This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

Adaptive Graph Filters in Reproducing Kernel Hilbert Spaces: Design and Performance Analysis

Vitor R. M. Elias, Vinay Chakravarthi Gogineni, *Member, IEEE,* Wallace A. Martins, *Senior Member, IEEE* and Stefan Werner, *Senior Member, IEEE*

***Abstract*—This paper develops adaptive graph filters that operate in reproducing kernel Hilbert spaces. We consider both centralized and fully distributed implementations. We first define nonlinear graph filters that operate on graph-shifted versions of the input signal. We then propose a centralized graph kernel least mean squares (GKLMS) algorithm to identify nonlinear graph filters' model parameters. To reduce the dictionary size of the centralized GKLMS, we apply the principles of coherence check and random Fourier features (RFF). The resulting algorithms have performance close to that of the GKLMS algorithm. Additionally, we leverage the graph structure to derive the distributed graph diffusion KLMS (GDKLMS) algorithms. We show that, unlike the coherence check-based approach, the GDKLMS based on RFF avoids the use of a pre-trained dictionary through its data-independent fixed structure. We conduct a detailed performance study of the proposed RFF-based GDKLMS, and the conditions for its convergence both in mean and mean-squared senses are derived. Extensive numerical simulations show that GKLMS and GDKLMS can successfully identify nonlinear graph filters and adapt to model changes. Furthermore, RFF-based strategies show faster convergence for model identification and exhibit better tracking performance in model-changing scenarios.**

***Index Terms*—Adaptive signal processing, distributed learning, kernel graph filters, kernel LMS, random Fourier features.**

I. Introduction

Graph signal processing (GSP) has recently received increased attention due to its wide applicability to model, process, and analyze signals and large data sets, ranging from daily-life social networks to sensor networks for industrial and military applications [1]–[5]. For instance, in the context of a wireless sensor network, graph nodes and edges represent sensors and communication links, respectively, while the so-called graph signal is the measurement snapshot across sensors [6]. Similar to traditional digital signal processing (DSP) techniques, the basic building block in GSP is the graph-shift operation, which captures node interconnections [7]. In the particular case of linear networks, the graph-shifted signal on a given node is a linear combination of adjacent node signals, where the weights relate to the edge values. This resemblance to DSP has sparked the development of a vast amount of GSP counterparts of methods related to spectral analysis [8]–[14] and traditional time-series analysis [15], [16].

One of the key research areas in GSP is modeling unknown relations between input and output graph signals through a filter [11]–[18]. The application of linear shift-invariant filter models is widely employed in the literature, e.g., to design graph spectral filters [11], [12] and model dynamic graph signals [15], [16]. Several works deal with adaptive learning of graph filters, see, e.g., [19]–[23]. These methods were later extended to multitask graphs [24], [25]. The previous works adopt the ideas of linear adaptive networks [26], [27] to estimate the graph filter through in-network processing. However, linear models cannot accurately represent many real-world systems that exhibit more sophisticated input-output relations. Prominent examples include the relations between air pressure and temperature [28], and wind speed and generated power in wind turbines [29].

In conventional DSP, several approaches to nonlinear system modeling exist in the literature [30]–[38]. In particular, methods based on reproducing kernel Hilbert spaces (RKHS) have gained popularity due to their efficacy and mathematical simplicity [36]–[53]. There is extensive literature on function estimation in RKHS for both single- and multi-node networks, see, e.g., [39]–[57]. Most works on adaptive networks treat each nodal signal as time series to estimate a common filter vector. In contrast, in GSP, the graph filter operates on an instantaneous topology-dependent snapshot of the network state by exploiting graph shifts. Although some of the prior works account for the input signals' network-related characteristics, such as smoothness across the graph, existing RKHS-based approaches do not consider graph-shifted signals. The shift operator and delayed versions of graph signals have been explored for linear adaptive graph filters [22], [23].

This paper introduces nonlinear graph filters and presents two adaptive methods for function estimation over graphs, namely the centralized graph kernel least mean squares (GKLMS) and the graph diffusion kernel least mean squares (GDKLMS). Preliminary results on this topic have been presented in [58]. The proposed nonlinear graph filters generalize conventional linear graph filters and consist of a nonlinearity applied to a combination of graph-shifted versions of the input signal. For the estimation methods, we consider two approaches for model reduction, namely coherence check (CC) [40], [52] that sparsifies the original dictionary of the GKLMS, and random Fourier features (RFF) [59] that ap-

This work was partly supported by: Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) –– Grant number: 88887.310189/2018-00, CNPq, ERCIM Alain Bensoussan Fellowship Programme, ERC project AGNOSTIC (Actively Enhanced Cognition based Framework for Design of Complex Systems – Grant number: 742648), FAPERJ, and the Research Council of Norway.

Vitor R. M. Elias, Vinay Chakravarthi Gogineni and Stefan Werner are with the Norwegian University of Science and Technology, Norway (e-mail: vitor.elias@ntnu.no, vc.gogineni@ntnu.no and stefan.werner@ntnu.no).

Wallace A. Martins is with the University of Luxembourg, Luxembourg (e-mail: wallace.alvesmartins@uni.lu).

Vitor R. M. Elias and Wallace A. Martins are also with the Federal University of Rio de Janeiro, Brazil.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

proximate kernel evaluations with inner products in a fixed-dimensional space. One of the main features of the CC-based implementation is the automatic tuning of the model order by selecting regressors based on a coherence measure [52]. On the other hand, RFF-based implementations use a data-independent mapping into a space where kernel evaluations can be approximated as inner-products, making them resilient to model changes. Building upon ideas of network diffusion [26], [27], the proposed RFF-based graph diffusion KLMS (GDKLMS) avoids the centralized processing and updates local estimates at each node through collaboration with neighbors. One of the main features of the RFF-based GDKLMS is its data-independent mapping that avoids using a pre-trained dictionary. This makes the GDKLMS more robust to changes in the underlying system since there is no need to retrain dictionaries associated with distributed CC-based solutions [52]. We analyze the performance of the GDKLMS and establish the convergence conditions in both mean and mean-squared senses.

This paper is organized as follows. Section II presents the necessary concepts and notations of GSP, including the conventional models of linear graph filters, and formulates the problem of modeling nonlinear graph filters. The proposed GKLMS and GDKLMS algorithms are presented in Section III. We first derive the GKLMS as a centralized solution for the modeling problem and present the implementations based on CC and RFF. Thereafter, the RFF-based GDKLMS is derived. In Section IV, we present the convergence analysis of the RFF-based GDKLMS, along with the conditions for convergence in the mean and mean-squared senses. In this section, we also study the steady-state mean-squared error. In Section VI, numerical experiments are conducted to demonstrate the performance of the proposed solutions for identifying and tracking the nonlinear graph filters. For this, we use both synthetic and real-life networks. Synthetic examples employ generic nonlinear functions, whereas real-life examples treat the modeling of relations between temperature and humidity data from sensor networks. Finally, in Section VII, we present the concluding remarks of this work.

II. Problem Formulation

Consider an undirected graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$, where $\mathcal{N} = \{1, 2, \ldots, K\}$ is the set of nodes and $\mathcal{E}$ is the set of edges such that $(k, l) \in \mathcal{E}$ if nodes k and l are connected. The graph is associated with a graph-shift operator, $\mathbf{S} \in \mathbb{R}^{K\times K}$, whose entries $[\mathbf{S}]_{k,l} = s_{kl}$ take non-zero values only if $(k, l) \in \mathcal{E}$ [1], [2]. The graph adjacency matrix [2] and the graph Laplacian matrix [1] are the most common choices for $\mathbf{S}$. At time instant n, the graph signal is defined by a vector $\mathbf{x}_n = [x_{1,n}\, x_{2,n}\, \ldots\, x_{K,n}]^{\mathrm{T}}$, with $x_{k,n}$ being the signal value at node k. The graph-shift operation $\mathbf{S}\mathbf{x}_n$ is performed locally at each node k by linearly combining the samples from neighboring nodes, namely $\sum_{l\in\mathcal{N}_k} s_{kl}x_{l,n}$, where $\mathcal{N}_k$ denotes the neighborhood of node k including k itself. In this work, we assume the graph topology and the shift matrix are known. For cases where $\mathbf{S}$ is not known, one can employ different techniques for learning the graph structure available in the GSP literature [11], [60]–[64].

A linear shift-invariant (LSI) graph filter of size $L \times 1$ combines shifted graph signals and is defined by

$$\mathbf{H} = \sum_{i=0}^{L-1} h_i \mathbf{S}^i, \tag{1}$$

where $[h_0\, h_1\, \ldots\, h_{L-1}]^{\mathrm{T}}$ is the linear graph filter coefficient vector [12], [22]. When streaming data is available, a two-dimensional graph-time filter [13] can be employed. The filter processes the signal $\mathbf{x}_n$ and yields the graph filtered vector $\mathbf{y}_n = [y_{1,n}\, y_{2,n}\, \ldots\, y_{K,n}]^{\mathrm{T}}$ as

$$\mathbf{y}_n = \sum_{i=0}^{L-1}\sum_{j=0}^{M-1} h_{i,j}\mathbf{S}^i\mathbf{x}_{n-j} + \boldsymbol{\upsilon}_n, \tag{2}$$

where $M-1$ is the filter memory in temporal domain, and $\boldsymbol{\upsilon}_n = [\upsilon_{1,n}\, \upsilon_{2,n}\, \ldots\, \upsilon_{K,n}]^{\mathrm{T}}$ is a zero-mean wide-sense stationary (WSS) noise with covariance matrix $\mathbf{R}_\upsilon = \mathrm{diag}\{\sigma^2_{\upsilon,1}, \sigma^2_{\upsilon,2}, \ldots, \sigma^2_{\upsilon,K}\}$. Also, $\boldsymbol{\upsilon}_n$ and $\boldsymbol{\upsilon}_m$ are i.i.d. for any $n \neq m$. The model (2) uses walks of up to length $L-1$ in the graph. Thus, it requires multihop communication in distributed implementations, which limits its usage in real-time applications.

A simplified model that avoids multihop communication can be constructed by combining time and graph domains into one, as

$$\mathbf{y}_n = \sum_{i=0}^{L-1} h_i\mathbf{S}^i\mathbf{x}_{n-i} + \boldsymbol{\upsilon}_n. \tag{3}$$

A graph diffusion LMS strategy using model (3) is proposed in [23]. In (3), samples $\{x_{k,n}, [\mathbf{S}\mathbf{x}_{n-1}]_k, \ldots, [\mathbf{S}^{L-1}\mathbf{x}_{n-L+1}]_k\}$ are available locally at node k. Thus, only one graph-shift operation is needed at each time instant. A crucial difference between our GSP approach and conventional single- and multi-variate DSP approaches lies in our assumption that the signals' spatio-temporal dynamics depend on the graph structure.

In many real-world applications, these linear models cannot fully capture the input-output relations [37]. For this purpose, we assume a nonlinear relation between input and output, at node k, given by

$$y_{k,n} = f(\mathbf{r}_{k,n}) + \upsilon_{k,n}, \tag{4}$$

where $f : \mathbb{R}^L \to \mathbb{R}$ is a continuous nonlinear function on $\mathbb{R}^L$, $\upsilon_{k,n}$ is the observation noise at node k, and

$$\mathbf{r}_{k,n} = \left[\, x_{k,n}\, [\mathbf{S}\mathbf{x}_{n-1}]_k\, \ldots\, [\mathbf{S}^{L-1}\mathbf{x}_{n-L+1}]_k \,\right]^{\mathrm{T}}. \tag{5}$$

The objective here is to identify $f(\cdot)$ at each node k given a set of data pairs $\{\mathbf{r}_{k,i}, y_{k,i}\}$, $i \in \{1, 2, \ldots, n\}$. In this paper, we characterize nonlinear graph filters using the principles of kernel adaptive filters.

III. Graph Kernel Adaptive Filters

In order to estimate the nonlinear function $f(\cdot)$ in (4), kernel methods first map the input regressors $\{\mathbf{r}_{k,i}\}_{i=1,k=1}^{n,K}$ into a higher dimensional feature space where $f(\cdot)$ takes a linear form [37], [49]. This mapping is denoted by $\kappa(\cdot, \mathbf{r}_{k,i})$, in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

which $\kappa(\cdot,\cdot) : \mathbb{R}^L \times \mathbb{R}^L \to \mathbb{R}$ is a reproducing kernel, which satisfies [37]

$$\kappa(\mathbf{r}_{k,n}, \mathbf{r}_{k,i}) = \langle \kappa(\cdot, \mathbf{r}_{k,n}), \kappa(\cdot, \mathbf{r}_{k,i}) \rangle_{\mathcal{H}}, \tag{6}$$

where $\mathcal{H}$ is the induced RKHS and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the corresponding inner product. In (6), $\kappa(\cdot, \mathbf{r}_{k,i})$ is a representer evaluation at $\mathbf{r}_{k,i}$ [51], [52]. The definition of the kernel function is sufficient to evaluate the inner product in (6) without explicitly mapping the data into RKHS.

A. Graph Kernel LMS

In the GSP context, K new data samples are available at each time instant. Then, given a set of regressors $\{\mathbf{r}_{k,i}\}_{i=1,k=1}^{n,K}$, the graph function $f(\cdot)$ can be expressed as a kernel expansion in terms of the mapped data as

$$f(\cdot) = \sum_{i=1}^{n} \sum_{k=1}^{K} \alpha_{ik}\, \kappa(\cdot, \mathbf{r}_{k,i}). \tag{7}$$

The model (7) can approximate any continuous function $f(\cdot)$ [37]. Hence, the corresponding estimate of $y_{l,n}$, at node l, is given by

$$\hat{y}_{l,n} = f(\mathbf{r}_{l,n}) = \sum_{i=1}^{n} \sum_{k=1}^{K} \alpha_{ik}\, \kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i}). \tag{8}$$

The coefficients of the expansion in (8) are obtained through the following minimization problem:

$$\min_{\alpha_{ik} \in \mathbb{R}} \sum_{l=1}^{K} \mathrm{E}\left[\left(y_{l,n} - \sum_{i=1}^{n} \sum_{k=1}^{K} \alpha_{ik}\, \kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i}) \right)^2 \right]$$

$$= \min_{\boldsymbol{\alpha} \in \mathbb{R}^{nK}} \mathrm{E}\left[\|\mathbf{y}_n - \mathbf{K}_n\, \boldsymbol{\alpha}\|_2^2 \right], \tag{9}$$

where $\mathrm{E}[\cdot]$ denotes the expected value of the argument, $\boldsymbol{\alpha}^{\mathrm{T}} = [\boldsymbol{\alpha}_1^{\mathrm{T}}\, \boldsymbol{\alpha}_2^{\mathrm{T}}\, \ldots\, \boldsymbol{\alpha}_n^{\mathrm{T}}]$, with $\boldsymbol{\alpha}_i^{\mathrm{T}} = [\alpha_{i1}\, \alpha_{i2}\, \ldots\, \alpha_{iK}]$, and the matrix

$$\mathbf{K}_n = [\mathbf{K}_{1,n}\, \mathbf{K}_{2,n}\, \ldots\, \mathbf{K}_{n,n}] \in \mathbb{R}^{K \times nK} \tag{10}$$

is a Gram matrix with $[\mathbf{K}_{i,n}]_{l,k} = \kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i})$ for $k, l \in \mathcal{N}$.

Considering the growing nature of the dictionary, access to the second-order statistics is impractical. Therefore, we use a stochastic-gradient approach and minimize the instantaneous value of (9) recursively. The update equation for the graph KLMS (GKLMS) is given by

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + \mu\, \mathbf{K}_n^{\mathrm{T}}\, (\mathbf{y}_n - \mathbf{K}_n \boldsymbol{\alpha}_n), \tag{11}$$

where $\mu > 0$ is the step size.

The proposed GKLMS algorithm is summarized in Algorithm 1.

Algorithm 1: GKLMS

Input: step size μ
Initialization: $\boldsymbol{\alpha}_0 =$ *empty vector*;
%Learning
for each time instant n **do**
  Input: $\mathbf{y}_n, \{\mathbf{r}_{k,n}\}_{k=1}^{K}$
  append K zeros to $\boldsymbol{\alpha}_n$;
  compute $\mathbf{K}_n = [\mathbf{K}_{1,n}\, \mathbf{K}_{2,n}\, \ldots\, \mathbf{K}_{n,n}]$;
  update $\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + \mu\, \mathbf{K}_n^{\mathrm{T}}\, (\mathbf{y}_n - \mathbf{K}_n \boldsymbol{\alpha}_n)$;
  store regressors $\{\mathbf{r}_{k,n}\}_{k=1}^{K}$;
end

B. Graph Kernel LMS using Coherence-check

As follows from (8), the model order grows with both time, n, and network size, K, when new data samples arrive. This increase makes this model unsuitable for real-time applications and large-scale networks. The growing dimensionality of the dictionary is a well-known issue in single-node kernel methods [40], [41], [49]–[53], where several solutions have been proposed that learn a sparse, or fixed-size dictionary. Of these, the coherence-based sparsification schemes use a coherence metric [40], [52] between a candidate regressor and the current dictionary to decide whether to include the candidate in the dictionary. Given a set of data samples $\{\mathbf{r}_{k,i}\}_{k=1,i=1}^{K,n-1}$, various approaches can be employed to construct a CC-based sparse dictionary adaptively. In a centralized manner, one can consider regressors from all nodes at each time instant and test the coherence metric for each regressor $\mathbf{r}_{l,n}$, given by

$$\delta_{l,n} = \max_{\mathbf{r}_j \in \mathcal{D}_n} |\kappa(\mathbf{r}_{l,n}, \mathbf{r}_j)|, \tag{12}$$

where $\mathcal{D}_n$ denotes the dictionary obtained before testing regressor $\mathbf{r}_{l,n}$; the dictionary starts empty before running the algorithm. Given a predefined threshold, $\delta > 0$, if $\delta_{l,n} < \delta$, the regressor is added to the dictionary. The process continues over the remaining regressors, accounting for previous regressors added to the dictionary, until a predefined dictionary size, D, is achieved, or all the data samples are used.

Therefore, using the coherence check criterion, $\hat{y}_{l,n}$ in (8) can be rewritten as

$$\hat{y}_{l,n} = \sum_{i \in \mathcal{M}_n} \sum_{k \in \mathcal{K}_i} \alpha_{ik}\, \kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i}), \tag{13}$$

where $\mathcal{M}_n$ is a set of time instants (up to time instant n) in which at least one input regressor is added to the dictionary, with $|\mathcal{M}_n| \leq n$, and $\mathcal{K}_i$ is a set of node indices of the regressors that passed the coherence check at time index i, with $|\mathcal{K}_i| \leq K$. Under the CC criterion, at time index n, the dictionary $\mathcal{D}_n$ contains $|\mathcal{D}_n| = \sum_{i \in \mathcal{M}_n} |\mathcal{K}_i|$ regressors.

Remark 1. Given a set of reasonable conditions on the threshold, δ, the maximum number of regressors in the dictionary is finite, i.e., $|\mathcal{D}_n|$ stops increasing after a certain time [52].

The coefficients of the expansion in (13) are obtained through the following minimization problem:

$$\min_{\widetilde{\alpha}_{ik} \in \mathbb{R}} \sum_{l=1}^{K} \mathrm{E}\Big[\Big(y_{l,n} - \sum_{i \in \mathcal{M}_n} \sum_{k \in \mathcal{K}_i} \widetilde{\alpha}_{ik}\, \kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i}) \Big)^2 \Big]$$

$$= \min_{\widetilde{\boldsymbol{\alpha}} \in \mathbb{R}^{|\mathcal{D}_n|}} \mathrm{E}\left[\|\mathbf{y}_n - \widetilde{\mathbf{K}}_n \widetilde{\boldsymbol{\alpha}}\|_2^2 \right], \tag{14}$$

where $\widetilde{\boldsymbol{\alpha}}^{\mathrm{T}} = [\widetilde{\boldsymbol{\alpha}}_1^{\mathrm{T}}\, \widetilde{\boldsymbol{\alpha}}_2^{\mathrm{T}}\, \ldots\, \widetilde{\boldsymbol{\alpha}}_{|\mathcal{M}_n|}^{\mathrm{T}}]$, with $\widetilde{\boldsymbol{\alpha}}_i^{\mathrm{T}} = [\widetilde{\alpha}_{i1}\, \widetilde{\alpha}_{i2}\, \ldots\, \widetilde{\alpha}_{i|\mathcal{K}_i|}] \in \mathbb{R}^{|\mathcal{K}_i|}$. The matrix $\widetilde{\mathbf{K}}_n$ is a Gram matrix given by

$$\widetilde{\mathbf{K}}_n = [\widetilde{\mathbf{K}}_{1,n}\, \widetilde{\mathbf{K}}_{2,n}\, \ldots\, \widetilde{\mathbf{K}}_{|\mathcal{M}_n|,n}] \in \mathbb{R}^{K \times |\mathcal{D}_n|}, \tag{15}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

Algorithm 2: GKLMS using coherence check

Input: training data $\{\tilde{\mathbf{r}}_{l,i}, \tilde{y}_{l,i}\}_{l=1,i=1}^{K,t}$, dictionary size D, threshold δ, and step size μ
Initialization: $\mathcal{D} = \emptyset$, $\boldsymbol{\alpha}_0 =$ *empty vector*;
%Learning
for each time instant n **do**
  Input: $\mathbf{y}_n, \{\mathbf{r}_{k,n}\}_{k=1}^{K}$
  for $k = 1, \ldots, K$ **do**
    if $|\mathcal{D}| < D$ **then**
      compute $\delta_{l,n} = \max_{\mathbf{r}_j \in \mathcal{D}} |\kappa(\mathbf{r}_{l,n}, \mathbf{r}_j)|$;
      if $\delta_{l,n} < \delta$ **then**
        add $\mathbf{r}_{l,n}$ to $\mathcal{D}$;
        add l to $\mathcal{K}_n$;
      end
    end
  end
  if $|\mathcal{K}_n| \neq 0$ **then**
    append $|\mathcal{K}_n|$ zeros to $\widetilde{\alpha}_n$;
    add n to $\mathcal{M}_n$;
  end
  compute $\widetilde{\mathbf{K}}_n = [\widetilde{\mathbf{K}}_{1,n}\, \widetilde{\mathbf{K}}_{2,n}\, \ldots\, \widetilde{\mathbf{K}}_{|\mathcal{M}_n|,n}]$;
  update $\widetilde{\boldsymbol{\alpha}}_{n+1} = \widetilde{\boldsymbol{\alpha}}_n + \mu\, \widetilde{\mathbf{K}}_n^{\mathrm{T}}\, (\mathbf{y}_n - \widetilde{\mathbf{K}}_n \widetilde{\boldsymbol{\alpha}}_n)$;
end

with $[\widetilde{\mathbf{K}}_{i,n}]_{l,k} = \kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i})$, for $l \in \mathcal{N}$ and $k \in \mathcal{K}_i$.

Using the stochastic-gradient approach and minimizing the instantaneous value of (14), we obtain the following update rule of the centralized GKLMS using coherence check:

$$\widetilde{\boldsymbol{\alpha}}_{n+1} = \widetilde{\boldsymbol{\alpha}}_n + \mu\, \widetilde{\mathbf{K}}_n^{\mathrm{T}}\, (\mathbf{y}_n - \widetilde{\mathbf{K}}_n \widetilde{\boldsymbol{\alpha}}_n). \tag{16}$$

Algorithm 2 summarizes the steps for pre-training the dictionary according to the CC criterion and the learning stage of the CC-based GKLMS algorithm.

Remark 2. If coherence check is employed in an online fashion, two events must be considered for each candidate regressor. If the regressor does not satisfy the CC criterion, the dictionary remains the same. Otherwise, $\widetilde{\mathbf{K}}_n$ gets one new column and a zero-valued entry must be appended to $\widetilde{\boldsymbol{\alpha}}_n$ [52]. At every time instant i, for $i \in \mathcal{M}_n$, $|\mathcal{K}_i|$ regressors are added to the dictionary. Hence, $|\mathcal{K}_i|$ zeros must be appended to $\widetilde{\boldsymbol{\alpha}}_n$.

C. Graph Kernel LMS using Random Fourier Features

An alternative to sparsification methods, like CC, is provided by RFF [59]. The shift-invariant kernel evaluation $\kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i}) = \kappa(\mathbf{r}_{l,n} - \mathbf{r}_{k,i})$ can be approximated as an inner product in the D-dimensional RFF space. This turns the problem into a finite-dimension linear filtering problem, while avoiding the evaluation of kernel functions [59]. Let $\mathbf{z}_{l,n}$ be the mapping of $\mathbf{r}_{l,n}$ into the RFF space $\mathbb{R}^D$, given by

$$\mathbf{z}_{l,n} = (D/2)^{-\frac{1}{2}} \left[\cos(\mathbf{v}_1^{\mathrm{T}}\mathbf{r}_{l,n} + b_1)\, \ldots\, \cos(\mathbf{v}_D^{\mathrm{T}}\mathbf{r}_{l,n} + b_D)\right]^{\mathrm{T}}, \tag{17}$$

Algorithm 3: GKLMS using RFF

Input: RFF-space dimension D, pdf $p(\mathbf{v})$, step size μ
Initialization:
draw vectors $\{\mathbf{v}_i\}_{i=1}^{D}$ from $p(\mathbf{v})$;
draw phase terms $\{b_i\}_{i=1}^{D}$ from $[0, 2\pi]$;
$\mathbf{h}_0 = \mathbf{0}_D$;
%Learning
for each time instant n **do**
  Input: $\mathbf{y}_n, \{\mathbf{r}_{k,n}\}_{k=1}^{K}$
  compute $\{\mathbf{z}_{l,n}\}_{l=1}^{K}$ using (17);
  construct matrix $\mathbf{Z}_n$ using (21);
  update $\mathbf{h}_{n+1} = \mathbf{h}_n + \mu \mathbf{Z}_n \mathbf{e}_n$;
end

where the phase terms $\{b_i\}_{i=1}^{D}$ are drawn from a uniform distribution on the interval $[0,\, 2\pi]$. Vectors $\{\mathbf{v}_i\}_{i=1}^{D}$ are drawn from the probability density function (pdf) $p(\mathbf{v})$ such that

$$k(\mathbf{r}_{l,n} - \mathbf{r}_{k,i}) = \int p(\mathbf{v}) \exp\left(\mathrm{j}\mathbf{v}^{\mathrm{T}}(\mathbf{r}_{l,n} - \mathbf{r}_{k,i})\right) \mathrm{d}\mathbf{v}, \tag{18}$$

where $\mathrm{j}^2 = -1$. In other words, the Fourier transform of $k(\mathbf{r}_{l,n} - \mathbf{r}_{k,i})$ is given by $p(\mathbf{v})$. From (17) and (18), it can be verified that $\mathrm{E}[\mathbf{z}_{k,i}^{\mathrm{T}}\mathbf{z}_{l,n}] = k(\mathbf{r}_{l,n}, \mathbf{r}_{k,i})$. Then, the kernel evaluation can be approximated as $\kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i}) \approx \mathbf{z}_{k,i}^{\mathrm{T}}\mathbf{z}_{l,n}$ and the estimate $\hat{y}_{l,n}$ in (8) can be approximated by

$$\hat{y}_{l,n} \approx \Big(\sum_{i=1}^{n}\sum_{k=1}^{K} \alpha_{ik}\, \mathbf{z}_{k,i}\Big)^{\mathrm{T}} \mathbf{z}_{l,n} = \mathbf{h}^{\mathrm{T}}\mathbf{z}_{l,n}, \tag{19}$$

where $\mathbf{h} \in \mathbb{R}^D$ is the representation of the function $f(\cdot)$ in the RFF space. A higher value of D improves the approximation of the kernel function. Therefore, the choice of D depends mostly on the application, as it represents a trade-off between performance and complexity.

We note that, if a Gaussian kernel given by $\kappa(\mathbf{r}_{l,n}, \mathbf{r}_{k,i}) = \exp\left(-\|\mathbf{r}_{l,n} - \mathbf{r}_{k,i}\|_2^2/(2\sigma^2)\right)$ is used, the pdf $p(\mathbf{v})$ is given in closed form as a normal distribution. See [59] for closed-form representations of $p(\mathbf{v})$ when other kernel functions are used.

The linear representation of $f(\cdot)$ in the RFF space, $\mathbf{h}$, can be estimated by solving the following optimization problem:

$$\min_{\mathbf{h} \in \mathbb{R}^D} \mathrm{E}\left[\|\mathbf{y}_n - \mathbf{Z}_n^{\mathrm{T}}\mathbf{h}\|_2^2\right], \tag{20}$$

where the matrix

$$\mathbf{Z}_n = [\mathbf{z}_{1,n}\, \mathbf{z}_{2,n}\, \ldots\, \mathbf{z}_{K,n}] \tag{21}$$

represents the RFF mapping of all input vectors at time n. Similar to (16), approximating the solution through stochastic-gradient descent iterations yields the RFF-based centralized graph kernel LMS (GKLMS) update rule

$$\mathbf{h}_{n+1} = \mathbf{h}_n + \mu \mathbf{Z}_n \mathbf{e}_n, \tag{22}$$

where $\mathbf{e}_n = \mathbf{y}_n - \mathbf{Z}_n^{\mathrm{T}}\mathbf{h}_n$. The proposed GKLMS using RFF is summarized in Algorithm 3.

Notice that the estimates $\widetilde{\boldsymbol{\alpha}}$ in (16) and $\mathbf{h}$ in (22) require knowledge of the input of the entire graph, which can be impractical in applications without a centralized processing

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

unit. Therefore, we propose a distributed implementation of the GKLMS, named graph diffusion KLMS (GDKLMS).

Remark 3. A CC-based distributed implementation requires a pre-trained dictionary available at each node [40]. The dictionary can be pre-trained in a centralized way and broadcasted to the entire network, or by a single node that shares its dictionary with all nodes. More importantly, the dictionary depends on available training data, and may be retrained whenever there are changes in the underlying model. Therefore, RFF-based algorithms seem more suitable for distributed implementations and robust to changes in model and data statistics.

D. Graph Diffusion Kernel LMS using RFF

In order to derive a distributed implementation, the global optimization problem (20) is expressed alternatively in the following separable form:

$$\underset{\boldsymbol{\psi}_1,\ldots,\boldsymbol{\psi}_K\in\mathbb{R}^D}{\arg\min} \sum_{k=1}^{K} \mathrm{E}\big[(y_{k,n}-\mathbf{z}_{k,n}^{\mathrm{T}}\boldsymbol{\psi}_k)^2\big], \tag{23}$$

where $\boldsymbol{\psi}_k$ is the local estimate of $\mathbf{h}$ at node k. The optimization problem in (23) can be solved in a distributed fashion by minimizing $\mathrm{E}\left[(y_{k,n}-\mathbf{z}_{k,n}^{\mathrm{T}}\boldsymbol{\psi}_k)^2\right]$ at each node. Let $e_{k,n} = y_{k,n} - \mathbf{z}_{k,n}^{\mathrm{T}}\boldsymbol{\psi}_k$. Following the similar lines of centralized GKLMS, the update rule for $\boldsymbol{\psi}_k$ is given by

$$\boldsymbol{\psi}_{k,n+1} = \boldsymbol{\psi}_{k,n} + \mu\, e_{k,n}\mathbf{z}_{k,n}. \tag{24}$$

We now leverage the graph structure and adopt the adapt-then-combine (ATC) strategy to improve individual estimates via graph diffusion [22], [23], [26], [40], [65]. The ATC strategy is one common diffusion strategy composed by two steps. At iteration n, the first step updates the local estimate, at a given node k, using the new input $\{\mathbf{r}_{k,n}, y_{k,n}\}$, generating an intermediate estimate. In the second step, nodes share and combine their intermediate estimates to generate the final estimate for that iteration. That is, the parameter update of $\mathbf{h}_{k,n}$ at node k is obtained by combining the estimates from its neighborhood. Note that the graph structure defines a node's neighborhood, and adjacent nodes relate to each other. The ATC update rule for the GDKLMS using RFF is given by

$$\begin{cases} \boldsymbol{\psi}_{k,n+1} = \mathbf{h}_{k,n} + \mu\, e_{k,n}\mathbf{z}_{k,n}, & \text{(25a)} \\ \mathbf{h}_{k,n+1} = \sum\limits_{l\in\mathcal{N}_k} a_{lk}\, \boldsymbol{\psi}_{l,n+1}, & \text{(25b)} \end{cases}$$

where the combination coefficients a_{lk} are non-negative and satisfy the condition $\sum_{l\in\mathcal{N}_k} a_{lk} = 1$ [26]. We could use a similar combine-then-adapt (CTA) strategy [65]. Both ATC and CTA strategies share fundamentally the same structure and yield similar results [27]. Algorithm 4 summarizes the steps of the GDKLMS implementation using RFF.

Algorithm 4: GDKLMS using RFF

Input: RFF-space dimension D, pdf $p(\mathbf{v})$, step size μ, combination coefficients a_{lk}
Initialization:
draw vectors $\{\mathbf{v}_i\}_{i=1}^{D}$ from $p(\mathbf{v})$;
draw phase terms $\{b_i\}_{i=1}^{D}$ from $[0, 2\pi]$;
$\mathbf{h}_{k,0} = \mathbf{0}_D$, $\forall k \in \{1, 2, \ldots, K\}$;
$\boldsymbol{\psi}_{k,0} = \mathbf{0}_D$, $\forall k \in \{1, 2, \ldots, K\}$;
%Learning
for each time instant n **do**
  for $k = 1, \ldots, K$ **do**
    compute $\mathbf{z}_{k,n}$ using (17);
    update $\boldsymbol{\psi}_{k,n+1} = \mathbf{h}_{k,n} + \mu\, e_{k,n}\mathbf{z}_{k,n}$;
  end
  for $k = 1, \ldots, K$ **do**
    update $\mathbf{h}_{k,n+1} = \sum\limits_{l\in\mathcal{N}_k} a_{lk}\, \boldsymbol{\psi}_{l,n+1}$;
  end
end

IV. Convergence Analysis

In this section, we study the performance of the proposed RFF-based GDKLMS and establish the conditions for its convergence both in mean and mean-squared senses. For this, at network level, we define the filter coefficient vector in the RFF space $\mathbf{h}_g = \mathbf{1}_K \otimes \mathbf{h}$, the estimated filter coefficient vector in RFF space $\mathbf{h}_{g,n} = [\mathbf{h}_{1,n}^{\mathrm{T}}\, \mathbf{h}_{2,n}^{\mathrm{T}}\, \ldots\, \mathbf{h}_{K,n}^{\mathrm{T}}]^{\mathrm{T}}$, and the (RFF-mapped) input data matrix $\boldsymbol{\mathcal{Z}}_n = \mathrm{blockdiag}\{\mathbf{z}_{1,n}, \mathbf{z}_{2,n}, \ldots, \mathbf{z}_{K,n}\}$. In these definitions, $\mathbf{1}_K$ is a vector of size $K\times 1$ with every entry taking the value one, $\otimes$ denotes the right Kronecker product operator, and blockdiag$\{\cdot\}$ denotes the block-diagonal-stacking operator. Using these definitions, the network-level data model is given by

$$\mathbf{y}_n = \boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}\mathbf{h}_g + \boldsymbol{\upsilon}_n. \tag{26}$$

From these definitions, the network-level recursion of the RFF-based GDKLMS can then be expressed as follows:

$$\mathbf{h}_{g,n+1} = \boldsymbol{\mathcal{A}}\big(\mathbf{h}_{g,n} + \mu\boldsymbol{\mathcal{Z}}_n\mathbf{e}_n\big), \tag{27}$$

where $\boldsymbol{\mathcal{A}} = \mathbf{A}^{\mathrm{T}} \otimes \mathbf{I}_D$. The matrix $\mathbf{A}$, with $[\mathbf{A}]_{l,k} = a_{lk}$, is a $K\times K$ left stochastic matrix (i.e., each column consists of non-negative real numbers whose sum is unity). In the following, we study the convergence behavior of the proposed RFF-based GDKLMS governed by the form (27). For this, we assume the following:

- **A1**: Given a node $k \in \mathcal{N}$, the RFF-mapped data signal $\mathbf{z}_{k,n}$ is drawn from a WSS multivariate random sequence with correlation matrix $\mathbf{R}_{z,k} = \mathrm{E}[\mathbf{z}_{k,n}\mathbf{z}_{k,n}^{\mathrm{T}}]$; in addition, the data vectors $\mathbf{z}_{k,n}$ and $\mathbf{z}_{l,m}$ are independent, for all $k \neq l \in \mathcal{N}$.
- **A2**: The observation noise $\boldsymbol{\upsilon}_n$ is a zero-mean WSS multivariate random sequence, with diagonal correlation matrix $\mathbf{R}_{\upsilon} = \mathrm{E}[\boldsymbol{\upsilon}_n\boldsymbol{\upsilon}_n^{\mathrm{T}}] = \mathrm{diag}\{\sigma_{\upsilon,1}^2, \sigma_{\upsilon,2}^2, \ldots, \sigma_{\upsilon,K}^2\}$, being independent of any other random signal in the model.
- **A3**: The weight vector $\mathbf{h}_{k,n}$ is taken to be independent of $\mathbf{z}_{k,n}$, for $k \in \mathcal{N}$.
- **A4**: The graph topology is assumed to be static, meaning the shift matrix $\mathbf{S}$ and the combiner coefficients a_{lk} are constant throughout the process.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

A5: The step size μ is sufficiently small so that the terms involving higher order powers of μ can be ignored.

The above assumptions are commonly used in the analysis of distributed adaptive schemes over networks.

Remark 4. Note that the vector $\mathbf{z}_{k,n}$ is the representation of $\mathbf{r}_{k,n}$ in the RFF space. Hence, $\mathbf{z}_{k,n}$ may not be normally distributed. If the basis of the RFF space is generated in a way such that the basis vectors $\mathbf{v}_i \neq \mathbf{v}_j$ for any $i \neq j$, the autocorrelation matrix $\mathbf{R}_{z,k}$, for $k \in \mathcal{N}$ will be strictly positive definite [47].

Apart from these assumptions, the analysis also requires properties of the block maximum norm of a matrix (i.e., $\|\cdot\|_{\mathrm{b},\infty}$), the block vectorization operator (i.e., $\mathrm{bvec}\{\cdot\}$) [27], and the block Kronecker product of two matrices (i.e., $\otimes_{\mathrm{b}}$) [66]. Details of these operators can be found in [27], [66], [67].

A. First-order Convergence Analysis

Denoting the global weight deviation vector of the proposed GDKLMS using RFF, at time instant n, as $\tilde{\mathbf{h}}_{g,n} = \mathbf{h}_g - \mathbf{h}_{g,n}$, recalling the fact that $\boldsymbol{\mathcal{A}}\mathbf{h}_g = \mathbf{h}_g$ (since the matrix $\mathbf{A}$ is left stochastic), from (27), the recursion for $\tilde{\mathbf{h}}_{g,n}$ can then be obtained as

$$\tilde{\mathbf{h}}_{g,n+1} = \boldsymbol{\mathcal{B}}_n \tilde{\mathbf{h}}_{g,n} - \mu\, \boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{Z}}_n\, \boldsymbol{\upsilon}_n, \tag{28}$$

where $\boldsymbol{\mathcal{B}}_n = \boldsymbol{\mathcal{A}}\big(\mathbf{I}_{DK} - \mu\, \boldsymbol{\mathcal{Z}}_n \boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}\big)$. In the following, we establish the condition for the convergence in mean.

Theorem 1. Assume the data model (26) and the assumptions **A1**-**A4** hold (assumption **A5** is not required here). Then a sufficient condition for the proposed RFF-based GDKLMS to converge in mean is given by

$$0 < \mu < \frac{2}{\max\limits_{1\leq k\leq K}\left\{\max\limits_{1\leq i\leq D}\{\lambda_i(\,\mathbf{R}_{z,k})\}\right\}}. \tag{29}$$

Proof. Taking the statistical expectation $\mathrm{E}[\cdot]$ on both sides of (28), and using the assumptions **A1**-**A4**, we obtain

$$\mathrm{E}[\tilde{\mathbf{h}}_{g,n+1}] = \overline{\boldsymbol{\mathcal{B}}}\;\; \mathrm{E}[\tilde{\mathbf{h}}_{g,n}], \tag{30}$$

with $\overline{\boldsymbol{\mathcal{B}}} = \mathrm{E}[\boldsymbol{\mathcal{B}}_n] = \boldsymbol{\mathcal{A}}\big(\mathbf{I}_{DK} - \mu\mathbf{R}_z\big)$, where $\mathbf{R}_z = \mathrm{E}[\boldsymbol{\mathcal{Z}}_n\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}]$ $= \mathrm{blockdiag}(\mathbf{R}_{z,1}, \mathbf{R}_{z,2}, \ldots, \mathbf{R}_{z,K})$.

From (30), it is easily seen that $\lim_{n\to\infty}\mathrm{E}\big[\tilde{\mathbf{h}}_{g,n}\big]$ attains a finite value if and only if $\|\overline{\boldsymbol{\mathcal{B}}}\| < 1$, where $\|\cdot\|$ denotes any matrix norm. We derive a convergence condition in terms of μ, by constraining the block maximum norm of the matrix $\overline{\boldsymbol{\mathcal{B}}}$ (i.e., $\|\overline{\boldsymbol{\mathcal{B}}}\|_{\mathrm{b},\infty}$). Using the properties of block maximum norm [26], we can write

$$\|\overline{\boldsymbol{\mathcal{B}}}\|_{\mathrm{b},\infty} \leq \|\boldsymbol{\mathcal{A}}\|_{\mathrm{b},\infty}\|\mathbf{I}_{DK} - \mu\mathbf{R}_z\|_{\mathrm{b},\infty}. \tag{31}$$

Since the matrix $\mathbf{A}$ is left stochastic, we have $\|\boldsymbol{\mathcal{A}}\|_{\mathrm{b},\infty} = \|\mathbf{A}^{\mathrm{T}}\otimes\mathbf{I}_D\|_{\mathrm{b},\infty} = 1$. Furthermore, as the matrix $(\mathbf{I}_{DK} - \mu\mathbf{R}_z)$ is block diagonal symmetric, using [26, Lemma D. 3(a), D. 5], a sufficient condition for $\mathrm{E}[\tilde{\mathbf{h}}_{g,n}]$ to converge in mean is given by $\rho(\mathbf{I}_{DK} - \mu\mathbf{R}_z) < 1$, or, equivalently, $|1 - \mu\lambda_j(\mathbf{R}_z))| < 1$ for $j \in \{1, 2, \ldots, DK\}$, where $\rho(\cdot)$ denotes the spectral radius of the argument matrix and $\lambda_j(\mathbf{R}_z)$ denotes the jth eigenvalue of $\mathbf{R}_z$. After solving this, we arrive at (29). □

B. Second-order Convergence Analysis

Next, we focus on the second-order convergence analysis of the proposed GDKLMS using RFF. Using the energy conservation approach, we investigate the steady-state MSE performance of the proposed scheme.

Defining the $\boldsymbol{\Sigma}$-weighted norm-square of $\tilde{\mathbf{h}}_{g,n}$ as $\|\tilde{\mathbf{h}}_{g,n}\|^2_{\boldsymbol{\Sigma}} = \tilde{\mathbf{h}}^{\mathrm{T}}_{g,n}\boldsymbol{\Sigma}\tilde{\mathbf{h}}_{g,n}$, where $\boldsymbol{\Sigma}$ is a positive semi-definite matrix that can be chosen arbitrarily, and using the assumptions **A1**-**A4**, one can write

$$\mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,n+1}\|^2_{\boldsymbol{\Sigma}}\right] = \mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,n}\|^2_{\boldsymbol{\Sigma}'}\right] + \mu^2\mathrm{E}[\boldsymbol{\upsilon}_n^{\mathrm{T}}\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}\boldsymbol{\mathcal{A}}^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{Z}}\boldsymbol{\upsilon}_n], \tag{32}$$

where the cross terms are zero since $\boldsymbol{\upsilon}_n$ is taken to be zero-mean and statistically independent of $\mathbf{z}_{k,n}$. The matrix $\boldsymbol{\Sigma}'$ is given by

$$\boldsymbol{\Sigma}' = \mathrm{E}[\boldsymbol{\mathcal{B}}_n^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}_n]. \tag{33}$$

Now, using the block Kronecker product denoted by $\otimes_{\mathrm{b}}$ [66] and the $\mathrm{bvec}\{\cdot\}$ operator [66], we can relate the vectors $\boldsymbol{\sigma} = \mathrm{bvec}\{\boldsymbol{\Sigma}\}$ and $\boldsymbol{\sigma}' = \mathrm{bvec}\{\boldsymbol{\Sigma}'\}$ as

$$\boldsymbol{\sigma}' = \boldsymbol{\mathcal{F}}^{\mathrm{T}}\boldsymbol{\sigma}, \tag{34}$$

with

$$\boldsymbol{\mathcal{F}} = \mathrm{E}[\boldsymbol{\mathcal{B}}_n \otimes_{\mathrm{b}} \boldsymbol{\mathcal{B}}_n] = (\boldsymbol{\mathcal{A}}\otimes\boldsymbol{\mathcal{A}})\boldsymbol{\mathcal{H}}, \tag{35}$$

where

$$\boldsymbol{\mathcal{H}} \approx \mathbf{I}_{D^2K^2} - \mu(\mathbf{R}_z \otimes_{\mathrm{b}} \mathbf{I}_{DK}) - \mu(\mathbf{I}_{DK} \otimes_{\mathrm{b}} \mathbf{R}_z). \tag{36}$$

In the above expression, using the assumption **A5**, the terms involving high-order powers of μ are ignored, and we continue our analysis with this approximation. Note that this approximation is standard in the analysis of many distributed schemes over networks [26], [27].

Now, consider the second term on the right-hand side of (32). We can write it as

$$\begin{aligned}
&\mathrm{E}[\boldsymbol{\upsilon}_n^{\mathrm{T}}\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}\boldsymbol{\mathcal{A}}^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{Z}}\boldsymbol{\upsilon}_n] \\
&\quad= \mathrm{Tr}\left(\mathrm{E}[\boldsymbol{\upsilon}_n^{\mathrm{T}}\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}\boldsymbol{\mathcal{A}}^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{Z}}\boldsymbol{\upsilon}_n]\right) \\
&\quad= \mathrm{Tr}\left(\boldsymbol{\mathcal{A}}\mathrm{E}[\boldsymbol{\mathcal{Z}}_n\boldsymbol{\upsilon}_n\boldsymbol{\upsilon}_n^{\mathrm{T}}\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}]\boldsymbol{\mathcal{A}}^{\mathrm{T}}\boldsymbol{\Sigma}\right) \\
&\quad= \mathrm{Tr}\left(\boldsymbol{\mathcal{A}}\mathrm{E}[\boldsymbol{\Phi}_n]\boldsymbol{\mathcal{A}}^{\mathrm{T}}\boldsymbol{\Sigma}\right),
\end{aligned} \tag{37}$$

where $\boldsymbol{\Phi}_n = \boldsymbol{\mathcal{Z}}_n\mathbf{R}_{\upsilon}\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}$.

Using the properties of block Kronecker product [66], we finally have

$$\mathrm{Tr}\left(\boldsymbol{\mathcal{A}}\mathrm{E}[\boldsymbol{\Phi}_n]\boldsymbol{\mathcal{A}}^{\mathrm{T}}\boldsymbol{\Sigma}\right) = \boldsymbol{\gamma}^{\mathrm{T}}\boldsymbol{\sigma}, \tag{38}$$

where $\boldsymbol{\gamma} = \mathrm{bvec}\{\boldsymbol{\mathcal{A}}\mathrm{E}[\boldsymbol{\Phi}_n]\boldsymbol{\mathcal{A}}^{\mathrm{T}}\} = (\boldsymbol{\mathcal{A}}\otimes\boldsymbol{\mathcal{A}})\boldsymbol{\gamma}_{\boldsymbol{\upsilon}}$, with

$$\begin{aligned}
\boldsymbol{\gamma}_{\boldsymbol{\upsilon}} &= \mathrm{bvec}\{\mathrm{E}[\boldsymbol{\Phi}_n]\} \\
&= \mathrm{bvec}\{\mathrm{E}[\boldsymbol{\mathcal{Z}}_n\mathbf{R}_{\upsilon}\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}]\} \\
&= \mathrm{E}[\boldsymbol{\mathcal{Z}}_n \otimes_{\mathrm{b}} \boldsymbol{\mathcal{Z}}_n]\cdot\mathrm{bvec}\{\mathbf{R}_{\upsilon}\}.
\end{aligned} \tag{39}$$

Combining these results, (32) can be expressed as

$$\mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,n+1}\|^2_{\mathrm{bvec}^{-1}\{\boldsymbol{\sigma}\}}\right] = \mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,n}\|^2_{\mathrm{bvec}^{-1}\{\boldsymbol{\mathcal{F}}^{\mathrm{T}}\boldsymbol{\sigma}\}}\right] + \mu^2\boldsymbol{\gamma}^{\mathrm{T}}\boldsymbol{\sigma}, \tag{40}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

where $\text{bvec}^{-1}\{\cdot\}$ rearranges the argument vector of size $D^2K^2 \times 1$ into a $DK \times DK$ matrix, i.e., $\boldsymbol{\Sigma} = \text{bvec}^{-1}\{\boldsymbol{\sigma}\}$.

Theorem 2. Assume the data model (26) and that assumptions **A1**-**A5** hold. Furthermore, assume that the step size μ is sufficiently small such that the approximation (36) is justified by ignoring the higher-order powers of μ, so that (40) can be used as a reasonable representation for studying the dynamics of the weighted mean-squared deviation (MSD). Then, the proposed RFF-based GDKLMS converges in mean-squared sense under

$$0 < \mu < \frac{1}{\max\limits_{1\le k\le K}\left\{\max\limits_{1\le i\le D}\{\lambda_i(\mathbf{R}_{z,k})\}\right\}}. \tag{41}$$

Proof. Iterating the recursion (40) backwards down to $n = 0$, we obtain

$$\begin{aligned}\mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,n+1}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}\right] =& \mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,0}\|^2_{\text{bvec}^{-1}\{(\boldsymbol{\mathcal{F}}^{\mathrm{T}})^{n+1}\boldsymbol{\sigma}\}}\right] \\ &+ \mu^2\boldsymbol{\gamma}^{\mathrm{T}}\left(\mathbf{I}_{D^2K^2} + \sum_{i=1}^{n}\left(\boldsymbol{\mathcal{F}}^{\mathrm{T}}\right)^i\right)\boldsymbol{\sigma},\end{aligned} \tag{42}$$

where $\tilde{\mathbf{h}}_{g,0} = \mathbf{h}_g - \mathbf{h}_{g,0}$. Note that under $\|\boldsymbol{\mathcal{F}}^{\mathrm{T}}\| = \|\boldsymbol{\mathcal{F}}\| < 1$, we will have $(\boldsymbol{\mathcal{F}}^{\mathrm{T}})^{n+1} \to \mathbf{0}_{D^2K^2}$ as $n \to \infty$. Hence, $\mathrm{E}[\|\tilde{\mathbf{h}}_{g,n}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}]$ attains a finite value. Therefore, a sufficient condition for convergence of $\mathrm{E}[\|\tilde{\mathbf{h}}_{g,n+1}\|^2_{\boldsymbol{\sigma}}]$ is then given by $\|\boldsymbol{\mathcal{F}}\| < 1$. To derive a convergence condition in terms of μ, we use the block maximum norm of the matrix $\boldsymbol{\mathcal{F}}$, i.e., $\|\boldsymbol{\mathcal{F}}\|_{\mathrm{b},\infty}$. From the properties of the block maximum norm [26], we can write

$$\|\boldsymbol{\mathcal{F}}\|_{\mathrm{b},\infty} = \|(\boldsymbol{\mathcal{A}} \otimes_{\mathrm{b}} \boldsymbol{\mathcal{A}})\boldsymbol{\mathcal{H}}\|_{\mathrm{b},\infty} \le \|(\boldsymbol{\mathcal{A}} \otimes_{\mathrm{b}} \boldsymbol{\mathcal{A}})\|_{\mathrm{b},\infty}\|\boldsymbol{\mathcal{H}}\|_{\mathrm{b},\infty}. \tag{43}$$

The term $(\boldsymbol{\mathcal{A}} \otimes_{\mathrm{b}} \boldsymbol{\mathcal{A}})$ can be written as $(\mathbf{A} \otimes \mathbf{A})^{\mathrm{T}} \otimes (\mathbf{I}_D \otimes \mathbf{I}_D)$. Again, from the properties of the block maximum norm, we have $\|\boldsymbol{\mathcal{A}} \otimes_{\mathrm{b}} \boldsymbol{\mathcal{A}}\|_{\mathrm{b},\infty} = \|(\mathbf{A} \otimes \mathbf{A})^{\mathrm{T}} \otimes \mathbf{I}_{D^2}\|_{\mathrm{b},\infty} = 1$. Now, substituting the definition of $\boldsymbol{\mathcal{H}}$ as given by (36), we have

$$\|\boldsymbol{\mathcal{F}}\|_{\mathrm{b},\infty} \le \|\mathbf{I}_{D^2K^2} - \mu(\mathbf{R}_z \otimes_{\mathrm{b}} \mathbf{I}_{DK}) - \mu(\mathbf{I}_{DK} \otimes_{\mathrm{b}} \mathbf{R}_z)\|_{\mathrm{b},\infty}. \tag{44}$$

Since the argument of the norm on the right-hand side of (44) is a block diagonal symmetric matrix, from the properties of block maximum norm, it is seen that $\mathrm{E}[\|\tilde{\mathbf{h}}_{g,n}\|^2_{\text{bvec}^{-1}\{\boldsymbol{\sigma}\}}]$ converges under

$$\rho(\|\mathbf{I}_{D^2K^2} - \mu(\mathbf{R}_z \otimes_{\mathrm{b}} \mathbf{I}_{DK}) - \mu(\mathbf{I}_{DK} \otimes_{\mathrm{b}} \mathbf{R}_z)\|_{\mathrm{b},\infty}) < 1, \tag{45}$$

or, equivalently,

$$|1 - \mu\lambda_p(\mathbf{R}_z) - \mu\lambda_q(\mathbf{R}_z)| < 1, \quad p, q \in \{1, 2, \ldots, DK\}. \tag{46}$$

Note that $(\mathbf{R}_z \otimes_{\mathrm{b}} \mathbf{I}_{DK})$ and $(\mathbf{I}_{DK} \otimes_{\mathrm{b}} \mathbf{R}_z)$ have the same set of eigenvectors and eigenvalues. Also, $\lambda_l(\mathbf{R}_z)$ has multiplicity of DK, for $l \in \mathcal{N}$. After solving the above condition, we obtain the mean-squared convergence condition on μ given in (41). □

Remark 5. We observe that the bounds established for μ are inversely proportional to the spectral radius of the covariance matrix of vectors $\mathbf{z}_k$. Hence, similar to conventional stochastic gradient algorithms, μ requires tuning according to the largest eigenmode.

C. Steady-State Mean-Squared Error

For μ under (41), letting $n \to \infty$ on both sides of (40), we have

$$\lim_{n\to\infty} \mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,n}\|^2_{\text{bvec}^{-1}\{(\mathbf{I}_{D^2K^2} - \boldsymbol{\mathcal{F}}^{\mathrm{T}})\boldsymbol{\sigma}\}}\right] = \mu^2\boldsymbol{\gamma}^{\mathrm{T}}\boldsymbol{\sigma}. \tag{47}$$

By selecting $\boldsymbol{\sigma} = (\mathbf{I}_{D^2K^2} - \boldsymbol{\mathcal{F}}^{\mathrm{T}})^{-1}\text{bvec}(\mathbf{R}_z)$, (47) becomes

$$\lim_{n\to\infty} \mathrm{E}\left[\|\tilde{\mathbf{h}}_{g,n}\|^2_{\mathbf{R}_z}\right] = \mu^2\boldsymbol{\gamma}^{\mathrm{T}}(\mathbf{I}_{D^2K^2} - \boldsymbol{\mathcal{F}}^{\mathrm{T}})^{-1}\text{bvec}(\mathbf{R}_z). \tag{48}$$

Using (48), the network-level steady-state mean-squared error (SMSE) of the proposed RFF-based GDKLMS is given by

$$\begin{aligned}\text{SMSE} &= \frac{1}{K}\lim_{n\to\infty}\mathrm{E}[\mathbf{e}_n^{\mathrm{T}}\mathbf{e}_n] \\ &= \frac{1}{K}\lim_{n\to\infty}\left[\mathrm{E}[\tilde{\mathbf{h}}_{g,n}^{\mathrm{T}}\boldsymbol{\mathcal{Z}}_n\boldsymbol{\mathcal{Z}}_n^{\mathrm{T}}\tilde{\mathbf{h}}_{g,n}] + \mathrm{E}[\boldsymbol{\upsilon}_n^{\mathrm{T}}\boldsymbol{\upsilon}_n]\right] \\ &= \frac{1}{K}\left[\lim_{n\to\infty}\mathrm{E}[\|\tilde{\mathbf{h}}_{g,n}\|^2_{\mathbf{R}_z}] + \lim_{n\to\infty}\mathrm{E}[\boldsymbol{\upsilon}_n^{\mathrm{T}}\boldsymbol{\upsilon}_n]\right] \\ &= \frac{1}{K}\left[\mu^2\boldsymbol{\gamma}^{\mathrm{T}}(\mathbf{I}_{D^2K^2} - \boldsymbol{\mathcal{F}}^{\mathrm{T}})^{-1}\text{bvec}(\mathbf{R}_z) + \text{tr}(\mathbf{R}_\upsilon)\right].\end{aligned} \tag{49}$$

V. Complexity Analysis

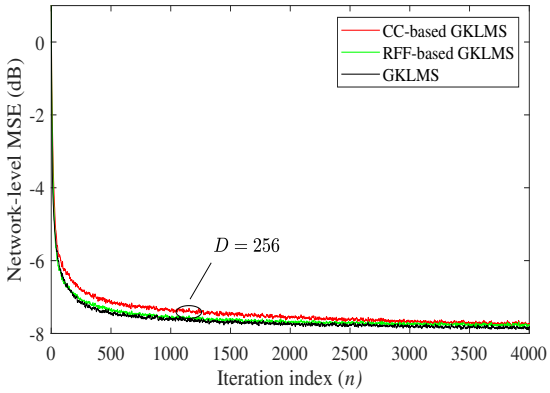
This section details the computational complexity of the proposed algorithms. For the GKLMS algorithm, the Gram matrix computation (10) requires a total of nK^2 kernel evaluations. The complexity of kernel evaluations is treated separately, as we do not consider a specific kernel function. The computational cost of (11) is $2nK^2 + nk$ multiplications and $2nK^2$ additions. These values reveal that kernel methods' complexity does not scale well with time and network size without using techniques to deal with the growing dictionary.

CC-based sparsification requires $K|\mathcal{D}|$ kernel evaluations per iterations for computing the Gram matrix, where $\mathcal{D}$ denotes the dictionary size, and $|\mathcal{D}|(2K+1)$ multiplications and $2K|\mathcal{D}|$ additions for the parameter update. The CC-based approach also requires dictionary training, and the minimum number of kernel evaluations for training is $|\mathcal{D}|(|\mathcal{D}|-1)/2$, assuming the first $|\mathcal{D}|$ regressors are added to the dictionary. An upper bound for the training process is $t|\mathcal{D}|$ kernel evaluations, where t is the number of training data samples.

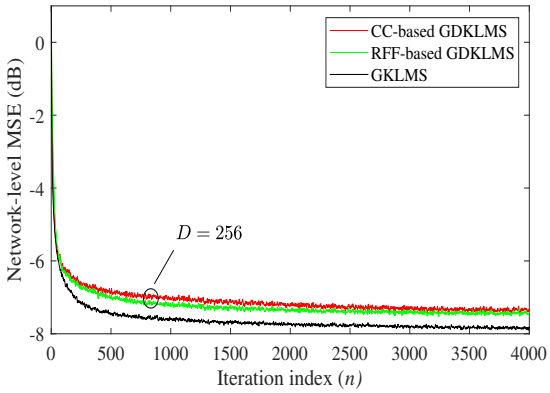
For the RFF-based computation, the mapping's complexity into RFF space is assumed similar to that of the kernel evaluation. In this case, the RFF-GKLMS requires KD kernel evaluations for the mapping, and $D(2K + 1)$ multiplications and $2KD$ additions for the update, where D denotes the dimension of the RFF space. Considering the case where $|\mathcal{D}|$ and D are the same for the CC- and RFF-based implementations, their complexities per iteration are also the same. The CC-based approach, however, has the added complexity of training the dictionary.

Finally, The GDKLMS using RFF requires, at each node, $D(|\mathcal{N}|+3)$ multiplications and $D(|\mathcal{N}|+1)$ additions, with $|\mathcal{N}|$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

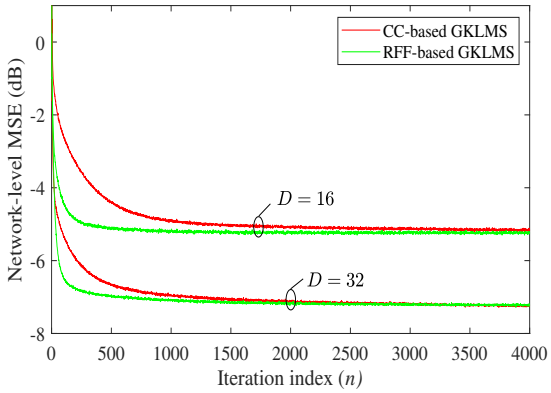


(a) Centralized solutions.

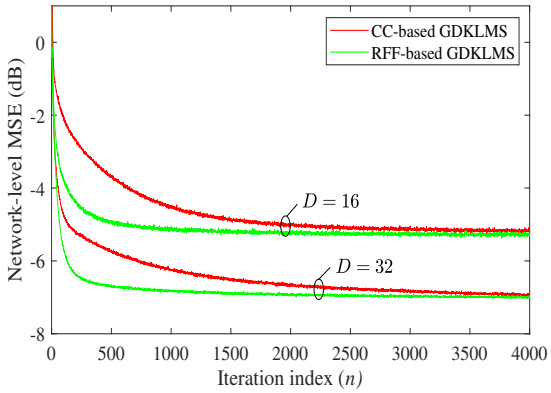


(b) Distributed solutions.

Fig. 1. Learning curves (network-level MSE vs iteration index) for the proposed algorithms with large dictionary size and RFF-space dimension.



(a) Centralized solutions.



(b) Distributed solutions.

Fig. 2. Learning curves (network-level MSE vs iteration index) for the proposed algorithms considering small values for D.

denoting the node's neighborhood cardinality. The mapping into the RFF space needs D kernel evaluations.

VI. Numerical Results

This section demonstrates the performance of the proposed algorithms through extensive numerical experiments under synthetic and real network data. This section demonstrates the performance of the proposed algorithms through extensive numerical experiments under synthetic and real network data. We exclude comparisons with state-of-the-art methods based on the linear model (3) because their performance in the considered setting will be poor. In all simulations, the value of δ is adjusted as a function of the target dictionary size, such that we can reach the target size while still having a representative dictionary.

A. Nonlinear Graph Filter Identification

First, we consider a connected Erdös-Renyi graph comprising $K = 20$ nodes with edge probability equal to 0.2. The shift matrix $\mathbf{S}$ is constructed as follows: first, the existing edges, according to the previously constructed graph, receive a weight value drawn from the uniform distribution in the interval $(0, 1]$; each entry s_{kl} receives the value of the corresponding edge weight or zero if the edge does not exist; the eigenvalues $\{\lambda_k\}_{k=1}^K$ of $\mathbf{S}$ are normalized by the largest eigenvalue such that $|\lambda_k| \leq 1$. Input signal $\mathbf{x}_n$ and observation noise $\boldsymbol{\upsilon}_n$ are drawn from zero-mean normal distributions with covariance matrices $\mathbf{R}_{\mathbf{x}} = \text{diag}\{\sigma_{x,k}^2\}$ and $\mathbf{R}_{\boldsymbol{\upsilon}} = \text{diag}\{\sigma_{\upsilon,k}^2\}$, respectively, where $\sigma_{x,k}^2$ are drawn from the uniform distribution in $[1, 1.5]$ and $\sigma_{\upsilon,k}^2$ from $[0.1, 0.15]$. For distributed implementations, the combination coefficients a_{kl} are computed according to the Metropolis rule [26]. We used a Gaussian kernel with $\sigma^2 = 1$. For a filter of length $L = 4$, we aim at estimating the time-invariant nonlinear function given by

$$f(\mathbf{r}_{k,n}) = \sqrt{r_{k,n,1}^2 + \sin^2(r_{k,n,4}\pi)} + (0.8 - 0.5\exp(-r_{k,n,2}^2)r_{k,n,3}. \tag{50}$$

The network-level instantaneous MSE, given by $\text{MSE}_n = \frac{1}{K}\sum_{k=1}^K e_{k,n}^2$, is considered as the performance metric and results are displayed by plotting MSE_n versus the iteration index n, averaging over 1000 independent runs.

In Fig. 1a, we present the learning curves of the centralized approaches based on CC and RFF. We limit the size of the dictionary and set the dimension of the RFF space to $D = 256$. Results show that, for large enough dictionary sizes and RFF-space dimensions, these implementations are able to reach similar performance to that of the GKLMS implementation without sparsification methods. In Fig 1b, we show similar results comparing the CC- and RFF-based GDKLMS against the GKLMS without sparsification. For the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

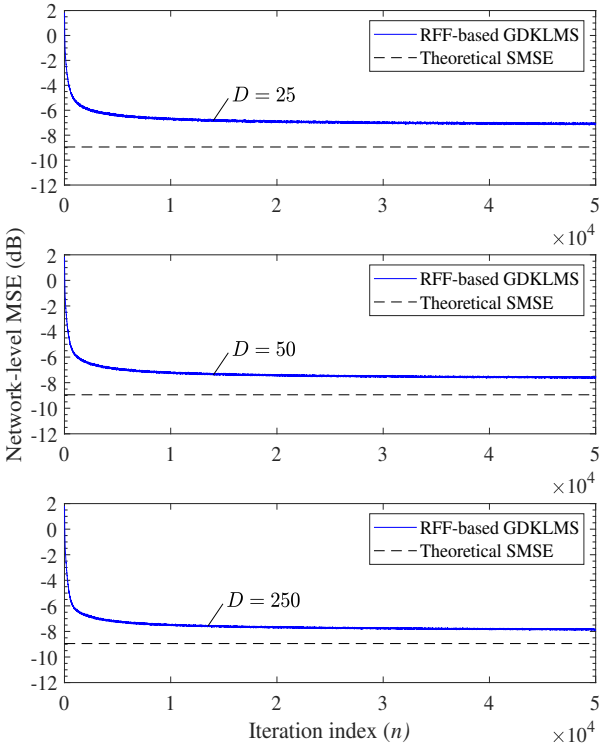
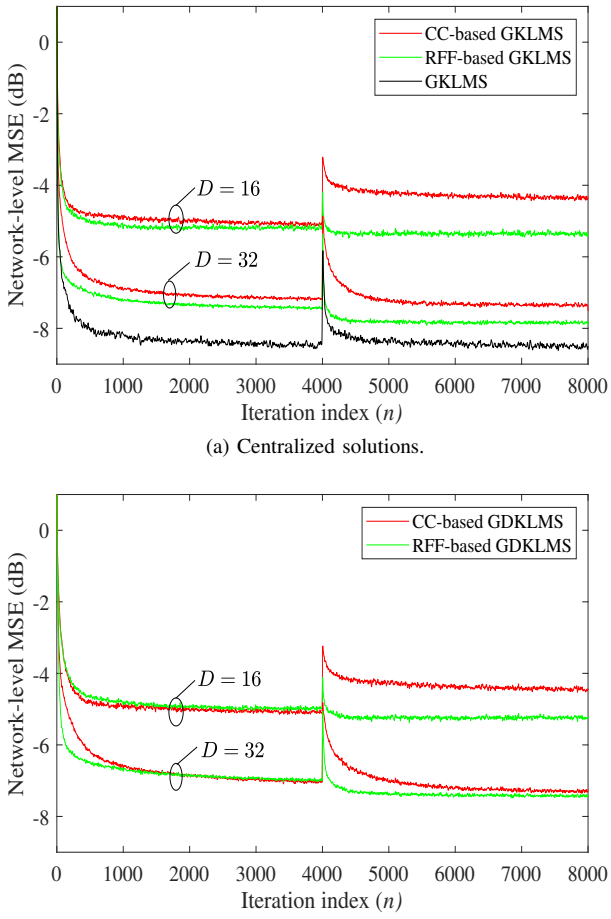


Fig. 3. Learning curves for the RFF-based GDKLMS with different values of D and the theoretical steady-state MSE values.



(a) Centralized solutions.

(b) Distributed solutions.

Fig. 4. Tracking performance of the proposed algorithms.

CC-based GDKLMS, we pre-train the dictionary before the learning process. The centralized implementations can better approximate the GKLMS without sparsification when compared to the GDKLMS. This is an expected result considering that data from the entire graph is available during the learning process of the centralized approaches.

In Fig. 2a we compare the proposed algorithms when smaller dictionaries and RFF-space dimensions are considered. Specifically, we compare the implementations based on RFF and coherence check against each other. For this purpose, we adjust the step-size μ and assess the convergence speed as the learning curves for both implementations achieve similar values of network-level steady-state MSE. Again, the value $D \in \{16, 32\}$ represents both the dimension of the RFF space and the size of the pre-trained dictionary for the coherence check approach. Results show that both CC- and RFF-based algorithms are capable of effectively representing the target function. Fig. 2a also shows that, for the same value of D and for similar values of network-level steady-state MSE, the RFF-based GKLMS converges faster than the CC-based one. Moreover, it can be observed that the performance of the implementations with fixed-size dictionaries greatly improves as D is increased from 16 to 32.

Fig. 2b shows the results for the distributed GDKLMS using CC and RFF. Similar to the centralized case, the plots show that both approaches can effectively represent the target function, achieving network-level MSE of approximately -5 dB for $D = 16$ and -7 dB for $D = 32$ for the noise scenario simulated. Again, the RFF-based solution exhibits faster convergence for both values of D when the network-level steady-state MSE is matched.

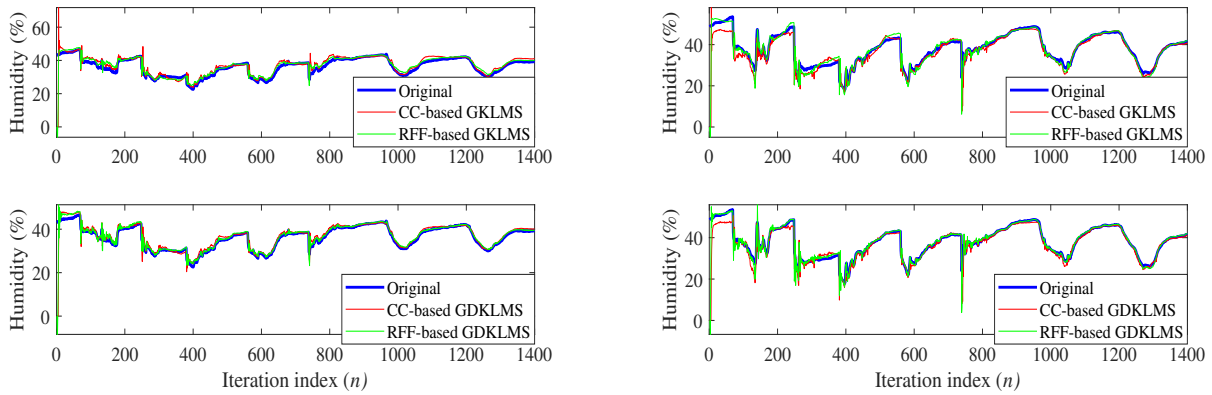
B. SMSE of the RFF-based GDKLMS

In this experiment, we observe the steady-state behavior of the proposed RFF-based GDKLMS. The network and data parameters employed in this simulation are the same used in Section VI-A. We run the RFF-based GDKLMS for a total of $T = 50000$ iterations, for different dimensions of the RFF space. In Fig. 3, we show the learning curves for $D \in \{25, 50, 250\}$ and the value of the SMSE computed using (49). The step-size is $\mu = 0.05$. Results show that increasing D reduces the gap between the numerical and theoretical results for the steady-state behavior of the algorithm. This observation is in line with the result presented in [47].

C. Tracking Performance of the Proposed Algorithms

In this section we study the performance of the algorithms subjected to an abrupt change in the underlying model. The

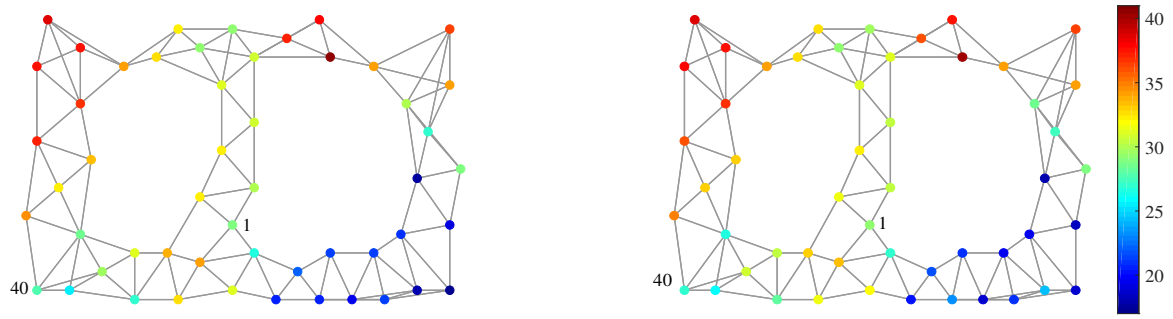
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks



(a) Time series for Sensor 1.

(b) Time series for Sensor 40.

Fig. 5. Time series of original and estimated humidity signals using the proposed algorithms for the Intel Lab dataset.



(a) Original.

(b) Estimated (RFF-based GDKLMS).

Fig. 6. Network structure for the Intel Lab simulation and snapshots of the original and estimated humidity signals.

simulation setup is the same as in Section VI-A. The nonlinear function to be estimated is given by

$$f_n(\mathbf{r}_{k,n}) = \begin{cases} \sqrt{r_{k,n,1}^2 + r_{k,n,4}^2} - r_{k,n,3}\mathrm{e}^{-r_{k,n,2}^2} & 0 < n \leq 4000 \\ \sqrt{r_{k,n,1}^2 + r_{k,n,2}^2 + r_{k,n,3}^2 + r_{k,n,4}^2} & 4000 < n. \end{cases} \tag{51}$$

Fig. 4 shows the learning curves for the centralized and distributed algorithms for two values of dictionary sizes and RFF-space dimension, namely, $D = 16$ and $D = 32$. We see that the RFF-based implementations are resilient to model changes, while the CC-based implementations suffer from noticeable performance losses, especially for small dictionaries. This is an expected behavior, since larger dictionaries can represent more functions. We also see that the GKLMS achieves the lowest MSE, however, at the cost of an unconstrained dictionary size.

D. Laboratory-monitoring Data

We consider the Intel Lab database [68] that contains temperature and humidity data, measured during March 2004, from 52 sensors spread across a laboratory and its common areas. The undirected graph is constructed by connecting each sensor to its four nearest neighbors. We consider the task of estimating humidity from the temperature signal. The data set comprises asynchronous sensor measurements, and we construct a snapshot of the graph signal by considering windows of 5 minutes from which we collect the first value available for each sensor. The model from temperature to humidity is expected to change with time. For instance, as workers arrive in the lab, the temperature and humidity are expected to change.

In our simulations, we used $L = 5$ and $D = 128$, for centralized and distributed implementations. The step sizes are 0.03 for CC- and RFF-based GKLMS, and 0.5 for GDKLMS implementations.

The humidity signals from Sensors 1 and 40 are plotted in Figs. 5a and 5b, respectively, together with the estimated signals from the graph filters. The variations in the plots are aligned with events that induce model changes. For example, the most notable peaks are aligned with the beginning and end of work shifts. The implementations based on CC and RFF have similar performances, while the latter exhibit slightly more resilience to changes in the model. Fig. 6 depicts the graph representation of the Intel Lab sensor network and presents snapshots of the humidity signals, both the original and the one estimated via RFF-based GDKLMS. These results confirm that the proposed algorithms can effectively estimate the humidity level from temperature readings.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSIPN.2020.3046217, IEEE Transactions on Signal and Information Processing over Networks

VII. Conclusion

This paper introduced nonlinear adaptive graph filters for model estimation in the reproducing kernel Hilbert space. To this end, a centralized graph kernel LMS (GKLMS) algorithm was derived. To overcome the growing dimension problem encountered in the centralized GKLMS algorithm, coherence check based dictionary-sparsification and random Fourier features (RFF) were proposed. Furthermore, diffusion-based distributed implementations of both coherence check and RFF-based graph KLMS algorithms were developed that update filter parameters through local communications and in-network processing. Mean and mean-square-error convergence conditions were established for the proposed GDKLMS using RFF. Numerical simulations were conducted to demonstrate the performance of the proposed algorithms. Simulations confirmed that coherence check and RFF-based approaches effectively estimate nonlinear graph filters, while the latter exhibits a faster convergence and is robust to model changes.

References

[1] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, pp. 83–98, May 2013.

[2] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, pp. 1644–1656, Apr. 2013.

[3] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, pp. 808–828, May 2018.

[4] E. Ceci and S. Barbarossa, "Graph signal processing in the presence of topology uncertainties," *IEEE Trans. Signal Process.*, 2020.

[5] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, pp. 80–90, Sep. 2014.

[6] I. Jablonski, "Graph signal processing in applications to sensor networks, smart grids, and smart cities," *IEEE Sensors J.*, vol. 17, pp. 7659–7666, Dec. 2017.

[7] A. Gavili and X. Zhang, "On the shift operator, graph frequency, and optimal filtering in graph signal processing," *IEEE Trans. Signal Process.*, vol. 65, pp. 6303–6318, Dec. 2017

[8] O. Teke, and P. Vaidyanathan, "Extending Classical Multirate Signal Processing Theory to Graphs - Part I: Fundamentals," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 409-422, Jan. 2017.

[9] S. K. Narang and A. Ortega, "Perfect Reconstruction Two-Channel Wavelet Filter Banks for Graph Structured Data," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2786-2799, June 2012.

[10] S. Chen, R. Varma, A. Sandryhaila and J. Kovačević, "Discrete Signal Processing on Graphs: Sampling Theory," *IEEE Trans. Signal Process.*, vol. 63, no. 24, pp. 6510-6523, Dec. 2015.

[11] J. Mei and J. M. F. Moura, "Signal processing on graphs: Causal modeling of unstructured data," *IEEE Trans. Signal Process.*, vol. 65, pp. 2077–2092, Apr. 2017.

[12] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Graph filters," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 6163–6166.

[13] E. Isufi, G. Leus and P. Banelli, "2-Dimensional finite impulse response graph-temporal filters," in *Proc. IEEE Global Conf. Signal Inf. Process.*, 2016, pp. 405–409.

[14] F. Hua, C. Richard, J. Chen, H. Wang, P. Borgnat and P. Gonçalves, "Learning Combination of Graph Filters for Graph Signal Modeling," *IEEE Signal Process. Lett.*, vol. 26, no. 12, pp. 1912-1916, Dec. 2019.

[15] A. Loukas, A. Simonetto, and G. Leus, "Distributed autoregressive moving average graph filters," *IEEE Signal Process. Lett.*, vol. 22, pp. 1931–1935, Nov. 2015.

[16] E. Isufi, A. Loukas, N. Perraudin and G. Leus, "Forecasting time series with VARMA recursions on graphs," *IEEE Trans. Signal Process.*, vol. 67, no. 18, pp. 4870-4885, Sep. 2019.

[17] X. Mao, K. Qiu, T. Li and Y. Gu, "Spatio-Temporal Signal Recovery Based on Low Rank and Differential Smoothness," *IEEE Trans. Signal Process.*, vol. 66, no. 23, pp. 6281-6296, Dec, 2018.

[18] K. Qiu, X. Mao, X. Shen, X. Wang, T. Li and Y. Gu, "Time-Varying Graph Signal Reconstruction," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 6, pp. 870-883, Sep. 2017.

[19] M. J. M. Spelta and W. A. Martins, "Normalized LMS algorithm and data-selective strategies for adaptive graph signal estimation," *Signal Process.*, vol. 167, 107326, Feb. 2020.

[20] F. Hua, R. Nassif, C. Richard, H. Wang and A. H. Sayed, "Diffusion LMS With Communication Delays: Stability and Performance Analysis," *IEEE Signal Process. Lett.*, vol. 27, pp. 730-734, 2020.

[21] P. Di Lorenzo, P. Banelli, S. Barbarossa, and S. Sardellitti, "Distributed adaptive learning of graph signals," *IEEE Trans. Signal Process.*, vol. 65, pp. 4193–4208, Aug. 2017.

[22] R. Nassif, C. Richard, J. Chen, and A. H. Sayed, "A graph diffusion LMS strategy for adaptive graph signal processing," in *Conf. Rec. Asilomar Conf. Signals Syst. Comput.*, pp. 1973–1976, Oct. 2017.

[23] R. Nassif, C. Richard, J. Chen, and A. H. Sayed, "Distributed diffusion adaptation over graph signals," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 4129–4133.

[24] F. Hua, R. Nassif, C. Richard, H. Wang and A. H. Sayed, "Online distributed learning over graphs with multitask graph-filter models," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 6, pp. 63–77, Jan. 2020.

[25] R. Nassif, S. Vlaski, C. Richard, J. Chen and A. H. Sayed, "Multitask Learning Over Graphs: An Approach for Distributed, Streaming Machine Learning," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 14-25, May 2020.

[26] A. H. Sayed, "Adaptation, learning, and optimization over networks," in *Foundations and Trends® in Mach. Learn.*, vol. 7, pp. 311–801, 2014.

[27] A. H. Sayed, "Diffusion Adaptation Over Networks," in *Acad. Press Libr. Signal Process.*, vol. 3, pp. 323–531, 2014.

[28] M. Paluš, "Detecting nonlinearity in multivariate time series," *Phys. Lett. A*, vol. 213, no. 3–4, pp. 138-147, Apr. 1996.

[29] J. F. Walker, N. Jenkins and N. Jenkins *Wind Energy Technology*. Wiley, June 1997.

[30] M. O. Franz and B. Schölkopf, "A unifying view of Wiener and Volterra theory and polynomial kernel regression," *Neural Comput.*, vol. 18, pp. 3097–3118, Dec. 2006.

[31] D. Comminiello and J. C. Príncipe. *Adaptive Learning Methods for Nonlinear System Modeling*. Elsevier, 2018.

[32] V. J. Mathews, "Adaptive polynomial filters," *IEEE Signal Process. Mag.*, vol. 8, no. 3, pp. 10–26, July 1991.

[33] Taiho Koh and E. Powers, "Second-order Volterra filtering and its application to nonlinear system identification," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 33, no. 6, pp. 1445–1455, Dec. 1985.

[34] D. J. Sebald and J. A. Bucklew, "Support vector machine techniques for nonlinear equalization," *IEEE Trans. Signal Process.*, vol. 48, no. 11, pp. 3217–3226, Nov. 2000.

[35] M. Scarpiniti, D. Comminiello, G. Scarano, R. Parisi and A. Uncini, "Steady-state performance of spline adaptive filters," *IEEE Trans. Signal Process.*, vol. 64, no. 4, pp. 816–828, Feb. 2016.

[36] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning Theory and Kernel Machines. Lecture Notes in Computer Science*, vol. 2777, pp. 144–158, Springer, 2003.

[37] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering*. Wiley, Feb. 2010.

[38] 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, June, 2014.

[39] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Process. Mag.*, vol. 28, pp. 97–123, Jan. 2011.

[40] W. Gao, J. Chen, C. Richard, and J. Huang, "Diffusion adaptation over networks with kernel least-mean-square," in *Proc. IEEE Int. Workshop Comput. Adv. Multisens. Adapt. Process.*, 2015, pp. 217–220.

[41] S. Chouvardas and M. Draief, "A diffusion kernel LMS algorithm for nonlinear adaptive networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 4164–4168.

[42] D. Romero, M. Ma, and G. B. Giannakis, "Kernel-based reconstruction of graph signals," *IEEE Trans. Signal Process.*, vol. 65, pp. 764–778, Feb. 2017.

[43] D. Romero, V. N. Ioannidis and G. B. Giannakis, "Kernel-based reconstruction of space-time functions on dynamic graphs," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 6, pp. 856–869, Sep. 2017.

[44] V. N. Ioannidis, D. Romero and G. B. Giannakis, "Inference of Spatio-Temporal Functions Over Graphs via Multikernel Kriged Kalman Filtering," *IEEE Trans. Signal Process.*, vol. 66, no. 12, pp. 3228-3239, June 2018.

[45] B.-S. Shin, M. Yukawa, R. L. G. Cavalcante, and A. Dekorsy, "Distributed adaptive learning with multiple kernels in diffusion networks," *IEEE Trans. Signal Process.*, vol. 66, pp. 5505–5519, Nov. 2018.

[46] A. Venkitaraman, S. Chatterjee and P. Händel, "Predicting graph signals using kernel regression where the input signal is agnostic to a graph," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 5, no. 4, pp. 698–710, Dec. 2019.

[47] 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, 2018.

[48] Y. Shen, G. Leus, and G. B. Giannakis, "Online graph-adaptive learning with scalability and privacy," *IEEE Trans. Signal Process.*, vol. 67, pp. 2471–2483, May 2019.

[49] S. Wang, L. Dang, B. Chen, S. Duan, L. Wang, and C. K. Tse, "Random Fourier filters under maximum correntropy criterion," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, pp. 3390–3403, Oct. 2018.

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

[51] K. Chen, S. Werner, A. Kuh, and Y.-F. Huang, "Nonlinear adaptive filtering with kernel set-membership approach," *IEEE Trans. Signal Process.*, pp. 1–1, 2020.

[52] C. Richard, J. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Process.*, vol. 57, pp. 1058–1067, Mar. 2009.

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

[54] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, May 1950.

[55] P. P. Pokharel, W. Liu and J. C. Príncipe, "Kernel least mean square algorithm with constrained growth", *Signal Process.*, vol. 89, no. 3, pp. 257–265 Mar. 2009.

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

[57] 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.

[58] V. G. Gogineni, V. R. M. Elias, W. A. Martins and S. Werner, "Graph diffusion kernel LMS using random Fourier Features," in *Conf. Rec. Asilomar Conf. Signals Syst. Comput.*, pp. 1–5, Nov. 2020.

[59] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. Adv. Neural Inf. Process. Syst.*, pp. 1177–1184, 2007.

[60] X. Dong, D. Thanou, P. Frossard and P. Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160-6173, Dec.1, 2016

[61] S. Segarra, A. G. Marques, G. Mateos and A. Ribeiro, "Network topology inference from spectral templates," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 467-483, Sep. 2017.

[62] G. B. Giannakis, Y. Shen and G. V. Karanikolas, "Topology identification and learning over graphs: accounting for nonlinearities and dynamics," *Proc. IEEE*, vol. 106, no. 5, pp. 787-807, May 2018.

[63] G. Mateos, S. Segarra, A. G. Marques and A. Ribeiro, "Connecting the dots: identifying network structure via graph signal processing," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 16-43, May 2019.

[64] M. Moscu, R. Nassif, F. Hua and C. Richard, "Learning Causal Networks Topology From Streaming Graph Signals," *Proc. Eur. Signal Process. Conf.*, 2019, pp. 1-5.

[65] A. H. Sayed, S. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, pp. 155–171, May 2013.

[66] R. H. Koning, H. Neudecker, and T. Wansbeek, "Block Kronecker products and the vecb operator," *Linear Algebra and Its Applications*, vol. 149(C), 165–184, 1991.

[67] A. H. Sayed, "Adaptive networks," *Proc. IEEE*, vol. 102, pp. 460–497, Apr. 2014.

[68] "Intel Lab Data." http://db.csail.mit.edu/labdata/labdata.html. Accessed: 2019-11-28.