaussian kernel-least-mean-square algorithm," *IEEE Trans. Signal Process.*, vol. 60, no. 5, pp. 2208–2222, 2012.

[11] C. Richard and J.-C. M. Bermudez, "Closed-form conditions for convergence of the gaussian kernel-least-mean-square algorithm," in *Proc. Asilomar*, Pacific Grove, CA, USA, 2012.

[12] W. Gao, J. Chen, C. Richard, J. Huang, and R. Flamary, "Kernel LMS algorithm with forward-backward splitting for dictionary learning," in *Proc. IEEE ICASSP*, Vancouver, Canada, 2013, pp. 5735–5739.

[13] W. Gao, J. Chen, C. Richard, and J. Huang, "Online dictionary learning for kernel LMS," *IEEE Trans. Signal Process.*, vol. 62, no. 11, pp. 2765–2777, 2014.

[14] P. Bouboulis and S. Theodoridis, "Extension of Wirtinger's calculus in reproducing kernel Hilbert spaces and the complex kernel LMS," *IEEE Trans. Signal Process.*, vol. 59, no. 3, pp. 964–978, 2011.

[15] T. K. Paul and T. Ogunfunmi, "Study of the convergence behavior of the complex kernel least mean square algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 9, pp. 1349–1363, 2013.

[16] D. P. Mandic, S. Javidi, S. L. Goh, A. Kuh, and K. Aihara, "Complex-valued prediction of wind profile using augmented complex statistics," *Renewable Energy*, vol. 34, no. 1, pp. 196–201, 2009.

[17] S. Y. Kung, "Kernel approaches to unsupervised and supervised machine learning," in *Advances in Multimedia Information Processing-PCM 2009*, pp. 1–32. Springer, 2009.

[18] P. Bouboulis, S. Theodoridis, and M. Mavroforakis, "The augmented complex kernel LMS," *IEEE Trans. Signal Process.*, vol. 60, no. 9, pp. 4962–4967, 2012.

[19] F. A. Tobar, A. Kuh, and D. P. Mandic, "A novel augmented complex valued kernel LMS," in *Proc. IEEE SAM*, 2012, pp. 473–476.

[20] F. A. Tobar and D. P. Mandic, "The quaternion kernel least squares," in *Proc. IEEE ICASSP*, 2013, pp. 6128–6132.

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

[22] F. A. Tobar, S.-Y Kung, and D. P. Mandic, "Multikernel least mean square algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 265–277, 2014.

[23] W. Gao, C. Richard, J.-C.M. Bermudez, and J. Huang, "Convex combinations of kernel adaptive filters," in *Proc. IEEE MLSP*, Reims, France, 2014, pp. 1–5.

[24] R. Pokharel, S. Seth, and J. C. Principe, "Mixture kernel least mean square," in *Proc. IJCNN*, Dallas, USA, 2013, pp. 1–7.

[25] I. Steinwart, D. Hush, and C. Scovel, "An explicit description of the reproducing kernel hilbert spaces of gaussian RBF kernels," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4635–4643, 2006.

[26] J. Chen, W. Gao, C. Richard, and J.-C. M. Bermudez, "Convergence analysis of kernel LMS algorithm with pre-tuned dictionary," in *Proc. IEEE ICASSP*, Florence, Italy, 2014, pp. 7243–7247.