Inference of Spatio-Temporal Functions Over Graphs via Multikernel Kriged Kalman Filtering

Vassilis N. Ioannidis, *Student Member, IEEE*, Daniel Romero, *Member, IEEE*, and Georgios B. Giannakis, *Fellow, IEEE*

***Abstract*—Inference of space-time varying signals on graphs emerges naturally in a plethora of network science related applications. A frequently encountered challenge pertains to reconstructing such dynamic processes, given their values over a subset of vertices and time instants. The present paper develops a graph-aware kernel-based kriged Kalman filter that accounts for the spatio-temporal variations, and offers efficient online reconstruction, even for dynamically evolving network topologies. The kernel-based learning framework bypasses the need for statistical information by capitalizing on the smoothness that graph signals exhibit with respect to the underlying graph. To address the challenge of selecting the appropriate kernel, the proposed filter is combined with a multikernel selection module. Such a data-driven method selects a kernel attuned to the signal dynamics on-the-fly within the linear span of a preselected dictionary. The novel multikernel learning algorithm exploits the eigenstructure of Laplacian kernel matrices to reduce computational complexity. Numerical tests with synthetic and real data demonstrate the superior reconstruction performance of the novel approach relative to state-of-the-art alternatives.**

***Index Terms*—Graph signal reconstruction, dynamic models on graphs, kriged Kalman filtering, multi-kernel learning.**

I. Introduction

A NUMBER of applications involve data that admit a natural representation in terms of node attributes over social, economic, sensor, communication, and biological networks, to name a few [12], [26]. An inference task that emerges in this context is to predict or extrapolate the attributes of all nodes in the network given the attributes of a subset of them. In a finance network, where nodes correspond to stocks and edges capture dependencies among them, one may be interested in predicting the price of all stocks in the network knowing the price of some. This is of paramount importance in applications where collecting the attributes of all nodes is prohibitive, as is the case when sampling large-scale graphs, or, when the attribute of interest is of sensitive nature, such as the transmission of HIV in a social network. This task was first formulated as reconstructing a *time-invariant* function on a graph [26], [27].

Follow-up reconstruction approaches leverage the notions of graph bandlimitedness [5], [24], sparsity and overcomplete dictionaries [29], smoothness over the graph [13], [27], all of which can be unified as approximations of nonparametric graph functions drawn from a reproducing kernel Hilbert space (RKHS) [21]; see also [10] for semi-parametric alternatives.

In various applications however, the network connectivity and node attributes change over time. Such is the case in e.g. a finance network, where not only the stock prices change over time, but also their inter-dependencies. Hence, maximizing reconstruction performance for these time-varying signals necessitates judicious modeling of the space-time dynamics, especially when samples are scarce.

Inference of *time-varying* graph functions has been so far pursued mainly for slow variations [9], [15], [30]. Temporal dynamics have been modeled in [18] by assuming that the covariance of the function to be reconstructed is available. On the other hand, spatio-temporal reconstruction of generally dynamic graphs has been approached using an extended graph kernel matrix model with a block tridiagonal structure that lends itself to a computationally tractable iterative solver [19]. However, [19] neither relies on a dynamic model of the function variability, nor it provides a tractable method to learn the "best" kernel that fits the data. Similarly, the algorithm in [11] assumes that an appropriate kernel is known. Furthermore, [18], [11], and [19] do not adapt to changes in the spatio-temporal dynamics of the graph function.

The present paper fills this gap by introducing online estimators for time-varying functions on generally dynamic graphs. Specifically, the contribution is threefold.

- C1. A deterministic model for time-varying network processes is proposed, where spatial dependencies are captured by the topology while spatio-temporal dynamics are described through a graph-aware state-space model.
- C2. Based on this model, an algorithm termed kernel kriged Kalman filter (KeKriKF) is developed to obtain function estimates by minimizing a kernel ridge regression (KRR) criterion in an online fashion. The proposed solver generalizes the traditional network kriged Kalman filter (KriKF) [17], [18], [31], which relies on a probabilistic

Manuscript received November 23, 2017; revised March 2, 2018 and March 25, 2018; accepted March 26, 2018. Date of publication April 20, 2018; date of current version May 10, 2018. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Oliver Lezoray. This work was supported by NSF under Grants 1442686, 1500713, and 1508993. This paper was presented in part at the 25th European Signal Processing Conference, Kos island, Greece, Aug.–Sep., 2017. *(Corresponding author: Vassilis N. Ioannidis.)*

V. N. Ioannidis and G. B. Giannakis are with the Department of Electrical and Computer Engineering and the Digital Technology Center, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: ioann006@umn.edu; georgios@umn.edu).

D. Romero is with the Department of ICT, University of Agder, Grimstad 4879, Norway (e-mail: daniel.romero@uia.no).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2018.2827328

model. The novel estimator forgoes with assumptions on data distributions and stationarity, by promoting space-time smoothness through dynamic kernels on graphs.

C3. To select the most appropriate kernel, a multi-kernel (M)KriKF is developed based on the multi-kernel learning (MKL) framework. This algorithm adaptively selects the kernel that "best" fits the data dynamics within the linear span of a prespecified kernel dictionary. The structure of Laplacian kernels is exploited to reduce complexity down to the order of KeKriKF. This complexity is linear in the number of time samples, which renders KeKriKF and MKriKF appealing for online operation.

The rest of the paper is structured as follows. Section II contains preliminaries and states the problem. Section III introduces the spatio-temporal model and develops the KeKriKF. Section IV endows the KeKriKF with an MKL module to obtain the MKriKF. Finally, numerical experiments and conclusions are presented in Sections V and VI, respectively.

Notation: Scalars are denoted by lowercase, column vectors by bold lowercase, and matrices by bold uppercase letters. Superscripts $^\top$ and † respectively denote transpose and pseudo-inverse; $\mathbf{1}_N$ stands for the $N \times 1$ all-one vector; $\mathrm{diag}\{\boldsymbol{x}\}$ corresponds to a diagonal matrix with the entries of $\boldsymbol{x}$ on its diagonal, while $\mathrm{diag}\{\boldsymbol{X}\}$ is a vector holding the diagonal entries of $\boldsymbol{X}$; and $\mathcal{N}(\mu, \sigma^2)$ a Gaussian distribution with mean μ and variance σ^2. Finally, if $\boldsymbol{A}$ is a matrix and $\boldsymbol{x}$ a vector, then $\|\boldsymbol{x}\|^2_{\boldsymbol{A}} := \boldsymbol{x}^\top \boldsymbol{A}^{-1} \boldsymbol{x}$ and $\|\boldsymbol{x}\|^2_2 := \boldsymbol{x}^\top \boldsymbol{x}$.

II. Problem Statement and Preliminaries

Consider a time-varying graph $\mathcal{G}_t := (\mathcal{V}, \boldsymbol{A}_t)$, $t = 1, 2, \ldots$, where $\mathcal{V} := \{v_1, \ldots, v_N\}$ denotes the vertex set, and $\boldsymbol{A}_t$ the $N \times N$ adjacency matrix, whose (n, n')-th entry $A_{n,n'}(t)$ is the nonnegative weight of the edge connecting vertices v_n and $v_{n'}$ at time t. The edge set is $\mathcal{E}_t := \{(v_n, v_{n'}) \in \mathcal{V} \times \mathcal{V} : A_{n,n'}(t) \neq 0\}$, and two vertices v and v' are *connected* at time t if $(v, v') \in \mathcal{E}_t$. The graphs $\{\mathcal{G}_t\}_t$ in this paper are undirected and have no self-loops, which means that $\boldsymbol{A}_t = \boldsymbol{A}_t^\top$ and $A_{n,n}(t) = 0$, $\forall t, n$. The Laplacian matrix is $\boldsymbol{L}_t := \mathrm{diag}\{\boldsymbol{A}_t \mathbf{1}_N\} - \boldsymbol{A}_t$, and is positive semidefinite provided that $A_{n,n'}(t) \geq 0, \forall n, n', t$.

A time-varying graph function is a map $f : \mathcal{V} \times \mathcal{T} \to \mathbb{R}$, where $\mathcal{T} := \{1, 2, \ldots\}$ is the set of time indices. Specifically, $f(v_n, t)$ represents the value of the attribute of interest at node n and time t, e.g. the closing price of the n-th stock on the t-th day. Vector $\boldsymbol{f}_t := [f(v_1, t), \ldots, f(v_N, t)]^\top \in \mathbb{R}^N$ collects the function values at time t.

Suppose that S_t noisy observations $y(v_{n_s}, t) = f(v_{n_s}, t) + e(v_{n_s}, t)$, $s = 1, \ldots, S_t$, are available at time t, where $\mathcal{S}_t := \{n_1, \ldots, n_{S_t}\}$ contains the sampled indices $1 \leq n_1 \leq \ldots \leq n_{S_t} \leq N$, and $e(v_{n_s}, t)$ captures the observation error.

With $\boldsymbol{y}_t := [y(v_{n_1}, t), \ldots, y(v_{n_{S_t}}, t)]^\top$ and $\boldsymbol{e}_t := [e(v_{n_1}, t), \ldots, e(v_{n_{S_t}}, t)]^\top$, the observation model in vector-matrix form is

$$\boldsymbol{y}_t = \boldsymbol{S}_t \boldsymbol{f}_t + \boldsymbol{e}_t, \quad t = 1, 2, \ldots \tag{1}$$

where $\boldsymbol{S}_t \in \{0, 1\}^{S_t \times N}$ selects the sampled entries of $\boldsymbol{f}_t$.

Given $\boldsymbol{y}_\tau$, $\boldsymbol{S}_\tau$, and $\boldsymbol{A}_\tau$ for $\tau = 1, \ldots, t$, the goal of this paper is to reconstruct $\boldsymbol{f}_t$ at each t. The estimators should operate in an online fashion, which means that the computational complexity per time slot t must not grow with t. Observe that no statistical information is assumed available in our formulation.

TABLE I
Examples of Laplacian Kernels and Their Associated Spectral Weight Functions

Kernel name	Function	Parameters
Diffusion kernel [13]	$r(\lambda) = \exp\{\sigma^2 \lambda/2\}$	$\sigma^2 \geq 0$
p-step random walk [27]	$r(\lambda) = (a - \lambda)^{-p}$	$a \geq 2$, p
Regularized Laplacian [26], [27], [32]	$r(\lambda) = 1 + \sigma^2 \lambda$	$\sigma^2 \geq 0$
Bandlimited [21]	$r(\lambda_n) = \begin{cases} 1/\beta & 1 \leq n \leq B \\ \beta & \text{otherwise} \end{cases}$	$\beta > 0$, B
Band-rejection	$r(\lambda_n) = \begin{cases} \beta & k \leq n \leq N - l \\ 1/\beta & \text{otherwise} \end{cases}$	$\beta > 0$, k, l

A. Kernel-Based Reconstruction

Aiming ultimately at the time-varying $\boldsymbol{f}_t$, it is instructive to outline the kernel-based reconstruction of a time-invariant $\boldsymbol{f} := [f_1, \ldots, f_N]$ given $\mathcal{G} := (\mathcal{V}, \boldsymbol{A})$, and using samples $\boldsymbol{y} = \boldsymbol{S}\boldsymbol{f} + \boldsymbol{e} \in \mathbb{R}^S$, where $\boldsymbol{S} \in \{0, 1\}^{S \times N}$ and $S < N$.

Relying on regularized least-squares (LS), we obtain

$$\hat{\boldsymbol{f}} = \arg\min_{\boldsymbol{f}} \|\boldsymbol{y} - \boldsymbol{S}\boldsymbol{f}\|_2^2 + \mu g(\boldsymbol{f}) \tag{2}$$

where $\mu > 0$ and the regularizer $g(\boldsymbol{f})$ promotes estimates with a certain structure.

For example, the so-called *Laplacian* regularizer

$$g_{\mathrm{LR}}(\boldsymbol{f}) := (1/2) \sum_{n=1}^{N} \sum_{n'=1}^{N} A_{n,n'} (f_n - f_{n'})^2 \tag{3}$$

promotes smooth function estimates with similar values at vertices connected by strong links (large $A_{n,n'}$), since $g_{\mathrm{LR}}(\boldsymbol{f})$ is small when $\boldsymbol{f}$ is smooth. It turns out that $g_{\mathrm{LR}}(\boldsymbol{f}) = \boldsymbol{f}^\top \boldsymbol{L} \boldsymbol{f}$; see e.g. [12, Ch. 2]. For a scalar function $r(\boldsymbol{L})$ a general graph kernel family of regularizers is obtained as $g_{\mathrm{KR}}(\boldsymbol{f}) = \boldsymbol{f}^\top \boldsymbol{K}^\dagger \boldsymbol{f} = \|\boldsymbol{f}\|^2_{\boldsymbol{K}}$, where the *Laplacian kernel* is defined as

$$\boldsymbol{K} := r^\dagger(\boldsymbol{L}) := \boldsymbol{U}^\top \mathrm{diag}\{r^\dagger(\boldsymbol{\lambda})\} \boldsymbol{U} \tag{4}$$

where $\boldsymbol{U} \in \mathbb{R}^{N \times N}$, and $\boldsymbol{\lambda} \geq \mathbf{0} \in \mathbb{R}^{N \times 1}$ denote the eigenvector matrix and the eigenvalues of $\boldsymbol{L}$, since $\boldsymbol{L} = \boldsymbol{U}\, \mathrm{diag}\{\boldsymbol{\lambda}\}\, \boldsymbol{U}^\top$. Clearly, $g_{\mathrm{KR}}(\boldsymbol{f})$ subsumes $g_{\mathrm{LR}}(\boldsymbol{f})$ for $r(\boldsymbol{L}) = \boldsymbol{L}$. Other special cases of $g_{\mathrm{KR}}(\boldsymbol{f})$ that will be tested in the simulations are collected in Table I, and the scalar functions are plotted in Fig 1. Prior knowledge about the properties of $\boldsymbol{f}$ may guide the selection of the appropriate $r(\cdot)$. For instance, a diffusion kernel accounts for smoothness of $\boldsymbol{f}$, as well as the prior that $\boldsymbol{f}$ is generated by a graph diffusion process. Data-driven selection techniques follow in Section IV.

Further broadening the scope of the generalized Laplacian kernel regularizers, one may set $g(\boldsymbol{f}) = \|\boldsymbol{f}\|^2_{\boldsymbol{K}}$ for an arbitrary positive semidefinite matrix $\boldsymbol{K}$, not necessarily a Laplacian kernel. These regularizers give rise to the family of *kernel ridge regression* (KRR) estimators

$$\hat{\boldsymbol{f}} := \arg\min_{\boldsymbol{f}} \frac{1}{S} \|\boldsymbol{y} - \boldsymbol{S}\boldsymbol{f}\|_2^2 + \mu \|\boldsymbol{f}\|^2_{\boldsymbol{K}} \tag{5}$$

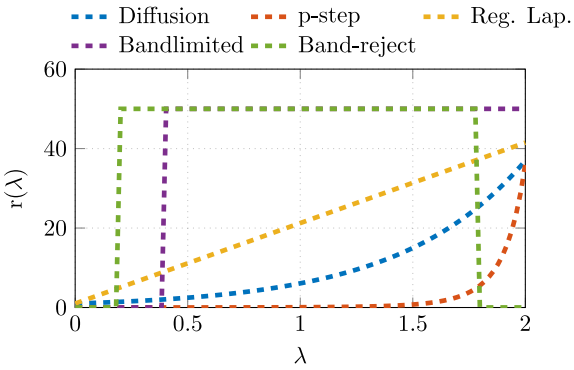


Fig. 1. Laplacian kernels (Diffusion $\sigma = 1.9$, p-step random walk $\alpha = 2.55$, $p = 6$, Regularized Laplacian $\sigma = 4.5$, $\beta = 50$, Bandwidth $B = 20$, $\beta = 50$, Band-reject $k = 10$, $l = 10$).

where $\mu > 0$ controls the effect of the regularizer with respect to the fitting term $S^{-1}||\boldsymbol{y} - \boldsymbol{S}\boldsymbol{f}||_2^2$. KRR estimators have well-documented merits and solid grounds on statistical learning theory; see e.g. [23].

So far, signal $\boldsymbol{f}$ was assumed deterministic. To present a probabilistic interpretation of KRR suppose that $\boldsymbol{f}$ is zero-mean with $\boldsymbol{C} := \mathbb{E}\left[\boldsymbol{f}\boldsymbol{f}^\top\right]$, and that the entries of $\boldsymbol{e}$ are uncorrelated with each other and with $\boldsymbol{f}$, and $\sigma_e^2 := S^{-1}\mathbb{E}\left[\|\boldsymbol{e}\|_2^2\right]$. In this setting, the KRR estimator (5) reduces to the linear minimum mean-square error (LMMSE) estimator if $\mu S = \sigma_e^2$ and $\boldsymbol{K} = \boldsymbol{C}$. Thus, KRR generalizes LMMSE and can be interpreted as the LMMSE estimator of a random signal $\boldsymbol{f}$ with covariance matrix $\boldsymbol{K}$; see [21, Proposition 2].

III. Kernel Kriged Kalman Filter

This section presents a space-time varying model that is capable of accommodating fairly general forms of spatio-temporal dynamics. Building on this model, a novel online KRR estimator will be subsequently developed for graph functions over time-varying graphs.

A. Spatio-Temporal Model

An immediate approach to estimate $\boldsymbol{f}_t$ is to apply (5) separately per slot t. This yields the instantaneous estimator

$$\hat{\boldsymbol{f}}_t^{(\nu)} := \arg\min_{\boldsymbol{f}} \frac{1}{S_t}||\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{f}||_2^2 + \mu\|\boldsymbol{f}\|_{\boldsymbol{K}_t}^2 \tag{6}$$

where $\boldsymbol{K}_t > \boldsymbol{0}$ is a per-slot preselected kernel matrix, and superscript ν will be explained later. Unfortunately, such an approach does not account for the possible dynamics relating $\boldsymbol{f}_t$ to $\boldsymbol{f}_{t-1}$. However, leveraging dependencies across slots can benefit the estimator of $\boldsymbol{f}_t$ from observations $\{\boldsymbol{y}_\tau\}_{\tau\neq t}$.

To circumvent the aforementioned limitation, consider modeling the function of interest as

$$f(v_n, t) = f^{(\nu)}(v_n, t) + f^{(\chi)}(v_n, t) \tag{7}$$

where $f^{(\nu)}$ captures arbitrary (even fast) temporal dynamics across sampling intervals and can be interpreted as an instantaneous component, while $f^{(\chi)}$ represents a structured (typically slow) varying component.

As an example, consider stock price prediction, where $f^{(\nu)}$ accounts for instantaneous changes caused e.g. by political statements or company announcements at t relative to $t-1$, while $f^{(\chi)}$ captures the steady evolution of the stock market, where stock prices at slot t are closely related to prices of (possibly) other stocks at $t-1$. Before delving into how these components are modeled, let $\boldsymbol{f}_t^{(\nu)} := [f^{(\nu)}(v_1, t), \ldots, f^{(\nu)}(v_N, t)]^\top$ and $\boldsymbol{f}_t^{(\chi)} := [f^{(\chi)}(v_1, t), \ldots, f^{(\chi)}(v_N, t)]^\top$, and note that (7) can be cast into vector form as

$$\boldsymbol{f}_t = \boldsymbol{f}_t^{(\nu)} + \boldsymbol{f}_t^{(\chi)}. \tag{8}$$

Vector $\boldsymbol{f}_t^{(\nu)}$ can be smooth over its entries $(\mathcal{G}_t)$, and captures instantaneous dependence among $\{f(v_n, t)\}_{n=1}^N$. This term models the component of the network process that is either uncorrelated over time or it is uncorrelated with respect to the presumed sampling interval. Indeed, one may consider nonlinear models to capture the dynamics of $\boldsymbol{f}^{(\nu)}$. However, such an approach goes beyond the scope of this submission. On the other hand, $\boldsymbol{f}_t^{(\chi)}$ is smooth not only over $\mathcal{G}_t$ but also over time, and models dependencies between $\{f(v_n, t)\}_{n=1}^N$ and their time-lagged versions $\{f(v_n, t-1)\}_{n=1}^N$.

The smooth evolution of $\boldsymbol{f}_t^{(\chi)}$ over time slots adheres to the state equation

$$\boldsymbol{f}_t^{(\chi)} = \boldsymbol{A}_{(t,t-1)}\boldsymbol{f}_{t-1}^{(\chi)} + \boldsymbol{\eta}_t, \quad t = 1, 2, \ldots \tag{9}$$

where $\boldsymbol{A}_{(t,t-1)}$ is a *graph* transition matrix, and $\boldsymbol{\eta}_t := [\eta(v_1, t), \ldots, \eta(v_N, t)]^\top \in \mathbb{R}^N$ is termed state noise. Vector $\boldsymbol{\eta}_t$ will be assumed smooth over $\mathcal{G}_t$, meaning $\eta(v_n, t)$ is expected to be similar to $\eta(v_{n'}, t)$ if $A_{n,n'}(t) \neq 0$. The recursion in (9) is the graph counterpart of a vector autoregressive model (VARM) of order one (see e.g. [16], [25]), and will lead to computationally efficient online KRR estimators of $\boldsymbol{f}_t$ that account for temporal dynamics [25].

Model (8) can be thought of as the graph counterpart of the model adopted in [31] to derive the kriged Kalman filter. In our context here, $\boldsymbol{f}_t^{(\nu)}$ describes small-scale *spatial fluctuations* within slot t, whereas $\boldsymbol{f}_t^{(\chi)}$ captures the so-called *trend* across slots. Furthermore, (8) generalizes the model used in [18], where $\boldsymbol{A}_{(t,t-1)} = \boldsymbol{I}_N$, for network delay prediction, where $\boldsymbol{f}_t^{(\nu)}$ represents the propagation, transmission, and processing delays and $\boldsymbol{f}_t^{(\chi)}$ the queuing delay at each router that is affected.

Remark 1: The transition matrix $\boldsymbol{A}_{(t,t-1)}$ can be interpreted as the $N \times N$ adjacency of a generally directed "transition graph" that relates $\{f^{(\chi)}(v_n, t-1)\}_{n=1}^N$ to $\{f^{(\chi)}(v_n, t)\}_{n=1}^N$. To avoid estimation of $\boldsymbol{A}_{(t,t-1)}$, the random walk model is motivated, where $\boldsymbol{A}_{(t,t-1)} = c\boldsymbol{I}_N$ with $c > 0$. On the other hand, adherence to the graph, prompts the selection $\boldsymbol{A}_{(t,t-1)} = c\boldsymbol{A}$, in which case (9) amounts to a diffusion process on a time-invariant $\mathcal{G}$ [24].

B. KeKriKF Algorithm

This section develops an online algorithm to estimate $\boldsymbol{f}_t$, given (1) and $\{\boldsymbol{y}_\tau, \boldsymbol{S}_\tau, \boldsymbol{A}_\tau, \boldsymbol{A}_{(\tau,\tau-1)}\}_{\tau=1}^t$ for the spatio-temporal model of $\boldsymbol{f}_t$ in (8) and (9). Unfortunately, $\{\boldsymbol{f}_\tau^{(\nu)}$ and $\boldsymbol{f}_\tau^{(\chi)}\}$ cannot be obtained by solving the system of equations comprising (1), (8), and (9) over time even if $\boldsymbol{e}_\tau = \boldsymbol{0}$ and $\boldsymbol{\eta}_\tau = \boldsymbol{0}$, $\forall\tau$; simply because after replacing $\boldsymbol{f}_\tau$ with

$\boldsymbol{f}_\tau^{(\chi)} + \boldsymbol{f}_\tau^{(\nu)}\ \forall \tau$, the estimation task involves $2Nt$ unknowns, namely $\{\boldsymbol{f}_\tau^{(\chi)}, \boldsymbol{f}_\tau^{(\nu)}\}_{\tau=1}^t$, and only $\tilde{S} + Nt$ equations, where $\tilde{S} := \sum_{\tau=1}^t S_\tau$ and $\tilde{S} \leq Nt$. To obtain a solution to this underdetermined problem, one must exploit the model structure. Extending the KRR estimator in (5) to time-varying functions, suppose we wish to

$$\underset{\{\boldsymbol{f}_\tau^{(\chi)}, \boldsymbol{f}_\tau^{(\nu)}\}_{\tau=1}^t}{\text{minimize}} \sum_{\tau=1}^t \frac{1}{S_\tau}\|\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\nu)}\|^2 + \mu_1 \sum_{\tau=1}^t \|\boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{A}_{(\tau,\tau-1)} \boldsymbol{f}_{\tau-1}^{(\chi)}\|^2_{\boldsymbol{K}_\tau^{(\chi)}} + \mu_2 \sum_{\tau=1}^t \|\boldsymbol{f}_\tau^{(\nu)}\|^2_{\boldsymbol{K}_\tau^{(\nu)}}. \tag{10}$$

where the scalars $\mu_1, \mu_2 \geq 0$ control the trade-off between smoothness and data fit. The first cost is an LS fitting error of the observations. The second cost is a weighted LS error of the slow-varying state (weight matrix promotes spatio-temporal smoothness over $\mathcal{G}_\tau$ and time), and the third term (regularizer) is the weighted ℓ_2-norm of the fast-varying state (with weight matrix promoting spatial smoothness over $\mathcal{G}_\tau$). When available, prior information about $\{\boldsymbol{f}_\tau^{(\nu)}, \boldsymbol{f}_\tau^{(\chi)}\}_{\tau=1}^t$ may steer the selection of suitable kernel matrices; when not available, one can resort to the algorithm in Section IV.

Directly solving (10) per t would not lead to an online algorithm since the complexity of such an approach grows with t; see Section II. However, we will develop next an efficient *online* algorithm to obtain per slot t estimates $\hat{\boldsymbol{f}}_{t|t}^{(\chi)}, \hat{\boldsymbol{f}}_{t|t}^{(\nu)}$ that still account for $\{\boldsymbol{y}_\tau, \boldsymbol{S}_\tau, \boldsymbol{A}_\tau\}_{\tau=1}^t$.

Given $\boldsymbol{f}_\tau^{(\chi)}$, the first-order necessary conditions for optimality of $\boldsymbol{f}_\tau^{(\nu)}$ yield [cf. (10)]

$$\boldsymbol{f}_\tau^{(\nu)} = \boldsymbol{K}_\tau^{(\nu)} \boldsymbol{S}_\tau^\top (\bar{\boldsymbol{K}}_\tau^{(\nu)} + \mu_2 S_\tau \boldsymbol{I}_{S_\tau})^{-1} (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)}) \tag{11}$$

where $\bar{\boldsymbol{K}}_\tau^{(\nu)} := \boldsymbol{S}_\tau \boldsymbol{K}_\tau^{(\nu)} \boldsymbol{S}_\tau^\top$. Notice that the overbar notation indicates $S_\tau \times S_\tau$ matrices or $S_\tau \times 1$ vectors, and recall that without overbar their counterparts have sizes $N \times N$ and $N \times 1$, respectively. Substituting (11) into (10), we arrive at an optimization problem that does not depend on $\boldsymbol{f}_\tau^{(\nu)}$ for $\tau = 1, \ldots, t$. Rewrite next the per slot τ measurement error in (10) using (11) as

$$\begin{aligned}
&\frac{1}{S_\tau}\|\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\nu)}\|^2 \\
&= \frac{1}{S_\tau}\|\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)} - \bar{\boldsymbol{K}}_\tau^{(\nu)} \\
&\quad \times (\bar{\boldsymbol{K}}_\tau^{(\nu)} + \mu_2 S_\tau \boldsymbol{I}_{S_\tau})^{-1} (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)})\|^2 \\
&= \frac{1}{S_\tau}\|\left[\boldsymbol{I}_{S_\tau} - \bar{\boldsymbol{K}}_\tau^{(\nu)} (\bar{\boldsymbol{K}}_\tau^{(\nu)} + \mu_2 S_\tau \boldsymbol{I}_{S_\tau})^{-1}\right] \\
&\quad \times (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)})\|^2.
\end{aligned} \tag{12a}$$

The matrix inversion lemma asserts for the matrix in square brackets of (12a) that

$$\left[\boldsymbol{I}_{S_\tau} - \bar{\boldsymbol{K}}_\tau^{(\nu)} (\bar{\boldsymbol{K}}_\tau^{(\nu)} + \mu_2 S_\tau \boldsymbol{I}_{S_\tau})^{-1}\right] = \left(\boldsymbol{I}_{S_\tau} + \frac{1}{\mu_2 S_\tau} \bar{\boldsymbol{K}}_\tau^{(\nu)}\right)^{-1}. \tag{12b}$$

Plugging (12b) into (12a) yields

$$\begin{aligned}
&= \frac{1}{S_\tau}\left\|\left(\frac{1}{\mu_2 S_\tau}\bar{\boldsymbol{K}}_\tau^{(\nu)} + \boldsymbol{I}_{S_\tau}\right)^{-1} (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)})\right\|^2 \\
&= (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)})^\top \left(\frac{1}{\mu_2}\bar{\boldsymbol{K}}_\tau^{(\nu)} + S_\tau \boldsymbol{I}_{S_\tau}\right)^{-\top} \\
&\quad \times S_\tau \boldsymbol{I}_{S_\tau} \left(\frac{1}{\mu_2}\bar{\boldsymbol{K}}_\tau^{(\nu)} + S_\tau \boldsymbol{I}_{S_\tau}\right)^{-1} (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)}).
\end{aligned} \tag{12c}$$

Next, we express the regularizer in (10) using (11) for each τ as

$$\begin{aligned}
&\mu_2 \|\boldsymbol{f}_\tau^{(\nu)}\|^2_{\boldsymbol{K}_\tau^{(\nu)}} \\
&= (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)})^\top \left(\frac{1}{\mu_2}\bar{\boldsymbol{K}}_\tau^{(\nu)} + S_\tau \boldsymbol{I}_{S_\tau}\right)^{-\top} \\
&\quad \times \frac{1}{\mu_2}\bar{\boldsymbol{K}}_\tau^{(\nu)} \left(\frac{1}{\mu_2}\bar{\boldsymbol{K}}_\tau^{(\nu)} + S_\tau \boldsymbol{I}_{S_\tau}\right)^{-1} (\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)})
\end{aligned} \tag{12d}$$

where the last equality follows from the definition of $\bar{\boldsymbol{K}}_\tau^{(\nu)}$.

Combining (12c) with (12d) yields

$$\begin{aligned}
&\frac{1}{S_\tau}\|\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\nu)}\|^2 + \mu_2 \|\boldsymbol{f}_\tau^{(\nu)}\|^2_{\boldsymbol{K}_\tau^{(\nu)}} \\
&= \|\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)}\|^2_{\check{\boldsymbol{K}}_\tau^{(\nu)}}
\end{aligned} \tag{13}$$

where $\check{\boldsymbol{K}}_\tau^{(\nu)} := \frac{1}{\mu_2}\bar{\boldsymbol{K}}_\tau^{(\nu)} + S_\tau \boldsymbol{I}_{S_\tau}$. Using (13) per slot, (10) boils down to

$$\{\hat{\boldsymbol{f}}_{\tau|t}^{(\chi)}\}_{\tau=1}^t := \underset{\{\boldsymbol{f}_\tau^{(\chi)}\}_{\tau=1}^t}{\arg\min} \sum_{\tau=1}^t \|\boldsymbol{y}_\tau - \boldsymbol{S}_\tau \boldsymbol{f}_\tau^{(\chi)}\|^2_{\check{\boldsymbol{K}}_\tau^{(\nu)}} + \mu_1 \sum_{\tau=1}^t \|\boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{A}_{(\tau,\tau-1)} \boldsymbol{f}_{\tau-1}^{(\chi)}\|^2_{\boldsymbol{K}_\tau^{(\chi)}}. \tag{14}$$

Since (14) is identical to the deterministic formulation of the Kalman filter (KF) applied to a state-space model with state noise covariance $\boldsymbol{K}_t^{(\chi)}$ and measurement noise covariance $\check{\boldsymbol{K}}_t^{(\nu)}$, we deduce that the KF algorithm, see e.g. [28, Ch. 17], applies readily to obtain sequentially the structured per slot t component $\{\hat{\boldsymbol{f}}_{\tau|\tau}^{(\chi)}\}_{\tau=1}^t$. After substituting $\{\hat{\boldsymbol{f}}_{\tau|\tau}^{(\chi)}\}_{\tau=1}^t$ into (11), we can find also the per slot instantaneous component $\{\hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}\}_{\tau=1}^t$. The t-th iteration of our so-termed KeKriKF is listed as Algorithm 1.

Summing up, we have established the following result.

Theorem 1: If $\left\{\{\hat{\boldsymbol{f}}_{\tau|t}^{(\chi)}, \hat{\boldsymbol{f}}_{\tau|t}^{(\nu)}\}_{\tau=1}^{\tau=t}\right\}_{t=1}^{t=t_1}$ *solves (10) for* $t = 1, \ldots, t_1$, *the KeKriKF iterations summarized in Algorithm 1 for* $t = 1, \ldots, t_1$ *generate the subset of solutions* $\{\hat{\boldsymbol{f}}_{t|t}^{(\chi)}, \hat{\boldsymbol{f}}_{t|t}^{(\nu)}\}_{t=1}^{t=t_1}$.

Clearly, the KeKriKF algorithm comprises two subprocedures: Kalman filtering (steps S1–S6), and kriging (step S7). S3–S5 specify $\boldsymbol{M}_{t|t-1}, \boldsymbol{M}_{t|t}$, and $\boldsymbol{G}_t$ that are known in the KF literature as the mean square-error matrices for prediction, correction, and the Kalman gain matrix.

The traditional KriKF has been employed to interpolate stationary processes defined over continuous spatial domains [17],

Algorithm 1: Kernel Kriged Kalman Filter (**KeKriKF**).

Input: $\boldsymbol{K}_t^{(\chi)}, \boldsymbol{K}_t^{(\nu)} \in \mathbb{S}_+^N$; $\boldsymbol{A}_{(t,t-1)} \in \mathbb{R}^{N\times N}$; $\boldsymbol{y}_t \in \mathbb{R}^{S_t}$; $\boldsymbol{S}_t \in \{0,1\}^{S_t\times N}$; $\hat{\boldsymbol{f}}_{t-1|t-1}^{(\chi)} \in \mathbb{R}^N$; $\boldsymbol{M}_{t-1|t-1} \in \mathbb{S}_+^N$.

S1. $\check{\boldsymbol{K}}_t^{(\nu)} = \frac{1}{\mu_2}\boldsymbol{S}_t\boldsymbol{K}_t^{(\nu)}\boldsymbol{S}_t^\top + S_t\boldsymbol{I}_{S_t}$

S2. $\hat{\boldsymbol{f}}_{t|t-1}^{(\chi)} = \boldsymbol{A}_{(t,t-1)}\hat{\boldsymbol{f}}_{t-1|t-1}^{(\chi)}$ (*prediction*)

S3. $\boldsymbol{M}_{t|t-1} = \boldsymbol{A}_{(t,t-1)}\boldsymbol{M}_{t-1|t-1}\boldsymbol{A}_{(t,t-1)}^\top + \frac{1}{\mu_1}\boldsymbol{K}_t^{(\chi)}$

S4. $\boldsymbol{G}_t = \boldsymbol{M}_{t|t-1}\boldsymbol{S}_t^\top(\check{\boldsymbol{K}}_t^{(\nu)} + \boldsymbol{S}_t\boldsymbol{M}_{t|t-1}\boldsymbol{S}_t^\top)^{-1}$ (*gain*)

S5. $\boldsymbol{M}_{t|t} = (\boldsymbol{I} - \boldsymbol{G}_t\boldsymbol{S}_t)\boldsymbol{M}_{t|t-1}$

S6. $\hat{\boldsymbol{f}}_{t|t}^{(\chi)} = \hat{\boldsymbol{f}}_{t|t-1}^{(\chi)} + \boldsymbol{G}_t(\boldsymbol{y}_t - \boldsymbol{S}_t\hat{\boldsymbol{f}}_{t|t-1}^{(\chi)})$ (*correction*)

S7. $\hat{\boldsymbol{f}}_{t|t}^{(\nu)} = \boldsymbol{K}_t^{(\nu)}\boldsymbol{S}_t^\top\check{\boldsymbol{K}}_t^{(\nu)^{-1}}(\boldsymbol{y}_t - \boldsymbol{S}_t\hat{\boldsymbol{f}}_{t|t}^{(\chi)})$ (*kriging*)

Output: $\hat{\boldsymbol{f}}_{t|t}^{(\chi)}$; $\hat{\boldsymbol{f}}_{t|t}^{(\nu)}$; $\boldsymbol{M}_{t|t}$.

[31], and its derivation follows from a probabilistic linear-minimum mean-square error (LMMSE) criterion that relies on knowledge of second-order statistics [17], [18], [31]. Here, our KeKriKF is derived from a deterministic kernel-based learning framework, which bypasses assumptions on data distributions and stationarity and replaces knowledge of second-order (cross-) covariances with knowledge of $\boldsymbol{K}_t^{(\nu)}$ and $\boldsymbol{K}_t^{(\chi)}$. Moreover, different from [7], [15], [18], [30], the novel KeKriKF can accommodate dynamic graph topologies provided $\{\boldsymbol{K}_t^{(\nu)}, \boldsymbol{K}_t^{(\chi)}\}_t$ are available.

Remark 2: The complexity of KeKriKF is $\mathcal{O}(N^3)$ per slot. When the underlying graph is large ($N \gg$), this complexity can be managed after splitting the graph into N_g subgraphs each with at most $\lceil N/N_g \rceil$ nodes, and employing consensus-based decentralized KF schemes along the lines of [22].

IV. Online Multi-Kernel Learning

This section broadens the scope of the KeKriKF algorithm by employing a multi-kernel learning scheme, to bypass the need for selecting an appropriate kernel.

The performance of KRR estimators is well known to heavily depend on the choice of the kernel matrix [21]. Unfortunately, it is difficult to know which kernel matrix is most appropriate for a given problem. To address this issue, an MKL approach is presented that selects a suitable kernel matrix within the linear span of a prespecified dictionary using the available data.

In the following, consider for simplicity that $\boldsymbol{K}_t^{(\nu)} = \boldsymbol{K}^{(\nu)}$, $\boldsymbol{K}_t^{(\chi)} = \boldsymbol{K}^{(\chi)}$, and $\boldsymbol{S}_t = \boldsymbol{S}$, $\forall t$. The kernels in the dictionaries $\mathcal{D}^{(\nu)} := \{\boldsymbol{K}^{(\nu)}[m] \in \mathbb{S}_+^N\}_{m=1}^{M_\nu}$, and $\mathcal{D}^{(\chi)} := \{\boldsymbol{K}^{(\chi)}[m] \in \mathbb{S}_+^N\}_{m=1}^{M_\chi}$ will be combined to generate $\boldsymbol{K}^{(\nu)} = \boldsymbol{K}^{(\nu)}(\boldsymbol{\theta}^{(\nu)}) := \sum_{m=1}^{M_\nu}\theta^{(\nu)}[m]\boldsymbol{K}^{(\nu)}[m]$ and $\boldsymbol{K}^{(\chi)} = \boldsymbol{K}^{(\chi)}(\boldsymbol{\theta}^{(\chi)}) := \sum_{m=1}^{M_\chi}\theta^{(\chi)}[m]\boldsymbol{K}^{(\chi)}[m]$, where $\boldsymbol{\theta}^{(\nu)} := [\theta^{(\nu)}[1], \ldots, \theta^{(\nu)}[M_\nu]]^\top$, $\boldsymbol{\theta}^{(\chi)} := [\theta^{(\chi)}[1], \ldots, \theta^{(\chi)}[M_\chi]]^\top \succeq \boldsymbol{0}$ are coefficients to be determined.

Next, consider expanding the optimization in (10) to obtain $\boldsymbol{\theta}^{(\nu)}, \boldsymbol{\theta}^{(\chi)}$ along with $\{\boldsymbol{f}_\tau^{(\chi)}, \boldsymbol{f}_\tau^{(\nu)}\}_{\tau=1}^t$, as follows

$$\begin{aligned}\underset{\substack{\{\boldsymbol{f}_\tau^{(\chi)}, \boldsymbol{f}_\tau^{(\nu)}\}_{\tau=1}^t,\\ \boldsymbol{\theta}^{(\chi)}\succeq\boldsymbol{0}, \boldsymbol{\theta}^{(\nu)}\succeq\boldsymbol{0}}}{\text{minimize}} & \frac{1}{t}\sum_{\tau=1}^t \frac{1}{S}\|\boldsymbol{y}_\tau - \boldsymbol{S}\boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{S}\boldsymbol{f}_\tau^{(\nu)}\|^2 \\ &+ \frac{\mu_1}{t}\sum_{\tau=1}^t \|\boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{A}_{(\tau,\tau-1)}\boldsymbol{f}_{\tau-1}^{(\chi)}\|^2_{\boldsymbol{K}^{(\chi)}(\boldsymbol{\theta}^{(\chi)})} \\ &+ \frac{\mu_2}{t}\sum_{\tau=1}^t \|\boldsymbol{f}_\tau^{(\nu)}\|^2_{\boldsymbol{K}^{(\nu)}(\boldsymbol{\theta}^{(\nu)})} + \rho_\nu\|\boldsymbol{\theta}^{(\nu)}\|_2^2 + \rho_\chi\|\boldsymbol{\theta}^{(\chi)}\|_2^2 \quad (15)\end{aligned}$$

where $\rho_\nu, \rho_\chi \geq 0$ are regularization parameters. The solution to (15) for each t will be denoted as $\{\hat{\boldsymbol{f}}_{\tau|t}^{(\chi)}, \hat{\boldsymbol{f}}_{\tau|t}^{(\nu)}\}_{\tau=1}^{\tau=t} \cup \{\hat{\boldsymbol{\theta}}_t^{(\chi)}, \hat{\boldsymbol{\theta}}_t^{(\nu)}\}$. Here, the data-dependent $\{\hat{\boldsymbol{\theta}}_t^{(\chi)}, \hat{\boldsymbol{\theta}}_t^{(\nu)}\}$ select the kernel matrices that "best" capture the data dynamics.

Due to the presence of the weighted norms, namely $\{\|\boldsymbol{f}_\tau^{(\chi)} - \boldsymbol{A}_{(\tau,\tau-1)}\boldsymbol{f}_{\tau-1}^{(\chi)}\|^2_{\boldsymbol{K}^{(\chi)}(\boldsymbol{\theta}^{(\chi)})}\}_{\tau=1}^t$ and $\{\|\boldsymbol{f}_\tau^{(\nu)}\|^2_{\boldsymbol{K}^{(\nu)}(\boldsymbol{\theta}^{(\nu)})}\}_{\tau=1}^t$, the problem in (15) is non-convex. Fortunately, (15) is separately convex in $\{\boldsymbol{f}_\tau^{(\chi)}, \boldsymbol{f}_\tau^{(\nu)}\}_{\tau=1}^t, \boldsymbol{\theta}^{(\nu)}, \boldsymbol{\theta}^{(\chi)}$, which motivates the use of alternating minimization (AM) strategies. AM algorithms minimize the objective with respect to every block of variables, while keeping the other variables fixed [8]. Conveniently, if $\boldsymbol{\theta}^{(\nu)}, \boldsymbol{\theta}^{(\chi)}$ are fixed, then (15) reduces to (10), which can be solved by Algorithm 1 for $\hat{\boldsymbol{f}}_{t|t}^{(\nu)}, \hat{\boldsymbol{f}}_{t|t}^{(\chi)}$ per slot t; see Theorem 1. Conversely, $\hat{\boldsymbol{\theta}}_t^{(\chi)}, \hat{\boldsymbol{\theta}}_t^{(\nu)}$ can be obtained for fixed $\{\boldsymbol{f}_\tau^{(\nu)}, \boldsymbol{f}_\tau^{(\chi)}\}_{\tau=1}^t$ as specified next.

Theorem 2: Consider minimizing (15) with respect to $\boldsymbol{\theta}^{(\chi)}$ and $\boldsymbol{\theta}^{(\nu)}$ for fixed $\boldsymbol{f}_\tau^{(\chi)} = \hat{\boldsymbol{f}}_{\tau|\tau}^{(\chi)}$ and $\boldsymbol{f}_\tau^{(\nu)} = \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}$, $\tau = 1, \ldots, t$, where $\{\hat{\boldsymbol{f}}_{\tau|\tau}^{(\chi)}, \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}\}_{\tau=1}^t$ are given and not necessarily the global minimizers of (15) with respect to $\{\boldsymbol{f}_\tau^{(\chi)}, \boldsymbol{f}_\tau^{(\nu)}\}_{\tau=1}^t$. Let $\tilde{\boldsymbol{f}}_{\tau|\tau}^{(\chi)} := \hat{\boldsymbol{f}}_{\tau|\tau}^{(\chi)} - \boldsymbol{A}_{(\tau,\tau-1)}\hat{\boldsymbol{f}}_{\tau-1|\tau-1}^{(\chi)}$, $\tau = 2, \ldots, t$, as well as $\boldsymbol{R}_t^{(\nu)} = \frac{1}{t}\sum_{\tau=1}^t \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}\hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)^\top}$ and $\boldsymbol{R}_t^{(\chi)} = \frac{1}{t}\sum_{\tau=1}^t \tilde{\boldsymbol{f}}_{\tau|\tau}^{(\chi)}\tilde{\boldsymbol{f}}_{\tau|\tau}^{(\chi)^\top}$. Then, the minimizers of (15) with respect to $\boldsymbol{\theta}^{(\nu)}$ and $\boldsymbol{\theta}^{(\chi)}$ are

$$\hat{\boldsymbol{\theta}}_t^{(\nu)} = \underset{\boldsymbol{\theta}^{(\nu)}\succeq\boldsymbol{0}}{\arg\min}\, \mathrm{Tr}\{\boldsymbol{R}_t^{(\nu)}\boldsymbol{K}^{(\nu)^{-1}}(\boldsymbol{\theta}^{(\nu)})\} + \frac{\rho_\nu}{\mu_2}\|\boldsymbol{\theta}^{(\nu)}\|_2^2 \quad (16a)$$

$$\hat{\boldsymbol{\theta}}_t^{(\chi)} = \underset{\boldsymbol{\theta}^{(\chi)}\succeq\boldsymbol{0}}{\arg\min}\, \mathrm{Tr}\{\boldsymbol{R}_t^{(\chi)}\boldsymbol{K}^{(\chi)^{-1}}(\boldsymbol{\theta}^{(\chi)})\} + \frac{\rho_\chi}{\mu_1}\|\boldsymbol{\theta}^{(\chi)}\|_2^2. \quad (16b)$$

Proof: To prove (16a), keep in (15) only those terms that depend on $\boldsymbol{\theta}^{(\nu)}$, and replace $\{\boldsymbol{f}_\tau^{(\nu)}\}_{\tau=1}^t$ with $\{\hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}\}_{\tau=1}^t$. Then, the objective in (15) reduces to $(1/t)\sum_{\tau=1}^t \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)^\top}\boldsymbol{K}^{(\nu)^{-1}}(\boldsymbol{\theta}^{(\nu)})\hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)} + (\rho_\nu/\mu_2)\|\boldsymbol{\theta}^{(\nu)}\|_2^2$. Next, using the linearity and cyclic invariance of the trace it follows that $\mathrm{Tr}\{(1/t)\sum_{\tau=1}^t \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)^\top}\boldsymbol{K}^{(\nu)^{-1}}(\boldsymbol{\theta}^{(\nu)})\hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}\} = \mathrm{Tr}\{(1/t)\sum_{\tau=1}^t \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}\hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)^\top}\boldsymbol{K}^{(\nu)^{-1}}(\boldsymbol{\theta}^{(\nu)})\} = \mathrm{Tr}\{\boldsymbol{R}_t^{(\nu)}\boldsymbol{K}^{(\nu)^{-1}}(\boldsymbol{\theta}^{(\nu)})\}$, which proves (16a). The proof of (16b) follows along the same lines. ■

Algorithm 2: Multi-kernel KriKF (**MKriKF**).

Input: $\mathcal{D}^{(\nu)}$; $\mathcal{D}^{(\chi)}$; $\boldsymbol{L}=\boldsymbol{U}^\top \operatorname{diag}\{\boldsymbol{\lambda}\}\boldsymbol{U}$.

Initialize: $\hat{\boldsymbol{\theta}}_0^{(\nu)}=\hat{\boldsymbol{\theta}}_0^{(\chi)}=[1,0,\ldots,0]$, $\hat{\boldsymbol{f}}_{0|0}^{(\chi)}=\mathbf{0}$,
$\boldsymbol{M}_{0|0}=\frac{1}{\mu_1}\boldsymbol{K}^{(\chi)}[1]$,
$\boldsymbol{\lambda}^{(\nu)}[m]:=\operatorname{diag}\left\{\boldsymbol{U}\boldsymbol{K}^{(\nu)}[m]\boldsymbol{U}^\top\right\}\forall m$,
$\boldsymbol{\lambda}^{(\chi)}[m]:=\operatorname{diag}\left\{\boldsymbol{U}\boldsymbol{K}^{(\chi)}[m]\boldsymbol{U}^\top\right\}\forall m$.
for $t=1,2,\ldots$ **do**
  Input: $\boldsymbol{A}_{(t,t-1)}\in\mathbb{R}^{N\times N}$; $\boldsymbol{y}_t\in\mathbb{R}^{S_t}$; $\boldsymbol{S}_t\in\{0,1\}^{S_t\times N}$.
  $\boldsymbol{K}_t^{(\nu)}=\boldsymbol{K}^{(\nu)}(\hat{\boldsymbol{\theta}}_t^{(\nu)})$
  $\boldsymbol{K}_t^{(\chi)}=\boldsymbol{K}^{(\chi)}(\hat{\boldsymbol{\theta}}_t^{(\chi)})$
  $\{\hat{\boldsymbol{f}}_{t|t}^{(\nu)},\hat{\boldsymbol{f}}_{t|t}^{(\chi)}\}=\text{KeKriKF}(\boldsymbol{K}_{t-1}^{(\chi)},\boldsymbol{K}_{t-1}^{(\nu)},\boldsymbol{A}_{(t,t-1)},\boldsymbol{y}_t,\boldsymbol{S}_t,\hat{\boldsymbol{f}}_{t-1|t-1}^{(\chi)},\boldsymbol{M}_{t-1|t-1})$
  Update $\boldsymbol{R}_t^{(\nu)}$ and $\boldsymbol{R}_t^{(\chi)}$
  $\boldsymbol{T}_t^{(\nu)}=\boldsymbol{U}^\top\boldsymbol{R}_t^{(\nu)}\boldsymbol{U}$
  $\boldsymbol{T}_t^{(\chi)}=\boldsymbol{U}^\top\boldsymbol{R}_t^{(\chi)}\boldsymbol{U}$
  $\hat{\boldsymbol{\theta}}_t^{(\nu)}=\text{OKM}(\{\boldsymbol{\lambda}^{(\nu)}[m]\}_{m=1}^{M_\nu},\boldsymbol{T}_t^{(\nu)},\hat{\boldsymbol{\theta}}_{t-1}^{(\nu)})$
  $\hat{\boldsymbol{\theta}}_t^{(\chi)}=\text{OKM}(\{\boldsymbol{\lambda}^{(\chi)}[m]\}_{m=1}^{M_\chi},\boldsymbol{T}_t^{(\chi)},\hat{\boldsymbol{\theta}}_{t-1}^{(\chi)})$
  Output: $\hat{\boldsymbol{f}}_{t|t}^{(\chi)}$; $\hat{\boldsymbol{f}}_{t|t}^{(\nu)}$; $\boldsymbol{M}_{t|t}$.
end for

Thus, Theorem 2 simplifies the objective that has to be minimized to find $\hat{\boldsymbol{\theta}}_t^{(\chi)}$ and $\hat{\boldsymbol{\theta}}_t^{(\nu)}$. With $\boldsymbol{K}(\boldsymbol{\theta})=\sum_{m=1}^M\theta[m]\boldsymbol{K}[m]$, problems (16a) and (16b) are of the form

$$\hat{\boldsymbol{\theta}}=\underset{\boldsymbol{\theta}\geq\mathbf{0}}{\arg\min}\ \operatorname{Tr}\{\boldsymbol{R}\boldsymbol{K}^{-1}(\boldsymbol{\theta})\}+\rho\|\boldsymbol{\theta}\|_2^2 \tag{17}$$

for some $\boldsymbol{R}\in\mathbb{R}^{N\times N}$, $\rho\geq 0$, and $\mathcal{D}=\{\boldsymbol{K}[m]\}_{m=1}^M$. Due to their resemblance to *covariance matching* [20], problem (17), and hence (16a) and (16b) will be referred to as *kernel matching*.

Theorem 2 suggests an online AM procedure to approximate the solution to (15), where Algorithm 1 and a solver for (17) termed online kernel matching (OKM) are executed alternatingly. This is summarized as Algorithm 2, and it is termed multi-kernel KriKF (MKriKF). Algorithm 2 does not generally find a global optimum of (15); yet, finding such an optimum may not be critical in practice, since it cannot be computed in polynomial time.

The rest of this section develops the OKM algorithm for solving (17) when $\mathcal{D}$ comprises Laplacian kernels. The first step is to exploit the fact that all Laplacian kernel matrices associated with a given graph have common eigenvectors.

Proposition 1: Consider the eigenvalue decompositions $\{\boldsymbol{K}[m]=\boldsymbol{U}\,diag\,\{\boldsymbol{\lambda}[m]\}\,\boldsymbol{U}^\top\}_{m=1}^M$ *and let* $\boldsymbol{T}:=\boldsymbol{U}^\top\boldsymbol{R}\boldsymbol{U}$. *Upon defining* $\boldsymbol{\Lambda}(\boldsymbol{\theta}):=diag\{\sum_{m=1}^M\theta[m]\boldsymbol{\lambda}[m]\}$ *and* $\phi(\boldsymbol{\theta}):=\operatorname{Tr}\left(\boldsymbol{T}\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})\right)+\rho\|\boldsymbol{\theta}\|_2^2$, *(17) can be equivalently written as*

$$\hat{\boldsymbol{\theta}}=\underset{\boldsymbol{\theta}\succeq\mathbf{0}}{\arg\min}\ \phi(\boldsymbol{\theta}) \tag{18}$$

Proof: Since $\boldsymbol{K}(\boldsymbol{\theta})=\sum_m^M\theta[m]\boldsymbol{U}\operatorname{diag}\{\boldsymbol{\lambda}[m]\}\boldsymbol{U}^\top=\boldsymbol{U}\boldsymbol{\Lambda}(\boldsymbol{\theta})\boldsymbol{U}^\top$, (18) follows by noting that $\operatorname{Tr}\{\boldsymbol{R}\boldsymbol{K}^{-1}(\boldsymbol{\theta})\}=\operatorname{Tr}\{\boldsymbol{R}\boldsymbol{U}\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})\boldsymbol{U}^\top\}=\operatorname{Tr}\{\boldsymbol{U}^\top\boldsymbol{R}\boldsymbol{U}\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})\}=\operatorname{Tr}\{\boldsymbol{T}\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})\}$. ■

Algorithm 3: Online Kernel Matching (**OKM**).

Input: $\{\boldsymbol{\lambda}[m]\}_{m=1}^M$; $\boldsymbol{T}_t\in\mathbb{S}_+^N$; $\hat{\boldsymbol{\theta}}_{t-1}\in\mathbb{R}_+^M$.
Initialize: $\boldsymbol{\theta}^0=\hat{\boldsymbol{\theta}}_{t-1}$,
while stopping_criterion not met **do**
  $\boldsymbol{\theta}^{k+1}=\left[\boldsymbol{\theta}^k-s^k\nabla\phi(\boldsymbol{\theta}^k)\right]^+$
  $k\leftarrow k+1$
end while
Output: $\hat{\boldsymbol{\theta}}_t$.

Proposition 1 establishes that (17) can be expressed as (18) when the kernels in $\mathcal{D}$ share eigenvectors, as is the case of Laplacian kernels; cf. Section II-A.

Proposition 2: When $\boldsymbol{\theta}\succeq\mathbf{0}$, *function* $\phi(\boldsymbol{\theta})$ *is strongly convex and differentiable with gradient*

$$\nabla\phi(\boldsymbol{\theta})=\boldsymbol{v}(\boldsymbol{\theta})+2\rho\boldsymbol{\theta} \tag{19}$$

where $\boldsymbol{v}(\boldsymbol{\theta}):=-[Tr\{diag\{\tilde{\boldsymbol{\lambda}}[1]\}\boldsymbol{T}\},\ldots,Tr\{diag\{\tilde{\boldsymbol{\lambda}}[M]\}\boldsymbol{T}\}]$, *with* $\tilde{\boldsymbol{\lambda}}[m]:=[\tilde{\lambda}_1[m],\ldots,\tilde{\lambda}_N[m]]^\top$ *and* $\tilde{\lambda}_n[m]:=\lambda_n[m]/(\sum_{\mu=1}^M\theta[\mu]\lambda_n[\mu])^2$.

Proof: Because $\boldsymbol{T}$ is a positive semidefinite matrix and $\boldsymbol{\lambda}[m]\succeq\mathbf{0}\,\forall m$, it can be easily seen that $\operatorname{Tr}\left\{\boldsymbol{T}\boldsymbol{\lambda}^{-1}(\boldsymbol{\theta})\right\}$ is convex over $\boldsymbol{\theta}\succeq\mathbf{0}$. And since $\rho\|\boldsymbol{\theta}\|_2^2$ is strongly convex, it follows by its definition that $\phi(\boldsymbol{\theta})$ is strongly convex. To obtain the gradient observe that

$$\frac{\partial\phi}{\partial\theta[m]}=-\operatorname{Tr}\left\{\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})\operatorname{diag}\{\boldsymbol{\lambda}[m]\}\,\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})\boldsymbol{T}\right\}+2\rho\theta[m] \tag{20}$$

and $\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})\operatorname{diag}\{\boldsymbol{\lambda}[m]\}\,\boldsymbol{\Lambda}^{-1}(\boldsymbol{\theta})=\operatorname{diag}\{\tilde{\boldsymbol{\lambda}}[m]\}$. ■

As (18) entails a strongly convex and differentiable objective, and projections on its feasible set are easy to obtain, we are motivate to solve (18) through projected gradient descent (PGD) [6]. Besides its simplicity, PGD converges linearly to the global minimum of (18). The general PGD iteration is

$$\boldsymbol{\theta}^{k+1}=\left[\boldsymbol{\theta}^k-s^k\nabla\phi(\boldsymbol{\theta}^k)\right]^+,\ k=0,1,\ldots \tag{21}$$

where s^k is the stepsize chosen e.g. by the Armijo rule [6], $\boldsymbol{\theta}^0$ is a feasible initial step, and $\left[\cdot\right]^+$ denotes projection on the nonnegative orthant $\{\boldsymbol{\theta}:\theta[m]\geq 0,\ m=1,\ldots,M\}$. The overall algorithm is termed OKM, and it is listed as Algorithm 3.

Observe that $\boldsymbol{\theta}^0$ in Algorithm 3 is initialized with its output in the previous iterate, namely $\hat{\boldsymbol{\theta}}_{t-1}$. This is a warm start that considerably speeds up convergence of Algorithm 3 since $\phi(\boldsymbol{\theta})$ is expected to change slowly across the iterations in Algorithm 2. An interesting byproduct of the OKM algorithm is its ability to adapt to changes in the spatio-temporal dynamics of the graph functions by adjusting the coefficients $\{\hat{\boldsymbol{\theta}}_t^{(\nu)},\hat{\boldsymbol{\theta}}_t^{(\chi)}\}_t$, and consequently the kernel matrices.

In view of Proposition 2, finding each entry of $\nabla\phi(\boldsymbol{\theta})$ in Algorithm 3 requires $\mathcal{O}(N)$ operations. Computing the gradient through (19) exploits the common eigenvectors of $\{\boldsymbol{K}[m]\}_{m=1}^M$, and avoids the inversion of the $N\times N$ matrix $\boldsymbol{K}(\boldsymbol{\theta})$ that is required when calculating the gradient for the general formulation (17), where $\{\boldsymbol{K}[m]\}_{m=1}^M$ need not share eigenvectors.

The complexity of evaluating the gradient is therefore reduced from a prohibitive $\mathcal{O}(N^3M)$ for general kernels to an affordable $\mathcal{O}(NM)$ for Laplacian kernels, which amounts to considerable computational savings especially for large-scale networks. With K denoting the number of PGD iterations for convergence, the overall computational complexity of OKM is therefore $\mathcal{O}(NMK)$. Typically, $N^3 \geq NMK$ and hence the complexity of Algorithm 2 is $\mathcal{O}(N^3)$, while learning the appropriate linear combination of kernels through MKL does not increase the complexity order that can be further reduced as suggested in Remark 2.

Selecting the dictionary and its size clearly depends on the amount of prior information available, and the complexity that can be afforded by the MKL optimization that follows up. Desirable attributes such as smoothness, bandlimitedness, and diffusion effects can prompt inclusion of corresponding kernels over a grid of their parameters - the case present in our simulation tests.

Remark 3: The algorithms in this section adopted a fixed kernel dictionary over time, namely $\mathcal{D} = \{\boldsymbol{K}[m] \in \mathbb{S}_+^N\}_{m=1}^M$. If the topology changes over time, the Laplacian kernel matrices change as well, cf. (4). To accommodate this scenario, one can restart Algorithm 2 whenever the topology changes, say at time t_c, and initialize $\hat{\boldsymbol{f}}_{0|0}^{(\chi)} \leftarrow \hat{\boldsymbol{f}}_{t_c|t_c}^{(\chi)}$, $\boldsymbol{M}_{0|0} \leftarrow \boldsymbol{M}_{t_c|t_c}$, as well as replace the Laplacian kernels in $\mathcal{D}$ with the ones corresponding to the new topology.

Remark 4: To accommodate a certain degree of nonstationarity one may consider using the following matrices

$$\tilde{\boldsymbol{R}}_t^{(\nu)} = \sum_{\tau=1}^{t} \gamma_\nu^{t-\tau} \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)} \hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)^\top} + \gamma_\nu^t \boldsymbol{I} \tag{22a}$$

$$\tilde{\boldsymbol{R}}_t^{(\chi)} = \sum_{\tau=1}^{t} \gamma_\chi^{t-\tau} \tilde{\boldsymbol{f}}_{\tau|\tau}^{(\chi)} \tilde{\boldsymbol{f}}_{\tau|\tau}^{(\chi)^\top} + \gamma_\chi^t \boldsymbol{I} \tag{22b}$$

instead of $\boldsymbol{R}_t^{(\nu)}$ and $\boldsymbol{R}_t^{(\chi)}$, where $\gamma_\chi, \gamma_\nu \in (0,1)$ are forgetting factors that weigh exponentially past observations, and ensure invertibility of matrices $\tilde{\boldsymbol{R}}_t^{(\nu)}$ and $\tilde{\boldsymbol{R}}_t^{(\chi)}$. Moreover, $\tilde{\boldsymbol{R}}_t^{(\nu)}$ and $\tilde{\boldsymbol{R}}_t^{(\chi)}$ can be updated recursively as

$$\tilde{\boldsymbol{R}}_t^{(\nu)} = \gamma_\nu \tilde{\boldsymbol{R}}_{t-1}^{(\nu)} + \hat{\boldsymbol{f}}_{t|t}^{(\nu)} \hat{\boldsymbol{f}}_{t|t}^{(\nu)^\top} \tag{23a}$$

$$\tilde{\boldsymbol{R}}_t^{(\chi)} = \gamma_\chi \tilde{\boldsymbol{R}}_{t-1}^{(\chi)} + \tilde{\boldsymbol{f}}_{t|t}^{(\chi)} \tilde{\boldsymbol{f}}_{t|t}^{(\chi)^\top} \tag{23b}$$

which significantly reduces the required memory for the computation with respect to (22), since $\{\hat{\boldsymbol{f}}_{\tau|\tau}^{(\nu)}, \tilde{\boldsymbol{f}}_{\tau|\tau}^{(\chi)}\}_{\tau=1}^{t-1}$ need not be stored.

Remark 5: The algorithms presented in this paper can be generalized to account for a VARM of order L, $\boldsymbol{f}_t^{(\chi)} = \sum_{l=1}^{L} \boldsymbol{A}_{(t,t-l)} \boldsymbol{f}_{t-l}^{(\chi)} + \boldsymbol{\eta}_t$. Towards that end, consider the $LN \times 1$ extended state vector $\bar{\boldsymbol{f}}_t^{(\chi)} := [(\bar{\boldsymbol{f}}_t^{(\chi)})^\top, \ldots (\bar{\boldsymbol{f}}_{t-L+1}^{(\chi)})^\top]^\top$, the $S_t \times LN$ matrix $\bar{\boldsymbol{S}}_t := [\boldsymbol{S}_t, \boldsymbol{0}, \ldots, \boldsymbol{0}]$, the $LN \times LN$ matrices $\bar{\boldsymbol{A}}_{(t,t-1)}$ with block entries $\{[\bar{\boldsymbol{A}}_{(t,t-1)}]_{1,l} = \boldsymbol{A}_{(t,t-l)}\}_{l=1}^{L}$, $\{[\bar{\boldsymbol{A}}_{(t,t-1)}]_{l,l} = \boldsymbol{I}_N\}_{l=2}^{L}$, and the rest zero, and $\bar{\boldsymbol{K}}_t^{(\chi)}$ with block entries $[\bar{\boldsymbol{K}}_t^{(\chi)}]_{1,1} = \boldsymbol{K}_t^{(\chi)}$, $\{[\bar{\boldsymbol{K}}_t^{(\chi)}]_{l,l} = \boldsymbol{I}_N\}_{l=2}^{L}$, and the rest zero. The KeKriKF algorithm can then be readily applied after replacing the pertinent matrices and vectors with their extended versions. Having established that Algorithm 1 can accommodate multi-lag dependencies, the extension of Algorithm 2 follows, as Algorithm 3 is not affected by the extended state.

Remark 6: One may ponder on the role of graphs in our KRR formulation with the regression coefficient vector obeying a linear dynamical model. Our graph-based formulation is well motivated not only because several physical networks are represented by *graphs* having known connectivity, but also because graphical models are known to represent effectively probabilistic dependencies among nodal vectors. Sure, one could have *a fortiori* assumed knowledge of the needed covariance (or kernel) matrices that are not generally available. Here, we employ kernel matrices that are functions of the graph adjacency matrices. In so doing, we further endow our formulation in the form of regularization terms with graph-related properties that can be present. Those include smoothness, (block) sparsity, low rank, and diffusion effects. Although the emphasis here is on leveraging these properties on graphs the advocated MKriKF approach can be indeed useful in various spatio-temporal estimation tasks involving signals not necessarily evolving over graphs, so long as the underlying (cross-)covariance matrices can become available. Finally, our OKM algorithm leverages the common eigenvectors of the *graph-induced* kernel matrix to reduce complexity. All in all, our graph-based formulation facilitates incorporation of graph-specific prior information.

V. Simulations

This section evaluates the performance of the developed algorithms by means of numerical tests with synthetic and real data. The proposed algorithms are compared with: (i) The least mean-square (LMS) algorithm in [15] with step size μ_{LMS}; and (ii) the distributed least-squares reconstruction (DLSR) algorithm [30] with step sizes μ_{DLSR} and β_{DLSR}. Both LMS and DLSR can track slowly time-varying B-bandlimited graph signals.

The performance of the aforementioned approaches is quantified through the normalized mean-square error (NMSE)

$$\text{NMSE} := \frac{\mathbb{E}\big[\sum_{\tau=1}^{t} \|\boldsymbol{S}_\tau^c(\boldsymbol{f}_\tau - \hat{\boldsymbol{f}}_{\tau|\tau})\|_2^2\big]}{\mathbb{E}\big[\sum_{\tau=1}^{t} \|\boldsymbol{S}_\tau^c \boldsymbol{f}_\tau\|_2^2\big]}$$

where the expectation is taken over the sample locations, and $\boldsymbol{S}_\tau^c$ is an $(N - S_\tau) \times N$ matrix comprising the rows of $\boldsymbol{I}_N$ whose indices are not in $\mathcal{S}_t$, which means that the test set in all experiments is $\mathcal{V} \setminus \mathcal{S}_t, \forall t$. Unless otherwise stated, $\mathcal{S}_t$ is chosen uniformly at random without replacement over $\mathcal{V}$, and kept constant over time; that is, $\mathcal{S}_t = \mathcal{S}$, per t, in order to compare on equal footing the competing algorithms DLSR [30] and LMS [15] that cannot cope with time-varying $\mathcal{S}_t$. For all tests 10 fold cross-validation has been employed. The training set $\mathcal{S}_t$ was split in 10 subsets, out of which 9 were used for training and 1 for validation. The validation error was averaged over the 10 subsets, and the set of parameters exhibiting the smallest error was selected. To reduce the search space, we set $\mu_1 = \mu_2$ and $\rho_\nu = \rho_\chi$. The test set is in all cases $\mathcal{V} \setminus \mathcal{S}_t, \forall t$. Notice that our MKriKF,

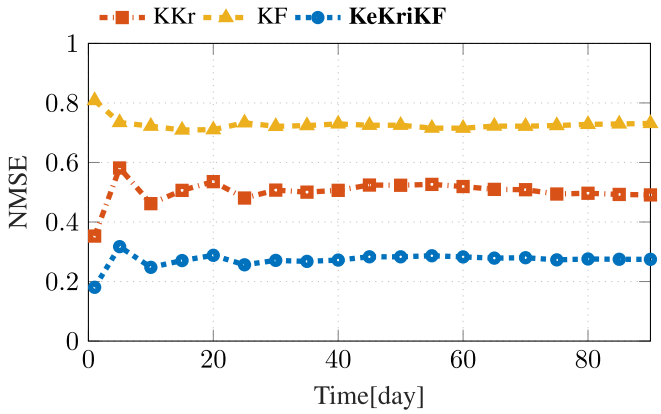


Fig. 2. NMSE of function estimates ($\mu_1 = \mu_2 = 1$).

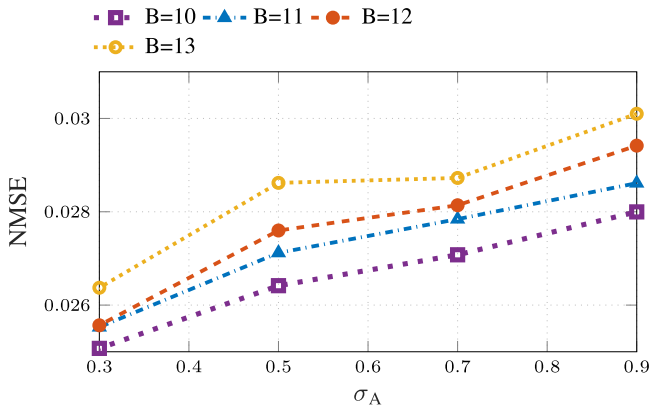


Fig. 3. NMSE of KeKriKF for different time-varying graphs ($S = 65$, $\mu_1 = \mu_2 = 1$).

which learns the kernel that "best" fits the data, requires minimal parameter tuning.

A. Numerical Tests on Synthetic Data

To construct a graph, consider the dataset in [4], which contains timestamped messages among students at the University of California, Irvine, exchanged over a social network during 90 days. The sampling interval t is one day. A graph is constructed such that the edge weight $A_{n,n'}(t)$ counts the number of messages exchanged between student n and n' in the k-th month, where $k = 1, 2, 3$ and $30(k-1)+1 \leq t \leq 30k$. Hence, $\boldsymbol{A}_t$ changes across months. A subset of $N = 310$ users for which $\boldsymbol{A}_t$ corresponds to a connected graph $\forall t$ is selected. At each t, $\boldsymbol{f}_t$ was generated by superimposing a B-bandlimited graph function with $B = 5$ and a spatio-temporally correlated signal. Specifically, $\boldsymbol{f}_t = \boldsymbol{f}_t^{(\nu)} + \boldsymbol{f}_t^{(\chi)} = \sum_{i=1}^{5} \gamma_t^i \boldsymbol{u}_t^i + \boldsymbol{f}_t^{(\chi)}$, where $\{\gamma_t^i\}_{i=1}^5 \sim \mathcal{N}(0,1)$ for all t, while $\{\boldsymbol{u}_t^i\}_{i=1}^5$ denote the eigenvectors associated with the 5 smallest eigenvalues of $\boldsymbol{L}_t$, and $\boldsymbol{f}_t^{(\chi)}$ is generated according to (9) with $\boldsymbol{A}_{(t,t-1)} = 0.03(\boldsymbol{A}_{t-1} + \boldsymbol{I}_N)$, $\boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_\eta)$, and $\boldsymbol{C}_\eta$ is a diffusion kernel with $\sigma = 0.5$. Function $f(v_n, t)$ is therefore smooth with respect to the graph and can be interpreted e.g. as the time that the n-th student spends on the specific social network during the t-th day.

The first experiment justifies the proposed decomposition by assessing the impact of dropping either $\boldsymbol{f}_t^{(\nu)}$ or $\boldsymbol{f}_t^{(\chi)}$ from the right hand side of (8). The KriKF algorithm uses diffusion kernels $\boldsymbol{K}_t^{(\nu)}$ and $\boldsymbol{K}_t^{(\chi)}$ with parameters $\sigma = 1.5$ and $\sigma = 0.5$, respectively.

Fig. 2 depicts the NMSE with $S = 217$ for the KeKriKF; the Kalman filter (KF) estimator, which results from setting $\boldsymbol{f}_t^{(\nu)} = \boldsymbol{0}$ for all t in the KeKriKF; as well as kernel Kriging (KKr), which the KeKriKF reduces to if $\boldsymbol{f}_t^{(\chi)} = \boldsymbol{0}$ for all t. As observed, KeKriKF, which accounts for both summands in (8), outperforms those algorithms that account for only one of them. Moreover, the low NMSE of KeKriKF in reconstructing the $N - S = 310 - 217 = 93$ unavailable node values reveals that this algorithm is capable of efficiently capturing the spatial as well as the temporal dynamics over time-varying topologies.

Next, the robustness of KeKriKF is evaluated when the connectivity of $\mathcal{G}_t$, captured by $\boldsymbol{A}_t$, exhibits abrupt changes over t. Synthetic time-varying networks of size $N = 81$ were generated using the Kronecker product model, which effectively captures properties of real graphs [14]. The prescribed "seed matrix"

$$\boldsymbol{D}_0 := \begin{bmatrix} 1 & 0.1 & 0.7 \\ 0.3 & 0.1 & 0.5 \\ 0 & 1 & 0.1 \end{bmatrix}$$

produces the $N \times N$ matrix $\boldsymbol{D} := \boldsymbol{D}_0 \otimes \boldsymbol{D}_0 \otimes \boldsymbol{D}_0 \otimes \boldsymbol{D}_0$, where $\otimes$ denotes the Kronecker product. An initial adjacency matrix $\boldsymbol{A}_0$ was constructed with entries $A_{n,n'}(0)\,\forall n$, $A_{n,n'}(0) \sim \text{Bernoulli}(D_{n,n'})$ for $n > n'$, and $A_{n,n'}(0) = A_{n',n}(0)$ for $n < n'$. Next, the following time-varying graph model was generated: at each $t_c = 10\kappa$, $\kappa = 1, 2, \ldots$, each entry of $\boldsymbol{A}_{t_c}$ changes with probability $p_{n,n'} = \sum_k A_{n,k}(t_c) \sum_l A_{l,n'}(t_c) / \sum_k \sum_l A_{k,l}(t_c)$ as $A_{n,n'}(t_c+1) = A_{n,n'}(t_c) + |\xi_{n,n'}(t_c)|$ for $n > n'$ where $\xi_{n,n'}(t_c) \sim \mathcal{N}(0, \sigma_A)$ and $A_{n',n}(t_c+1) = A_{n,n'}(t_c+1)$ for $n < n'$. This choice of $p_{n,n'}$ is based on the "rich get richer" attribute of real networks, where new connections are formed between nodes with high degree [14]. Moreover, the edge $(v_n, v_{n'})$ is deleted at each $t_d = 20\kappa$, $\kappa = 1, 2, \ldots$ with probability 0.1; that is, $A_{n',n}(t_d+1) = A_{n,n'}(t_d+1) = 0$, as long as the graph remains connected. By varying σ_A, we obtain different time-varying graphs. A graph function was generated for each time-varying graph as follows

$$\boldsymbol{f}_t = \delta \boldsymbol{A}_t \boldsymbol{f}_{t-1} + \sum_{i=1}^{10} \gamma_t^{(i)} \boldsymbol{u}_t^{(i)} \tag{24}$$

where $\delta = 10^{-2}$ is a forgetting factor, $\sum_{i=1}^{10} \gamma_t^{(i)} \boldsymbol{u}_t^{(i)}$ is a graph-bandlimited component with $\gamma_t^{(i)} \sim \mathcal{N}(0,1)$, and $\{\boldsymbol{u}_t^{(i)}\}_{i=1}^{10}$ are the eigenvectors associated with the 10 smallest eigenvalues of $\boldsymbol{L}_t$. Algorithm 1 employs a bandlimited kernel with $\beta = 10^3$ and B for $\boldsymbol{K}_t^{(\nu)}$, a diffusion kernel with $\sigma = 0.5$ for $\boldsymbol{K}_t^{(\chi)}$, and $\boldsymbol{A}_{(t,t-1)} = 10^{-3}(\boldsymbol{A}_{t-1} + \boldsymbol{I}_N)$. Fig. 3 plots the NMSE of the KeKriKF algorithm as a function of σ_A, which determines how

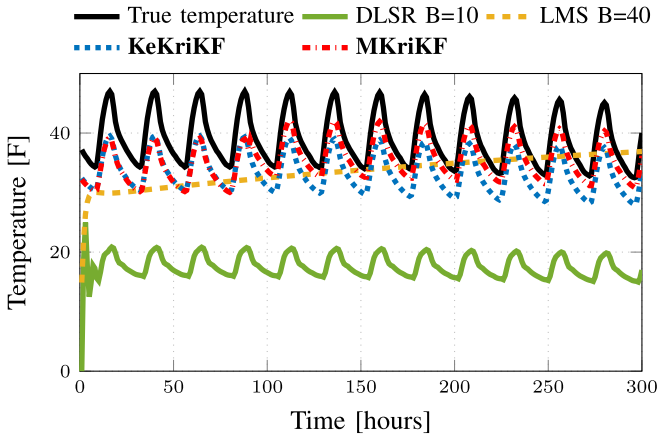


Fig. 4. True and estimated temperature values ($B = 5$, $\mu_{\text{DLSR}} = 1.2$, $\beta_{\text{DLSR}} = 0.5$, $\mu_{\text{LMS}} = 1.5$, $\mu_1 = \mu_2 = 1$, $\rho_\nu = 10^5$, $\rho_\chi = 10^5$).

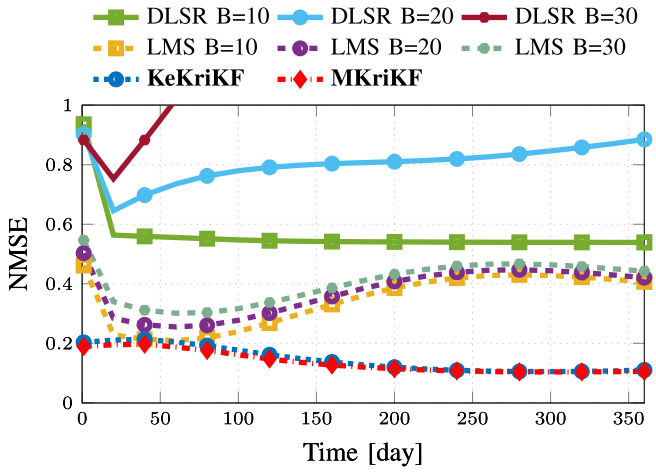


Fig. 5. NMSE of temperature estimates ($\mu_{\text{DLSR}} = 1.6$, $\beta_{\text{DLSR}} = 0.5$, $\mu_{\text{LMS}} = 1.5$, $\rho_\nu = 10^5$, $\rho_\chi = 10^5$).

rapidly the graph changes. As observed, the KeKriKF algorithm can effectively cope with different degrees of time variation.

B. Temperature Prediction

Consider the dataset [1] provided by the National Climatic Data Center, which comprises hourly temperature measurements at $N = 109$ measuring stations across the continental United States in 2010. A time-invariant graph was constructed as in [19], based on geographical distances. The value $f(v_n, t)$ represents the t-th temperature sample recorded at the n-th station. The sampling interval is one hour for the first experiment, and one day for the second.

KeKriKF employs diffusion kernels with parameter $\sigma = 1.8$ for $\boldsymbol{K}_t^{(\nu)} \boldsymbol{K}_t^{(\chi)} = 10^{-5} \boldsymbol{I}_N$, and a transition matrix $\boldsymbol{A}_{(t,t-1)} = 5 \cdot 10^{-4} (\boldsymbol{A}_{t-1} + \boldsymbol{I}_N)$. MKriKF is configured as follows: $\mathcal{D}^{(\nu)}$ contains $M_\nu = 40$ diffusion kernels with parameters $\{\sigma[m]\}_{m=1}^{40}$ with $\sigma[m] \sim \mathcal{N}(2, 0.5), \forall m$; $\mathcal{D}^{(\chi)}$ contains 44 diffusion kernels with parameters $\{\sigma[m]\}_{m=1}^{44}$, where $\sigma[m] \sim \mathcal{N}(1, 0.2), \forall m$, and an identity kernel $\boldsymbol{K}^{(\chi)}[45] = \boldsymbol{I}_N$.

Fig. 4 depicts the true temperature along with its estimates for a station n that is not sampled, meaning $n \notin \mathcal{S}$, with $S = 44$. Clearly, KeKriKF accurately tracks the temperature by exploiting spatial and temporal dynamics, but MKriKF outperforms KeKriKF by learning those dynamics from the data. The random sampling set selection heavily affects performance of the LMS algorithm; for adaptive selection of $\mathcal{S}$ see [15].

Fig. 5 compares the NMSE of all considered approaches for $S = 44$. Observe the superior performance of the proposed reconstruction methods, which in this scenario exhibit roughly the same NMSE.

C. GDP Prediction

The next dataset is provided by the World Bank Group [2], and comprises gross domestic product (GDP) per capita for $N = 127$ countries for the years 1960–2016. A time-invariant graph was constructed using the correlation between the GDP of different countries for the first 25 years. The graph function

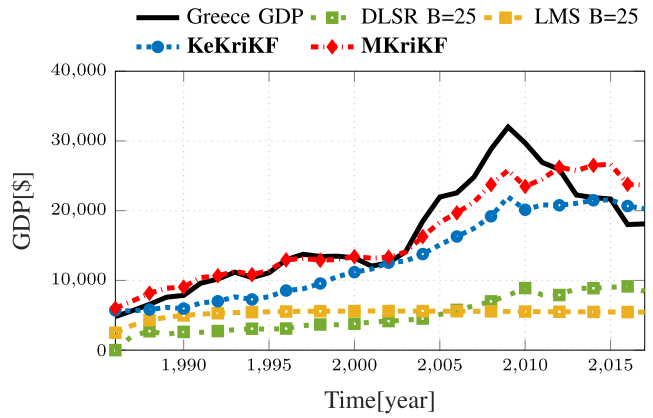


Fig. 6. Greece GDP values along with the estimated ones ($S = 38$, $\mu_{\text{DLSR}} = 1.6$, $\beta_{\text{DLSR}} = 0.4$, $\mu_{\text{LMS}} = 1.2$, $\rho_\nu = 10^4$, $\rho_\chi = 10^4$).

$f(v_n, t)$ denotes the GDP reported at the n-th country and t-th year for $t = 1985, \ldots, 2016$.

The graph Fourier transform of the GDP in the first 25 years defined as $\check{f}_n := \boldsymbol{u}_n^\top \boldsymbol{f} \; \forall n$, where $\boldsymbol{u}_n$ denotes the n-th eigenvector of the Laplacian matrix; see [26], shows that the graph frequencies $\check{f}_k$ take small values for $4 < k < 123$, and large values otherwise. Motivated by the aforementioned observation, the KeKriKF is configured with a band-reject kernel $\boldsymbol{K}^{(\nu)}$ with $k = 6, l = 6, \beta = 15$; see Table I, $\boldsymbol{K}^{(\chi)} = 10^{-3} \boldsymbol{I}_N$, and $\boldsymbol{A}_{(t,t-1)} = 10^{-5} (\boldsymbol{A}_{t-1} + \boldsymbol{I}_N)$. MKriKF adopts a $\mathcal{D}^{(\nu)}$ with $M_\nu = 16$ band-reject kernels with $k \in [2, 5]$, $l \in [1, 4]$, $\beta = 15$, and a $\mathcal{D}^{(\chi)}$ with 60 diffusion kernels with parameters $\{\sigma[m]\}_{m=1}^{60}$, where $\sigma[m] \sim \mathcal{N}(2, 0.5), \forall m$, and an identity kernel $\boldsymbol{K}^{(\chi)}[61] = \boldsymbol{I}_N$.

Fig. 6 depicts the actual GDP as well as its estimates for Greece, which is not contained in the sampled countries. Clearly, both MKriKF and KeKriKF, track the GDP evolution over the years with greater accuracy than the considered alternatives. This is expected because the graph function does not adhere to the graph bandlimited model assumed by DLSR and LMS.

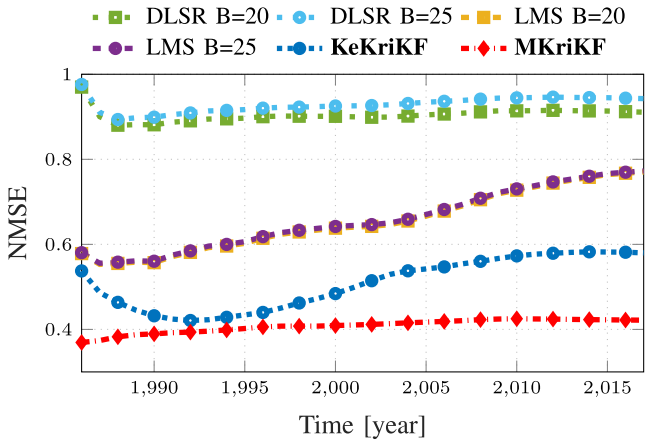


Fig. 7. NMSE of GDP estimates ($S = 38$, $\mu_{\text{DLSR}} = 1.6$, $\beta_{\text{DLSR}} = 0.4$, $\mu_{\text{LMS}} = 1.6$, $\rho_\nu = 10^4$, $\rho_\chi = 10^4$).

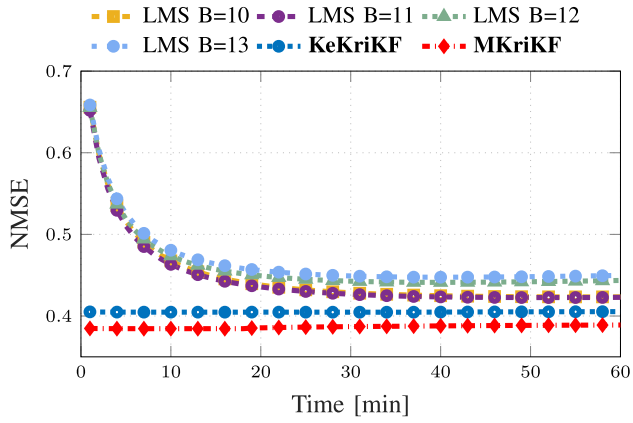


Fig. 8. NMSE of network delay estimates ($\mu_{\text{LMS}} = 1.5$, $c = 0.0005$, $\rho_\nu = 100$, $\rho_\chi = 100$, $\mu_1 = \mu_2 = 1$).

Fig. 7 reports NMSE over time, where the proposed algorithms achieve the smallest NMSE. The data-driven MKriKF outperforms KeKriKF, which is configured manually.

D. Network Delay Prediction

The last dataset records measurements of path delays on the Internet2 backbone [3]. The network comprises 9 end-nodes and 26 directed links. The delays are available for $N = 70$ paths at every minute. The paths connect origin-destination nodes by a series of links described by the path-link routing matrix $\mathbf{\Pi} \in \{0,1\}^{N\times 26}$, whose (n,l) entry is $\Pi_{n,l} = 1$ if path n' traverses link l, and 0 otherwise. A graph is constructed with each vertex corresponding to one of these paths, and with the time-invariant adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ given by

$$A_{n,n'} = \frac{\sum_{l=1}^{26} \Pi_{n,l}\Pi_{n',l}}{\sum_{l=1}^{26} \Pi_{n,l} + \sum_{l=1}^{26} \Pi_{n',l} - \sum_{l=1}^{26} \Pi_{n,l}\Pi_{n',l}} \quad (25)$$

for $n, n' = 1, \ldots, N$, $n \neq n'$. Expression (25) was selected to assign a greater weight to edges connecting vertices whose associated paths share a large number of links. This is intuitively reasonable since paths with common links usually experience similar delays [7]. Function $f(v_n, t)$ denotes the delay in milliseconds measured at the n-th path and t-th minute.

The KeKriKF algorithm employs a diffusion kernel with parameter $\sigma = 2.5$ for $\boldsymbol{K}_t^{(\nu)}$, $\boldsymbol{K}_t^{(\chi)} = 0.002\boldsymbol{I}_N$, and $\boldsymbol{A}_{(t,t-1)} = 0.005(\boldsymbol{A}_{t-1} + \boldsymbol{I}_N)$. The MKriKF is configured as follows: $\mathcal{D}^{(\nu)}$ contains $M_\nu = 40$ diffusion kernels with parameters $\{\sigma[m]\}_{m=1}^{40}$ with $\sigma[m] \sim \mathcal{N}(4, 0.5), \forall m$; $\mathcal{D}^{(\chi)}$ contains $M_\chi = 60$ diffusion kernels with parameters $\{\sigma[m]\}_{m=1}^{60}$ with $\sigma[m] \sim \mathcal{N}(1, 0.1), \forall m$, and an identity kernel $\boldsymbol{K}^{(\chi)}[61] = \boldsymbol{I}_N$.

Fig. 8 depicts the NMSE when $S = 20$. KeKriKF and MKriKF are seen to outperform competing methods. Using the parameters of Fig. 8, the MKriKF algorithm is tested for $\mathcal{S}_t$ chosen at random per slot t with $S_t = S, \forall t$, but $\mathcal{S}_t \neq \mathcal{S}_{t'}, \forall t \neq t'$. Fig. 9 shows the NMSE of MKriKF with variable S and as expected the performance improves as the number of samples increases. For the same configuration, Fig. 10 depicts the NMSE

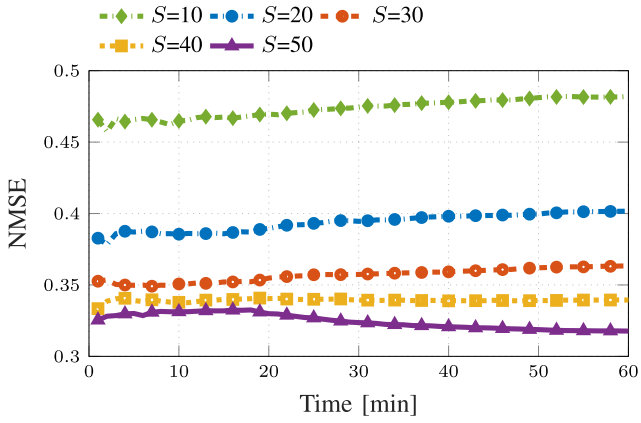


Fig. 9. NMSE of MKriKF with varying sampling set size ($\mu_1 = \mu_2 = 1$, $\rho_\nu = 100$, $\rho_\chi = 100$).

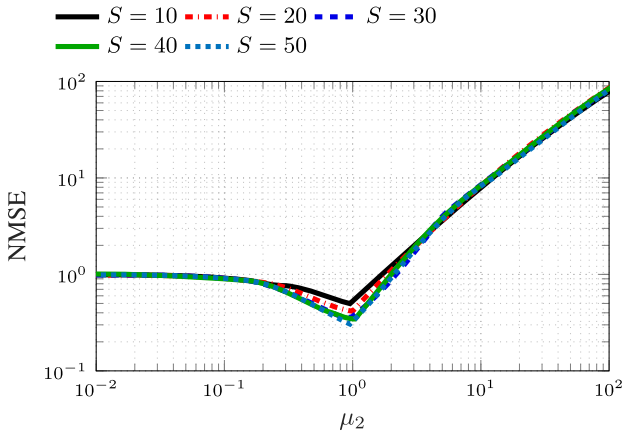
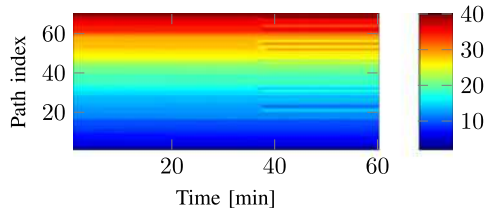


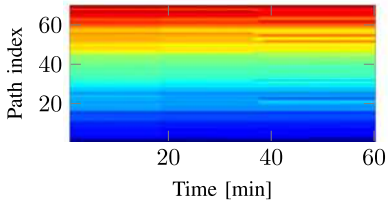
Fig. 10. NMSE of MKriKF for different μ_2 ($\mu_1 = 1$, $\rho_\nu = 100$, $\rho_\chi = 100$).

as μ_2 varies, and shows that the minimum NMSE for all S is at $\mu_2 = 1$.

Finally, the proposed MKriKF will be evaluated in tracking the delay over the network from $S = 56$ randomly sampled path delays. To that end, delay maps are traditionally employed, which depict the network delay per path over time and enable operators to perform troubleshooting; see also [18]. The paths



(a) True delay



(b) MKriKF

Fig. 11. True and estimated network delay map for $N = 70$ paths ($\rho_\chi = 100$, $\mu_1 = \mu_2 = 1$). (a) True delay (b) MKriKF.

for the delay maps in Fig. 11 are sorted in increasing order of the true delay at $t = 1$. Clearly, the delay map recovered by MKriKF in Fig. 11 (b) visually resembles the true delay map in Fig. 11 (a).

VI. Conclusion

This paper introduced online estimators to reconstruct dynamic functions over (possibly dynamic) graphs. In this context, the function to be estimated was decomposed in two parts: one capturing the spatial dynamics, and the other jointly modeling spatio-temporal dynamics by means of a state-space model. A novel kernel kriged Kalman filter was developed using a deterministic RKHS approach. To accommodate scenarios with limited prior information, an online multi-kernel learning technique was also developed to allow tracking of the spatio-temporal dynamics of the graph function. The structure of Laplacian kernels was exploited to achieve low computational complexity. Through numerical tests with synthetic as well as real-data, the novel algorithms were observed to perform markedly better than existing alternatives. Future work includes distributed implementations of the proposed algorithms, data-driven learning of $\boldsymbol{A}_{(t,t-1)}$, and exploring nonlinear dynamical models such as the sampled Brownian motion, the extended KF, unscented KF, or particle filters.

References

[1] "1981–2010 U.S. climate normals," [Online]. Available: https://www.ncdc.noaa.gov/data-access/land-based-station-data/land-based-datasets/climate-normals/1981-2010-normals-data, Accessed on: Sep. 2016.

[2] "GDP per capita (current US)," [Online]. Available: https://data.worldbank.org/indicator/NY.GDP.PCAP.CD, Accessed on: Sep. 2017.

[3] "One-way ping internet2," [Online]. Available: http://software.internet2.edu/owamp/, Accessed on: Sep. 2017.

[4] "Snap temporal networks: Collegemsg," [Online]. Available: http://snap.stanford.edu/data/CollegeMsg.html, Accessed Sep. 2017.

[5] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Trans. Signal Process.*, vol. 64, no. 14, pp. 3775–3789, Jul. 2016.

[6] D. Bertsekas, *Nonlinear Programming*. Belmont, MA, USA: Athena Scientific, 1999.

[7] D. B. Chua, E. D. Kolaczyk, and M. Crovella, "Network kriging," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 12, pp. 2263–2272, Dec. 2006.

[8] I. Csiszár and G. Tusnády, "Information geometry and alternating minimization procedures," *Stat. Decis., Supplement Issue, 1*, pp. 205–237, 1984.

[9] P. A. Forero, K. Rajawat, and G. B. Giannakis, "Prediction of partially observed dynamical processes over networks via dictionary learning," *IEEE Trans. Signal Process.*, vol. 62, no. 13, pp. 3305–3320, Jul. 2014.

[10] V. N. Ioannidis, A. N. Nikolakopoulos, and G. B. Giannakis, "Semi-parametric graph kernel-based reconstruction," in *Proc. Global Conf. Signal Inf. Process.*, Montreal, QC, Canada, Nov. 2017, pp. 588–592.

[11] V. N. Ioannidis, D. Romero, and G. B. Giannakis, "Inference of spatiotemporal processes over graphs via kernel kriged Kalman filtering," in *Proc. Eur. Signal Process. Conf.*, Kos, Greece, Aug. 2017, pp. 1679–1683.

[12] E. D. Kolaczyk, *Statistical Analysis of Network Data: Methods and Models*. New York, NY, USA: Springer, 2009.

[13] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *Proc. Int. Conf. Mach. Learn.*, Sydney, NSW, Australia, Jul. 2002, pp. 315–322.

[14] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *J. Mach. Learn. Res.*, vol. 11, pp. 985–1042, Feb. 2010.

[15] P. D. Lorenzo, S. Barbarossa, P. Banelli, and S. Sardellitti, "Adaptive least mean-square estimation of graph signals," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 4, pp. 555–568, Sep. 2016.

[16] H. Lütkepohl, *New Introduction to Multiple Time Series Analysis*. New York, NY, USA: Springer, 2005.

[17] K. V. Mardia, C. Goodall, E. J. Redfern, and F. J. Alonso, "The kriged Kalman filter," *Test*, vol. 7, no. 2, pp. 217–282, 1998.

[18] K. Rajawat, E. Dall'Anese, and G. B. Giannakis, "Dynamic network delay cartography," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2910–2920, Mar. 2014.

[19] D. Romero, V. N. Ioannidis, and G. B. Giannakis, "Kernel-based reconstruction of space-time functions on dynamic graphs," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 1–14, Sep. 2017.

[20] D. Romero and G. Leus, "Wideband spectrum sensing from compressed measurements using spectral prior information," *IEEE Trans. Signal Process.*, vol. 61, no. 24, pp. 6232–6246, Dec. 2013.

[21] D. Romero, M. Ma, and G. B. Giannakis, "Kernel-based reconstruction of graph signals," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 764–778, Feb. 2017.

[22] I. D. Schizas, G. B. Giannakis, S. I. Roumeliotis, and A. Ribeiro, "Consensus in ad hoc WSNs with noisy links—Part II: Distributed estimation and smoothing of random signals," *IEEE Trans. Signal Process.*, vol. 56, no. 4, pp. 1650–1666, Apr. 2008.

[23] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2002.

[24] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, "Reconstruction of graph signals through percolation from seeding nodes," *IEEE Trans. Signal Process.*, vol. 64, no. 16, pp. 4363–4378, Aug. 2016.

[25] Y. Shen, B. Baingana, and G. B. Giannakis, "Nonlinear structural vector autoregressive models for inferring effective brain network connectivity," arXiv:1610.06551v1, 2016.

[26] 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, no. 3, pp. 83–98, May 2013.

[27] A. J. Smola and R. I. Kondor, "Kernels and regularization on graphs," in *Learning Theory and Kernel Machines*, New York, NY, USA: Springer, 2003, pp. 144–158.

[28] G. Strang and K. Borre, *Linear Algebra, Geodesy, and GPS*. Philadelphia, PA, USA: SIAM, 1997.

[29] D. Thanou, D. I. Shuman, and P. Frossard, "Learning parametric dictionaries for signals on graphs," *IEEE Trans. Signal Process.*, vol. 62, no. 15, pp. 3849–3862, Aug. 2014.

[30] X. Wang, M. Wang, and Y. Gu, "A distributed tracking algorithm for reconstruction of graph signals," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 4, pp. 728–740, Feb. 2015.

[31] C. K. Wikle and N. Cressie, "A dimension-reduced approach to space-time Kalman filtering," *Biometrika*, vol. 86, pp. 815–829, 1999.

[32] D. Zhou and B. Schölkopf, "A regularization framework for learning from graph data," in *Proc. ICML Workshop Stat. Relational Learn. Connections Other Fields*, Banff, AB, Canada, Jul. 2004, vol. 15, pp. 67–68.

Vassilis N. Ioannidis (S'16) received the Diploma in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 2015, and the M.Sc. degree in electrical engineering from the University of Minnesota, Minneapolis, MN, USA, in 2017. He is currently working toward the Ph.D. degree at the Department of Electrical and Computer Engineering, University of Minnesota. From 2014 to 2015, he worked as a Middleware Consultant for Oracle in Athens, Greece. His research interests include machine learning, big data analytics, and network science. He was the recipient of the Performance Excellence award.

Daniel Romero (M'16) received the M.Sc. and Ph.D. degrees in signal theory and communications from the University of Vigo, Vigo, Spain, in 2011 and 2015, respectively. From July 2015 to November 2016, he was a Post-Doctoral Researcher with the Digital Technology Center and Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN, USA. In December 2016, he was an Associate Professor with the Department of Information and Communication Technology, University of Agder, Kristiansand, Norway. His main research interests include signal processing, communications, and machine learning.

Georgios B. Giannakis (F'97) received the Diploma in electrical engineering from the National Technical University of Athens, Athens, Greece, 1981. From 1982 to 1986, he was with the University of Southern California, where he received the MSc. degree in electrical engineering, 1983, the MSc. degree in mathematics, 1986, and the Ph.D. degree in electrical engineering, 1986.

He was with the University of Virginia from 1987 to 1998, and since 1999, he has been a Professor with the University of Minnesota, Minneapolis, MN, USA, where he holds an Endowed Chair in Wireless Telecommunications, a University of Minnesota McKnight Presidential Chair in ECE, and serves as Director of the Digital Technology Center. His interests include communications, networking and statistical signal processing—subjects on which he has authored or coauthored more than 400 journal papers, 700 conference papers, 25 book chapters, 2 edited books and 2 research monographs (h-index 125). His current research interest focuses on learning from big data, wireless cognitive radios, and network science with applications to social, brain, and power networks with renewables. He is a Fellow of EURASIP, and has served the IEEE in a number of posts, including that of a Distinguished Lecturer for the IEEE-SP Society. He is the (co-) inventor of 30 patents issued, and the (co-) recipient of 8 best paper awards from the IEEE Signal Processing (SP) and Communications Societies, including the G. Marconi Prize Paper Award in Wireless Communications. He was also the recipient of the Technical Achievement Awards from the SP Society (2000), from EURASIP (2005), a Young Faculty Teaching Award, the G. W. Taylor Award for Distinguished Research from the University of Minnesota, and the IEEE Fourier Technical Field Award (2015).