Predicting Graph Signals using Kernel Regression where the Input Signal is Agnostic to a Graph

Arun Venkitaraman, Saikat Chatterjee, Peter Händel
Department of Information Science and Engineering
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden
arunv@kth.se, sach@kth.se, ph@kth.se

***Abstract*—We propose a kernel regression method to predict a target signal lying over a graph when an input observation is given. The input and the output could be two different physical quantities. In particular, the input may not be a graph signal at all or it could be agnostic to an underlying graph. We use a training dataset to learn the proposed regression model by formulating it as a convex optimization problem, where we use a graph-Laplacian based regularization to enforce that the predicted target is a graph signal. Once the model is learnt, it can be directly used on a large number of test data points one-by-one independently to predict the corresponding targets. Our approach employs kernels between the various input observations, and as a result the kernels are not restricted to be functions of the graph adjacency/Laplacian matrix. We show that the proposed kernel regression exhibits a smoothing effect, while simultaneously achieving noise-reduction and graph-smoothness. We then extend our method to the case when the underlying graph may not be known apriori, by simultaneously learning an underlying graph and the regression coefficients. Using extensive experiments, we show that our method provides a good prediction performance in adverse conditions, particularly when the training data is limited in size and is noisy. In graph signal reconstruction experiments, our method is shown to provide a good performance even for a highly under-determined subsampling.**

***Index Terms*—Linear model, regression, kernels, machine learning, graph signal processing, graph-Laplacian.**

EDICS−ADEL-SIPOG

I. Introduction

Graph signal processing (GSP) has emerged recently as a framework which employs graph-structural information in the analysis and processing of vector-valued signals [1], [2]. The framework has been shown to exhibit great potential in a wide range of real-world applications that deal with data over networks or graphs. By actively making use of the graph or the network structure, GSP deals with the extension of several traditional signal processing and machine learning concepts to a graph signal setting. In this article, our contribution to GSP is the development of a supervised kernel regression method for predicting graph signal outputs from general input observations. In the next two subsections, we provide a review of the existing literature, followed by our contributions placed in the context of the existing methods.

A. Literature review

The extension of traditional signal processing methods to graph signal processing includes many conventional spectral analysis concepts such as the windowed Fourier transforms, filterbanks, multiresolution analysis, and wavelets [1]–[18]. Spectral clustering approaches based on graph signal filtering have also been proposed [19], [20]. The problems of subsampling and interpolation of signals lying over graphs have been considered extensively in diverse settings [21]–[31]. Techniques for compression and representation of signals such as the principal component analysis (PCA) [32], [33] and dictionary learning approaches [34]–[37] have also been proposed for graph signals. Many researchers have considered the statistical analysis of graph signals particularly in the context of stationarity [38]–[43]. The reconstruction and estimation of graph signals have also been steadily gaining interest in the community. Berger et al. [44] and Chen et al. [45] considered the recovery of graph signals based on a total-variation minimization formulated as a convex optimization problem. Wang et al. [46] considered a distributed reconstruction of time-varying bandlimited graph signals. Di Lorenzo et al. [47] proposed a least mean squares approach for the adaptive estimation and tracking of bandlimited graph signals. Several approaches have also been proposed for learning an underlying graph structure from the given graph signal data [48]–[58]. GSP is a rich and continually expanding area of research and we refer the reader to [59] for a more comprehensive review of the developments.

We now proceed to briefly survey the relevant literature in kernel regression and kernel methods. Kernel regression constitutes one of the fundamental building blocks of supervised and semi-supervised learning strategies, be it in simple regression tasks or in the more advanced settings. Kernel regression lies at the core of support vector machines [60] and Gaussian processes [61], and finds applications in deep neural networks [62], [63] and extreme learning machines [64], [65]. Kernel regression in the setting of graphs or manifolds has been investigated in the labeling and coloring of graphs and in the context of graph clustering [66]–[73]. These works generally deal with signals which are binary-valued. Kernel regression has been employed in image deblurring by using a graph-based constraint on the pixel intensities of the deblurred image [74]. Kernel regression was recently used in object saliency detection and spatial attention modeling in images, wherein the kernel matrix was simultaneously used to define a Laplacian matrix, in order to recover the smooth images [75]. These prior graph-based approaches incorporate a graph-

Laplacian based regularization by defining a graph between the various observations/datapoints and are concerned with an output/target that is scalar valued, such as the node label or the pixel intensity. Kernels have also been extensively employed in the smoothing and regression of brain signals, where the functional connectivity or the topology of the brain surface is described using meshes [76]–[81].

Kernel-based reconstruction strategies specific to graph signals were proposed recently by Romero et al. in the framework of reproducing kernel Hilbert spaces [82], [83]. Using the notion of joint space-time graphs, Romero et al. have also proposed a kernel based reconstruction of graph signals and an extension of the Kalman filter for kernel-based learning [84], [85]. Along the same lines of thought, Ioannidis et al. proposed a more general approach for inferring functions over graphs in both static and dynamic settings [86]. Kernel regression combined with diffusion wavelets have been employed in the modeling of mandible growth in CT images [87]. Shen et al. used kernels in structural equation models for the identification of network topologies from graph signals [88]. The prior works of [82]–[86] use kernels across the nodes of a graph, while considering the input comprising the signal values over a subset of the nodes of the graph. In these prior works, the setting is that all the observed inputs and the corresponding outputs to be predicted lie jointly over a composite or an augmented graph. As a result, the setting results in large-sized graphs which may not provide a scalable solution when the number of inputs and outputs becomes moderately large. Further, the setting naturally requires that the input and the output variables are of the same physical quantities. Therefore, these prior works suffer from limitations when the input is a fundamentally different physical quantity from the predicted output, or when the input is not a graph signal. For example, consider a scenario where we observe the air pressure of several cities in a country as the input, and the task is to predict the temperature of those cities as the output.

B. Our contributions vis-a-vis existing works

We propose a kernel regression method for graph signals that can handle scenarios where the input and the output may be entirely different physical quantities, or when the input is not a graph signal or is agnostic to a graph. For example, our method is applicable to the case when the hourly air pressure measurements over the cities in a country is taken as the input, and the predicted output is the temperature of those cities. (Such a real signal example is indeed demonstrated in the numerical experiments section.) This is possible because we treat the input variables without any graph constraints, or as being agnostic to a graph. The graph-awareness is employed only for the output: that the predicted output is a vector lying over a graph. Since we do not use kernels between the nodes of the underlying graph but only across the different observations of the graph signal, our kernel is not necessarily defined or dictated by the underlying graph. This is in contrast with the prior works where the kernel matrix is an explicit function of the graph adjacency matrix and the kernel is across the different nodes of the same graph. In several applications where the input is also a graph signal, experiments with real-world datasets show that our method performs better than those which exploit the graph structure in the input.

The success of our method can be attributed to a standard machine learning concept where a set of training data is used to learn a regression model and then the trained model is used to make predictions on the test data. The prediction at each test datapoint is made independent of the other test datapoints, and is based only on the kernels between the training datapoints and the relevant test datapoint (and not all the test datapoints together). In contrast, in the prior works involving kernels and graph signals [82], [85], the estimation of the graph signal value at even one of the nodes involves the computation of the entire kernel matrix for all the nodes over the graph, and not just over the input or the training nodes. In other words, they employ the entire kernel matrix across all the available training and test datapoints and do not treat a relevant test datapoint independently with respect to the other test datapoints. This is true even for the cases where one predicts a subset of the unobserved nodes or test datapoints.

Further, the independent treatment of the test datapoints allows us to use our regression method on any number of test datapoints. The method does not assume that the number of test datapoints is known from the beginning. Therefore, our method scales well with a large amount of test datapoints and naturally extends to a dynamic tracking setup, such as the Gaussian process model [89]. On the other hand, the works of [82]–[85] assume that the number of test datapoints is specified from the beginning.

C. Signal processing over graphs

We next briefly review some of the basic concepts from graph signal processing. Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{A})$ denote a graph with M nodes indexed by the vertex set $\mathcal{V} = \{1, \cdots, M\}$. Let $\mathcal{E}$ and $\mathbf{A}$ denote the edge set containing pairs of nodes, and the weighted adjacency matrix, respectively. The (i,j)th entry of the adjacency matrix $\mathbf{A}(i,j)$ denotes the strength of the edge between the ith and jth nodes. There exists an edge between the ith and the jth nodes if $\mathbf{A}(i,j) > 0$ and the edge pair $(i,j) \in \mathcal{E} \iff \mathbf{A}(i,j) \neq 0$. In our analysis, we consider only undirected graphs with symmetric edge-weights or $\mathbf{A} = \mathbf{A}^\top$. The graph-Laplacian matrix $\mathbf{L}$ of the graph G is then defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A},$$

where $\mathbf{D}$ is the diagonal degree matrix with the ith diagonal element given by the sum of the elements in the ith row of $\mathbf{A}$. A vector $\mathbf{x} = [x(1)\, x(2)\, \cdots x(M)]^\top \in \mathbb{R}^M$ is said to be a graph signal if $x(i)$ denotes the value of the signal at the ith node of $\mathcal{G}$. The quadratic form of $\mathbf{x}$ with $\mathbf{L}$ is given by

$$\mathbf{x}^\top \mathbf{L} \mathbf{x} = \sum_{(i,j)\in\mathcal{E}} \mathbf{A}(i,j)(x(i) - x(j))^2.$$

We observe that $\mathbf{x}^\top \mathbf{L} \mathbf{x}$ is minimized when the signal $\mathbf{x}$ takes the same value across all the connected nodes, which agrees with the intuitive concept of a smooth signal. In general, a graph signal is said to be smooth or a low-frequency signal if it has similar values across all the connected nodes in a graph,

and is said to be a high-frequency signal if it has dissimilar values across the connected nodes, $\mathbf{x}^\top\mathbf{L}\mathbf{x}$ being the measure of similarity. This motivates the use of $\mathbf{x}^\top\mathbf{L}\mathbf{x}$ as a constraint in the applications where either the signal $\mathbf{x}$ or the graph-Laplacian $\mathbf{L}$ is to be estimated [48], [50]. The eigenvectors of $\mathbf{L}$ are referred to as the graph Fourier transform basis for G, and the corresponding eigenvalues are referred to as the graph frequencies. The smaller eigenvalues (the smallest being zero by construction) are referred to as the low frequencies since the corresponding eigenvectors result in small values of the quadratic form of $\mathbf{L}$, and vary smoothly over the nodes. Similarly, the larger eigenvalues are referred to as the high frequencies. Then, a smooth graph signal is one which has the energy of the GFT coefficients predominantly in the low graph frequencies.

II. Kernel Regression over Graphs

A. Linear basis model for regression over graphs

Let $\{\mathbf{x}_n\}_{n=1}^N$ denote a set of N input observations. Each input $\mathbf{x}_n$ is paired with a target $\mathbf{t}_n \in \mathbb{R}^M$. Our goal is to model the target $\mathbf{t}_n$ with $\mathbf{y}_n$ given by:

$$\mathbf{y}_n = \mathbf{W}^\top\boldsymbol{\phi}(\mathbf{x}_n), \tag{1}$$

where $\boldsymbol{\phi}(\mathbf{x}_n) \in \mathbb{R}^K$ is a known function of $\mathbf{x}_n$ and $\mathbf{W} \in \mathbb{R}^{K\times M}$ denotes the regression coefficient matrix. Equation (1) is referred to as the linear basis model for regression, often shortened to linear regression in the machine learning parlance (cf. Chapters 3 and 6 in [90]). For brevity, we hereafter follow this shortened nomenclature and refer to the outcome of using (1) as linear regression. Our central assumption is that the target $\mathbf{y}_n$ is a smooth signal over an underlying graph G with M nodes. We learn the optimal parameter matrix $\mathbf{W}$ by minimizing the following cost function with respect to $\mathbf{W}$:

$$C(\mathbf{W}) = \sum_{n=1}^{N}\|\mathbf{t}_n-\mathbf{y}_n\|_2^2+\alpha\mathrm{tr}(\mathbf{W}^\top\mathbf{W})+\beta\sum_{n=1}^{N}\mathbf{y}_n^\top\mathbf{L}\mathbf{y}_n, \tag{3}$$

where the regularization coefficients $\alpha, \beta \geq 0$, $\mathrm{tr}(\cdot)$ denotes the trace operator, and $\|\mathbf{x}\|_2$ denotes the ℓ_2 norm of $\mathbf{x}$, and we emphasize that $\mathbf{y}_n$ is a function of $\mathbf{W}$. The cost in (3) is convex in $\mathbf{W}$, since $\mathbf{L}$ is positive semidefinite on the virtue of it being the graph-Laplacian matrix. The choice of α, β depends on the problem, and in our analysis we compute these parameters through crossvalidation. The penalty $\mathrm{tr}(\mathbf{W}^\top\mathbf{W}) = \|\mathbf{W}\|_F^2$ ensures that $\mathbf{W}$ remains bounded. The penalty or regularization $\mathbf{y}_n^\top\mathbf{L}\mathbf{y}_n$ enforces $\mathbf{y}_n$ to be smooth over G. We note that the smoothness over a graph could be quantified in a number of alternative ways, specially if domain-specific knowledge is available. However, since the graph-Laplacian based regularization has been the most popular metric in the graph signal processing literature with respect to undirected graphs, we employ the same in our work. Another aspect is that $\mathbf{y}_n^\top\mathbf{L}\mathbf{y}_n$ being quadratic in $\mathbf{y}_n$ helps us arrive at a unique and closed-form solution for the regression model, as we show next. We define matrices $\mathbf{T}$, $\mathbf{Y}$ and $\boldsymbol{\Phi}$ as follows:

$$\begin{aligned}\mathbf{T} &= [\mathbf{t}_1\,\mathbf{t}_2\,\cdots\mathbf{t}_N]^\top \in \mathbb{R}^{N\times M},\\ \mathbf{Y} &= [\mathbf{y}_1\,\mathbf{y}_2\,\cdots\mathbf{y}_N]^\top \in \mathbb{R}^{N\times M},\\ \boldsymbol{\Phi} &= [\boldsymbol{\phi}(\mathbf{x}_1)\;\boldsymbol{\phi}(\mathbf{x}_2)\;\cdots\boldsymbol{\phi}(\mathbf{x}_N)]^\top \in \mathbb{R}^{N\times K}.\end{aligned} \tag{4}$$

Using (1) and (4), the cost function (3) is expressible as (2) where we use the properties of the matrix trace operation. Since the cost function is quadratic and convex in $\mathbf{W}$, we get the optimal and unique solution by setting the gradient of C with respect to $\mathbf{W}$ equal to zero. Using the following matrix derivative relations

$$\begin{aligned}\frac{\partial}{\partial\mathbf{W}}\mathrm{tr}\left(\mathbf{M}_1\mathbf{W}\right) &= \mathbf{M}_1^\top,\\ \frac{\partial}{\partial\mathbf{W}}\mathrm{tr}\left(\mathbf{W}^\top\mathbf{M}_1\mathbf{W}\mathbf{M}_2\right) &= \mathbf{M}_1^\top\mathbf{W}\mathbf{M}_2^\top+\mathbf{M}_1\mathbf{W}\mathbf{M}_2,\end{aligned}$$

where $\mathbf{M}_1$ and $\mathbf{M}_2$ are matrices, and setting $\frac{\partial C}{\partial\mathbf{W}} = 0$ we get that

$$\begin{aligned}&-\boldsymbol{\Phi}^\top\mathbf{T}+\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\mathbf{W}+\alpha\mathbf{W}+\beta\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\mathbf{W}\mathbf{L} = \mathbf{0},\\ \text{or,}\quad &\left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi}+\alpha\mathbf{I}_K\right)\mathbf{W}+\beta\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\mathbf{W}\mathbf{L} = \boldsymbol{\Phi}^\top\mathbf{T}.\end{aligned} \tag{5}$$

On vectorizing both sides of (5), we get that

$$\mathrm{vec}(\boldsymbol{\Phi}^\top\mathbf{T}) = \left[(\mathbf{I}_M\otimes(\boldsymbol{\Phi}^\top\boldsymbol{\Phi}+\alpha\mathbf{I}_K))+(\beta\mathbf{L}\otimes\boldsymbol{\Phi}^\top\boldsymbol{\Phi})\right]\mathrm{vec}(\mathbf{W}),$$

where $\mathrm{vec}(\cdot)$ denotes the standard vectorization operator and $\otimes$ denotes the Kronecker product operation [91]. Then, the optimal $\mathbf{W}$, denoted by $\overset{\star}{\mathbf{W}}$, follows the relation:

$$\mathrm{vec}(\overset{\star}{\mathbf{W}}) = \left[(\mathbf{I}_M\otimes(\boldsymbol{\Phi}^\top\boldsymbol{\Phi}+\alpha\mathbf{I}_K))+(\beta\mathbf{L}\otimes\boldsymbol{\Phi}^\top\boldsymbol{\Phi})\right]^{-1}\mathrm{vec}(\boldsymbol{\Phi}^\top\mathbf{T}).$$

The predicted target for a new input $\mathbf{x}$ is then given by

$$\overset{\star}{\mathbf{t}} = \overset{\star}{\mathbf{W}}{}^\top\boldsymbol{\phi}(\mathbf{x}). \tag{6}$$

From (6), it appears that the proposed target prediction approach requires the explicit knowledge of the function $\boldsymbol{\phi}(\cdot)$. We next show that using the 'kernel trick' or 'kernel substitution' this explicit requirement of $\boldsymbol{\phi}(\cdot)$ is circumvented and that the target prediction may be done using only the knowledge of the inner products $\boldsymbol{\phi}(\mathbf{x}_m)^\top\boldsymbol{\phi}(\mathbf{x}_n)$, $\forall m, n$. Towards this end, we next discuss a dual representation of the cost in (3). We hereafter refer to (6) as the output of the linear regression over graphs (LRG).

B. Dual representation of cost using kernel trick

We now use the substitution $\mathbf{W} = \boldsymbol{\Phi}^\top\boldsymbol{\Psi}$ and express the cost function in terms of the parameter $\boldsymbol{\Psi} \in \mathbb{R}^{N\times M}$. This substitution is motivated by observing that on rearranging the terms in (5), we get that

$$\mathbf{W} = \boldsymbol{\Phi}^\top\boldsymbol{\Psi},$$

where $\boldsymbol{\Psi} = \left[\frac{1}{\alpha}\left(\mathbf{T}-\beta\boldsymbol{\Phi}\mathbf{W}\mathbf{L}-\boldsymbol{\Phi}\right)\right]$. On substituting $\mathbf{W} = \boldsymbol{\Phi}^\top\boldsymbol{\Psi}$ in (2), where $\boldsymbol{\Psi}$ becomes the dual parameter matrix that we wish to learn, and omitting terms that do not depend on $\mathbf{W}$, we get that

$$\begin{aligned}C(\boldsymbol{\Psi}) =& -2\mathrm{tr}\left(\mathbf{T}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Psi}\right)+\mathrm{tr}\left(\boldsymbol{\Psi}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Psi}\right)\\ &+\alpha\,\mathrm{tr}(\boldsymbol{\Psi}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Psi})+\beta\,\mathrm{tr}\left(\boldsymbol{\Psi}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Psi}\mathbf{L}\right)\\ =& -2\mathrm{tr}\left(\mathbf{T}^\top\mathbf{K}\boldsymbol{\Psi}\right)+\mathrm{tr}\left(\boldsymbol{\Psi}^\top\mathbf{K}\mathbf{K}\boldsymbol{\Psi}\right)\\ &+\alpha\,\mathrm{tr}(\boldsymbol{\Psi}^\top\mathbf{K}\boldsymbol{\Psi})+\beta\,\mathrm{tr}\left(\boldsymbol{\Psi}^\top\mathbf{K}\mathbf{K}\boldsymbol{\Psi}\mathbf{L}\right),\end{aligned} \tag{7}$$

where $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \in \mathbb{R}^{N\times N}$ denotes the kernel matrix for the training samples such that its (m, n)th entry is given by

$$k_{m,n} = \boldsymbol{\phi}(\mathbf{x}_m)^\top\boldsymbol{\phi}(\mathbf{x}_n).$$

$$
\begin{aligned}
C(\mathbf{W}) &= \sum_n \|\mathbf{t}_n - \mathbf{W}^\top \boldsymbol{\phi}(\mathbf{x}_n)\|_2^2 + \alpha \,\mathrm{tr}(\mathbf{W}^\top \mathbf{W}) + \beta \sum_n \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{W}\mathbf{L}\mathbf{W}^\top \boldsymbol{\phi}(\mathbf{x}_n) \\
&= \sum_n \|\mathbf{t}_n\|_2^2 - 2\,\mathrm{tr}\left(\sum_n \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{W}\mathbf{t}_n\right) + \mathrm{tr}\left(\sum_n \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{W}\mathbf{W}^\top \boldsymbol{\phi}(\mathbf{x}_n)\right) + \alpha\,\mathrm{tr}(\mathbf{W}^\top\mathbf{W}) \\
&\quad +\beta\,\mathrm{tr}\left(\sum_n \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{W}\mathbf{L}\mathbf{W}^\top \boldsymbol{\phi}(\mathbf{x}_n)\right) \\
&= \sum_n \|\mathbf{t}_n\|_2^2 - 2\,\mathrm{tr}\left(\sum_n \mathbf{t}_n \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{W}\right) + \mathrm{tr}\left(\mathbf{W}\mathbf{W}^\top \sum_n \boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\phi}(\mathbf{x}_n)^\top\right) + \alpha\,\mathrm{tr}(\mathbf{W}^\top\mathbf{W}) \\
&\quad +\beta\,\mathrm{tr}\left(\mathbf{W}\mathbf{L}\mathbf{W}^\top \sum_n \boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\phi}(\mathbf{x}_n)^\top\right) \\
&= \sum_n \|\mathbf{t}_n\|_2^2 - 2\,\mathrm{tr}\left(\mathbf{T}^\top \boldsymbol{\Phi}\mathbf{W}\right) + \mathrm{tr}(\mathbf{W}^\top \boldsymbol{\Phi}^\top \boldsymbol{\Phi}\mathbf{W}) + \alpha\,\mathrm{tr}(\mathbf{W}^\top\mathbf{W}) + \beta\,\mathrm{tr}\left(\mathbf{W}^\top \boldsymbol{\Phi}^\top \boldsymbol{\Phi}\mathbf{W}\mathbf{L}\right).
\end{aligned} \tag{2}
$$

Equation (7) is referred to as a dual representation of (2) in the kernel regression literature (cf. Chapter 6 of [90]). Taking the derivative of $C(\boldsymbol{\Psi})$ with respect to $\boldsymbol{\Psi}$ and setting it to zero, we get that

$$
\begin{aligned}
&(\mathbf{I}_M \otimes (\mathbf{K} + \alpha\mathbf{I}_N))\mathrm{vec}(\boldsymbol{\Psi}) + (\beta\mathbf{L}\otimes\mathbf{K})\mathrm{vec}(\boldsymbol{\Psi}) = \mathrm{vec}(\mathbf{T}), \text{ or} \\
&\quad [(\mathbf{I}_M \otimes (\mathbf{K} + \alpha\mathbf{I}_N)) + (\beta\mathbf{L}\otimes\mathbf{K})]\,\mathrm{vec}(\boldsymbol{\Psi}) = \mathrm{vec}(\mathbf{T}), \quad \text{or} \\
&\quad \mathrm{vec}(\boldsymbol{\Psi}) = [(\mathbf{I}_M \otimes (\mathbf{K} + \alpha\mathbf{I}_N)) + (\beta\mathbf{L}\otimes\mathbf{K})]^{-1}\,\mathrm{vec}(\mathbf{T}).
\end{aligned}
$$

We define the matrices

$$
\begin{aligned}
\mathbf{B} &= (\mathbf{I}_M \otimes (\mathbf{K} + \alpha\mathbf{I}_N)), \\
\mathbf{C} &= (\beta\mathbf{L}\otimes\mathbf{K}).
\end{aligned} \tag{8}
$$

Then, we have that

$$
\mathrm{vec}(\boldsymbol{\Psi}) = (\mathbf{B}+\mathbf{C})^{-1}\,\mathrm{vec}(\mathbf{T}). \tag{9}
$$

Once $\boldsymbol{\Psi}$ is computed, the predicted output of the kernel regression for a new test input $\mathbf{x}$ is given by

$$
\begin{aligned}
\mathbf{y} &= \mathbf{W}^\top \boldsymbol{\phi}(\mathbf{x}) = \boldsymbol{\Psi}^\top \boldsymbol{\Phi}\boldsymbol{\phi}(\mathbf{x}) \\
&= \boldsymbol{\Psi}^\top \mathbf{k}(\mathbf{x}) \\
&= \left(\mathrm{mat}\left((\mathbf{B}+\mathbf{C})^{-1}\,\mathrm{vec}(\mathbf{T})\right)\right)^\top \mathbf{k}(\mathbf{x}),
\end{aligned} \tag{10}
$$

where $\mathbf{k}(\mathbf{x}) = [k_1(\mathbf{x}), k_2(\mathbf{x}), \cdots, k_N(\mathbf{x})]^\top$ and $k_n(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x}_n)^\top\boldsymbol{\phi}(\mathbf{x})$. Here mat$(\cdot)$ denotes the reshaping operation of an argument vector into an appropriate matrix of size $N \times M$ by concatenating the subsequent N length sections as the columns. We refer to (10) as the output of the method named *kernel regression over graphs* (KRG). The kernel regression is arrived at by noting that the entire formulation remains valid if the inner products $\boldsymbol{\phi}(\mathbf{x}_m)^\top\boldsymbol{\phi}(\mathbf{x}_n)$ are replaced with a general kernel function associating pairs of the inputs $\mathbf{x}_m$ and $\mathbf{x}_n$.

In general, a variety of valid kernel functions may be employed. Any kernel function $k(\mathbf{x}, \mathbf{x}')$ is a valid kernel as long as it can be expressed in the form $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top\boldsymbol{\phi}(\mathbf{x}')$, and the associated kernel matrix $\mathbf{K}$ is positive semi-definite for all observation sizes [90]. The Gaussian kernel is a popularly employed kernel and we use the same in our experiments later. We note that KRG is a generalization over the conventional kernel regression (KR), where the latter does not use any knowledge of the underlying graph structure. On setting $\beta = 0$, the KRG output (10) reduces to the conventional KR output as follows

$$
\begin{aligned}
\mathbf{y} &= \left(\mathrm{mat}\left((\mathbf{B}+\mathbf{C})^{-1}\,\mathrm{vec}(\mathbf{T})\right)\right)^\top \mathbf{k}(\mathbf{x}) \\
&= \left(\mathrm{mat}\left((\mathbf{B})^{-1}\,\mathrm{vec}(\mathbf{T})\right)\right)^\top \mathbf{k}(\mathbf{x}) \\
&= \left(\mathrm{mat}\left((\mathbf{I}_M \otimes (\mathbf{K}+\alpha\mathbf{I}_N))^{-1}\,\mathrm{vec}(\mathbf{T})\right)\right)^\top \mathbf{k}(\mathbf{x}) \\
&= \left(\mathrm{mat}\left((\mathbf{I}_M^{-1} \otimes (\mathbf{K}+\alpha\mathbf{I}_N)^{-1})\mathrm{vec}(\mathbf{T})\right)\right)^\top \mathbf{k}(\mathbf{x}) \\
&= \left((\mathbf{K}+\alpha\mathbf{I}_N)^{-1}\mathbf{T}\right)^\top \mathbf{k}(\mathbf{x}) \\
&= \mathbf{T}^\top(\mathbf{K}+\alpha\mathbf{I}_N)^{-1}\mathbf{k}(\mathbf{x}),
\end{aligned} \tag{11}
$$

where we have used the Kronecker product equality: vec$((\mathbf{K}+\alpha\mathbf{I}_N)^{-1}\mathbf{T}) = (\mathbf{I}_M^{-1}\otimes(\mathbf{K}+\alpha\mathbf{I}_N)^{-1})vec(\mathbf{T})$. Further, we note that KRG reduces to LRG on setting $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$.

C. Interpretation of KRG – a smoothing effect

We next discuss how the output of KRG is smooth across the training samples $\{1, \cdots, N\}$ and over the M nodes of the graph. Before proceeding with KRG, we review a similar property exhibited by the KR output [92]. Using (11) and concatenating the KR outputs for the N training samples, we get that

$$
\begin{aligned}
[\mathbf{y}_1\,\mathbf{y}_2\,\cdots\,\mathbf{y}_N] &= \mathbf{T}^\top(\mathbf{K}+\alpha\mathbf{I}_N)^{-1}[\mathbf{k}(\mathbf{x}_1)\,\mathbf{k}(\mathbf{x}_2)\,\ldots\mathbf{k}(\mathbf{x}_N)] \\
\text{or,}\quad \mathbf{Y} &= \mathbf{K}(\mathbf{K}+\alpha\mathbf{I}_N)^{-1}\mathbf{T},
\end{aligned}
$$

where we use $\mathbf{K} = [\mathbf{k}(\mathbf{x}_1)\,\mathbf{k}(\mathbf{x}_2)\,\ldots\mathbf{k}(\mathbf{x}_N)]$. Assuming $\mathbf{K}$ is diagonalizable, let $\mathbf{K} = \mathbf{U}\mathbf{J}_K\mathbf{U}^\top$, where $\mathbf{J}_K$ and $\mathbf{U}$ denote the eigenvalue and the eigenvector matrices of $\mathbf{K}$, respectively. Let $\tilde{\mathbf{y}}(m) = [y_1(m), y_2(m), \cdots, y_N(m)]^\top$ ($m \in \{1, \cdots, M\}$) denote the mth column of $\mathbf{Y}$. We note that $\tilde{\mathbf{y}}(m)$ consists of values of the mth component of $\mathbf{y}$ collected over all the time instances. Then, we have that

$$
\tilde{\mathbf{y}}(m) = \sum_{i=1}^{N} \frac{\theta_i}{\theta_i + \alpha}[\mathbf{u}_i^\top \tilde{\mathbf{t}}(m)]\mathbf{u}_i,
$$

where θ_i and $\mathbf{u}_i$ denote the ith eigenvalue and eigenvector of $\mathbf{K}$, respectively, and $\tilde{\mathbf{t}}(m)$ denotes the vector containing the mth component of the target vector for all the training

samples (mth column of $\mathbf{T}$). Thus, we observe that the KR output performs a shrinkage of $\tilde{\mathbf{t}}(m)$ along the various eigenvector directions $\mathbf{u}_i$ for each m. The contribution from the eigenvectors corresponding to the smaller eigenvalues $\theta_i < \alpha$ are effectively eliminated, and only those corresponding to the larger eigenvalues $\theta_i > \alpha$ are retained. Since the eigenvectors corresponding to the larger eigenvalues of $\mathbf{K}$ represent smooth variations across the observations $\{1, \cdots, N\}$, we observe that KR performs a smoothing of $\mathbf{t}_n$. We next show that such is also the case with KRG: KRG acts as a smoothing filter across both the observations and the graph nodes. Using (10) and concatenating the KRG outputs, we have that

$$\begin{aligned}[\mathbf{y}_1\, \mathbf{y}_2\, \ldots\, \mathbf{y}_N] &= \mathbf{\Psi}^\top [\mathbf{k}(\mathbf{x}_1)\, \mathbf{k}(\mathbf{x}_2)\, \ldots \mathbf{k}(\mathbf{x}_N)] \quad (12)\\ \text{or,}\quad \mathbf{Y} &= \mathbf{K}\mathbf{\Psi}.\end{aligned}$$

On vectorizing both sides of (12), we get that

$$\begin{aligned}\text{vec}(\mathbf{Y}) &= (\mathbf{I}_M \otimes \mathbf{K})\text{vec}(\mathbf{\Psi})\\ &\overset{(a)}{=} (\mathbf{I}_M \otimes \mathbf{K})\,(\mathbf{B}+\mathbf{C})^{-1}\,\text{vec}(\mathbf{T}),\end{aligned} \quad (13)$$

where we have used (9) in (a). Let $\mathbf{L}$ be diagonalizable with the eigendecomposition:

$$\mathbf{L} = \mathbf{V}\mathbf{J}_L\mathbf{V}^\top,$$

where $\mathbf{J}_L$ and $\mathbf{V}$ denote the eigenvalue and the eigenvector matrices of $\mathbf{L}$, respectively. We also assume that $\mathbf{K}$ is diagonalizable as earlier. Let λ_i and $\mathbf{v}_i$ denote the ith eigenvalue and the ith eigenvector of $\mathbf{L}$, respectively. Then, we have that

$$\begin{aligned}\mathbf{V} &= [\mathbf{v}_1\, \mathbf{v}_2 \cdots \mathbf{v}_N] \text{ and } \mathbf{J}_L = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_N),\\ \mathbf{U} &= [\mathbf{u}_1\, \mathbf{u}_2 \cdots \mathbf{u}_M] \text{ and } \mathbf{J}_K = \text{diag}(\theta_1, \theta_2, \cdots, \theta_M).\end{aligned}$$

Now, using (8), we have

$$\begin{aligned}\mathbf{B}+\mathbf{C} &= (\mathbf{I}_M \otimes (\mathbf{K} + \alpha\mathbf{I}_N)) + (\beta\mathbf{L} \otimes \mathbf{K})\\ &= [(\mathbf{V}\mathbf{I}_M\mathbf{V}^\top) \otimes (\mathbf{U}(\mathbf{J}_K + \alpha\mathbf{I}_N)\mathbf{U}^\top)]\\ &\quad +[\beta(\mathbf{V}\mathbf{J}_L\mathbf{V}^\top) \otimes (\mathbf{U}\mathbf{J}_K\mathbf{U}^\top)]\\ &\overset{(a)}{=} [(\mathbf{V}\otimes\mathbf{U})(\mathbf{I}_M \otimes (\mathbf{J}_K + \alpha\mathbf{I}_N))(\mathbf{V}^\top \otimes \mathbf{U}^\top)]\\ &\quad +[\beta(\mathbf{V}\otimes\mathbf{U})(\mathbf{J}_L \otimes \mathbf{J}_K)(\mathbf{V}^\top \otimes \mathbf{U}^\top)]\\ &= \mathbf{Z}\mathbf{J}\mathbf{Z}^\top, \end{aligned} \quad (14)$$

where $\mathbf{Z} = \mathbf{V} \otimes \mathbf{U}$ is the eigenvector matrix and $\mathbf{J}$ is the diagonal eigenvalue matrix given by

$$\mathbf{J} = (\mathbf{I}_M \otimes (\mathbf{J}_K + \alpha\mathbf{I}_N)) + \beta(\mathbf{J}_L \otimes \mathbf{J}_K). \quad (15)$$

In (14)(a), we have used the distributivity property of the Kronecker product: $(\mathbf{M}_1 \otimes \mathbf{M}_2)(\mathbf{M}_3 \otimes \mathbf{M}_4) = \mathbf{M}_1\mathbf{M}_3 \otimes \mathbf{M}_2\mathbf{M}_4$ where $\{\mathbf{M}_i\}_{i=1}^4$ are four matrices. We note that $\mathbf{J}$ is a diagonal matrix of size MN. Let $\mathbf{J} = \text{diag}(\eta_1, \eta_2, \ldots, \eta_{MN})$. Then, η_i is a function of $\{\lambda_j\}, \{\theta_j\}, \alpha, \beta$. On dropping the subscripts for simplicity, we observe that any eigenvalue η_i has the form

$$\eta = (\theta + \alpha) + \beta(\lambda\theta).$$

where θ and λ are the appropriate eigenvalues of $\mathbf{K}$ and $\mathbf{L}$, respectively. Similarly, we have that

$$\begin{aligned}(\mathbf{I}_M \otimes \mathbf{K}) &= (\mathbf{V}\mathbf{I}_M\mathbf{V}^\top) \otimes (\mathbf{U}\mathbf{J}_K\mathbf{U}^\top)\\ &= (\mathbf{V}\otimes\mathbf{U})(\mathbf{I}_M \otimes \mathbf{J}_K)(\mathbf{V}^\top \otimes \mathbf{U}^\top)\\ &= \mathbf{Z}(\mathbf{I}_M \otimes \mathbf{J}_K)\mathbf{Z}^\top,\end{aligned} \quad (16)$$

and note that $(\mathbf{I}_M \otimes \mathbf{J}_K)$ is also a diagonal matrix of size MN. Then, on substituting (15) and (16) in (13), we get that

$$\begin{aligned}\text{vec}(\mathbf{Y}) &= (\mathbf{I}_M \otimes \mathbf{K})\,(\mathbf{B}+\mathbf{C})^{-1}\,\text{vec}(\mathbf{T})\\ &= (\mathbf{Z}(\mathbf{I}_M \otimes \mathbf{J}_K)\mathbf{Z}^\top)(\mathbf{Z}\mathbf{J}^{-1}\mathbf{Z}^\top)\text{vec}(\mathbf{T})\\ &= (\mathbf{Z}(\mathbf{I}_M \otimes \mathbf{J}_K)\mathbf{J}^{-1}\mathbf{Z}^\top)\text{vec}(\mathbf{T}).\end{aligned} \quad (17)$$

We note again that $(\mathbf{I}_M \otimes \mathbf{J}_K)\mathbf{J}^{-1}$ is a diagonal matrix with size MN. Let $(\mathbf{I}_M \otimes \mathbf{J}_K)\mathbf{J}^{-1} = \text{diag}(\zeta_1, \zeta_2, \ldots, \zeta_{MN})$. Then, on dropping the subscripts, any η_i is of the form

$$\zeta = \frac{\theta}{\eta} = \frac{\theta}{(\theta+\alpha) + \beta(\lambda\theta)}.$$

From (17), we have that

$$\text{vec}\,(\mathbf{Y}) = \sum_{i=1}^{MN} \zeta_i \mathbf{z}_i\mathbf{z}_i^\top\,\text{vec}\,(\mathbf{T}),$$

where $\mathbf{z}_i$ are the column vectors of $\mathbf{Z}$. In the case when $\zeta_i \ll 1$, the component in vec($\mathbf{T}$) along $\mathbf{z}_i$ is effectively eliminated. For most covariance or kernel functions $k(\cdot,\cdot)$ used in practice, the eigenvectors corresponding to the largest eigenvalues of $\mathbf{K}$ are the low-frequency components across time or observations. Similarly, the eigenvectors corresponding to the smaller eigenvalues of $\mathbf{L}$ are smooth over the graph [1]. We observe that the condition $\zeta \ll 1$ is achieved when θ is small and/or λ is large. This condition in turn corresponds to effectively retaining only the components of vec($\mathbf{T}$) which vary smoothly across the samples $\{1, \cdots, N\}$ as well as over the M nodes of the graph.

D. Learning an underlying graph

In developing KRG, we have so far assumed that the underlying graph is known *apriori* in terms of the graph-Laplacian matrix $\mathbf{L}$. Such an assumption may not hold true in many practical applications, since there is not necessarily one best graph to describe the given networked data. This motivates us to develop a joint learning approach where we learn both the graph-Laplacian $\mathbf{L}$ and the KRG parameter matrix $\mathbf{W}$ (or its dual representation parameter $\mathbf{\Psi}$). Our goal in this section is to provide a simple means of estimating a graph that helps enhance the prediction performance, if a graph is not known apriori. We note that a vast and expanding body of literature exists in the domain of estimating graphs from graph signals [48]–[54], [93]–[95], and that many of these techniques may be used in the learning approach proposed in this section. Nevertheless, we pursue the particular approach taken in this section due to the ease of implementation and the minimal assumptions involved.

We propose the minimization of the following cost function to achieve our goal:

$$\begin{aligned}C(\mathbf{W}, \mathbf{L}) &= \sum_n \|\mathbf{t}_n - \mathbf{y}_n\|_2^2 + \alpha\,\text{tr}(\mathbf{W}^\top\mathbf{W})\\ &\quad +\beta\sum_n \mathbf{y}_n^\top\mathbf{L}\mathbf{y}_n + \nu\,\text{tr}(\mathbf{L}^\top\mathbf{L}),\end{aligned}$$

where $\nu \geq 0$. Since our goal is to recover an undirected graph, we impose the appropriate constraints [96]. Firstly, any non-trivial graph has a graph-Laplacian matrix $\mathbf{L}$ which is

symmetric and positive semi-definite [96]. Secondly, the vector of all ones $\mathbf{1}$ forms the eigenvector of the graph-Laplacian corresponding to the zero eigenvalue. Since $\mathbf{L} = \mathbf{D} - \mathbf{A}$, we have that $\mathbf{L}$ being positive semi-definite is equivalent to the constraint that all the off-diagonal elements of $\mathbf{L}$ are non-positive. This is a simpler constraint than the direct positive semi-definiteness constraint. Then, the solution to the joint estimation of $\mathbf{W}$ and $\mathbf{L}$ is obtained by solving the following:

$$\min_{\mathbf{W},\mathbf{L}} C(\mathbf{W},\mathbf{L}) \quad \text{such that} \quad \mathbf{L}(i,j) \leq 0 \; \forall i \neq j, \\ \mathbf{L} = \mathbf{L}^\top, \mathbf{L}\mathbf{1} = \mathbf{0}. \tag{18}$$

The optimization problem (18) is jointly non-convex over $\mathbf{W}$ and $\mathbf{L}$, but convex on $\mathbf{W}$ given $\mathbf{L}$ and vice-versa. Hence, we adopt an alternating minimization approach and solve (18) in two steps alternatingly as follows:

- For a given $\mathbf{L}$, solve $\min_{\mathbf{W}} C(\mathbf{W})$ using the KRG approach of Section II.
- Given the matrix $\mathbf{W}$, solve $\min_{\mathbf{L}} C(\mathbf{L})$ such that $\mathbf{L}(i,j) \leq 0 \; \forall i \neq j, \mathbf{L} = \mathbf{L}^\top$, and $\mathbf{L}\mathbf{1} = \mathbf{0}$. Here $C(\mathbf{L}) = \beta \sum_n \mathbf{y}_n^\top \mathbf{L}\mathbf{y}_n + \nu \, \mathrm{tr}(\mathbf{L}^\top \mathbf{L})$.

We start the alternating optimization using a suitable initialization; initializing $\mathbf{L} = \mathbf{0}$ yields the KR. In order to keep the successive $\mathbf{L}$ estimates comparable, we scale $\mathbf{L}$ such that the largest eigenvalue modulus is unity at every iteration.

III. Experimental results

We evaluate the performance of the relevant methods under adverse conditions where we use limited training data and noisy training data. Our hypothesis is that KRG and LRG provide better prediction performance than KR and LR, respectively. A state-of-the-art method is also compared with in the experiments of graph signal reconstruction. We experiment with both synthesized and real-world signal examples. The experiment with the synthesized data is carried out using small-world graphs; this being a standard practice in several existing works to demonstrate the efficiency of a model. For real applications, we consider three different real-world data experiments:

(D1) Prediction of the temperature as the output, using the air-pressure observations as the input, for the cities in Sweden.

(D2) Temperature prediction for the cities in Sweden from the current day to the next day.

(D3) Prediction for the fMRI voxel intensities of the cerebellum region.

Among these three experiments, D1 is the experiment where the input observation and the output to be predicted are two different physical quantities. To the best of our knowledge, none of the existing graph-signal processing approaches address such a dataset and we therefore, make comparisons only with conventional linear/kernel regression. The other two experiments D2 and D3 are performed for two reasons. The first reason is to compare our method against the kernel-ridge regression (KRR) method of [82], [85]. In these experiments, the input and the output both lie on the same graph and are the same physical quantities, making it applicable to the KRR method. We choose KRR as the competing method in these two experiments because it has been claimed to provide a state-of-the-art performance [82]. The second reason is to investigate the performance of our method when we simultaneously learn an underlying graph from the available data.

We use the normalized-mean-square-error (NMSE) as the measure of the prediction performance:

$$\text{NMSE} = 10 \log_{10} \left(\frac{\mathrm{E}\|\mathbf{Y} - \mathbf{T}_0\|_F^2}{\mathrm{E}\|\mathbf{T}_0\|_F^2} \right),$$

where $\mathbf{Y}$ denotes the regression output matrix and $\mathbf{T}_0$ the true value of target matrix, meaning that $\mathbf{T}_0$ does not contain any noise. The expectation operation $\mathrm{E}(\cdot)$ is realized as the sample average over multiple experiment trials. In the case of real-world examples, we compare the performance of the following four regression approaches:

1) linear regression (LR): $k_{m,n} = \boldsymbol{\phi}(\mathbf{x}_m)^\top \boldsymbol{\phi}(\mathbf{x}_n)$, where $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$ and $\beta = 0$,
2) linear regression over graphs (LRG): $k_{m,n} = \boldsymbol{\phi}(\mathbf{x}_m)^\top \boldsymbol{\phi}(\mathbf{x}_n)$ and $\beta \neq 0$, where $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$,
3) kernel regression (KR): Using $\beta = 0$ and the radial basis function (RBF) kernel $k_{m,n} = \exp\left(-\frac{\|\mathbf{x}_m - \mathbf{x}_n\|_2^2}{\sigma^2 \sum_{m',n'} \|\mathbf{x}_{m'} - \mathbf{x}_{n'}\|_2^2 / N} \right)$, and
4) kernel regression over graphs (KRG): Using $\beta \neq 0$ and the RBF kernel $k_{m,n} = \exp\left(-\frac{\|\mathbf{x}_m - \mathbf{x}_n\|_2^2}{\sigma^2 \sum_{m',n'} \|\mathbf{x}_{m'} - \mathbf{x}_{n'}\|_2^2 / N} \right)$.

The regularization parameters α, β, σ^2 and ν for different training data sizes are found through a five-fold crossvalidation on the training set. While performing the crossvalidation, we assume that clean target vectors are available. We wish to emphasize that our goal here is to illustrate that LRG and KRG are better than LR and KR, respectively. Depending on the choice of the kernel used, one kernel may perform better than the other and that there is generally no guarantee that a Gaussian kernel always outperforms the linear kernel in practice. As regards determining which kernel is best suited to an application, there is often no direct answer than a trial-and-error approach of using different kernels and comparing their performance. An alternative is to use a kernel selection approach in the form of multi-kernel regression [83], [97], [98] which 'selects' the best kernels from a bag of kernels.

A. Regression for synthesized data

We perform regression for the synthesized dataset where the target vectors to be predicted are smooth over a specified graph. We generate synthesized data where we know the ground truth. A part of the data is used for the training in presence of additive noise, and our task is to predict for the remaining part of the data, given the information about correlations in form of kernels. In order to generate the synthesized data, we use random small-world graphs from the Erdős-Rényi and the Barabási-Albert models [99] with

the number of nodes equal to $M = 50$. We generate a total of S target vector realizations. We adopt the following data generation strategy: We first pick M independent vector realizations from an S-dimensional Gaussian vector source $\mathcal{N}(\mathbf{0}, \mathbf{C}_S)$, where $\mathbf{C}_S$ is an S-dimensional covariance matrix drawn from the inverse Wishart distribution with an identity hyperparameter matrix. We use a highly correlated covariance matrix $\mathbf{C}_S$ such that each S-dimensional vector has strongly correlated components. Thus, we create a data matrix with M columns such that each column is an S-dimensional Gaussian vector. Each row vector of the data matrix has a size M and the row vectors of the data matrix are correlated to each other. We denote a row vector by $\mathbf{r}$. We then select the row vectors $\{\mathbf{r}\}$ one-by-one and project them onto the specified graph to generate target vectors $\{\mathbf{t}\}$ that are smooth over the graph while maintaining the correlation between observations, by solving the following optimization problem:

$$\mathbf{t} = \arg\min_{\mathbf{z}} \left\{ \|\mathbf{r} - \mathbf{z}\|_2^2 + \mathbf{z}^\top \mathbf{L}\mathbf{z} \right\}.$$

We randomly divide the S data samples into training and test sets of equal size $N_{tr} = N_{ts} = S/2$. We define the kernel function between the ith and jth data samples $\mathbf{z}_i$ and $\mathbf{z}_j$ to be

$$k_{i,j} \triangleq \mathbf{C}_S(i,j),$$

considering the same kernel for all the graph nodes. The choice of the kernel is motivated by the assumed generating model. Given the training set of size N_{tr}, we choose a subset of N data samples to make predictions for the N_{ts} test data samples using the kernel regression over graphs. The training target vectors are corrupted with additive white Gaussian noise at varying levels of signal-to-noise ratio (SNR). We repeat our experiments over 100 realizations of the graphs and noise realizations. We compare the performance of KRG with KR. We observe from Figure 1 that for a fixed training data size of $N = 50$, KRG outperforms KR by a significant margin at low SNR levels (below 10dB). As the SNR-level increases, the NMSE of KRG and KR almost coincide. A similar trend is also observed in the case of Barabási-Albert graphs, which is not reported for brevity. In Figure 2, we show the NMSE obtained with KRG and KR on both the graph models, as a function of the training data size at an SNR-level of 5 dB. We observe that KRG consistently outperforms KR and that the gap between the NMSE of KRG and KR reduces as the training data size increases. The results shown in Figure 1 and 2 verify our hypothesis.

B. Experiment D1: Prediction of the temperature of the cities using air-pressure observations

We now consider the experiment where the input and the output of the kernel regression are two different physical quantities. The task is to predict the temperature of the cities in Sweden from the air pressure observations at those cities. We predict the temperature as the average daily temperature for 24 hours of a day; the input consists of the air pressure observations collected on an hourly basis for the same day.

For the experiment, we collected the temperature and air-pressure measurements from the 25 most populated cities in

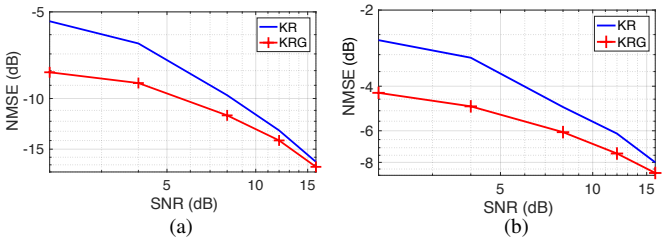

Fig. 1. Performance for the synthesized dataset using Erdős-Rényi graphs. We plot the NMSE against SNR for the training data size $N = 50$. (a) NMSE for training data, and (b) NMSE for test data.

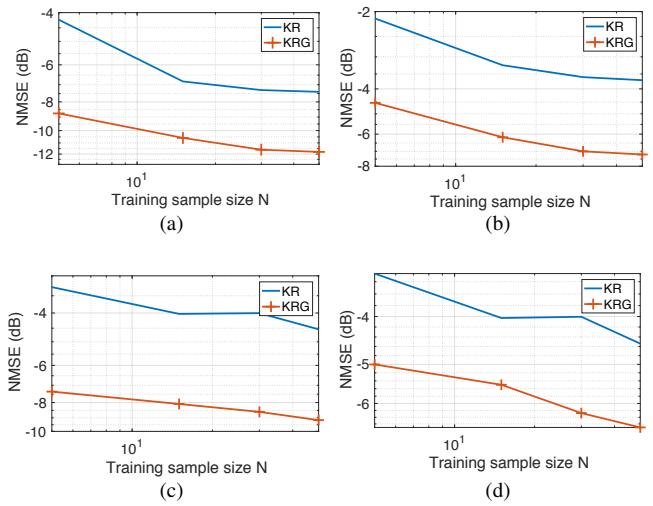

Fig. 2. Performance for the synthesized dataset at a 5 dB SNR level. We use Erdős-Rényi graphs for subfigures (a) and (b). (a) NMSE for training data, and (b) NMSE for test data. Then we use Barabási-Albert graphs for subfigures (c) and (d). (c) Training data performance and (d) Test data performance.

Sweden. The data was collected for a period of two months from February to March of 2018. In Figure 3(a), we indicate the 45 most populated cities in Sweden. We consider 25 of these 45 cities in this experiment since the relevant data was not available at the remaining cities. The data is available publicly from the Swedish Meteorological and Hydrological Institute [100]. We predict the temperature of the 25 cities as the output vector or the target. The input is taken to be the air pressure measurement at all those 25 cities collected on hourly basis. This results in an input vector with $24 \times 25 = 600$ components, which means that we have $\mathbf{t}_n \in \mathbb{R}^{25}$ and $\mathbf{x}_n \in \mathbb{R}^{600}$. The data from the first 48 days is taken as the training set, and the data from the remaining 12 days is used for testing. Let d_{ij} denote the geodesic distance between the ith and jth cities in kilometres, $\forall i, j \in \{1, \cdots, 25\}$. Then, we construct the adjacency matrix $\mathbf{A}$ for the graph by setting

$$\mathbf{A}(i,j) = \exp\left(-\frac{d_{ij}^2}{\sum_{i,j} d_{ij}^2} \right).$$

In our experiments, we randomly sample for the training observations from the full training set for various $N \leq N_{tr} = 48$. We consider the case when the training targets are corrupted with additive white Gaussian noise at a 10 dB SNR level. We compare the performances of LR, LRG, KR, and KRG. For

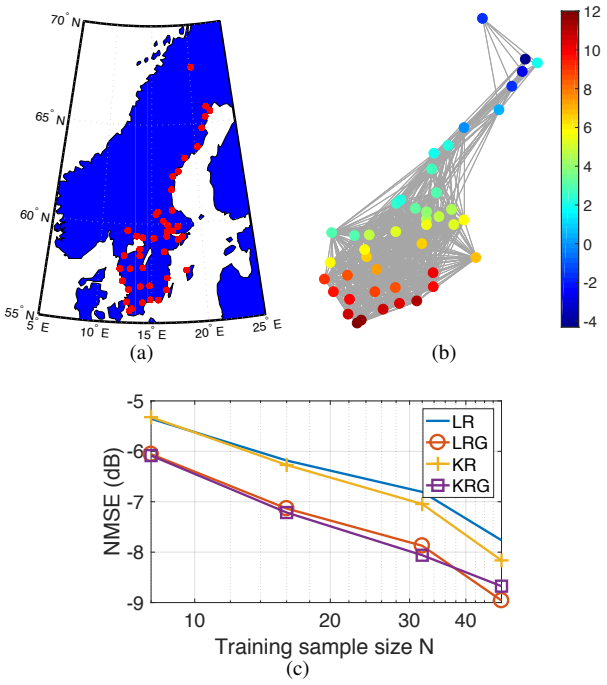

Fig. 3. Results for the experiment (D1). (a) The map of Sweden with the major 45 cities indicated, (b) True temperature measurement signal for a particular day in the dataset shown in units of degree celsius (The graph is the same as that in subfigure (a) but without the underlying map.). (c) NMSE as a function of the training sample size, with additive white Gaussian noise at a 10 dB SNR-level.

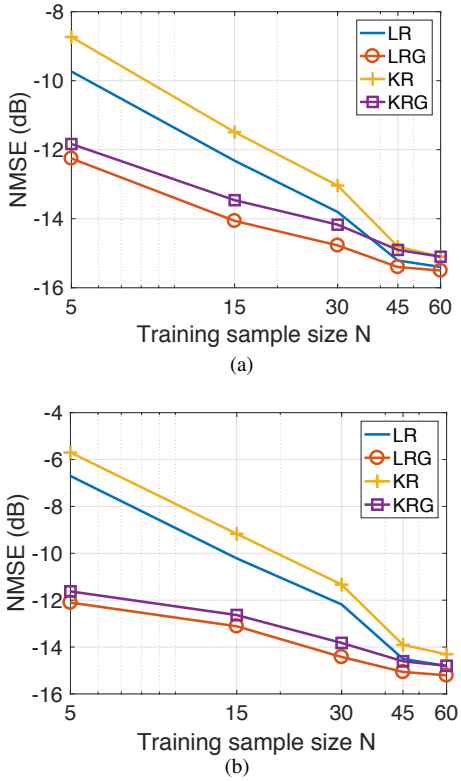

Fig. 4. Results for experiment (D2). (a) NMSE for test data with additive white Gaussian noise at a 5dB SNR level, (b) NMSE for test data with additive white Gaussian noise at a 0dB SNR level.

the test data, the NMSE as a function of the training sample size N is shown in Figure 3. We observe that LRG and KRG outperform LR and KR, respectively, by a significant margin, particularly at smaller training sample sizes. In addition to this, we find that KRG outperforms LRG for this experiment, though this is not necessarily guaranteed for all the datasets and under all experimental conditions.

C. Experiment D2: Temperature prediction from the current day to the next day

In this experiment, the task is to predict the temperature of several Swedish cities for the next day from the temperature observations of the current day. For the experiment, we consider the temperature measurements from the 45 most populated cities in Sweden taken for a period of three months from September to November 2017. Since both the input and the output are temperatures, this experiment represents a graph signal reconstruction/recovery problem and hence, we compare our method with the KRR method. We have already mentioned that KRR is a state-of-the-art method in graph signal recovery [82], [85]. Further, we also consider the case when the underlying graph is not known a-priori. In this case, we learn an underlying graph and compare the performance of our approach against the case where the graph is known a-priori. The data is available publicly from the Swedish Meteorological and Hydrological Institute [100]. The cities are indicated in the map of Sweden in Figure 3(a). We consider the target vector $\mathbf{t}_n$ to be the temperature measurement of a particular day, and $\mathbf{x}_n$ to be the temperature measurements

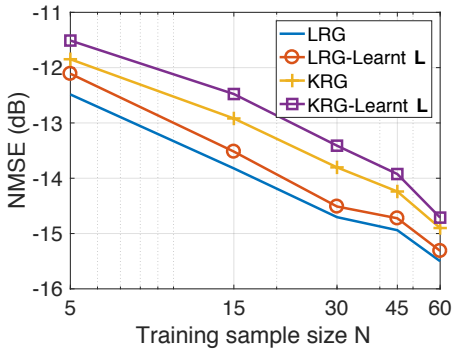

Fig. 5. Results for experiment (D2) with graph learning, for training data samples at a 5dB SNR-level for $N = 45$.

(in degree celsius units) from the previous day. We have 90 input-target data pairs in total, divided into the training set and the test set of sizes $N_{tr} = 60$ and $N_{ts} = 30$, respectively. Once again, we consider the geodesic distance based graph. For each training dataset size N, we compute the NMSE by averaging over 50 different random training subsets of size N drawn from the full training set of size N_{tr}. In Figure 4, we show the NMSE for the test set at SNR-levels of 5 dB and 0 dB. We observe that KRG outperforms other regression methods by a significant margin, particularly at low sample sizes N. Next, we compare our methods with KRR.

1) Comparison with KRR: KRR deals with a sub-sampling problem where the signal values are predicted at a set of nodes from the signal values given at the remaining set of nodes.

TABLE I
PERFORMANCE COMPARISON OF THE PROPOSED METHODS AND KRR FOR THE EXPERIMENT D2 (USING NMSE IN dB)

Training sample size N	Kernel Ridge Regression (KRR)			Proposed Methods			
	Diffusion kernel	Covariance kernel at 5dB SNR	Covariance kernel at 0dB SNR	LRG at 5dB SNR	KRG at 5dB SNR	LRG at 0dB SNR	KRG at 0dB SNR
5	2.8×10^{-4}	-6.5	-3.5	-12.3	-11.9	-12.1	-11.6
15	”	-7.2	-4.9	-14.0	-13.4	-13.1	-12.7
30	”	-9.8	-6.7	-14.8	-14.2	-14.4	-13.8
45	”	-10.5	-7.8	-15.4	-14.9	-15.0	-14.6
60	”	-12.2	-9.5	-15.5	-15.1	-15.2	-14.8

Therefore, we formulate the one-day temperature prediction problem in a suitable sub-sampling setup where KRR can be used. In the sub-sampling setup, KRR minimizes the following convex cost:

$$\arg\min_{\boldsymbol{\alpha}} \left\|\mathbf{x} - \mathbf{S}\bar{\mathbf{K}}\mathbf{S}\boldsymbol{\alpha}\right\|_2^2 + \mu\boldsymbol{\alpha}^\top\bar{\mathbf{K}}\boldsymbol{\alpha},\ \text{s.t. } \bar{\mathbf{K}}\boldsymbol{\alpha} = \begin{bmatrix}\mathbf{y}\\ \hat{\mathbf{x}}\end{bmatrix} \quad (19)$$

where $\mathbf{x} \in \mathbb{R}^s$ is the input observation signal corresponding to a subset Ω with s nodes, from the total set of M nodes. Here, $\mathbf{S}$ denotes the sampling matrix obtained by concatenating the zero matrix and the s-dimensional identity matrix; $\bar{\mathbf{K}}$ is the kernel matrix across all nodes of the graph, and $\boldsymbol{\alpha}$ is the vector of KRR coefficients, and $\hat{\mathbf{x}}$ is the estimate of the graph signal produced by KRR at Ω. The estimate of the entire graph signal is then given by:

$$\begin{bmatrix}\mathbf{y}\\ \hat{\mathbf{x}}\end{bmatrix} = \bar{\mathbf{K}}\mathbf{S}^\top(\mathbf{S}\bar{\mathbf{K}}\mathbf{S}^\top + \mu s\mathbf{I}_s)^{-1}\mathbf{x}. \quad (20)$$

Thus, KRR achieves an extrapolation of the graph signal from the nodes in Ω to those outside it using the graph topoplogy employed in the extrapolation kernel. The parameters related to the above prediction and kernels are found by cross-validation. In all the experiments employing KRR [82], [84], we have used the same diffusion kernels and the covariance kernels used by the authors in the corresponding articles [82], [84].

Since we consider the one-day temperature prediction problem, we use the space-time variant of the KRR proposed in [84], by taking the adjacency matrix given by the Cartesian product of the geodesic graph $\mathbf{A}$ and the temporal dynamics graph for one time step $\mathbf{B} = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$, meaning that each node at time n is connected to the corresponding node at time $n+1$ by an edge with unity weight. The composite or augmented graph [84] for the two days is then given by the Cartesian product $\mathbf{A} \oplus \mathbf{B} = \begin{bmatrix}\mathbf{A} & \mathbf{I}\\ \mathbf{I} & \mathbf{A}\end{bmatrix}$. We observe from Table I, that our approach significantly outperforms KRR for both the covariance and the diffusion kernels. We note that the performance of the covariance kernel is better than that of the diffusion kernel, and this trend is in agreement with the results reported in [84]. We also observe that the performance of KRR and our approaches improve as more training data becomes available.

The performance of our approach is significantly better than that of KRR. This can be attributed to two factors. The first factor is that KRR deals with an under-determined setup where the subsampling matrix has a special structure. The special structure is formed by concatenating the identity matrix and the zero matrix. This sampling matrix structure may not be well suited for sub-sampling. The second aspect or factor pertains to our approach: we use the advantage of explicit training and testing. This assumes the availability of a training dataset for our approach, whereas KRR does not have that as a requirement.

2) Learning an underlying graph: We next consider the performance of our approach when the graph is also simultaneously learnt from the training data. This experiment serves the purpose of illustrating the effectiveness of our approach in inferring a graph suited to the prediction task even in the absence of a prior graph. We use the alternating optimization strategy of Section II-D initialized with $\mathbf{L} = \mathbf{0}$. We consider the training data at an SNR level of 5 dB at different sample sizes. We find experimentally that the algorithm converges typically after five to ten iterations. In Figure 5, we plot the NMSE values obtained for the test data using both the fixed $\mathbf{L}$ based on the geodesic distances, and with the learnt graph. We observe that our approach learns a graph which provides agreeable performance even when initialized with the zero graph. This validates our intuition that the graph signal holds sufficient information to both infer both a meaningful underlying graph structure and perform target predictions.

D. Experiment D3: Prediction for the fMRI voxel intensities of the cerebellum region

Finally, we consider the prediction of voxel intensities in the functional magnetic resonance imaging (fMRI) data obtained for the cerebellum region of the brain. We apply our approach to predict the intensities at some of the voxels of the MRI, given the intensities at the other voxels. As before, we consider the training samples to be corrupted with additive white Gaussian noise. In the beginning, the graph is constructed from the voxels at the different slices of the MRI scans connected together to form a composite graph as shown in Figure 6(a). The details of the image acquisition and the dataset may be found publicly at https://openfmri.org/dataset/ds000102. Each voxel is considered as a node of the graph and the voxel intensity to be the signal. The full data graph is of dimension 4000 obtained by mapping the 4000 cerebellum voxels anatomically following the atlas template [102]. We refer the reader to [103] for further details on the construction of the voxel graph. In our analysis, we consider only the first 100 voxels from the 4000 voxels to construct the dataset in our experiments. This is to reduce the computational complexity in performing the experiments. We use the intensity values at

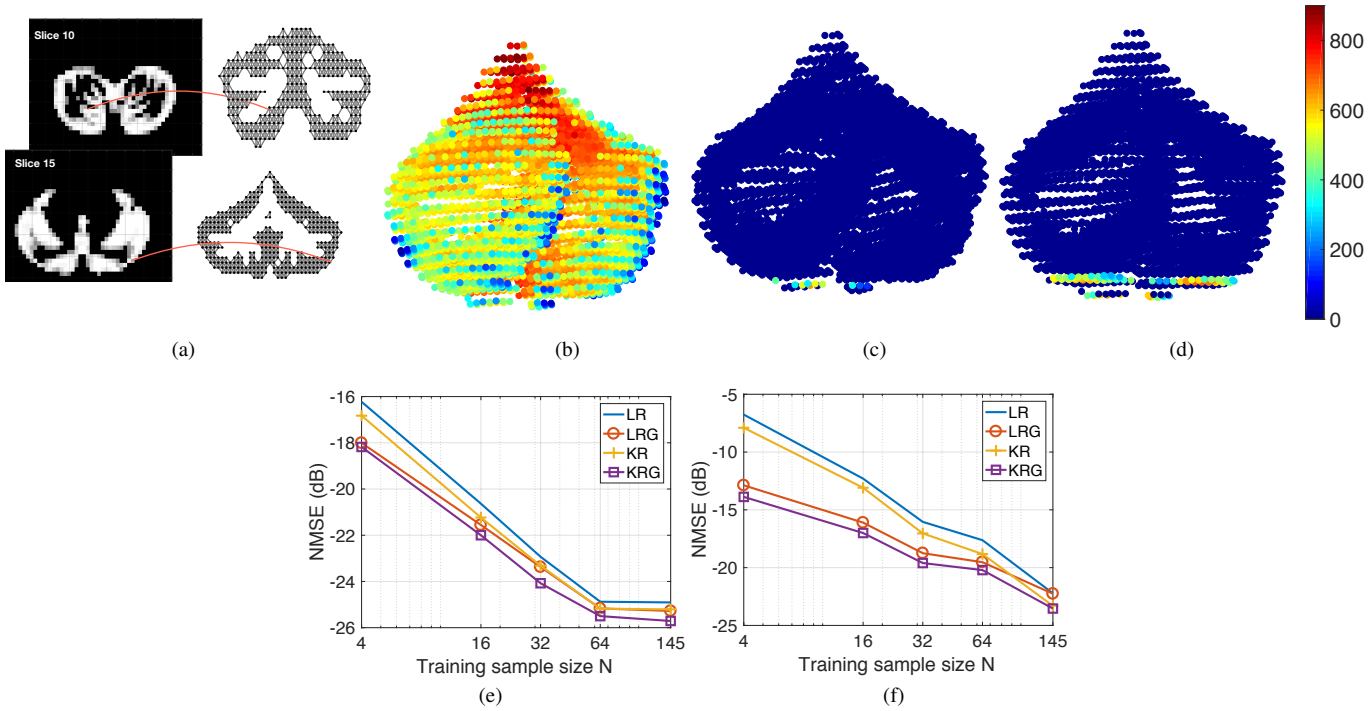

Fig. 6. Results for the cerebellum data (D3). (a) Representation of how the graph is constructed from the voxels at the different slices (b) The entire graph with an instance of the graph signal, and the corresponding intensities (c) at only the voxels used as the input (d) at only the voxels used as the output (the edges are omitted for clarity), (e) NMSE for test data at a 10dB SNR-level, (f) NMSE for test data at a 0dB SNR-level.

TABLE II
COMPARISON OF THE PROPOSED METHODS AND KRR FOR THE EXPERIMENT D3

Training sample size N	Kernel Ridge Regression (KRR)			Proposed Methods			
	Diffusion kernel	Covariance kernel at 10dB SNR	Covariance kernel at 0dB SNR	LRG at 10dB SNR	KRG at 10dB SNR	LRG at 0dB SNR	KRG at 0dB SNR
145	0.01	-1.0	-0.7	-25.3	-25.7	-22.2	-23.5

10 of the voxels from the first slice as the input $\mathbf{x} \in \mathbb{R}^{10}$ to make predictions for the output $\mathbf{t} \in \mathbb{R}^{90}$ comprising the voxel instensities at 90 voxels present in the first and second slice. In Figure 6(b), we show an instance of the voxel intensity signal over the full 4000 voxel graph. In Figures 6(c) and (d), we show the corresponding input and output signals used in our experiments. The dataset consists of 290 input-target data points or observation pairs. We use one half of the data for training and the other half for testing. We construct noisy training data at SNR levels of 10 dB and 0 dB. The NMSE is obtained by computing the average over 100 different random partitions of the entire data into the training and test sets. The results are shown in Figure 6(e)-(f). We observe that LRG and KRG have a superior performance over their conventional counterparts, particularly for small N and at low SNR-levels.

The performance of our methods and that of KRR with the maximum number of training samples is reported in Table II. We observe that KRR performs poorly in comparison with our approaches. The poor performance of KRR may be attributed to the relatively small number of samples available for reconstruction: only 10% of the total number of nodes are observed. Further, we note that KRR does not explicitly employ training data other than that used in the construction of the covariance kernel. The covariance kernel approach in turn also requires sufficient number of samples for a reliable reconstruction, which is not the case in our experiments owing to the adverse training data. All these factors explain why KRR performs rather poorly in this experiment.

We now consider the experiment with the learning of an underlying graph. We observe from Figure 7(a) that when initialized with the zero graph, the prediction performance of our method is comparable to that obtained using the fixed atlas-template graph (given graph). We further consider the case when the graph learning iterations are initalized with $\mathbf{L}$ corresponding to the atlas-template. This is motivated by the observation that in many applications such as biomedical data, there is no single graph which is guaranteed to work the best for prediction. The goal of this experiment is then to investigate if a graph better suited to the regression task could be learnt starting from an existing non-trivial graph. The prediction NMSE obtained for the test data in this case is shown in Figure 7(b). We observe that our method learns a graph better suited to the prediction in terms of the NMSE. This in turn shows that for this dataset, it is more suitable to jointly learn an underlying graph.

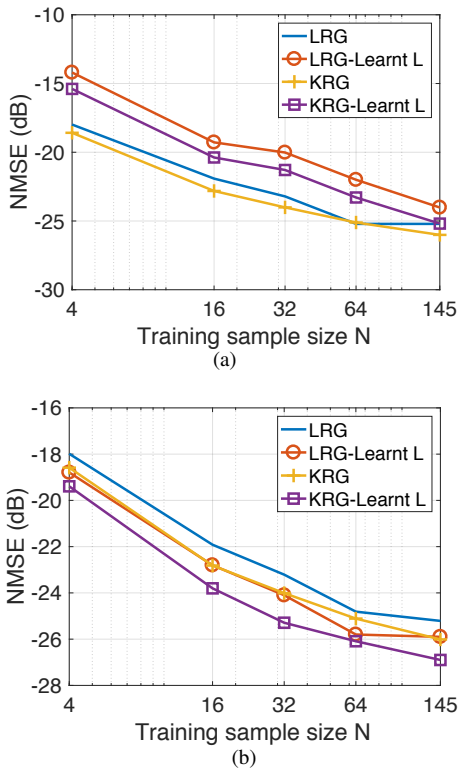

Fig. 7. Results for (D3) with graph learning for the fMRI data, when the training data samples are corrupted with additive white Gaussian noise at 10dB SNR-level for $N = 45$. (a) NMSE for the test data when initialized with $\mathbf{L} = \mathbf{0}$, and (b) when initialized with $\mathbf{L}$ corresponding to the graph based on atlas template [101].

IV. Conclusions

We proposed a kernel regression method for predicting graph signal outputs from inputs that are not necessarily lying over a graph or for inputs agnostic to a graph. The resulting problem was shown to be a convex one resulting in an analytically tractable solution. Our approach presents a generalization of the standard kernel regression for graph signals. Experiments with synthesized and real-world graph signal datasets demonstrated the merit of our approach, particularly in the adverse scenarios of training with noise and limited datasizes. Our approach was shown to outperform a state-of-the-art method in real-world graph signal reconstruction problems. We further showed that our approach is also applicable in cases where an underlying graph is simultaneously estimated from the training data.

V. Reproducible research

In the spirit of reproducible research, all the codes relevant to the experiments in this article are made available at https://www.researchgate.net/profile/Arun_Venkitaraman and https://www.kth.se/ise/research/reproducibleresearch-1.433797.

References

[1] D. I. Shuman, S. 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, 2013.

[2] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, 2013.

[3] ——, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, 2014.

[4] ——, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, 2014.

[5] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "A windowed graph Fourier transform," *IEEE Statist. Signal Process. Workshop (SSP)*, pp. 133–136, Aug 2012.

[6] S. K. Narang and A. Ortega, "Local two-channel critically sampled filter-banks on graphs," *Proc. IEEE Int. Conf. Image Process. (ICIP)*, pp. 333–336, 2010.

[7] ——, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2786–2799, 2012.

[8] ——, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4673–4685, 2013.

[9] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Appl. Computat. Harmonic Anal.*, vol. 21, no. 1, pp. 53–94, 2006.

[10] D. Ganesan, B. Greenstein, D. Estrin, J. Heidemann, and R. Govindan, "Multiresolution storage and search in sensor networks," *ACM Trans. Storage*, vol. 1, no. 3, pp. 277–315, 2005.

[11] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Computat. Harmonic Anal.*, vol. 30, no. 2, pp. 129–150, 2011.

[12] R. Wagner, V. Delouille, and R. Baraniuk, "Distributed wavelet de-noising for sensor networks," *Proc. 45th IEEE Conf. Decision Control*, pp. 373–379, 2006.

[13] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "Vertex-frequency analysis on graphs," *Appl. Comput. Harmonic Anal.*, vol. 40, no. 2, pp. 260 – 291, 2016.

[14] D. I. Shuman, M. J. Faraji, and P. Vandergheynst, "A multiscale pyramid transform for graph signals," *IEEE Trans. Signal Process.*, vol. 64, no. 8, pp. 2119–2134, April 2016.

[15] O. Teke and P. P. Vaidyanathan, "Extending classical multirate signal processing theory to graphs-part I: Fundamentals," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 409–422, Jan 2017.

[16] ——, "Extending classical multirate signal processing theory to graphs-part II: M-channel filter banks," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 423–437, Jan 2017.

[17] A. Venkitaraman, S. Chatterjee, and P. Handel, "On Hilbert transform of signals on graphs," *Proc. Sampling Theory Appl.*, 2015. [Online]. Available: http://w.american.edu/cas/sampta/papers/a13-venkitaraman.pdf

[18] A. Venkitaraman, S. Chatterjee, and P. Händel, "On Hilbert transform, analytic signal, and modulation analysis for signals over graphs," *Signal Process.*, vol. 156, pp. 106–115, 2019.

[19] N. Tremblay, G. Puy, P. Borgnat, R. Gribonval, and P. Vandergheynst, "Accelerated spectral clustering using graph filtering of random signals," *IEEE Int. Conf. Acoust. Speech Signal Process.*, 2016.

[20] N. Tremblay and P. Borgnat, "Joint filtering of graph and graph-signals," *Asilomar Conf. Signals Syst. Comput.*, pp. 1824–1828, Nov 2015.

[21] S. Chen, R. Varma, A. Sandryhaila, and J. Kovacevic, "Discrete signal processing on graphs: Sampling theory," *IEEE Trans. Signal Process.*, vol. 63, no. 24, pp. 6510–6523, Dec 2015.

[22] F. Gama, A. G. Marques, G. Mateos, and A. Ribeiro, "Rethinking sketching as sampling: Linear transforms of graph signals," *Proc. Asilomar Conf. Signals Syst. Comput.*, pp. 522–526, Nov 2016.

[23] S. P. Chepuri and G. Leus, "Subsampling for graph power spectrum estimation," *Proc. IEEE Sensor Array Multichannel Signal Process. Workshop (SAM)*, pp. 1–5, July 2016.

[24] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal recovery on graphs: Fundamental limits of sampling strategies," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 4, pp. 539–554, Dec 2016.

[25] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1832–1843, April 2016.

[26] H. Q. Nguyen and M. N. Do, "Downsampling of signals on graphs via maximum spanning trees," *IEEE Trans. Signal Process.*, vol. 63, no. 1, pp. 182–191, Jan 2015.

[27] M. Tsitsvero, S. Barbarossa, and P. D. Lorenzo, "Signals on graphs: Uncertainty principle and sampling," *IEEE Trans. Signal Process.*, vol. 64, no. 18, pp. 4845–4860, 2016.

[28] 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, July 2016.

[29] L. F. O. Chamon and A. Ribeiro, "Greedy sampling of graph signals," *IEEE Trans. Sig. Process.*, vol. 66, no. 1, pp. 34–47, Jan 2018.

[30] S. P. Chepuri and G. Leus, "Graph sampling for covariance estimation," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 451–466, Sep. 2017.

[31] S. K. Narang, A. Gadde, and A. Ortega, "Signal processing techniques for interpolation in graph structured data," in *IEEE Int. Conf. Acoust. Speech Signal Process.*, May 2013, pp. 5445–5449.

[32] N. Shahid, V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Robust principal component analysis on graphs," *IEEE Int. Conf. Comput. Vision (ICCV)*, pp. 2812 – 2820, 2015.

[33] N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst, "Fast robust pca on graphs," *IEEE J. Selected Topics Signal Process.*, vol. 10, no. 4, pp. 740–756, June 2016.

[34] 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, 2014.

[35] D. Thanou and P. Frossard, "Multi-graph learning of spectral graph dictionaries," in *IEEE Int. Conf. Acoust. Speech Signal Process.*, April 2015, pp. 3397–3401.

[36] 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.

[37] Y. Yankelevsky and M. Elad, "Dual graph regularized dictionary learning," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 4, pp. 611–624, Dec 2016.

[38] B. Girault, "Stationary graph signals using an isometric graph translation," *Proc. Eur. Signal Process. Conf. (EUSIPCO)*, pp. 1516–1520, Aug 2015.

[39] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, "Stationary graph processes: Nonparametric spectral estimation," *Proc. IEEE Sensor Array Multichannel Signal Process. Workshop (SAM)*, pp. 1–5, July 2016.

[40] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *IEEE Trans. Signal Process.*, vol. 65, no. 22, pp. 5911–5926, Nov 2017.

[41] N. Perraudin and P. Vandergheynst, "Stationary signal processing on graphs," *IEEE Trans. Sig. Proc.*, vol. 65, no. 13, pp. 3462–3477, Jul. 2017.

[42] B. Girault, P. Goncalves, and E. Fleury, "Translation on graphs: An isometric shift operator," *IEEE Signal Process. Lett.*, vol. 22, no. 12, pp. 2416–2420, Dec. 2015.

[43] B. Girault, "Stationary graph signals using an isometric graph translation," *Proc. Eur. Signal Process. Conf. (EUSIPCO)*, 2015.

[44] P. Berger, G. Hannak, and G. Matz, "Graph signal recovery via primal-dual algorithms for total variation minimization," *IEEE J. Selected Topics Signal Process.*, vol. 11, no. 6, pp. 842–855, Sept 2017.

[45] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovacevic, "Signal recovery on graphs: Variation minimization," *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4609–4624, Sept 2015.

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

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

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

[49] Y. Shen, B. Baingana, and G. B. Giannakis, "Tensor decompositions for identifying directed graph topologies and tracking dynamic networks," *IEEE Trans. Signal Process.*, vol. 65, no. 14, pp. 3675–3687, July 2017.

[50] C. Hu, L. Cheng, J. Sepulcre, G. E. Fakhri, Y. M. Lu, and Q. Li, "A graph theoretical regression model for brain connectivity learning of Alzheimer's disease," *Proc. IEEE Int. Symp. Biomed. Imag.*, pp. 616–619, April 2013.

[51] S. I. Daitch, J. A. Kelner, and D. A. Spielman, "Fitting a graph to vector data," *Proc. Int. Conf. Mach. Learn.*, pp. 201–208, 2009.

[52] N. Leonardi and D. Van De Ville, "Wavelet frames on graphs defined by fMRI functional connectivity," in *IEEE Int. Symp. Biomed. Imag.*, March 2011, pp. 2136–2139.

[53] N. Leonardi, J. Richiardi, M. Gschwind, S. Simioni, J.-M. Annoni, M. Schluep, P. Vuilleumier, and D. Van De Ville, "Principal components of functional connectivity: A new approach to study dynamic brain connectivity during rest," *NeuroImage*, vol. 83, pp. 937–950, 2013.

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

[55] S. P. Chepuri, S. Liu, G. Leus, and A. Hero, "Learning sparse graphs under smoothness prior," *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, pp. 6508–6512, 2017.

[56] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, "Characterization and inference of graph diffusion processes from observations of stationary signals," *IEEE Trans. Signal Inf. Process. Netw.*, pp. 1–1, 2017.

[57] 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.

[58] A. Venkitaraman, H. P. Maretic, S. Chatterjee, and P. Frossard, "Supervised linear regression for graph learning from graph signals," *CoRR*, vol. abs/1811.01586, 2018. [Online]. Available: http://arxiv.org/abs/1811.01586

[59] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.

[60] C. Cortes and V. Vapnik, "Support-vector networks," *J. Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[61] M. Seeger, "Gaussian processes for machine learning," *Int. J. Neural Syst.*, vol. 14, no. 02, pp. 69–106, 2004.

[62] L. Yann, Y. Bengio, , and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[63] Y. Cho and L. K. Saul, "Kernel methods for deep learning," in *Adv. Neural Inf. Process. Syst.*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds. Curran Associates, Inc., 2009, pp. 342–350.

[64] G.-B. Huang, "An insight into extreme learning machines: Random neurons, random features and kernels," *Cogn. Computat.*, vol. 6, pp. 376–390, 2014.

[65] A. Iosifidis, A. Tefas, and I. Pitas, "On the kernel extreme learning machine classifier," *Patt. Recognit. Lett.*, vol. 54, pp. 11 – 17, 2015.

[66] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," *Proc. Int. Conf. Mach. Learn. ICML*, pp. 315–322, 2002.

[67] A. J. Smola and R. Kondor, *Kernels and Regularization on Graphs*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 144–158.

[68] M. Belkin, I. Matveeva, and P. Niyogi, "Tikhonov regularization and semi-supervised learning on large graphs," *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, May 2004.

[69] ——, "Regularization and semi-supervised learning on large graphs," in *Proc. 17th Annual Conf. Learn. Theory*, J. Shawe-Taylor and Y. Singer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 624–638.

[70] A. Argyriou, M. Herbster, and M. Pontil, "Combining graph Laplacians for semi-supervised learning," *Proc. Neural Info. Process. Syst.*, pp. 67–74, 2005.

[71] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Dec. 2006.

[72] K. Zhang, L. Lan, J. T. Kwok, S. Vucetic, and B. Parvin, "Scaling up graph-based semisupervised learning via prototype vector machines," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 3, pp. 444–457, March 2015.

[73] K. I. Kim, J. Tompkin, H. Pfister, and C. Theobalt, "Context-guided diffusion for label propagation on graphs," in *IEEE Int. Conf. Comput. Vision (ICCV)*, Dec 2015, pp. 2776–2784.

[74] H. Takeda, S. Farsiu, and P. Milanfar, "Deblurring using regularized locally adaptive kernel regression," *IEEE Trans. Image Process.*, vol. 17, no. 4, pp. 550–563, 2008.

[75] H. Dou, D. Ming, Z. Yang, Z. Pan, Y. Li, and J. Tian, "Object-based visual saliency via Laplacian regularized kernel regression," *IEEE Trans. Multimedia*, vol. 19, no. 8, pp. 1718–1729, 2017.

[76] M. K. Chung and J. Taylor, "Diffusion smoothing on brain surface via finite element method," *Proc. IEEE Int. Symp. Biomed. Imag.*, pp. 432–435, 2004.

[77] M. K. Chung, P. Bubenik, and P. T. Kim, "Persistence diagrams of cortical surface data," *Int. Conf. Inf. Proc. Med. Imag.*, pp. 386–397, 2009.

[78] S. Seo, M. K. Chung, and H. K. Vorperian, "Heat kernel smoothing using Laplace-Beltrami eigenfunctions," *Proc. Med. Image. Comput. Assist. Interv.*, pp. 505–512, 2010.

[79] D. Pachauri, C. Hinrichs, M. K. Chung, S. C. Johnson, and V. Singh, "Topology-based kernels with application to inference problems in Alzheimer's disease," *IEEE Trans. Med. Imag.*, vol. 30, no. 10, pp. 1760–1770, Oct 2011.

[80] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt, "A stable multi-scale kernel for topological machine learning," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 4741–4748, Jun. 2015.

[81] V. Solo, J. Poline, M. A. Lindquist, S. L. Simpson, F. D. Bowman, M. K. Chung, and B. Cassidy, "Connectivity in fMRI: Blind spots and breakthroughs," *IEEE Trans. Med. Imag.*, vol. 37, no. 7, pp. 1537–1550, July 2018.

[82] 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.

[83] ——, "Estimating signals over graphs via multi-kernel learning," in *IEEE Statist. Signal Process. Workshop (SSP)*, June 2016, pp. 1–5.

[84] V. N. Ioannidis, D. Romero, and G. B. Giannakis, "Kernel-based reconstruction of space-time functions via extended graphs," in *Asilomar Conf. Signals Syst. Comput.*, Nov 2016, pp. 1829–1833.

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

[86] V. N. Ioannidis, M. Ma, A. N. Nikolakopoulos, G. B. Giannakis, and D. Romero, *Kernel-Based Inference of Functions Over Graphs*. Butterworth-Heinemann, 2018, pp. 173 – 198.

[87] M. K. Chung, A. Qiu, S. Seo, and H. K. Vorperian, "Unified heat kernel regression for diffusion, kernel smoothing and wavelets on manifolds and its application to mandible growth modeling in CT images," *Med. Image Anal.*, vol. 22, no. 1, pp. 63 – 76, 2015.

[88] Y. Shen, B. Baingana, and G. B. Giannakis, "Kernel-based structural equation models for topology identification of directed networks," *IEEE Trans. Signal Process.*, vol. 65, no. 10, pp. 2503–2516, May 2017.

[89] A. Venkitaraman, S. Chatterjee, and P. Händel, "Gaussian processes over graphs," *CoRR https://arxiv.org/abs/1803.05776*, vol. abs/1803.05776, 2018.

[90] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[91] C. F. V. Loan, "The ubiquitous Kronecker product," *J. Comput. Appl. Math.*, vol. 123, no. 1–2, pp. 85–100, 2000.

[92] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[93] P. K. Shivaswamy and T. Jebara, *Laplacian Spectrum Learning*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 261–276.

[94] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Laplacian matrix learning for smooth graph signal representation," in *IEEE Int. Conf. Acoust. Speech Signal Process.*, April 2015, pp. 3736–3740.

[95] V. Kalofolias, "How to learn a graph from smooth signals," *Proc. Int. Conf. Artif. Intell. Statist.*, vol. 51, pp. 920–929, May 2016.

[96] F. R. K. Chung, *Spectral Graph Theory*. AMS, 1996.

[97] C. Cortes, M. Mohri, and A. Rostamizadeh, "L2 regularization for learning kernels," in *Proc. Conf. Uncertainty in Artif. Intell.*, ser. UAI '09, 2009, pp. 109–116.

[98] A. Venkitaraman, S. Chatterjee, and P. Händel, "Multi-kernel regression for graph signal processing," in *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, April 2018, pp. 4644–4648.

[99] M. E. J. Newman, *Networks: An Introduction*. Oxford University Press, 2010.

[100] Swedish meteorological and hydrological institute (smhi). [Online]. Available: http://opendata-download-metobs.smhi.se/

[101] H. Behjat, U. Richter, D. Van De Ville, and L. Sörnmo, "Signal-adapted tight frames on graphs," *IEEE Trans. Signal Process.*, vol. 64, no. 22, pp. 6017–6029, Nov 2016.

[102] J. Diedrichsen, J. H. Balsters, J. Flavell, E. Cussans, and N. Ramnani, "A probabilistic MR atlas of the human cerebellum," *NeuroImage*, vol. 46, no. 1, pp. 39 – 46, 2009.

[103] H. Behjat, N. Leonardi, L. Sörnmo, and D. Van De Ville, "Anatomically-adapted graph wavelets for improved group-level fMRI activation mapping," *NeuroImage*, vol. 123, pp. 185 – 199, 2015.