Graph Diffusion Kernel LMS using Random Fourier Features

Vinay Chakravarthi Gogineni [†], Vitor R. M. Elias[*‡], Wallace A. Martins[‡§], Stefan Werner[*]
[†] Department of Machine Intelligence, SimulaMet, Simula Research Laboratory-Oslo, Norway
[*] Department of Electronic Systems, NTNU – Norwegian University of Science and Technology, Trondheim
[‡] Electrical Engineering Program, UFRJ – Federal University of Rio de Janeiro
[§] Interdisciplinary Centre for Security, Reliability and Trust, UniLu – University of Luxembourg

Abstract—**This work introduces kernel adaptive graph filters that operate in the reproducing kernel Hilbert space. We propose a centralized graph kernel least mean squares (GKLMS) approach for identifying the nonlinear graph filters. The principles of coherence-check and random Fourier features (RFF) are used to reduce the dictionary size. Additionally, we leverage the graph structure to derive the graph diffusion KLMS (GDKLMS). The proposed GDKLMS requires only single-hop communication during successive time instants, making it viable for real-time network-based applications. In the distributed implementation, usage of RFF avoids the requirement of a centralized pre-trained dictionary in the case of coherence-check. Finally, the performance of the proposed algorithms is demonstrated in modeling a nonlinear graph filter via numerical examples. The results show that centralized and distributed implementations effectively model the nonlinear graph filters, whereas the random-feature-based solutions are shown to outperform coherence-check based solutions.**

I. Introduction

Recently, graph signal processing (GSP) has received increased attention due to its wide applicability to model, process, and analyze network signals and large data sets [1]–[4]. For instance, in the context of a wireless sensor network, graph nodes and edges represent sensors and communication links, respectively. Similar to conventional digital signal processing (DSP) techniques, the basic building block in GSP is the graph-shift operation, which captures node interconnections. 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. The development of tools for GSP has been extensively studied over the last few years [1]–[9].

A key area of GSP research is to model the unknown relations between input and output graph signals through a filter [1], [3], [7], [8], [10]. The application of linear shift-invariant filter models is widely employed in the literature, e.g., to design graph spectral filters [7], [10] and model dynamic graph signals [8], [9]. More recently, several works deal with adaptive learning of graph filters, see, e.g., [11]–[15]. However, linear models cannot accurately model many real-world systems that exhibit more sophisticated input-output relations. Prominent examples include the relations between air pressure and temperature, and wind speed and generated power in wind turbines [16], [17].

In this work, we introduce the concept of nonlinear adaptive filtering of graph signals. Adaptive filtering in reproducing kernel Hilbert spaces (RKHS) has proven to be an effective method for modeling nonlinear relations [18]–[28]. Therefore, drawing upon the ideas of kernel methods, we propose graph kernel adaptive filters that effectively capture the nonlinear input-output relations of graph signals. We first derive the centralized graph kernel least mean square (GKLMS) for nonlinear graph system identification. To tackle the growing dimension problem in GKLMS, we first consider a coherence-check approach to construct a fixed-size dictionary. However, this approach requires a centralized pre-trained dictionary and, therefore, does not render an efficient distributed implementation. To overcome this issue, using random Fourier features [29], we propose centralized GKLMS in RFF space. By extending the concepts of distributed learning over networks [12], [25], [30]–[32], we also propose a graph diffusion KLMS (GDKLMS) using RFF that solely depends on local information exchange. Furthermore, we establish the conditions for the mean convergence of the proposed RFF based GDKLMS. Finally, numerical experiments are conducted to demonstrate the performance of the proposed algorithms.

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 and only if nodes k and l are connected. The graph is equipped with the graph shift operator, defined by a symmetric matrix $\mathbf{S} \in \mathbb{R}^{K \times K}$ whose entries $[\mathbf{S}]_{k,l} = s_{kl}$ take non-zero values only if $(k, l) \in \mathcal{E}$ [1], [2]. The graph Laplacian matrix [1] and the graph adjacency matrix [2] are the common choices for $\mathbf{S}$. At time-index n, a graph signal is defined by the mapping $x(n) : \mathcal{N} \rightarrow \mathbb{R}$ and represented by a vector $\mathbf{x}(n) = [x_1(n)\, x_2(n) \ldots x_K(n)]^{\mathrm{T}}$, where $x_k(n)$ represents the signal value at the kth node. 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.

This work was partly supported by the 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, FAPERJ, and the Research Council of Norway.

A length-L linear shift-invariant (LSI) graph filter linearly combines these shifted versions of a graph signal and yields an output $\mathbf{y}(n) = \sum_{i=0}^{L-1} h_i \mathbf{S}^i \mathbf{x}(n-i)$, for $n \geq L-1$, where $h_0, h_1, \ldots, h_{L-1}$ are the coefficients of the graph filter [12]. This model embeds time-evolution and is an alternative to the initial linear graph filter designs [10]–[12]. By retaining the samples $\{x_k(n), [\mathbf{S}\mathbf{x}(n-1)]_k, \ldots, [\mathbf{S}^{L-1}\mathbf{x}(n-L+1)]_k\}$, only one graph shift operation needs to be performed at each time-instant n, which makes this model suitable for real-time applications. For this model, linear graph diffusion LMS strategies have been proposed in [12] for adaptive graph signal processing.

However, in many real-world scenarios, the limited capabilities of linear models fail to represent systems with more sophisticated input-output relations reasonably [18]. This limitation on linear models is discussed in many problems such as channel regression and time-series prediction [18], [26], [28]. Adopting a nonlinear model proved to be effective when tackling this class of problems. In this context, at every node k, we assume a nonlinear relation between the input and output as given below:

$$y_k(n) = f(\mathbf{r}_k(n)) + \upsilon_k(n), \tag{1}$$

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

$$\mathbf{r}_k(n) = [x_k(n) \; [\mathbf{S}\mathbf{x}(n-1)]_k \; \ldots \; [\mathbf{S}^{L-1}\mathbf{x}(n-L+1)]_k]^{\mathrm{T}}. \tag{2}$$

Here, the goal is to estimate the function $f(\cdot)$ at each node k given a set of data pairs $\{\mathbf{r}_k(i), y_k(i)\}$ for $i \in \{1, 2, \ldots, n\}$; this refers to a nonlinear system identification task. While a linear filter can be uniquely defined by its coefficients $h_0, h_1, \ldots, h_{L-1}$, the characterization of a nonlinear filter admits a range of approaches. Several methods exist in literature to estimate the non-linear functions in an adaptive fashion [18], [28], [33]. Of these, kernel methods take a linear form in high-dimensional feature space and, thus, gained much popularity in modeling the nonlinear input-output relations [18]–[25], [27]. Thus, we characterize the nonlinear relations on graphs as graph kernel adaptive filters.

III. Graph Kernel Filters

In order to estimate the nonlinear function $f(\cdot)$ in (1), kernel methods first map the input regressor $\mathbf{r}_k(i) \in \mathbb{R}^L$ into a high-dimensional feature space where $f(\cdot)$ takes a linear form [18], [26]. This mapping is denoted by $\kappa(\cdot, \mathbf{r}_k(i))$, where $\kappa(\cdot,\cdot)$ is a reproducing kernel. The reproducing kernel $\kappa(\cdot,\cdot) : \mathbb{R}^L \times \mathbb{R}^L \to \mathbb{R}$ satisfies [18]

$$\kappa(\mathbf{r}_k(n), \mathbf{r}_k(i)) = \langle \kappa(\cdot, \mathbf{r}_k(n)), \kappa(\cdot, \mathbf{r}_k(i)) \rangle_{\mathcal{H}}, \tag{3}$$

where $\mathcal{H}$ is the induced RKHS and $\langle \cdot,\cdot \rangle_{\mathcal{H}}$ denotes the corresponding inner product. In (3), $\kappa(\cdot, \mathbf{r}_k(i))$ is a representer evaluation at $\mathbf{r}_k(i)$ [27], [28].

A. Centralized Graph Kernel LMS

In the graph setting, at every time-instant n, K new data samples are available. Then, at time-instant n, using the representer theorem [20], the estimate of $y_l(n)$ (i.e., $\hat{y}_l(n)$), given data pairs $\{\mathbf{r}_k(i), y_k(i)\}_{i=1,k=1}^{n-1,K} \bigcup \{\mathbf{r}_k(n)\}_{k=1}^{K}$, can be expressed as

$$\hat{y}_l(n) = f(\mathbf{r}_l(n)) = \sum_{i=1}^{n} \sum_{k=1}^{K} \alpha_{ik} \, \kappa(\mathbf{r}_l(n), \mathbf{r}_k(i)). \tag{4}$$

The model in (4) grows with both time, n, and network size, K. This is a well-known issue with single-node kernel methods [19], [26]–[28], [34]–[36], where several solutions have been proposed that learn a sparse, or fixed-size dictionary. Of these, the coherence-check (CC) methods use a coherence metric [19], [28] between a candidate regressor and the current dictionary to decide whether to include the candidate in the dictionary. Using coherence-check criterion, $\hat{y}_l(n)$ can be written as

$$\hat{y}_l(n) = f(\mathbf{r}_l(n)) = \sum_{i \in \mathcal{M}(n)} \sum_{k \in \mathcal{K}(i)} \alpha_{ik} \, \kappa\big(\mathbf{r}_l(n), \mathbf{r}_k(i)\big), \tag{5}$$

where $\mathcal{M}(n)$ is a set of time instants in which at least one input regressor is added to the dictionary until time-index n, with $|\mathcal{M}(n)| \leq n$, and $\mathcal{K}(i)$ is a set of node indices of regressors that passed the coherence-check at time-index i, with $|\mathcal{K}(i)| \leq K$. Under the coherence-check 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 reasonable conditions on the coherence-metric threshold, the maximum number of regressors in the dictionary is finite, i.e., $|\mathcal{D}|$ stops increasing after a certain time [28].

The coefficients of the function expansion in (5) are obtained through the following minimization problem

$$\begin{aligned}\min_{\alpha_{ik}} \sum_{l=1}^{K} \mathbb{E}\Big[\Big(y_l(n) - \sum_{i \in \mathcal{M}(n)} \sum_{k \in \mathcal{K}(i)} \alpha_{ik} \, \kappa\big(\mathbf{r}_l(n), \mathbf{r}_k(i)\big)\Big)^2\Big] \\ = \min_{\boldsymbol{\alpha} \in \mathbb{R}^{|\mathcal{D}(n)|}} \mathbb{E}\left[\|\mathbf{y}(n) - \mathbf{K}(n)\boldsymbol{\alpha}\|_2^2\right],\end{aligned} \tag{6}$$

where $\boldsymbol{\alpha} \triangleq [\boldsymbol{\alpha}_1^{\mathrm{T}} \, \boldsymbol{\alpha}_2^{\mathrm{T}} \, \ldots \, \boldsymbol{\alpha}_{|\mathcal{M}(n)|}^{\mathrm{T}}]^{\mathrm{T}}$, with $\boldsymbol{\alpha}_i^{\mathrm{T}} = [\alpha_{i1} \, \alpha_{i2} \, \ldots \, \alpha_{i|\mathcal{K}(i)|}] \in \mathbb{R}^{|\mathcal{K}(i)|}$ and $\mathbf{K}(n) = [\mathbf{K}_1(n) \, \mathbf{K}_2(n) \, \ldots \, \mathbf{K}_{|\mathcal{M}(n)|}] \in \mathbb{R}^{K \times |\mathcal{D}(n)|}$, with $[\mathbf{K}_i(n)]_{lk} = \kappa(\mathbf{r}_l(n), \mathbf{r}_k(i))$, for $l \in \mathcal{N}$ and $k \in \mathcal{K}(i)$.

Considering the growing nature of the dictionary, access to second-order statistics is impractical. Therefore, we use a stochastic-gradient approach and minimize the instantaneous value of (6) recursively. The update equation for the graph KLMS (GKLMS) is given by

$$\boldsymbol{\alpha}(n+1) = \boldsymbol{\alpha}(n) + \mu \, \mathbf{K}^{\mathrm{T}}(n) \, \big(\mathbf{y}(n) - \mathbf{K}(n)\boldsymbol{\alpha}(n)\big), \tag{7}$$

where μ is a positive adaptation step size.

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 coherence-check criteria, the dictionary remains the same. Otherwise, $\mathbf{K}(n)$ gets one new column and a zero-valued entry must be appended to $\boldsymbol{\alpha}(n)$ [28]. At every time instant i, $|\mathcal{K}(i)|$ regressors are added to $\mathcal{D}$. Hence, $|\mathcal{K}(i)|$ zeros must be appended to $\boldsymbol{\alpha}(n)$.

B. Centralized GKLMS using RFF

An alternative to sparsification methods is provided by random Fourier features (RFF) [29]. RFF are used to approximate the evaluation of a shift-invariant kernel $\kappa(\mathbf{r}(n), \mathbf{r}(i)) = \kappa(\mathbf{r}(n) - \mathbf{r}(i))$ as an inner-product in the D-dimensional RFF space. This turns the problem into a finite-dimension linear problem while removing the need to evaluate kernel functions [29]. Let $\mathbf{z}_k(n)$ be the mapping of $\mathbf{r}_k(n)$ into the RFF space $\mathbb{R}^D$. Then, the kernel evaluation can be approximated as $\kappa(\mathbf{r}_l(n), \mathbf{r}_k(i)) \approx \mathbf{z}_k^{\mathrm{T}}(i)\mathbf{z}_l(n)$, and the estimate $\hat{y}_l(n)$ in (4) can be approximated by

$$\hat{y}_l(n) \approx \Big(\sum_{i=1}^{n}\sum_{k=1}^{K} \alpha_{ik}\, \mathbf{z}_k(i)\Big)^{\mathrm{T}} \mathbf{z}_l(n) = \mathbf{h}^{\mathrm{T}}\mathbf{z}_l(n), \qquad (8)$$

where $\mathbf{h} \in \mathbb{R}^D$ is the representation of the function $f(\cdot)$ in the RFF space. Let the matrix $\mathbf{Z}(n) = [\mathbf{z}_1(n)\, \mathbf{z}_2(n) \,\ldots\, \mathbf{z}_K(n)]$ describe the RFF mapping of all input vectors at time n. Now, the optimization problem becomes

$$\mathbf{h}^*(n) = \arg\min_{\mathbf{h}\in\mathbb{R}^D} \mathbb{E}\left[\|\mathbf{y}(n) - \mathbf{Z}^{\mathrm{T}}(n)\mathbf{h}\|_2^2\right]. \qquad (9)$$

Similar to (7), approximating the solution through stochastic gradient descent iterations yields the update rule

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu\mathbf{Z}(n)\mathbf{e}(n), \qquad (10)$$

where $\mathbf{e}(n) \triangleq \mathbf{y}(n) - \mathbf{Z}^{\mathrm{T}}(n)\mathbf{h}(n)$.

The estimates $\boldsymbol{\alpha}$ in (7) and $\mathbf{h}$ in (10) require knowledge of the input of the entire graph, which can be impractical in applications without a centralized processing unit. Therefore, we consider now a distributed implementation of the GKLMS, named graph diffusion KLMS (GDKLMS).

C. GDKLMS using RFF

The global optimization problem (9) can be rewritten as the following separable problem

$$(\boldsymbol{\psi}_1^*(n), \ldots, \boldsymbol{\psi}_K^*(n)) = \arg\min_{\boldsymbol{\psi}_1,\ldots,\boldsymbol{\psi}_K\in\mathbb{R}^D} \sum_{k=1}^{K} \mathbb{E}[(y_k(n) - \mathbf{z}_k^{\mathrm{T}}(n)\boldsymbol{\psi}_k)^2], \qquad (11)$$

where $\boldsymbol{\psi}_k$ is the local estimate of $\mathbf{h}$ at node k. Problem (11) is solved in a distributed fashion by minimizing $\mathbb{E}\left[(y_k(n) - \mathbf{z}_k^{\mathrm{T}}(n)\boldsymbol{\psi}_k)^2\right]$ at each node. Similar to the centralized case, denoting $e_k(n) = y_k(n) - \mathbf{z}_k^{\mathrm{T}}(n)\boldsymbol{\psi}_k(n)$, the update rule for $\boldsymbol{\psi}_k$ is given by

$$\boldsymbol{\psi}_k(n+1) = \boldsymbol{\psi}_k(n) + \mu\, e_k(n)\mathbf{z}_k(n). \qquad (12)$$

We now adopt the adapt-then-combine (ATC) strategy to improve individual estimates via graph diffusion [11], [12], [19], [30], [32]. The parameter update of $\mathbf{h}_k(n)$ at node k is obtained by combining the local estimates from its neighborhood. The ATC update rule for the GDKLMS using RFF is given by

$$\boldsymbol{\psi}_k(n+1) = \mathbf{h}_k(n) + \mu\, e_k(n)\mathbf{z}_k(n), \qquad (13a)$$

$$\mathbf{h}_k(n+1) = \sum_{l\in\mathcal{N}_k} a_{lk}\, \boldsymbol{\psi}_l(n+1), \qquad (13b)$$

where combination coefficients a_{lk} are non-negative and satisfy the condition $\sum_{l\in\mathcal{N}_k} a_{lk} = 1$ [32].

We note that if coherence-check is implemented for individual nodes, it can lead to unequal dictionaries across the graph, making it challenging to implement the algorithm in a distributed fashion [28]. As an alternative, we consider the construction of a pre-trained centralized dictionary [19]. The ATC approach using coherence-check proposed in [19] can be generalized for graph kernel filters, such that each node k updates its coefficient vector $\boldsymbol{\alpha}_k$ by combining the local estimates from its neighborhood. The dictionary can be obtained in a centralized way and broadcasted to the entire network or by a single dedicated node that shares its dictionary with all nodes. Moreover, the pre-training of the dictionary depends on available training data. Therefore, we note that RFF offer more flexibility for distributed implementations than the coherence-check approach, as the dimension of the RFF space can be set equally for all nodes.

IV. Mean Convergence Analysis

In this section, we study the mean convergence of the GDKLMS using RFF. For this, at network-level, we define the filter coefficient vector in RFF space $\mathbf{h}_g = \mathbf{1}_K \otimes \mathbf{h}$, the estimated filter coefficient vector in RFF space $\mathbf{h}_g(n) = [\mathbf{h}_1^{\mathrm{T}}(n)\, \mathbf{h}_2^{\mathrm{T}}(n) \ldots \mathbf{h}_K^{\mathrm{T}}(n)]^{\mathrm{T}}$, and the input data matrix $\boldsymbol{\mathcal{Z}}(n) = \text{blockdiag}\{\mathbf{z}_1(n), \mathbf{z}_2(n), \ldots, \mathbf{z}_K(n)\}$, where blockdiag$\{\cdot\}$ denotes the block-diagonal-stacking operator. The symbol $\mathbf{1}_K$ is a column vector of size $K \times 1$ with every element taking the value one and $\otimes$ denotes the right Kronecker product operator. Combination coefficients are gathered into a stochastic matrix $\mathbf{A}$ such that $[\mathbf{A}]_{k,l} = a_{kl}$. From these definitions, the network-level data model is given by $\mathbf{y}(n) = \boldsymbol{\mathcal{Z}}^{\mathrm{T}}(n)\mathbf{h}_g + \boldsymbol{v}(n)$, where $\boldsymbol{v}(n) = [v_1(n)\, v_2(n) \,\ldots\, v_K(n)]^{\mathrm{T}}$. Using these definitions, the network-level update recursion of the GD-KLMS using RFF can be stated as

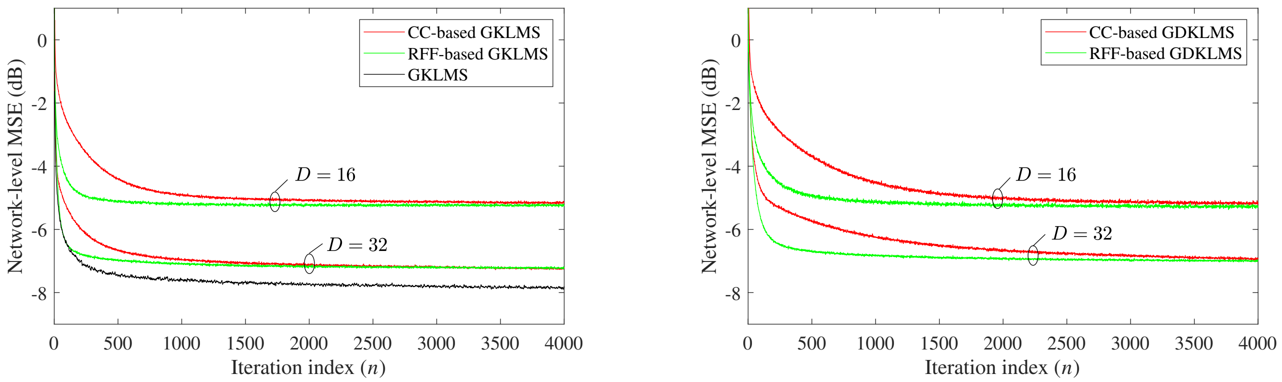
$$\mathbf{h}_g(n+1) = \boldsymbol{\mathcal{A}}\left(\mathbf{h}_g(n) + \mu\boldsymbol{\mathcal{Z}}(n)\mathbf{e}(n)\right), \qquad (14)$$

where $\boldsymbol{\mathcal{A}} = \mathbf{A}^{\mathrm{T}} \otimes \mathbf{I}_D$. Denoting the global weight deviation vector of the proposed GDKLMS using RFF at time index n as $\tilde{\mathbf{h}}_g(n) = \mathbf{h}_g - \mathbf{h}_g(n)$, and considering that $\boldsymbol{\mathcal{A}}\mathbf{h}_g = \mathbf{h}_g$ (since the matrix $\mathbf{A}$ is left stochastic), from (14), the recursion for $\tilde{\mathbf{h}}_g(n)$ can then be obtained as

$$\tilde{\mathbf{h}}_g(n+1) = \boldsymbol{\mathcal{B}}(n)\tilde{\mathbf{h}}_g(n) - \mu\, \boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{Z}}(n)\boldsymbol{v}(n), \qquad (15)$$

where $\boldsymbol{\mathcal{B}}(n) = \boldsymbol{\mathcal{A}}\big(\mathbf{I}_{DK} - \mu\,\boldsymbol{\mathcal{Z}}(n)\boldsymbol{\mathcal{Z}}^{\mathrm{T}}(n)\big)$.

Taking the statistical expectation on both sides of (15), assuming statistical independence between $\mathbf{h}_k(n)$ and $\mathbf{z}_k(n)$, $\forall k \in \mathcal{N}$ [31], and considering that the observation noise $v_k(n)$



(a) Centralized solutions.

(b) Distributed solutions.

Fig. 1. Learning curves (network-level MSE vs iteration index) for the proposed algorithms.

is a zero mean i.i.d. random sequence, which is taken to be independent of any other signal, we obtain

$$\mathbb{E}[\tilde{\mathbf{h}}_g(n+1)] = \overline{\boldsymbol{\mathcal{B}}}\ \mathbb{E}[\tilde{\mathbf{h}}_g(n)], \tag{16}$$

where $\overline{\boldsymbol{\mathcal{B}}} = \mathbb{E}[\boldsymbol{\mathcal{B}}(n)] = \boldsymbol{\mathcal{A}}\big(\mathbf{I}_{DK} - \mu\mathbf{R}_z\big)$ with $\mathbf{R}_z = \mathbb{E}[\boldsymbol{\mathcal{Z}}(n)\boldsymbol{\mathcal{Z}}^{\mathrm{T}}(n)] = \text{blockdiag}(\mathbf{R}_{z,1}, \mathbf{R}_{z,2}, \ldots, \mathbf{R}_{z,K})$, with $\mathbf{R}_{z,k} = \mathbb{E}[\mathbf{z}_k(n)\mathbf{z}_k^{\mathrm{T}}(n)]$. Note that the vector $\mathbf{z}_k(n)$ is the representation of $\mathbf{r}_k(n)$ in the RFF space. So the input vectors $\mathbf{z}_k(n)$, for $k \in \mathcal{N}$, may not satisfy both zero-mean and Gaussian distribution conditions [24]. However, 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 [24]. Therefore, from (16), it is easily seen that $\lim_{n\to\infty} \mathbb{E}\big[\tilde{\mathbf{h}}_g(n)\big]$ 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., $\|\overline{\boldsymbol{\mathcal{B}}}\|_{b,\infty}$). Using the properties of block maximum norm [32], we can write

$$\|\overline{\boldsymbol{\mathcal{B}}}\|_{b,\infty} \leq \|\boldsymbol{\mathcal{A}}\|_{b,\infty}\|\mathbf{I}_{DK} - \mu\mathbf{R}_z\|_{b,\infty}. \tag{17}$$

Using [32, Lemma D. 3(a), D. 5], a sufficient condition for $\mathbb{E}[\tilde{\mathbf{h}}_g(n)]$ to converge asymptotically 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 obtain the following condition on μ:

$$0 < \mu < \frac{2}{\max\limits_{\forall k\in\mathcal{N}}\big\{\max\limits_{\forall i}\{\lambda_i(\ \mathbf{R}_{z,k})\}\big\}}. \tag{18}$$

V. Numerical Results

We validate the performance of the proposed algorithms on a connected Erdös-Renyi graph consisting of 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 [32]. We used a Gaussian kernel with $\sigma^2 = 1$. For a filter of length $L = 4$, we aim at estimating the nonlinear function given by

$$\begin{aligned} f(\mathbf{r}_k(n)) = \sqrt{r_{k,1}(n)^2 + \sin^2(r_{k,4}(n)\pi)} \\ + (0.8 - 0.5\exp(-r_{k,2}(n)^2)r_{k,3}(n) \end{aligned} \tag{19}$$

The network-level MSE given by $\text{MSE}(n) = \frac{1}{K}\sum_{k=1}^{N} e_k^2(n)$ was considered as the performance metric, and results are displayed by plotting the MSE versus iteration index n, averaging over 1000 independent experiments. In order to compare coherence-check-based approaches with RFF-based approaches, the adaptation step size μ was adjusted so that the learning curves achieve similar steady network-level MSE. A centralized training dataset was used for the coherence-check simulations to pre-train the dictionary and broadcast it to all nodes before the learning iterations. The number of training samples used in the pre-training is not considered in the results. Moreover, we note that the linear approaches, namely the graph LMS and the graph diffusion LMS [12], could not model the target function reasonably.

Fig. 1a shows the learning curves for centralized solutions. Specifically, we compare the GKLMS without dictionary sparsification with $\mu = 0.1$ to the solutions using RFF and coherence-check. 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 the GKLMS without dictionary sparsification, coherence-check, and RFF based algorithms can effectively represent the target function. Fig. 1a also shows that, for the same D and similar values of steady-state network-level MSE, the RFF based algorithm converges faster than the coherence-check-based one. Moreover, comparing the plots for $D = 32$ to the GKLMS plot shows that both the coherence-check and RFF based algorithms can approximate the graph KLMS without sparsification as D increases.

Fig. 1b shows the results for the distributed GDKLMS using coherence-check and RFF. Similar to the centralized case, the plots show that the coherence-check and RFF-based approaches can effectively represent the target function, achieving network-level MSE of approximately -10 dB for $D = 16$ and -14 dB for $D = 32$. Again, the RFF-based solution exhibits faster convergence for both values of D when the steady-state network-level MSE is matched.

VI. Conclusions

This paper introduced nonlinear graph filters that operate 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 feature (RFF) based approaches were proposed. Furthermore, diffusion-based distributed implementations of coherence-check and RFF-based graph KLMS algorithms were developed to update filter parameters through local communications and in-network processing. Mean convergence conditions on the adaptation step size were established for the proposed GDKLMS using RFF. Numerical simulations were conducted to demonstrate the performance of the proposed algorithms. Although the coherence-check and RFF-based approaches effectively estimate the nonlinear graph filter, the RFF-based approach exhibits a faster convergence rate than the coherence-check based approach.

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,” in *IEEE Trans. Signal Process.*, 2020.

[5] 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.

[6] 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.

[7] 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.

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

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

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

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

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

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

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

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

[30] 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.

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

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

[33] 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.

[34] 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.

[35] 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.

[36] 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.