Kernel Regression for Signals over Graphs

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

Abstract

We propose kernel regression for signals over graphs. The optimal regression coefficients are learnt using a constraint that the target vector is a smooth signal over an underlying graph. The constraint is imposed using a graph-Laplacian based regularization. We discuss how the proposed kernel regression exhibits a smoothing effect, simultaneously achieving noise-reduction and graph-smoothness. We further extend the kernel regression to simultaneously learn the underlying graph and the regression coefficients. We validate our theory by application to various synthesized and real-world graph signals. Our experiments show that kernel regression over graphs outperforms conventional regression, particularly for small sized training data and under noisy training. We also observe that kernel regression reveals the structure of the underlying graph even with a small number of training samples.

Index Terms

Linear model, regression, kernels, machine learning, graph signal processing, graph-Laplacian.

EDICS—NEG-SPGR, NEG-ADLE, MLR-GRKN.

I. INTRODUCTION

Kernel regression constitutes one of the fundamental building blocks in supervised and semi-supervised learning strategies, be it for simple regression tasks or in more advanced settings such as support vector machines [1], Gaussian processes [2], deep neural networks [3], [4] and extreme learning machines [5], [6]. Kernel regression is a data-driven approach which works by projecting the input vector into a higher dimensional space using kernel functions and performs a linear regression in that space [7], [8]. The projection is achieved typically by using a nonlinear transformation. The target output $y$ (for both scalar
and vector cases) for a given test input $x$ is then expressible as a linear combination of the kernels between the test input and the training inputs $\{x_i\}$:

$$y = \sum_i \rho_i k(x, x_i),$$

where $k(\cdot, \cdot)$ denotes the kernel function. $\rho_i$s denote the coefficient vectors learnt from the training input-output pairs $\{(x_i, y_i)\}$. Given sufficiently large number of samples of reliable training data, kernel regression leads to accurate target predictions. However, in many cases, such as medical experiments or epidemic studies, it might not be feasible to collect large amounts of clean data. Insufficient or noisy training data often leads to deterioration in the prediction performance of kernel regression, just as in the case of any data-driven approach. In such cases, incorporating additional structure or regularization during the training phase can boost performance. In this article, we propose kernel regression for signals over graphs, where it is assumed that target signals are smooth over graphs.

Recently, graph signal processing has emerged as a promising area for incorporating graph-structural information in the analysis of vector signals [9], [10]. This takes particular relevance in contemporary applications that deal with data over networks or graphs. Several traditional signal processing methods have been extended to graphs [9]–[12] including many conventional spectral analysis concepts such as the windowed Fourier transforms, filterbanks, multiresolution analysis, and wavelets, [13]–[26]. Spectral clustering approaches based on graph signal filtering have been proposed recently [27], [28]. A number of graph-aware sampling schemes have also been developed [29]–[36]. Principal component analysis (PCA) techniques were proposed for graph signals by Shahid et al. for recovery of low-rank matrices [37], [38] and were shown to perform on par with nuclear-norm based approaches. Parametric dictionary learning approaches have demonstrated that graph specific information can help to outperform standard approaches like the $K$-SVD at small sample sizes [39]–[43]. Statistical analysis of graph signals particularly with respect to stationarity have been considered [44]–[48]. A number of approaches have also been proposed for learning the graph-Laplacian directly from data using different metrics [49]–[53].

Graph-based kernels have been investigated for labeling/coloring of graph nodes and graph clustering [54]–[61]. They deal with binary valued signals or data over the nodes. Kernel-based reconstruction strategies for graph signals were proposed recently by Romero et al. in the framework of reproducing kernel Hilbert spaces [62], [63]. Using the notion of joint space-time graphs, Romero et al. have also proposed kernel based reconstruction of graph signals and an extension of the Kalman filter for kernel-based learning [64], [65]. Recently, Shen et al. used kernels in structural equation models for identification of network topologies from graph signals [66].
A. Our contributions

We assume that the output of kernel regression and the target vector are smooth over a graph with a specified graph-Laplacian matrix. By noting that the input observations enter the regression model only in the form of inner-products, we develop kernel regression over graphs (KRG) using a dual representation or kernel trick. We extend kernel regression over graphs to the case when the graph-Laplacian is not specified apriori. In this case, we jointly learn the graph-Laplacian and the regression coefficients by formulating an alternating minimization algorithm. In other words, we perform both prediction for unseen inputs and learning of underlying graph topology. Experiments with both synthesized and real-world data confirm our hypothesis that a significant gain in prediction performance vis-a-vis conventional kernel regression can be obtained by regularizing the target vector to be a smooth signal over an underlying graph.

B. Signal processing over graphs

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

$$L = D - A,$$

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

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

We observe that $x^\top Lx$ is minimized when the signal $x$ takes the same value across all 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 connected nodes in a graph, and is said to be a high-frequency signal if it has dissimilar values across connected nodes, $x^\top Lx$ being the measure of similarity. This motivates the use of $x^\top Lx$ as a constraint in applications where either the signal $x$ or the graph-Laplacian $L$ is to be estimated [49], [50]. The eigenvectors of $L$ 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 low frequencies since the corresponding eigenvectors minimize the quadratic form of $L$ and vary smoothly over the the nodes. Similarly, the larger eigenvalues are referred to as the high frequencies. Then, a smooth graph signal is one which has GFT coefficients predominantly in the low graph frequencies.

II. KERNEL REGRESSION OVER GRAPHS

A. Linear regression over graphs

Let $\{x_n\}_{n=1}^N$ denote the set of $N$ input observations. Each $x_n$ is paired with target $t_n \in \mathbb{R}^M$. Our goal is to obtain the closest fit to the target $t_n$ with the linear regression model $y_n$ given by:

$$y_n = W^T \phi(x_n),$$

(1)

where $\phi(x_n) \in \mathbb{R}^K$ is a function of $x_n$ and $W \in \mathbb{R}^{K \times M}$ denotes the regression coefficient matrix. Our central assumption is that target $y_n$ is a smooth signal over an underlying graph $G$ with $M$ nodes. We learn the optimal parameter matrix $W$ by minimizing the following cost function:

$$C(W) = \sum_{n=1}^N \|t_n - y_n\|_2^2 + \alpha \text{tr}(W^T W) + \beta \sum_{n=1}^N y_n^T Ly_n,$$

(2)

where regularization coefficients $\alpha, \beta \geq 0$, $\text{tr}(\cdot)$ denotes the trace operator, and $\|x\|_2$ the $\ell_2$ norm of $x$. The choice of $\alpha, \beta$ depends on the problem, and in our analysis we compute these parameters through exhaustive grid search. The penalty $\text{tr}(W^T W) = \|W\|_F^2$ ensures that $W$ remains bounded. The penalty or regularization $y_n^T Ly_n$ enforces $y_n$ to be smooth over $G$. We define matrices $T$, $Y$ and $\Phi$ as follows:

$$T = [t_1 \ t_2 \ \cdots \ t_N]^T \in \mathbb{R}^{N \times M},$$

$$Y = [y_1 \ y_2 \ \cdots \ y_N]^T \in \mathbb{R}^{N \times M},$$

(3)

$$\Phi = [\phi(x_1) \ \phi(x_2) \ \cdots \ \phi(x_N)]^T \in \mathbb{R}^{N \times K}.$$

Using (1) and (3), the cost function (2) is expressible as (4) where we use properties of the matrix trace operation.

Since the cost function is quadratic in $W$, we get the optimal and unique solution by setting the gradient of $C$ with respect to $W$ equal to zero. Using the following matrix derivative relations

$$\frac{\partial}{\partial W} \text{tr}(M_1 W) = M_1^T,$$

$$\frac{\partial}{\partial W} \text{tr} \left( W^T M_1 WM_2 \right) = M_1^T WM_2^T + M_1 WM_2,$$

where $M_1$ and $M_2$ are matrices, and setting $\frac{\partial C}{\partial W} = 0$ we get that

$$-\Phi^T T + \Phi^T \Phi W + \alpha W + \beta \Phi^T \Phi WL = 0,$$

or,

$$\left( \Phi^T \Phi + \alpha I_K \right) W + \beta \Phi^T \Phi WL = \Phi^T T.$$

(5)
\[
C(W) = \sum_n \|t_n - W^\top \phi(x_n)\|^2_2 + \alpha \text{tr}(W^\top W) + \beta \sum_n \phi(x_n)^\top WLW^\top \phi(x_n)
\]
\[
= \sum_n \|t_n\|^2_2 - 2 \text{tr} \left( \sum_n \phi(x_n)^\top Wt_n \right) + \text{tr} \left( \sum_n \phi(x_n)^\top WW^\top \phi(x_n) \right) + \alpha \text{tr}(W^\top W)
+ \beta \text{tr} \left( \sum_n \phi(x_n)^\top WLW^\top \phi(x_n) \right)
\]
\[
= \sum_n \|t_n\|^2_2 - 2 \text{tr} \left( \sum_n t_n \phi(x_n)^\top W \right) + \text{tr} \left( WW^\top \sum_n \phi(x_n)\phi(x_n)^\top \right) + \alpha \text{tr}(W^\top W)
+ \beta \text{tr} \left( WLW^\top \sum_n \phi(x_n)\phi(x_n)^\top \right)
\]
\[
= \sum_n \|t_n\|^2_2 - 2 \text{tr} \left( T^\top \Phi W \right) + \text{tr}(W^\top \Phi^\top \Phi W) + \alpha \text{tr}(W^\top W) + \beta \text{tr} \left( W^\top \Phi^\top \Phi WL \right).
\]

On vectorizing both sides of (5), we have that
\[
\text{vec}(\Phi^\top T) = \left[ I_M \otimes (\Phi^\top \Phi + \alpha I_K) + (\beta L \otimes \Phi^\top \Phi) \right] \text{vec}(W),
\]
where \(\text{vec}(\cdot)\) denotes the standard vectorization operator and \(\otimes\) denotes the Kronecker product operation [67]. Then, the optimal \(W\) denoted by \(\hat{W}\) follows the relation:
\[
\text{vec}(\hat{W}) = \left[ I_M \otimes (\Phi^\top \Phi + \alpha I_K) + (\beta L \otimes \Phi^\top \Phi) \right]^{-1} \text{vec}(\Phi^\top T).
\]
The predicted target for a new input \(x\) is then given by
\[
\hat{t} = \hat{W}^\top \phi(x).
\]
From (6), it appears that the proposed target prediction approach requires the explicit knowledge about the function \(\phi(\cdot)\). We next show that using the ‘kernel trick’ this explicit requirement of \(\phi(\cdot)\) is circumvented and that the target prediction may be done using only the knowledge of inner products \(\phi(x_m)^T\phi(x_n)\), \(\forall m, n\), in other words, using the matrix \(\Phi\Phi^\top\). Towards this end, we next discuss a dual representation of the cost in (2). We hereafter refer to (6) as the output of linear regression over graphs (LRG).

B. Dual representation of cost using kernel trick

In dual representation [7] of the cost function (2), we use the substitution \(W = \Phi^\top \Psi\) and express the cost function in terms of the parameter \(\Psi \in \mathbb{R}^{N \times M}\). This substitution is motivated by observing that on rearranging the terms in (5), we get that
\[
W = \Phi^\top \Psi,
\]
where $\Psi = \left[ \frac{1}{\alpha} (T - \beta \Phi W L - \Phi) \right]$. On substituting $W = \Phi^T \Psi$ in (4), where $\Psi$ becomes the dual parameter matrix that we wish to learn, and omitting terms that do not depend on $W$ we get that

$$
C(\Psi) = -2 \text{tr} \left( T^T \Phi \Phi^T \Psi \right) + \text{tr} \left( \Psi^T \Phi \Phi^T \Phi \Phi^T \Psi \right) + \alpha \text{tr} (\Psi^T \Phi \Phi^T \Psi) + \beta \text{tr} (\Psi^T \Phi \Phi^T \Phi \Phi^T \Psi L) = -2 \text{tr} \left( T^T K \Psi \right) + \text{tr} \left( \Psi^T K K \Psi \right) + \alpha \text{tr} (\Psi^T K \Psi) + \beta \text{tr} (\Psi^T K K \Psi L),
$$

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

$$k_{m,n} = \phi(x_m)^T \phi(x_n).$$

Taking the derivative of $C(\Psi)$ with respect to $\Psi$ and setting it to zero, we get that

$$
(I_M \otimes (K + \alpha I_N)) \text{vec}(\Psi) + (\beta L \otimes K) \text{vec}(\Psi) = \text{vec}(T),
$$

or

$$
[(I_M \otimes (K + \alpha I_N)) + (\beta L \otimes K)] \text{vec}(\Psi) = \text{vec}(T),
$$

or

$$
\text{vec}(\Psi) = [(I_M \otimes (K + \alpha I_N)) + (\beta L \otimes K)]^{-1} \text{vec}(T).
$$

We define the matrices

$$
B = (I_M \otimes (K + \alpha I_N)),
$$

$$
C = (\beta L \otimes K).
$$

Then, we have that

$$
\text{vec}(\Psi) = (B + C)^{-1} \text{vec}(T). \tag{8}
$$

Once $\Psi$ is computed, the kernel regression output for a new test input $x$ is given by

$$
y = W^T \phi(x) = \Psi^T \Phi \phi(x) = \Psi^T k(x) = \left( \text{mat} \left( (B + C)^{-1} \text{vec}(T) \right) \right)^T k(x),
$$

where $k(x) = [k_1(x), k_2(x), \cdots, k_N(x)]^T$ and $k_n(x) = \phi(x_n)^T \phi(x)$. mat(·) denotes the reshaping operation of an argument vector into an appropriate matrix of size $N \times M$ by concatenating subsequent $N$ length sections as columns. We hereafter refer (9) as the output of *kernel regression over graphs* (KRG). We note that KRG is a generalization over the conventional kernel regression which does not
use any knowledge of the underlying graph structure. On setting $\beta = 0$, the KRG output (9) reduces to the conventional KR output as follows

$$
y = \left( \text{mat} \left( (B + C)^{-1} \text{vec}(T) \right) \right)^T k(x)
= \left( \text{mat} \left( (B)^{-1} \text{vec}(T) \right) \right)^T k(x)
= \left( \text{mat} \left( (I_M \otimes (K + \alpha I_N))^{-1} \text{vec}(T) \right) \right)^T k(x)
= \left( \text{mat} \left( (I_M^{-1} \otimes (K + \alpha I_N)^{-1}) \text{vec}(T) \right) \right)^T k(x)
= \left( (K + \alpha I_N)^{-1} T \right)^T k(x)
= T^T (K + \alpha I_N)^{-1} k(x), \quad (10)
$$

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

**C. Interpretation of KRG – a smoothing effect**

We next discuss how the output of KRG for training input 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 [8]. Using (10) and concatenating KR outputs for the $N$ training samples, we get that

$$
[y_1 y_2 \cdots y_N] = T^T (K + \alpha I_N)^{-1} [k(x_1) k(x_2) \cdots k(x_N)]
$$

or

$$
Y = K (K + \alpha I_N)^{-1} T,
$$

where we use $K = [k(x_1) k(x_2) \cdots k(x_N)]$. Assuming $K$ is diagonalizable, let $K = U J_K U^T$, where $J_K$ and $U$ denote the eigenvalue and eigenvector matrices of $K$, respectively. Then, the $n$th training input is expressible as

$$
y_n = \sum_{i=1}^{N} \frac{\theta_i}{\theta_i + \alpha} u_i^T t_n,
$$

where $\theta_i$ and $u_i$ denote the $i$th eigenvalue and eigenvector of $K$, respectively. Thus, we observe that the KR output performs a shrinkage of the components of $t_n$ along various eigenvectors. The components along the smaller eigenvalues $\theta_i < \alpha$ are effectively eliminated, and only those corresponding to larger eigenvalues $\theta_i > \alpha$ are effectively retained. Since the eigenvectors corresponding to larger eigenvalues of $K$ represent smooth variations across observations \{1, \cdots, N\}, we observe that KR performs a smoothing
of \( t_n \). We next show that such is also the case with KRG: KRG acts as a smoothing filter across both observations and graph nodes. Using (9) and concatenating KRG outputs, we have that

\[
[\mathbf{y}_1 \mathbf{y}_2 \ldots \mathbf{y}_N] = \Psi^T [k(x_1) k(x_2) \ldots k(x_N)]
\]

or,

\[
\mathbf{Y} = \mathbf{K}\Psi.
\]

On vectorizing both sides of (11), we get that

\[
\text{vec}(\mathbf{Y}) = (I_M \otimes \mathbf{K}) \text{vec}(\Psi) \overset{(a)}{=} (I_M \otimes \mathbf{K})(\mathbf{B} + \mathbf{C})^{-1} \text{vec}(\mathbf{T}),
\]

where we have used (8) in (a). Let \( \mathbf{L} \) be diagonalizable with the eigendecomposition:

\[
\mathbf{L} = \mathbf{VJ}_L \mathbf{V}^T,
\]

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

\[
\mathbf{V} = [\mathbf{v}_1 \mathbf{v}_2 \cdots \mathbf{v}_N] \text{ and } \mathbf{J}_L = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_N),
\]

\[
\mathbf{U} = [\mathbf{u}_1 \mathbf{u}_2 \cdots \mathbf{u}_M] \text{ and } \mathbf{J}_K = \text{diag}(\theta_1, \theta_2, \cdots, \theta_M).
\]

Now, using (7), we have

\[
\mathbf{B} + \mathbf{C} = (I_M \otimes (\mathbf{K} + \alpha I_N)) + (\beta \mathbf{L} \otimes \mathbf{K})
\]

\[
= [(\mathbf{V} \mathbf{I}_M \mathbf{V}^T) \otimes (\mathbf{U}(\mathbf{J}_K + \alpha I_N) \mathbf{U}^T)]
\]

\[
+ [\beta (\mathbf{V} \mathbf{J}_L \mathbf{V}^T) \otimes (\mathbf{U}\mathbf{J}_K \mathbf{U}^T)] \overset{(a)}{=} [(\mathbf{V} \otimes \mathbf{U})(I_M \otimes (\mathbf{J}_K + \alpha I_N))(\mathbf{V}^T \otimes \mathbf{U}^T)]
\]

\[
+ [\beta (\mathbf{V} \otimes \mathbf{U})(\mathbf{J}_L \otimes \mathbf{J}_K)(\mathbf{V}^T \otimes \mathbf{U}^T)] = \mathbf{Z}\mathbf{J}\mathbf{Z}^T,
\]

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

\[
\mathbf{J} = (I_M \otimes (\mathbf{J}_K + \alpha I_N)) + \beta (\mathbf{J}_L \otimes \mathbf{J}_K).
\]

In (13)(a), we have used the distributivity 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 \( \mathbf{M}_i \) are 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 each \( \eta_i \) is a function of \( \{\lambda_j\}, \{\theta_j\}, \alpha, \beta \). On dropping the subscripts for simplicity, we observe that any eigenvalue \( \eta_i \) has the form

\[
\eta = (\theta + \alpha) + \beta (\lambda \theta).
\]
where $\theta$ and $\lambda$ are the appropriate eigenvalues of $K$ and $L$, respectively. Similarly, we have that

\[
(I_M \otimes K) = (VI_M V^T) \otimes (UJ_K U^T)
\]

\[
= (V \otimes U) (I_M \otimes J_K) (V^T \otimes U^T)
\]

\[
= Z (I_M \otimes J_K) Z^T,
\]

and note that $(I_M \otimes J_K)$ is also a diagonal matrix of size $MN$. Then, on substituting (14) and (15) in (12), we get that

\[
\text{vec}(Y) = (I_M \otimes K) (B + C)^{-1} \text{vec}(T)
\]

\[
= (Z (I_M \otimes J_K) Z^T) (Z J^{-1} Z^T) \text{vec}(T)
\]

\[
= (Z (I_M \otimes J_K) J^{-1} Z^T) \text{vec}(T).
\]

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

\[
\zeta = \frac{\theta}{\eta_i} = \frac{\theta}{(\theta + \alpha) + \beta(\lambda \theta)}.
\]

From (16), we have that

\[
\text{vec}(Y) = \sum_{i=1}^{MN} \zeta_i z_i z_i^T \text{vec}(T),
\]

where $z_i$ are column vectors of $Z$. In the case when $\zeta_i \ll 1$, the component in vec$(T)$ along $z_i$ is effectively eliminated. For most covariance or kernel functions $k(\cdot, \cdot)$ used in practice, the eigenvectors corresponding to the largest eigenvalues of $K$ correspond to the low-frequency components. Similarly, the eigenvectors corresponding to the smaller eigenvalues of $L$ are smooth over the graph [9]. We observe that the condition $\zeta \ll 1$ is achieved when corresponding $\theta$ is small and/or $\lambda$ is large. This condition in turn corresponds to effectively retaining only components in vec$(T)$ which vary smoothly across the samples $\{1, \cdots, N\}$ as well as over the $M$ nodes of the graph.

III. 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 $L$. Such an assumption may not hold for many practical applications since there is not necessarily one best graph for given networked data. This motivates us to develop a joint learning approach where we learn $L$ and KRG parameter matrix $W$ (or its dual representation parameter $\Psi$). We propose the minimization of the following cost function to achieve our goal:

\[
C(W, L) = \sum_n \|t_n - y_n\|^2_2 + \alpha \text{tr}(W^T W)
\]

\[
+ \beta \sum_n y_n^T L y_n + \nu \text{tr}(L^T L),
\]
where \( \nu \geq 0 \). Since our goal is to recover an undirected graph, we impose appropriate constraints [68]. Firstly, a connected graph has a graph-Laplacian matrix \( L \) which is symmetric and positive semi-definite. Secondly, the vector of all ones \( 1 \) forms the eigenvector of the graph-Laplacian corresponding to the zero eigenvalue. Then, the solution to joint estimation of \( W \) and \( L \) is obtained by solving the following:

\[
\min_{W,L} C(W,L) \quad \text{such that} \quad L \succeq 0, \quad L = L^\top, \quad L1 = 0.
\]

The optimization problem (17) is jointly non-convex over \( W \) and \( L \), but conditionally convex on both. Hence, we adopt an alternating minimization approach and solve (17) in two steps alternatingly as follows:

- For a given \( L \), solve \( \min_W C(W) \) using KRG approach of Section II.
- For a given \( W \), solve \( \min_L C(L) \) such that \( L \succeq 0, \quad L = L^\top, \quad L1 = 0 \). Here \( C(L) = \beta \sum_n y_n^\top Ly_n + \nu \text{tr}(L^\top L) \). This is a quadratic semidefinite program and may be solved using standard convex solvers.

We start the alternating optimization using the initialization \( L = 0 \), which corresponds to KR. In order to keep the successive \( L \) estimates comparable, we scale \( L \) such that the largest eigenvalue modulus is unity at each iteration.

IV. EXPERIMENTS

We consider application of KRG to various synthesized and real-world signal examples. We use normalized-mean-square-error (NMSE) defined as follows to evaluate the performance:

\[
\text{NMSE} = 10 \log_{10} \left( \frac{\|Y - T_0\|_F^2}{\|T_0\|_F^2} \right),
\]

where \( Y \) denotes the regression output matrix and \( T_0 \) the true value of target matrix, that means \( T_0 \) does not contain any noise. In the case of real-world examples, we compare the performance of the following four approaches:

1) Linear regression (LR): \( k_{m,n} = \phi(x_m)^\top \phi(x_n) \), where \( \phi(x) = x \) and \( \beta = 0 \),

2) Linear regression over graphs (LRG): \( k_{m,n} = \phi(x_m)^\top \phi(x_n) \) and \( \beta \neq 0 \), where \( \phi(x) = x \),

3) Kernel regression (KR): Using radial basis function (RBF) kernel \( k_{m,n} = \exp\left(-\frac{\|x_m - x_n\|^2}{\sigma^2}\right) \)
and \( \beta = 0 \), and

4) Kernel regression over graphs (KRG): Using RBF kernel \( k_{m,n} = \exp\left(-\frac{\|x_m - x_n\|^2}{\sigma^2}\right) \) and \( \beta \neq 0 \).

The regularization parameters \( \alpha, \beta, \) and \( \nu \) for different training data sizes are found by exhaustive grid-search. The RBF kernel parameter \( \sigma \) is set as \( \sigma^2 = 1/N \sum_{m',n'} \|x_{m'} - x_n\|^2 \) in our experiments, where the summation is made over the training inputs. The choice of \( \sigma \) is made arbitrarily and is not necessarily optimized to the training datasets.
Fig. 1. Results for synthesized data using Erdős-Rényi graphs. Normalized MSE at different SNR levels with training data size of $N = 50$.

A. Regression for synthesized data

Fig. 2. Results for synthesized data using Normalized MSE at 5 dB SNR level: (a) for training data, and (b) for testing data for Erdős-Rényi graphs. Normalized MSE at 5 dB SNR level: (c) for training data, and (d) for testing data for Barabási-Albert graphs.

We perform regression for synthesized data where the target vectors are known to lie on a specified graph, and the various target vectors are correlated among each other. A part of the data is used for training in presence of additive noise, and our task is to predict for the remaining part of the data, given the information of the correlations in form of kernels. In order to generate synthesized data, we use random small-world graphs from Erdős-Rényi and Barabási-Albert models [69] with the number of nodes is fixed 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 $\mathcal{N}(0, C_S)$ where $C_S$ is $S$-dimensional covariance matrix. We use a highly correlated covariance matrix $C_S$ 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 between each other. We denote a row vector by $\mathbf{r}$. We then select row vectors $\{\mathbf{r}\}$ one-by-one and project them on the specified graph to generate target vectors $\{\mathbf{z}\}$ that lie on the graph while maintaining the correlation between observations, by solving the following optimization problem:

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

Out of the $S$ target vectors, we use one half for training and the remaining as test data with which we compare our predictions. Since we use a kernel regression approach, we perform the regression by assuming that we know the true kernel to be $\mathbf{K} = \mathbf{C}_S$, that is, by setting the kernel $k_{m,n} = \mathbf{C}_S(m,n)$. 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 normalized MSE of KRG and KR become closer. This is expected since KR is able to make better predictions due to availability of cleaner training observations. A similar trend is observed for also the case of Barabási-Albert graphs, which is not reported. In Figure 2 we show the normalized MSE obtained with KRG and KR on both graph models as a function of training data size at an SNR-level of 5dB. We observe that KRG consistently outperforms KR and that the gap between NMSE of KRG and KR reduces as training data size increases as expected. The trend is similar for large dimensional graphs as well and is not reported here to avoid repetition.

B. Temperature prediction for cities in Sweden

We collect temperature measurements from 45 most populated cities in Sweden for a period of two months from November to December 2016. The data is available publicly from the Swedish Meteorological and Hydrological Institute [70]. 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 from the previous day. We have 62 input-target data pairs in total, and a part of the data is used for training and the rest for testing. We report the results by averaging over 100 experiments where each experiment uses a random partition of the data into training and testing datasets. Let $d_{ij}$ denote the geodesic distance between cities $i$ and $j$, $\forall i, j \in \{1, \cdots, 45\}$. We construct the adjacency matrix by setting

$$\mathbf{A}(i, j) = \exp \left( -\frac{d_{ij}^2}{\sum_{i,j} d_{ij}^2} \right).$$
In order to remove self loops, the diagonal of $A$ is set to zero. We simulate noisy training data by adding zero-mean white Laplacian noise to the targets. The Laplacian distribution is heavy-tailed and simulates impulsive noise at various weather stations. In Figure 3, we show the NMSE testing data at SNR-levels of 5dB and 0dB. We observe that KRG outperforms other regression methods by a significant margin, particularly at low sample sizes. We also consider the joint learning of the graph from training data using the approach described in Section III. We perform the training at an SNR level of 10 dB. We find experimentally that the flip-flop iterations converge typically after five to ten iterations. The NMSE for training and testing data obtained using oracle or true $L$ and the iteratively learnt $L$ for both LRG and KRG are shown in Figure 4. We observe that the NMSE obtained with the learnt graph is higher than with the true graph, which is expected given the lack of knowledge of the true graph. We also observe that the estimated Laplacian matrices from both LRG and KRG are close to each other and are in reasonable agreement with that of the geodesic graph. This validates our intuition that the graph signal holds sufficient information to both infer the underlying graph structure and perform predictions.

C. Prediction on EEG signals

We repeat the prediction experiment on 64-channel EEG signal data taken from the Physionet database [71]. The signals are obtained by placing electrodes at various locations of a subject as shown in the
schematic in Figure 5(a). At each time instant, the target graph signal is the $M = 64$-dimensional vector with components given by signal value at the electrodes. The corresponding input signal $x_n$ is taken as the EEG signal at the preceding time-stamp. Our goal is then to predict the signal at the $M$ electrodes for the next time-instant given the current values. Since the graph is not given apriori, we construct the graph for the $M$ electrodes from a part of noiseless data using Gaussian kernels as follows:

$$A(i,j) = \exp \left( -\frac{\|r_k - r_l\|^2}{\sum_{k,l} \|r_k - r_l\|^2} \right) \quad \forall k, l \in \{1, \cdots, M\},$$

where $r_l$ denotes the vector with first 100 time-samples for the $l$th electrode. Thus, we use the first 100 time-samples to generate the diffusion or Gaussian kernel graph for the electrodes. We then consider training with noisy targets corrupted with additive white Gaussian noise at SNR levels of 0dB and 5dB. In Figure 5 we show the NMSE for LR, LRG, KR, and KRG for different training data sizes.
observe once again that LRG and KRG consistently outperform LR and KR, respectively, particularly at low training data sizes.

D. Prediction on flow-cytometry data

We next perform prediction on flow-cytometry data used by Sachs et al. [72] consisting of response or signalling level of 11 proteins in different experiment cells. We use the symmetricized version of the directed unweighted acyclic graph derived by Sachs et al. [72] in our analysis. Among the 11 proteins, we select the four proteins with largest number of graph edges (proteins 3, 8, 9, and 10, cf. Figure 6) and use the corresponding responses as the input $x_n \in \mathbb{R}^4$, where $n$ denotes the $n$th experiment cell or instance. The corresponding graph adjacency matrix has entries of one or zero depending on the presence or absence of edges. The names of the proteins corresponding to the node numbers are listed in Table I. The response of the remaining 7 proteins then forms the target vector $y_n \in \mathbb{R}^7$ lying on the $M = 7$ node subgraph of the full graph shown in Figures 6(a)-(b). The corresponding Laplacian matrix is shown
in Figure 7(a). We note that the subgraph has two connected components, one comprising six nodes and the other isolated node 11.

In Figure 6 we compare the NMSE of LR, LRG, KR, and KRG as a function of training data size. We observe again that LRG and KRG consistently outperform LR and KR, respectively, at small training data sizes and low SNR levels. In Figure 7 we show the results obtained by applying the graph-learning approach. We observe that the estimates of the graph-Laplacian matrix from both LRG and KRG nearly coincide and that the NMSE is higher for learnt graph cases than with the underlying graph as expected. However, unlike in the case of Swedish temperature, the learnt graph-Laplacians differ to a greater extent from the graph of Sachs et al.

V. CONCLUSIONS

Our hypothesis was that incorporating the graph smoothness constraint would help kernel regression to perform better, particularly when we lack sufficient and reliable training data. Our experiments illustrate that this is indeed the case in practice. Through experiments we also conclude that graph signals carry sufficient information about the underlying graph structure which may be extracted in the regression setting even with moderately small number of samples in comparison with the graph dimension. Thus, our approach helps both predict and infer the underlying topology of the network or graph.
Fig. 7. Results with graph-learning for flow-cytometry data at SNR of 10dB. (a) NMSE for test data, (b) True Laplacian, (c) Estimated Laplacian from LRG, (d) Estimated Laplacian from KRG, (e) vectorized entries of the different Laplacians. The Laplacians have been scaled to have maximum eigenvalue equal to unity.

REFERENCES

[1] C. Cortes and V. Vapnik, “Support-vector networks,” Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.
[2] M. Seeger, “Gaussian processes for machine learning,” Int. J. Neural Syst., vol. 14, no. 02, pp. 69–106, 2004.
[3] Y. B. L. Yann and G. Hinton., “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.
[4] Y. Cho and L. K. Saul, “Kernel methods for deep learning,” in Adv. Neural Inform. Process. Syst. Curran Associates, Inc., 2009, pp. 342–350.
[5] G.-B. Huang, “An insight into extreme learning machines: Random neurons, random features and kernels,” Cognitive Computation, vol. 6, pp. 376–390, 2014.
[6] A. Iosifidis, A. Tefas, and I. Pitas, “On the kernel extreme learning machine classifier,” Patt. Recognition Lett., vol. 54, pp. 11 – 17, 2015.
[7] C. M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics). Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
[8] C. E. Rasmussen and C. K. I. Williams, Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2005.
[9] 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.
[10] A. Sandryhaila and J. M. F. Moura, “Discrete signal processing on graphs,” IEEE Trans. Signal Process., vol. 61, no. 7, pp. 1644–1656, 2013.
[11] ——, “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.
[12] ——, “Discrete signal processing on graphs: Frequency analysis,” IEEE Trans. Signal Process., vol. 62, no. 12, pp. 3042–3054, 2014.
[13] D. I. Shuman, B. Ricaud, and P. Vandergheynst, “A windowed graph Fourier transform,” IEEE Statist. Signal Process. Workshop (SSP), pp. 133–136, Aug 2012.
[14] 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.
[15] ——, “Perfect reconstruction two-channel wavelet filter banks for graph structured data,” IEEE Trans. Signal Process., vol. 60, no. 6, pp. 2786–2799, 2012.
[16] ——, “Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs,” IEEE Trans. Signal Process., vol. 61, no. 19, pp. 4673–4685, 2013.
[17] R. R. Coifman and M. Maggioni, “Diffusion wavelets,” Appl. Computat. Harmonic Anal., vol. 21, no. 1, pp. 53–94, 2006.
[18] 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.
[19] 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.
[20] R. Wagner, V. Delouille, and R. Baraniuk, “Distributed wavelet de-noising for sensor networks,” Proc. 45th IEEE Conf. Decision Control, pp. 373–379, 2006.
[21] D. I. Shuman, B. Ricaud, and P. Vandergheynst, “Vertex-frequency analysis on graphs,” Applied and Computational Harmonic Analysis, vol. 40, no. 2, pp. 260 – 291, 2016.
[22] 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. 2199–2134, April 2016.
[23] 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.
[24] ——, “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.
[25] 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](http://w.american.edu/cas/sampta/papers/a13-venkitaraman.pdf)
[26] A. Venkitaraman, S. Chatterjee, and P. Händel, “Hilbert transform, analytic signal, and modulation analysis for graph signal processing,” CoRR, vol. abs/1611.05269, 2016. [Online]. Available: [http://arxiv.org/abs/1611.05269](http://arxiv.org/abs/1611.05269)
[27] 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.
[28] N. Tremblay and P. Borgnat, “Joint filtering of graph and graph-signals,” Asilomar Conference on Signals, Systems and Computers, pp. 1824–1828, Nov 2015.
[29] S. Chen, R. Varma, A. Sandryhaila, and J. Kovacevic, “Discrete signal processing on graphs: Sampling theory,” IEEE Transactions on Signal Processing, vol. 63, no. 24, pp. 6510–6523, Dec 2015.
[30] F. Gama, A. G. Marques, G. Mateos, and A. Ribeiro, “Rethinking sketching as sampling: Linear transforms of graph signals,” Proc. Asilomar Conf. Signals, Systems Comput., pp. 522–526, Nov 2016.
[31] 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.
[32] S. Chen, R. Varma, A. Singh, and J. Kovačević, “Signal recovery on graphs: Fundamental limits of sampling strategies,” IEEE Trans. Signal Inform. Process. over Networks, vol. 2, no. 4, pp. 539–554, Dec 2016.
[33] 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.

[34] 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.

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

[36] 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.

[37] 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.

[38] N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst, “Fast robust pca on graphs,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 740–756, June 2016.

[39] 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.

[40] D. Thanou and P. Frossard, “Multi-graph learning of spectral graph dictionaries,” in *IEEE Intl. Conf. Acoust. Speech Signal Processing*, April 2015, pp. 3397–3401.

[41] ——, “Multi-graph learning of spectral graph dictionaries,” *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, pp. 3397–3401, April 2015.

[42] 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.

[43] Y. Yankelevsky and M. Elad, “Dual graph regularized dictionary learning,” *IEEE Trans. Signal Info. Process. Networks*, vol. 2, no. 4, pp. 611–624, Dec 2016.

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

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

[46] N. Perraudin and P. Vandergheynst, “Stationary signal processing on graphs,” *CoRR*, vol. abs/1601.02522, 2016. [Online]. Available: http://arxiv.org/abs/1601.02522

[47] 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.

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

[49] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, “Learning graphs from signal observations under smoothness prior,” *CoRR*, vol. abs/1406.7842, 2014. [Online]. Available: http://arxiv.org/abs/1406.7842

[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,” in *2013 IEEE 10th International Symposium on Biomedical Imaging*, April 2013, pp. 616–619.

[51] S. I. Daitch, J. A. Kelner, and D. A. Spielman, “Fitting a graph to vector data,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML ’09. New York, NY, USA: ACM, 2009, pp. 201–208.

[52] N. Leonardi and D. V. D. Ville, “Wavelet frames on graphs defined by fMRI functional connectivity,” in *IEEE International Symposium on Biomedical Imaging*, March 2011, pp. 2136–2139.
[53] N. Leonardi, J. Richiardi, M. Gschwind, S. Simioni, J.-M. Annoni, M. Schluep, P. Vuilleumier, and D. V. D. Ville, “Principal components of functional connectivity: A new approach to study dynamic brain connectivity during rest,” NeuroImage, vol. 83, pp. 937–950, 2013.

[54] R. I. Kondor and J. Lafferty, “Diffusion kernels on graphs and other discrete structures,” Proc. of the ICML, pp. 315–322, 2002.

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

[56] M. Belkin, I. Matveeva, and P. Niyogi, “Tikhonov regularization and semi-supervised learning on large graphs,” Proc. IEEE Intl. Conf. Acoust.Speech Signal Process., May 2004.

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

[58] A. Argyriou, M. Herbster, and M. Pontil, “Combining graph laplacians for semi-supervised learning,” NIPS’05, pp. 67–74, 2005.

[59] 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.

[60] 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 Networks Learning Systems, vol. 26, no. 3, pp. 444–457, March 2015.

[61] K. I. Kim, J. Tompkin, H. Pfister, and C. Theobalt, “Context-guided diffusion for label propagation on graphs,” in IEEE Intl. Conf. Computer Vision (ICCV), Dec 2015, pp. 2776–2784.

[62] 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.

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

[64] 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.

[65] D. Romero, V. N. Ioannidis, and G. B. Giannakis, “Kernel-based reconstruction of space-time functions on dynamic graphs,” CoRR, vol. abs/1612.03615, 2016. [Online]. Available: http://arxiv.org/abs/1612.03615

[66] 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.

[67] C. F. V. Loan, “The ubiquitous Kronecker product,” J. Computat. Appl. Mathematics, vol. 123, no. 1–2, pp. 85–100, 2000.

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

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

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

[71] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, “Physiobank, physiotoolkit, and physionet,” vol. 101, no. 23, pp. e215–e220, 2000.

[72] K. Sachs, O. Perez, D. Pe’er, D. A. Lauffenburger, and G. P. Nolan, “Causal protein-signaling networks derived from multiparameter single-cell data,” Science, vol. 308, no. 5721, pp. 523–529, 2005.