Adaptive Random Fourier Features Kernel LMS

Wei Gao, *Member, IEEE*, Jie Chen, *Senior Member, IEEE*, Cédric Richard *Senior Member, IEEE*, Wentao Shi, *Member, IEEE*, and Qunfei Zhang, *Member, IEEE*

Abstract—We propose the adaptive random Fourier features Gaussian kernel LMS (ARFF-GKLMS). Like most kernel adaptive filters based on stochastic gradient descent, this algorithm uses a preset number of random Fourier features to save computation cost. However, as an extra flexibility, it can adapt the inherent kernel bandwidth in the random Fourier features in an online manner. This adaptation mechanism allows to alleviate the problem of selecting the kernel bandwidth beforehand for the benefit of an improved tracking in non-stationary circumstances. Simulation results confirm that the proposed algorithm achieves a performance improvement in terms of convergence rate, error at steady-state and tracking ability over other kernel adaptive filters with preset kernel bandwidth.

Index Terms—Kernel LMS, random Fourier features, Gaussian kernel, stochastic gradient descent.

I. INTRODUCTION

THE kernel least-mean-square (KLMS) algorithm was first introduced in [1] by reformulating the LMS algorithm in reproducing kernel Hilbert spaces (RKHS). Since the KLMS can be easily implemented and has good tracking performance, it has become central in the family of kernel adaptive filters. The Gaussian kernel is commonly used with kernel adaptive filters because it has universal modeling capability, desirable smoothness, and numerical stability [2], [3]. In particular, the Gaussian kernel LMS (GKLMS) has attracted substantial research interests as well as its variants [4]–[9], and its theoretical performance has been extensively analyzed [10]–[14]. However, the selection of an appropriate bandwidth for the Gaussian kernel to ensure good performance still remains a problem with GKLMS-type algorithms in practical use, especially for non-stationary environments.

Multi-kernel LMS (MKLMS) algorithms use a collection of kernels with predefined bandwidths. They were developed to alleviate the issue of kernel bandwidth selection, unfortunately at the cost of an extra computational overhead [15], [16]. The GKLMS algorithm with adaptive kernel bandwidth, in parametric vector-valued form, and in non-parametric functional form, was introduced independently in [17], [18], but without considering any non-negative constraint for the kernel bandwidth. On the other hand, the random Fourier features

GKLMS (RFF-GKLMS) algorithm was proposed to make the GKLMS algorithms computationally more efficient with low performance penalty [19]. Random Fourier features (RFF) were also considered in [20] for distributed learning over networks and graphs with kernel adaptive filters in order to address nonlinear regression and classification tasks. The RFF principle was used with the kernel conjugate gradient algorithm [21]. The Cauchy-loss conjugate gradient method based on multiple RFF was proposed in [22] to improve robustness and reduce computational cost in the presence of non-Gaussian noises. Recently, several RFF kernel regression algorithms over graphs were proposed in [23], and their conditions for convergence in the mean and mean-square sense were also studied.

To the best of our knowledge, no RFF-based algorithm has been proposed yet to adapt random Fourier features. In this letter, we overcome this lack by devising the adaptive random Fourier features GKLMS (ARFF-GKLMS) algorithm. Based on stochastic gradient descent, it updates the vectors and phase factors of the RFF in an online manner. The ARFF-GKLMS algorithm outperforms the RFF-GKLMS and the GKLMS in terms of convergence rate, steady-state error and tracking ability. More importantly, the proposed simple but effective principle of adaptive RFF can be readily incorporated into all existing RFF filtering algorithms to enhance their performance.

II. GAUSSIAN KERNEL-BASED METHODS WITH RFF

Consider an unknown system with input-output relation characterized by the following nonlinear model:

$$y_n = f^\star(\boldsymbol{x}_n) + z_n \qquad (1)$$

where $f^\star(\cdot)$ is an unknown function to be identified in a given RKHS $\mathcal{H}$ endowed with a kernel $\kappa(\cdot, \cdot)$, and $\boldsymbol{x}_n \in \mathbb{R}^L$ is the original input data. The nonlinear desired output $y_n \in \mathbb{R}$ is corrupted by a zero-mean white Gaussian observation noise z_n. Given input and noisy output data pairs $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, we consider the following functional optimization problem:

$$\min_{f \in \mathcal{H}} \sum_{n=1}^N \ell\big(f(\boldsymbol{x}_n), y_n\big) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \qquad (2)$$

where $\ell(\cdot, \cdot)$ denotes any convex loss function, $\lambda > 0$ is the regularization parameter, and $\|\cdot\|_{\mathcal{H}}$ is the RKHS norm induced by its inner product. The Representer Theorem ensures that the solution $f(\cdot)$ can be represented by a kernel expansion in terms of training data [24]:

$$f(\cdot) = \sum_{n=1}^N w_n \kappa(\cdot, \boldsymbol{x}_n) = \boldsymbol{w}_N^\top \boldsymbol{\kappa}_N(\cdot) \qquad (3)$$

with $\boldsymbol{w}_N = [w_1, \ldots, w_N]^\top$ the coefficient vector to determine and $\boldsymbol{\kappa}_N(\cdot) = \left[\kappa(\cdot, \boldsymbol{x}_1), \ldots, \kappa(\cdot, \boldsymbol{x}_N)\right]^\top$ the kernelized input.

Instead of using the kernel trick [25], [26], which implicitly maps the data into a feature space, the input data can be explicitly mapped to a finite low-dimensional Euclidean space by a random Fourier feature nonlinear map, $\boldsymbol{z} : \mathbb{R}^L \to \mathbb{R}^D$. Hence, the kernel evaluation step can be approximated as follows [27]:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}') \rangle \approx \boldsymbol{z}(\boldsymbol{x})^\top \boldsymbol{z}(\boldsymbol{x}'). \tag{4}$$

A continuous and shift-invariant kernel $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{\delta})$ with $\boldsymbol{\delta} = \boldsymbol{x} - \boldsymbol{x}'$ defined on $\mathbb{R}^L$ is positive definite if, and only if, $\kappa(\boldsymbol{\delta})$ is the Fourier transform of a non-negative measure [28]. When the kernel $\kappa(\boldsymbol{\delta})$ is properly scaled, Bochner's theorem guarantees that its Fourier transform:

$$p(\boldsymbol{\omega}) = \frac{1}{(2\pi)^L} \int_{\mathbb{R}^L} \kappa(\boldsymbol{\delta}) \exp\left(-j\boldsymbol{\omega}^\top \boldsymbol{\delta}\right) d\boldsymbol{\delta} \tag{5}$$

is a proper probability distribution [27], where $j = \sqrt{-1}$. Defining $\zeta_{\boldsymbol{\omega}}(\boldsymbol{x}) = \exp(j\boldsymbol{\omega}^\top \boldsymbol{x})$, we obtain:

$$\begin{aligned} \kappa(\boldsymbol{x} - \boldsymbol{x}') &= \int_{\mathbb{R}^L} p(\boldsymbol{\omega}) \exp\left(j\boldsymbol{\omega}^\top (\boldsymbol{x} - \boldsymbol{x}')\right) d\boldsymbol{\omega} \\ &= \mathbb{E}_{\boldsymbol{\omega}}\left[\zeta_{\boldsymbol{\omega}}(\boldsymbol{x})^\mathrm{H} \zeta_{\boldsymbol{\omega}}(\boldsymbol{x}')\right] \end{aligned} \tag{6}$$

where $(\cdot)^\mathrm{H}$ denotes the Hermitian transpose operator. When $\boldsymbol{\omega}$ is drawn from $p(\boldsymbol{\omega})$, $\zeta_{\boldsymbol{\omega}}(\boldsymbol{x})^\mathrm{H} \zeta_{\boldsymbol{\omega}}(\boldsymbol{x}')$ then provides an unbiased estimate of $\kappa(\boldsymbol{x}, \boldsymbol{x}')$. Because $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ is real-valued, replacing $\exp\left(j\boldsymbol{\omega}^\top (\boldsymbol{x} - \boldsymbol{x}')\right)$ by its real part $\cos\left(\boldsymbol{\omega}^\top (\boldsymbol{x} - \boldsymbol{x}')\right)$ leads to a real-valued random feature for kernel κ. By defining mapping $z_{\boldsymbol{\omega},b}(\boldsymbol{x}) = \sqrt{2}\cos(\boldsymbol{\omega}^\top \boldsymbol{x} + b)$, then the real-valued kernel function can be expressed as [27]:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{\omega},b}\left[z_{\boldsymbol{\omega},b}(\boldsymbol{x})^\top z_{\boldsymbol{\omega},b}(\boldsymbol{x}')\right] \tag{7}$$

where b is drawn from the uniform distribution on $[0, 2\pi]$. Using (7) with the Gaussian kernel mentioned before, the latter can be approximated by D random Fourier features and random phase factors:

$$\begin{aligned} \kappa(\boldsymbol{x}, \boldsymbol{x}') &= \exp\left(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2 / 2\xi^2\right) \\ &\approx \frac{1}{D} \sum_{m=1}^{D} z_{\boldsymbol{\omega}_m, b_m}(\boldsymbol{x}) z_{\boldsymbol{\omega}_m, b_m}(\boldsymbol{x}') \end{aligned} \tag{8}$$

with $\xi > 0$ the kernel bandwidth. Each $\boldsymbol{\omega}_m$ is obtained by sampling $p(\boldsymbol{\omega}) = \left(\xi/\sqrt{2\pi}\right)^D \exp\left(-\xi^2\|\boldsymbol{\omega}\|^2/2\right)$ beforehand, that is, $\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{0}_D, \xi^{-2}\boldsymbol{I}_D)$. On the other hand, each b_m is obtained by sampling $\mathcal{U}([0, 2\pi])$ [19], [20]. A high order D can improve the approximation of (8). However, a trade-off between accuracy and complexity has to be reached. Assume that the feature map $\boldsymbol{z}_\Omega : \mathbb{R}^L \to \mathbb{R}^D$ is defined as:

$$\boldsymbol{z}_\Omega(\boldsymbol{x}) = \sqrt{2/D}\left[\cos(\boldsymbol{\omega}_1^\top \boldsymbol{x} + b_1), \ldots, \cos(\boldsymbol{\omega}_D^\top \boldsymbol{x} + b_D)\right]^\top. \tag{9}$$

Substituting (9) into (8), the kernelized input vector in (3) can be approximated by:

$$\boldsymbol{\kappa}_N(\cdot) \approx \left[\boldsymbol{z}_\Omega(\boldsymbol{x}_1)^\top \boldsymbol{z}_\Omega(\cdot), \ldots, \boldsymbol{z}_\Omega(\boldsymbol{x}_N)^\top \boldsymbol{z}_\Omega(\cdot)\right]. \tag{10}$$

By using approximation (10) and a sufficiently large order D,

function (3) can be reformulated as:

$$f(\cdot) \approx \boldsymbol{w}_N^\top \left[\boldsymbol{z}_\Omega(\boldsymbol{x}_1), \ldots, \boldsymbol{z}_\Omega(\boldsymbol{x}_N)\right]^\top \boldsymbol{z}_\Omega(\cdot) \tag{11}$$

We shall rewrite (3) as follows:

$$f(\cdot) = \boldsymbol{\alpha}^\top \boldsymbol{z}_\Omega(\cdot) \tag{12}$$

with the modified $(D \times 1)$-dimensional weight vector $\boldsymbol{\alpha}$ and the RFF nonlinear transformation $\boldsymbol{z}_\Omega(\cdot)$:

$$\boldsymbol{\alpha} = \left[\boldsymbol{z}_\Omega(\boldsymbol{x}_1), \ldots, \boldsymbol{z}_\Omega(\boldsymbol{x}_N)\right] \boldsymbol{w}_N \tag{13}$$

Based on model (12), we can now derive a linear adaptive filtering strategy based on the LMS for updating $\boldsymbol{\alpha}$ based on the $(D \times 1)$-dimensional RFF representation $\boldsymbol{z}_\Omega(\cdot)$ of data.

III. ADAPTIVE RANDOM FOURIER FEATURES GKLMS

In this section, we introduce the proposed ARFF-GKLMS algorithm. Model (12) shows that, although it no longer required to evaluate Gaussian kernel functions, the preset kernel bandwidth ξ still plays a prominent role through the Gaussian vectors $\boldsymbol{\omega}_m$ sampled from $\mathcal{N}(\boldsymbol{0}_D, \xi^{-2}\boldsymbol{I}_D)$. Note that a parallel can be drawn between these vectors $\boldsymbol{\omega}_m$ and the dictionary elements usually considered with KAF algorithms; see, e.g., [1], [4], [29]. We shall now consider adjusting vectors $\{\boldsymbol{\omega}_m\}_{m=1}^D$ to improve the performance of RFF-based algorithms.

Consider the mean-square error cost function defined by:

$$\mathrm{Ł}(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega) = \mathbb{E}\left\{\left|y_n - \boldsymbol{\alpha}^\top \boldsymbol{z}_\Omega(\boldsymbol{x}_n)\right|^2\right\} \tag{14}$$

with $\boldsymbol{\omega}_\Omega = (\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_D)$ and $\boldsymbol{b}_\Omega = (b_1, \cdots, b_D)$. We aim to estimate these optimal variables $\boldsymbol{\alpha}$, $\boldsymbol{\omega}_\Omega$, and $\boldsymbol{b}_\Omega$ by solving the following optimization problem of identifying the nonlinear system described by model (12):

$$\min_{\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega} \mathrm{Ł}(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega). \tag{15}$$

Following to the steepest-descent principle, the weight vector $\boldsymbol{\alpha}_{n+1}$ at time $n+1$ can be evaluated by updating the weight vector $\boldsymbol{\alpha}_n$ at time n as follows:

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + \frac{1}{2}\eta_\alpha \left[-\frac{\partial \mathrm{Ł}(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)}{\partial \boldsymbol{\alpha}}\right] \tag{16}$$

where $\eta_\alpha > 0$ denotes the learning step-size. The gradient vector of $\mathrm{Ł}(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)$ with respect to $\boldsymbol{\alpha}$ is approximated by its instantaneous value, i.e.,

$$\frac{\partial \mathrm{Ł}(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)}{\partial \boldsymbol{\alpha}} \approx -2e_n \boldsymbol{z}_{\Omega,n}(\boldsymbol{x}_n). \tag{17}$$

with $e_n = y_n - \boldsymbol{\alpha}_n^\top \boldsymbol{z}_{\Omega,n}(\boldsymbol{x}_n)$ the instantaneous estimation error. Substituting the stochastic subgradient (17) into (16), we arrive at the update relation of the ARFF-GKLMS algorithm:

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + \eta_\alpha e_n \boldsymbol{z}_{\Omega,n}(\boldsymbol{x}_n) \tag{18}$$

with $\boldsymbol{\alpha}_n = [\alpha_{1,n}, \ldots, \alpha_{D,n}]^\top$ the weight vector, and $\boldsymbol{z}_{\Omega,n}(\boldsymbol{x}_n)$ the adaptive random Fourier features transformation vector:

$$\boldsymbol{z}_{\Omega,n}(\boldsymbol{x}_n) = \left[\cos(\boldsymbol{\omega}_{1,n}^\top \boldsymbol{x}_n + b_{1,n}), \ldots, \cos(\boldsymbol{\omega}_{D,n}^\top \boldsymbol{x}_n + b_{D,n})\right]^\top. \tag{19}$$

Now we apply the steepest-descent principle to (14) in order to update the m-th vector $\boldsymbol{\omega}_{m,n}$:

$$\boldsymbol{\omega}_{m,n+1} = \boldsymbol{\omega}_{m,n} + \frac{1}{2}\eta_\omega \left[-\frac{\partial Ł(\boldsymbol{\alpha}_n, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)}{\partial \boldsymbol{\omega}_m} \right] \quad (20)$$

for $m = 1, \ldots, D$, where $\eta_\omega > 0$ is the corresponding learning step-size. Applying the chain rule to take the partial derivative of (14) with respect to $\boldsymbol{\omega}_m$, we obtain:

$$\frac{\partial Ł(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)}{\partial \boldsymbol{\omega}_m} = \frac{\partial Ł(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)}{\partial z_{\Omega,m,n}(\boldsymbol{x}_n)} \cdot \frac{\partial z_{\Omega,m,n}(\boldsymbol{x}_n)}{\partial \boldsymbol{\omega}_m} \quad (21)$$
$$\approx 2e_n \alpha_{m,n} \sin(\boldsymbol{\omega}_{m,n}^\top \boldsymbol{x}_n + b_{m,n})\boldsymbol{x}_n$$

where the subgradient vector $\partial Ł(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)/\partial \boldsymbol{\omega}_m$ is replaced by its instantaneous estimate, i.e., the stochastic subgradient. Substituting (21) into (20), the update equation is given by:

$$\boldsymbol{\omega}_{m,n+1} = \boldsymbol{\omega}_{m,n} - \eta_\omega e_n \alpha_{m,n} \sin(\boldsymbol{\omega}_{m,n}^\top \boldsymbol{x}_n + b_{m,n})\boldsymbol{x}_n. \quad (22)$$

Likewise, we can obtain the recursive relation of the m-th phase factor $b_{m,n}$:

$$b_{m,n+1} = b_{m,n} - \eta_b e_n \alpha_{m,n} \sin(\boldsymbol{\omega}_{m,n}^\top \boldsymbol{x}_n + b_{m,n}) \quad (23)$$

for $m = 1, \ldots, D$, with the learning step-size $\eta_b > 0$. The procedures of the ARFF-GKLMS are listed in Algorithm 1.

Before going further, two important points need to be given attention. First, problem (15) is no longer convex with respect to variables $(\boldsymbol{\alpha}, \boldsymbol{\omega}_\Omega, \boldsymbol{b}_\Omega)$. We shall however observe in the next section that, thanks to the adaptation steps (22) and (23), the ARFF-GKLMS algorithm offers a fast convergence rate, low steady-state error, and good tracking ability, in particular when processing with non-stationary systems. More importantly, due to its simplicity, the VRFF method can be readily applied to other RFF-based algorithms. Secondly, the adaptation steps (22) and (23) no longer guaranty that the $\{\boldsymbol{\omega}_{m,n}\}$ are driven by any Gaussian distribution $\mathcal{N}(\boldsymbol{0}_D, \xi^{-2}\boldsymbol{I}_D)$ as the algorithm progresses. This does not allow us to establish a correspondence between the $\{\boldsymbol{\omega}_{m,n}\}$ and the bandwidth ξ_n of a Gaussian kernel. Further work will be carried out to give a better insight in understanding the properties of the nonlinear mapping.

Algorithm 1 ARFF-GKLMS algorithm

1: **Initialization:**
2: Set the step-sizes η_α, η_ω, η_b, and the kernel bandwidth ξ.
3: Generate random $\boldsymbol{\omega}_{m,1}$ and $b_{m,1}$ for $m = 1, 2, \ldots, D$.
4: **Input:** $\{(\boldsymbol{x}_n, y_n)\}$, $n = 1, 2, \ldots, N$.
5: **for** $n = 1, 2, \cdots, N$ **do**
6:     Update $\boldsymbol{\alpha}_{n+1}$ via (18).
7:     **for** $m = 1, 2, \ldots, D$ **do**
8:         Update $\boldsymbol{\omega}_{m,n+1}$ via (22);
9:         Update $b_{m,n+1}$ via (23).
10:    **end for**
11: **end for**
12: **Output:** $f(\boldsymbol{x}_N)$.

IV. SIMULATION RESULTS

In this section, we shall present two simulation examples to validate the improved performance of the ARFF-GKLMS

algorithm compared to its RFF-GKLMS counterpart on the one hand, and to the classical GKLMS algorithm with coherence sparsification (CS) criterion [4] on the other hand. All the simulated curves were obtained by averaging over 200 independent Monte Carlo runs.

A. Stationary Nonlinear System Identification

Consider first the stationary nonlinear system defined by:

$$y_n = \boldsymbol{\kappa}_{\xi^\star}^\top(\boldsymbol{x}_n)\boldsymbol{w}^\star + z_n$$

where z_n denotes a zero-mean Gaussian observation noise at a SNR = 15dB, and $\boldsymbol{w}^\star$ the optimal weight vector is given by

$$\boldsymbol{w}^\star = [0.756, -1.384, -0.101, 0.445, -0.565, 0.134]^\top.$$

The kernelized input vector $\boldsymbol{\kappa}_{\xi^\star}(\boldsymbol{x}_n)$ was constructed based on the Gaussian kernel with bandwidth $\xi^\star = 0.95$ and the dictionary elements defined by:

$$\mathcal{D} = \left\{ \begin{bmatrix} 0.17 \\ -1.92 \end{bmatrix}, \begin{bmatrix} -1.62 \\ -0.18 \end{bmatrix}, \begin{bmatrix} 0.52 \\ 1.55 \end{bmatrix}, \begin{bmatrix} 2.90 \\ 1.92 \end{bmatrix}, \begin{bmatrix} -2.01 \\ -2.47 \end{bmatrix}, \begin{bmatrix} 2.66 \\ -0.82 \end{bmatrix} \right\}.$$

The input sequence $x_n = \rho x_{n-1} + \sqrt{1 - \rho^2} u_n$ was generated with correlation coefficient $\rho = 0.5$ and u_n a random sequence governed by the i.i.d. standard normal distribution. The input data vector was defined as $\boldsymbol{x}_n = [x_n, x_{n-1}]^\top$. The step-size η_α was set to 0.2 for the GKLMS-CS, 0.01 for the RFF-GKLMS, and 0.005 for the ARFF-GKLMS, respectively. Both step-sizes η_ω and η_b were set to 1. The number of RFF was set to $D = 48$. The kernel bandwidth of the Gaussian kernel used by the GKLMS-CS was set to $\xi = 0.95$, and the threshold of the CS criterion δ_κ was set to 0.7 to obtain the final dictionary size $M = 51$ for the comparisons of learning curves of transient EMSE.

Fig. 1(a) shows that ARFF-GKLMS algorithm significantly outperformed the RFF-GKLMS and the GKLMS-CS algorithms in terms of convergence rate and steady-state excess-mean-square error (EMSE), which is the mean of the last 5×10^3 entries of the ensemble-average learning curve of EMSE. Correspondingly, Figs. 1(c) and 1(d) show that the weight coefficients of the ARFF-GKLMS converged faster than those of the RFF-GKLMS. Fig. 1(b) shows the location of vectors $\boldsymbol{\omega}_n$ at the beginning and at the end of the optimization process. Fig. 1(e) shows that ξ setting has a strong effect on the performance of the GKLMS-CS algorithm, which thus needs to be carefully initialized based on side information or preliminary tests. After a transient stage, the dictionary size reaches a maximum value M determined by the CS criterion. Fig. 1(f) shows that M gradually decreases as the kernel bandwidth ξ increases. We observe on Fig. 1(e) that the EMSE at steady state of the RFF-GKLMS algorithm is sensitive to large orders D and to kernel bandwidth ξ setting. In contrast, we can notice that the EMSE at steady-state of the ARFF-GKLMS algorithm is robust with respect to kernel bandwidth ξ initialization, particularly for large ξ. Nevertheless, the algorithm suffers from performance degradation when both D and the kernel bandwidth are small, as a result of the poor approximation capacities of the kernel model in that case.
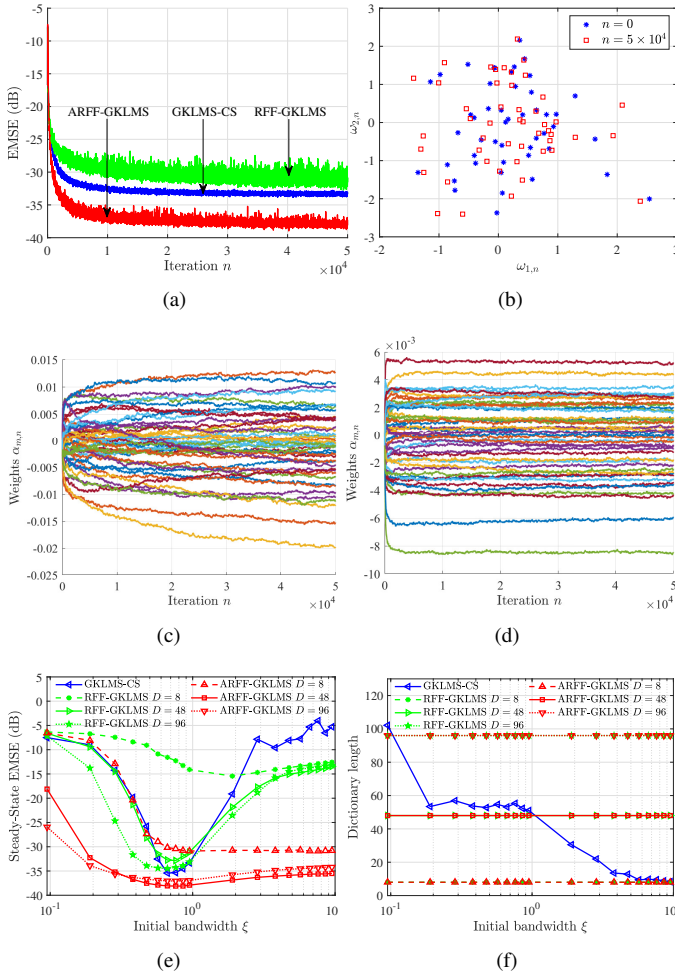
Fig. 1. Simulation results for stationary nonlinear system identification. (a) Learning curves of transient EMSE for $\xi = 0.95$, $M = 51$, and $D = 48$. (b) Location of vectors $\boldsymbol{\omega}_n$ ($D = 48$). (c) Weights evolution for the RFF-GKLMS ($D = 48$). (d) Weights evolution for the ARFF-GKLMS ($D = 48$). (e) Steady-state EMSE versus ξ initial setting. (f) Dictionary length versus different initial ξ.

B. Non-stationary Nonlinear System Identification

Consider the following non-stationary nonlinear system with an abrupt change at time instant $n = 1 \times 10^4$:

$$
\begin{cases}
d_n = \big[0.8 - 0.5\exp(-d_{n-1}^2)\big]d_{n-1} + 0.1\sin(d_{n-1}\pi) \\
\qquad - \big[0.3 + 0.9\exp(-d_{n-1}^2)\big]d_{n-2}, \\
\qquad\qquad \text{for } 0 \le n \le 5 \times 10^3, \\
d_n = \big[0.2 - 0.7\exp(-d_{n-1}^2)\big]d_{n-1} + 0.2\sin(d_{n-1}\pi) \\
\qquad - \big[0.8 + 0.8\exp(-d_{n-1}^2)\big]d_{n-2}, \\
\qquad\qquad \text{for } 5 \times 10^3 < n \le 1 \times 10^4, \\
y_n = d_n + z_n,
\end{cases}
$$

with $d_{-1} = d_{-2} = 0.1$, and z_n a zero-mean white Gaussian noise at SNR = 25dB. The input data vector consists of the nonlinear delayed system outputs $\boldsymbol{x}_n = [d_{n-1}, d_{n-2}]^\top$. The step-size η_α was set to 0.05 for the GKLMS-CS, and to 0.005 for the RFF-GKLMS and the ARFF-GKLMS, respectively. Both step-sizes η_ω and η_b were set to 0.05. The threshold of the CS criterion δ_κ and the order D were set to 0.9 and 96,

respectively. In order to test the ability of the ARFF-GKLMS to track nonstationary systems, the kernel bandwidth ξ was set to 0.3661 for all algorithms.

Fig. 2(a) shows that the ARFF-GKLMS has a good tracking ability, characterized by the lowest steady-state EMSE and the fastest convergence rate after the abrupt change, thanks to the online adaptation of $\boldsymbol{\omega}_n$. Fig. 2(b) shows that the dictionary length of the ARFF-GKLMS remains significantly smaller than that of the GKLMS-CS, which allows to save computation overhead. We can observe on Fig. 2(c) that the locations of vectors $\boldsymbol{\omega}_n$ remained unchanged during the first stationary phase because the kernel bandwidth was carefully initialized for it, and then almost half of the $\boldsymbol{\omega}_n$ changed in order to adapt to the abrupt change.
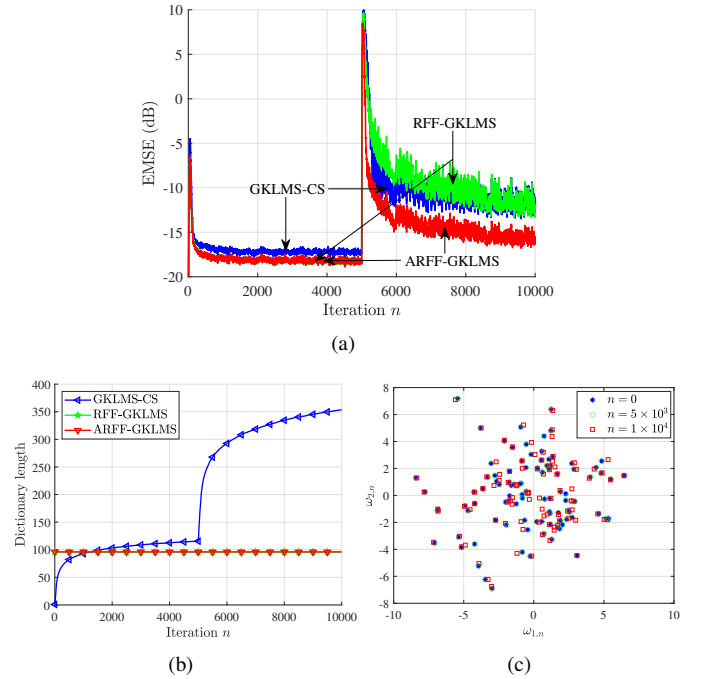

Fig. 2. Simulation results for nonstationary nonlinear system identification. (a) Learning curves of transient EMSE . (b) Evolution of the dictionary length. (c) Variation in the locations of vectors $\boldsymbol{\omega}_n$ ($D = 96$).

V. CONCLUSION

In this letter, we proposed a novel ARFF-GKLMS algorithm to adapt random Fourier features. This extra flexibility endows the algorithm with robustness and good tracking ability in non-stationary environments. The simulation results showed a significant performance improvement, both in transient and steady state. Since the step-sizes η_ω and η_b have an important impact on the performance of the ARFF-GKLMS, variable step-size methods will be considered in future work. We will also apply a forward-backward splitting framework to eliminate the features with negligible contribution to the estimation performance. Finally, as mentioned before, further work will be carried out to give a better insight in understanding the properties of the nonlinear mapping resulting from the adaptation process.

References

[1] 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, 2008.

[2] S. S. Keerthi and C. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Computation*, vol. 15, no. 7, pp. 1667–1689, 2003.

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

[4] 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.

[5] 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.

[6] 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*, 2013.

[7] 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, July 2015.

[8] 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.

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

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

[11] 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, May 2014, pp. 7243–7247.

[12] 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, Jun. 2014.

[13] W. Gao and J. Chen, "Transient performance analysis of zero-attracting Gaussian kernel LMS algorithm with pre-tuned dictionary," *IEEE Access*, vol. 7, pp. 135 770–135 779, 2019.

[14] W. Gao, M. Song, and J. Chen, "Tracking analysis of Gaussian kernel signed error algorithm for time-variant nonlinear systems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 10, pp. 2289–2293, 2020.

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

[16] 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.

[17] H. Fan, Q. Song, and S. B. Shrestha, "Kernel online learning with adaptive kernel width," *Neurocomputing*, vol. 175, pp. 233–242, 2016.

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

[19] P. Bouboulis, S. Pougkakiotis, and S. Theodoridis, "Efficient KLMS and KRLS algorithms: A random Fourier feature perspective," in *Proc. IEEE SSP*, 2016, pp. 1–5.

[20] P. Bouboulis, S. Chouvardas, and S. Theodoridis, "Online distributed learning over networks in RKH spaces using random Fourier features," *IEEE Trans. Signal Process.*, vol. 66, no. 7, pp. 1920–1932, Apr. 2018.

[21] K. Xiong and S. Wang, "The online random Fourier features conjugate gradient algorithm," *IEEE Signal Process. Lett.*, vol. 26, no. 5, pp. 740–744, 2019.

[22] H. Zhang, B. Yang, L. Wang, and S. Wang, "General Cauchy conjugate gradient algorithms based on multiple random Fourier features," *IEEE Trans. Signal Process.*, vol. 69, pp. 1859–1873, 2021.

[23] V. R. M. Elias, V. C. Gogineni, W. A. Martins, and S. Werner, "Kernel regression over graphs using random Fourier features," *IEEE Trans. Signal Process.*, vol. 70, pp. 936–949, 2022.

[24] B. Schölkopf, R. Herbrich, and R. Williamson, "A generalized representer theorem," NeuroCOLT, Royal Holloway College, University of London, UK, Tech. Rep. NC2-TR-2000-81, 2000.

[25] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2001.

[26] R. Herbrich, *Learning Kernel Classifiers: Theory and Algorithms*. Cambridge, MA: MIT Press, 2001.

[27] A. Rahimi and B. Recht, "Random features for large-scale kernel machines." in *NIPS*, vol. 3, no. 4, 2007, pp. 1–8.

[28] W. Rudin, *Fourier analysis on groups*. Courier Dover Publications, 2017.

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