Finding GEMS: Multi-Scale Dictionaries For High-Dimensional Graph Signals

Yael Yankelevsky, *Student Member, IEEE*, and Michael Elad, *Fellow, IEEE*

***Abstract*—Modern data introduces new challenges to classic signal processing approaches, leading to a growing interest in the field of graph signal processing. A powerful and well-established model for real world signals in various domains is sparse representation over a dictionary, combined with the ability to train the dictionary from signal examples. This model has been successfully applied to graph signals as well by integrating the underlying graph topology into the learned dictionary. Nonetheless, dictionary learning methods for graph signals are typically restricted to small dimensions due to the computational constraints that the dictionary learning problem entails, and due to the direct use of the graph Laplacian matrix. In this paper, we propose a graph-enhanced multi-scale dictionary learning algorithm that applies to a broader class of graph signals, and is capable of handling much higher dimensional data. We incorporate the underlying graph topology both implicitly, by forcing the learned dictionary atoms to be sparse combinations of graph-wavelet functions, and explicitly, by adding direct graph constraints to promote smoothness in both the feature and manifold domains. The resulting atoms are thus adapted to the data of interest, while adhering to the underlying graph structure and possessing a desired multi-scale property. Experimental results on several datasets, representing both synthetic and real network data of different nature, demonstrate the effectiveness of the proposed algorithm for graph signal processing even in high dimensions.**

***Index Terms*—Sparse representation, dictionary learning, graph signal processing, graph Laplacian, double-sparsity, manifold structure, graph wavelets.**

I. Introduction

IN RECENT years, the field of graph signal processing has been gaining momentum. By merging concepts of spectral graph theory and harmonic analysis, it aims at extending classical signal processing approaches to signals having a complex and irregular underlying structure. Such signals emerge in numerous modern applications of diverse sources, such as transportation, energy, biological-, social-, and sensor-networks [1], [2]. In all these cases and many others, the underlying structure of the data could be represented using a weighted graph, such that its vertices (or nodes) represent the discrete data domain, and the edge weights reflect the pairwise similarities between these vertices. The data itself resides on the graph, that is, every graph signal is a function assigning a real value to each vertex.

As in classical signal processing, a model for graph signals is key for handling various processing tasks, such as solving inverse problems, sampling, compression, and more. A popular and highly effective such model for real world signals in different domains is sparse representation [3]. This model assumes the availability of a dictionary, which could be either analytic (constructed) or trained from signal examples. Indeed, the work reported in [4]–[6] has deployed this breed of models to graph signals, and this paper aims at extending these contributions by proposing better performing algorithms, while also allowing the processing of high-dimensional graphs, which earlier methods fail to handle.

A fundamental ingredient in the use of the sparse representations model is dictionary learning. Classic dictionary learning methods such as the method of optimal directions (MOD) [7] and K-SVD [8] are generally structure agnostic. In order to better support graph signals, the work in [4]–[6] extended these methods by integrating the underlying graph topology into the learned dictionary. More specifically, the work reported in [4], [5] imposed a parametric structure on the trained dictionary, relying on the graph topology. Whereas [4] learns a collection of shift-invariant graph filters, [5] restricts the dictionary to a concatenation of polynomials of the graph Laplacian matrix.

In [6], we have developed a framework for dictionary learning with graph regularity constraints in both the feature and manifold domains, which we referred to as Dual Graph Regularized Dictionary Learning (DGRDL). Furthermore, our proposed scheme suggests the additional ability of inferring the graph topology within the dictionary learning process. This is important in cases where this structure is not given, yet known to exist.

The DGRDL algorithm and its extensions to a supervised setting [9], [10] already exhibit very good performance in various applications. Nevertheless, a significant limitation of these methods is their poor scalability to high dimensional data, which is limited by the complexity of the training problem as well as by the use of the large graph Laplacian matrices. This problem is in fact not specific to DGRDL but common to all current dictionary learning methods for graph signals, that were shown to accommodate graphs with no more than a few hundreds of nodes. For instance, the scalability of the polynomial method [5] is also limited by the complexity of the involved optimization

Manuscript received June 13, 2018; revised November 18, 2018 and February 1, 2019; accepted February 1, 2019. Date of publication February 15, 2019; date of current version March 4, 2019. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mark A. Davenport. This work was supported in part by the European Research Council under European Union's Seventh Framework Program, ERC under Grant Agreement 320649, and in part by the Israel Science Foundation under Grant 1770/14. *(Corresponding author: Yael Yankelevsky.)*

The authors are with the Department of Computer Science, Technion - Israel Institute of Technology, Haifa 3200003, Israel (e-mail: yaelyan@cs.technion.ac.il; elad@cs.technion.ac.il).

Digital Object Identifier 10.1109/TSP.2019.2899822

problem as well as by the need to compute different powers of the large graph Laplacian matrix.

This limitation might be addressed by constructing analytic multi-scale transforms. Indeed, incorporating multi-scale properties in the dictionary is vital for representing large signals, and could reveal structural information about the signals at different resolution levels. Following this reasoning, classical wavelets have been generalized from the Euclidean domain to the graph setting in a number of different ways. Examples include the diffusion wavelets [11], spectral graph wavelets [12], lifting based wavelets [13], multi-scale wavelets on balanced trees [14], permutation based wavelets [15], wavelets on graphs via deep learning [16] and a multi-scale pyramid transform for graph signals [17].

Such transform-based dictionaries offer an efficient implementation that makes them less costly to apply than structure-agnostic trained dictionaries. However, while accounting for the underlying topology and possessing the desired multi-scale property, these transforms are not adapted to the given data, limiting their performance in real life applications.

In order to combine both the adaptability and the multi-scale property, while enabling treatment of higher dimensional signals, we propose infusing structure into the learned dictionary by harnessing the double sparsity framework [18] with a graph-Haar wavelet base dictionary. As such, the proposed approach benefits from the multi-scale structure and the topology-awareness that this base dictionary brings, along with the ability to adapt to the signals. It can thus be viewed as a fusion of the analytic and the trainable paradigms.

Beyond its implicit presence through the constructed wavelet basis, the underlying data geometry is also added explicitly via direct graph regularization constraints, promoting smoothness in both the feature and manifold domains. Finally, we devise a complete scheme for joint learning of the graph, and hence the graph-wavelet basis, along with the dictionary. By doing so, we essentially replace the pre-constructed wavelet basis with an adaptive one, iteratively tuned along the dictionary learning process. The resulting algorithm, termed Graph Enhanced Multi-Scale dictionary learning (GEMS), leads to atoms that adhere to the underlying graph structure and possess a desired multi-scale property, yet they are adapted to capture the prominent features of the data of interest. A special configuration of this method, termed GEMS-HD, further reduces the computational complexity to support a high-dimensional setting, thus enabling treatment of graphs that are an order of magnitude larger compared with current graph dictionary learning techniques.

An early version of this work appeared in [19], introducing the core idea of graph sparse-dictionary learning accompanied with preliminary experiments. This work extends the above in several important ways: (i) The introduction of the explicit regularity along with the modifications to the overall algorithm; (ii) The derivation of a joint-learning of the topology; and (iii) The addition of extensive new experiments demonstrating the strengths of the new algorithms. As these experiments show, the proposed dictionary structure brings along piece-wise smoothness and localization properties, making it more suitable for modeling graph data of different nature and different dimensions.

To summarize, in this paper we propose GEMS - a novel dictionary learning algorithm for graph signals, improving over the algorithms presented in [6], [19], and enabling treatment of much larger graphs. For medium-sized graphs consisting of hundreds of nodes, GEMS yields superior results to other graph dictionary learning methods. For larger graphs consisting of thousands or even tens-of-thousands of nodes, all existing dictionary learning methods fail. For such dimensions, we offer a simplified and efficient configuration termed GEMS-HD, significantly increasing the range of treatable data dimensions.

The outline of the paper is as follows: In Section II, we commence by delineating the background for graph signal processing. Consequently, we revisit our DGRDL algorithm for graph signals, and present the incorporation of a sparse dictionary model, including a detailed description of the base dictionary construction procedure. In Section III we consider the task of training the dictionary from examples and derive the foundation of the GEMS algorithm for doing so. Section IV introduces the adaptation of the graph Laplacian, as well as the wavelet base dictionary, along the learning process. All these components together assemble the complete GEMS framework. We then evaluate the performance of the proposed algorithm in Section V, and conclude in Section VI.

II. Sparse Dictionary Learning For Graph Signals

A. Preliminaries

A weighted and undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$ consists of a finite set $\mathcal{V}$ of N vertices (or nodes), a finite set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ of weighted edges, and a weighted adjacency matrix $W \in \mathbb{R}^{N \times N}$. The entry w_{ij} represents the weight of the edge $(i, j) \in \mathcal{E}$, reflecting the similarity between the nodes i and j. In general, w_{ij} is non-negative, and $w_{ij} = 0$ if the nodes i and j are not directly connected in the graph. Additionally, for undirected weighted graphs with no self-loops, W is symmetric and $w_{ii} = 0 \ \forall i$.

The graph degree matrix Δ is the diagonal matrix whose i-th diagonal entry computes the sum of weights of all edges incident to the i-th node, i.e. having $\Delta_{ii} = \sum_j w_{ij}$. The combinatorial graph Laplacian matrix, representing the second-order differential operator on the graph, is then given by $L = \Delta - W$.

Given a topological graph, we refer to graph signals as functions $f : \mathcal{V} \to \mathbb{R}$ assigning a real value to each vertex. Any graph signal is therefore a vector in $\mathbb{R}^N$, whose i-th entry is the measurement corresponding to the i-th graph node.

The regularity of a graph signal f can be measured using the Laplacian L [20] in terms of the graph Dirichlet energy,

$$f^T L f = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} (f_i - f_j)^2. \tag{1}$$

When this measure of variation is small, indicating that strongly connected nodes have similar signal values, the signal is considered smooth with respect to the given graph.

B. Introducing the Sparse Dictionary Model

The standard dictionary learning problem is formulated as

$$\begin{aligned} \arg\min_{D,X} \quad & \|Y - DX\|_F^2 \\ \text{s.t.} \quad & \|x_i\|_0 \le T \quad \forall i, \quad \|d_j\|_2 = 1 \quad \forall j, \end{aligned} \tag{2}$$

where $Y \in \mathbb{R}^{N\times M}$ is the data matrix containing the training examples in its columns, $X \in \mathbb{R}^{K\times M}$ is the corresponding sparse coefficients matrix, $D \in \mathbb{R}^{N\times K}$ is an overcomplete dictionary with normalized columns (atoms), and T is a sparsity threshold. The i-th column of the matrix X is denoted x_i.

In order to account for the data geometry, the dual graph regularized dictionary learning (DGRDL) algorithm [6] introduces graph regularity constraints in both the feature and manifold domains. The DGRDL problem is thus given by

$$\begin{aligned} \arg\min_{D,X} \quad & \|Y - DX\|_F^2 + \alpha Tr(D^T L D) + \beta Tr(X L_c X^T) \\ \text{s.t.} \quad & \|x_i\|_0 \le T \quad \forall i, \end{aligned} \tag{3}$$

where $L \in \mathbb{R}^{N\times N}$ denotes the topological graph Laplacian, accounting for the underlying inner structure of the data, and $L_c \in \mathbb{R}^{M\times M}$ is the manifold Laplacian, representing correlation between different signals within the training set. Imposing smoothness with respect to both graphs encourages the atoms to preserve the underlying geometry of the signals and the representations to preserve the data manifold structure.

As mentioned in the introductory section, a significant limitation of DGRDL is poor scalability to high dimensional data, which is limited by the complexity of training, storing, and deploying the explicit dictionary D. To better accommodate higher dimensional graphs, we leverage the double sparsity approach [18] and propose employing a sparsity model of the dictionary atoms over a base dictionary, i.e. defining the dictionary as a product $D = \Phi A$, where Φ is some known (perhaps analytic or structured) base dictionary, and A is a learned sparse matrix, having P non-zeros per column.

Integrating this structure into the DGRDL scheme, we obtain the following graph-enhanced multi-scale (GEMS) dictionary learning problem:

$$\begin{aligned} \arg\min_{A,X} \quad & \|Y - \Phi A X\|_F^2 + \alpha Tr(A^T \Phi^T L \Phi A) \\ & + \beta Tr(X L_c X^T) \\ \text{s.t.} \quad & \|x_i\|_0 \le T \quad \forall i, \\ & \|a_j\|_0 \le P \quad \forall j, \quad \|\Phi a_j\|_2 = 1 \quad \forall j. \end{aligned} \tag{4}$$

Despite the formulation resemblance, the combination of the atom sparsity constraint and the graph regularization introduces new challenges, leading to a new algorithm derivation that is significantly different from both [6] and [18]. The solution can be obtained by alternating optimization over A and X, as will be detailed in the next section.

While the double sparsity framework allows flexibility in the dimensions of Φ and A and it is not generally necessary for Φ to be square, we here choose to use an orthogonal transform. Therefore, in our setting, $\Phi \in \mathbb{R}^{N\times N}$ is the base dictionary and $A \in \mathbb{R}^{N\times K}$ is a redundant ($K > N$) column-wise sparse matrix.

We emphasize that while A is a redundant matrix, identical in size to the general unstructured dictionary D in (3), the dictionary update is now constrained by the number of non-zeros in the columns of A. Consequently, the sparse dictionary requires training of merely $P \cdot K$ parameters rather than $N \cdot K$, where $P \ll N$. Hence learning in this case is feasible even given limited training data or high signal dimensions.

Overall, the sparse dictionary has a compact representation and provides efficient forward and adjoint operators, yet it can be effectively trained from given data even when the dimensions are very large. Therefore, it naturally bridges the gap between analytic dictionaries, which have efficient implementations yet lack adaptability, and standard trained dictionaries, which are fully adaptable but non-efficient and costly to deploy.

C. Graph-Haar Wavelet Construction

The success of the sparse dictionary model heavily depends on a proper choice of the base dictionary Φ. In order to bring the double sparsity idea to the treatment of graph signals, we ought to define Φ such that it reflects the graph topology. Following our previous work [19], we choose to construct a Haar-like graph wavelet basis. As an initial step, and in order to expose the inherent multi-scale structure of the data, the underlying graph should be converted to a hierarchical tree by spectral partitioning.

Spectral graph partitioning methods are commonly based on the Fiedler vector [21], which is the eigenvector corresponding to the smallest non-zero eigenvalue of the graph Laplacian matrix L. The Fiedler vector bisects the graph into two disjoint yet covering sets of nodes based on the sign of the corresponding vector entry. Explicitly, denote the Fiedler vector for the ℓ-th partition by v_f^ℓ, then the bisection results in two separate sets:

$$\begin{aligned} \Omega_1^\ell &= \left\{ i | v_f^\ell[i] \ge 0 \right\}, \\ \Omega_2^\ell &= \left\{ i | v_f^\ell[i] < 0 \right\}. \end{aligned} \tag{5}$$

By applying the spectral bisection procedure recursively, in a coarse-to-fine manner (until reaching individual nodes or a constant-polarity Fiedler vector), full partitioning is obtained and the graph can be traversed into a hierarchical tree [22].

We note that the Fiedler vector itself may be efficiently computed using the power-method or Lanczos algorithm [23], without having to compute the full eigendecomposition of L. Furthermore, only a few iterations of these methods typically suffice as the bisection only depends on the sign pattern of the Fiedler vector and not on its precise values.

The proposed bisection approach is demonstrated in Figure 1, where the first two hierarchies of partition are presented for the Minnesota road network graph.

Equipped with the tree representation of the given data, we can now construct an orthonormal Haar-like wavelet basis. That is, each basis function consists of constant values in each set,

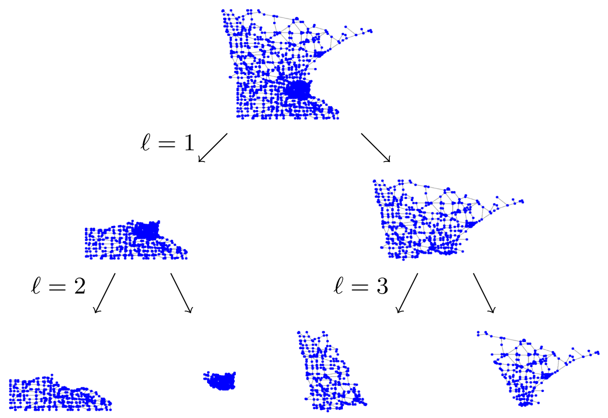


Fig. 1. The first partition hierarchies illustrated on the Minnesota road network graph.

with the constants chosen so as to satisfy the orthogonality (meaning, in this case, that the sum of all entries should be zero) and normalization requirements. Explicitly, the first function is constant over the graph

$$\phi_0[i] = \frac{1}{\sqrt{N}} \quad \forall i, \tag{6}$$

and the ℓ-th partition induces the function

$$\phi_\ell[i] = \begin{cases} \frac{\sqrt{|\Omega_2^\ell|}}{\sqrt{|\Omega_1^\ell|}\sqrt{|\Omega_1^\ell|+|\Omega_2^\ell|}} & i \in \Omega_1^\ell, \\ -\frac{\sqrt{|\Omega_1^\ell|}}{\sqrt{|\Omega_2^\ell|}\sqrt{|\Omega_1^\ell|+|\Omega_2^\ell|}} & i \in \Omega_2^\ell, \\ 0 & \text{else.} \end{cases} \tag{7}$$

The accumulated set of basis functions $\{\phi_\ell\}_\ell$ constitutes the columns of the matrix Φ that will serve as our base dictionary.

Not only is Φ orthogonal by construction, but also the data geometry was captured by a hierarchical tree of increasingly refined partitions. This achieves the desired localization of the constructed basis functions, and consequently, of their sparse linear combinations, which constitute the atoms of $D = \Phi A$.

We note that a similar graph-Haar wavelet basis was proposed in [14]. However, the work in [14] assumes the given data is readily encoded as a balanced hierarchical tree, whereas we propose an efficient partitioning to traverse the graph into a tree, which is additionally not required to be balanced. This enables using such graph-Haar basis in practice.

Finally, we emphasize again that while our general scheme could incorporate any overcomplete or more sophisticated graph-wavelet construction, these would be significantly more costly to implement, especially as larger graphs are concerned. The orthogonal basis here proposed introduces the desired multi-scale properties while being simple and efficient in terms of both construction and deployment. Though Φ itself is non-redundant, when combined with a redundant sparse matrix A it leads to an effective dictionary D that can represent a wide class of problems.

III. The Graph Enhanced Multi-Scale Dictionary Learning Algorithm (GEMS)

A. The Overall Learning Formulation

For derivation simplicity, we assume in this section that the graph Laplacian and wavelet basis are given, and focus on the task of training the dictionary. In the next section, we shall consider the complementary task of learning the graph Laplacian.

To solve the graph-enhanced multi-scale dictionary learning problem posed in (4), we develop a K-SVD like learning scheme, based on an alternating minimization approach. Recall that the K-SVD iteration consists of two main steps. The first is sparse coding of the signals in Y, given the current dictionary $D = \Phi A$, to obtain X. Optimizing (4) over X yields the graph regularized sparse coding problem:

$$\begin{aligned} \arg\min_{X} \quad & \|Y - \Phi A X\|_F^2 + \beta Tr(X L_c X^T) \\ \text{s.t.} \quad & \|x_i\|_0 \leq T \quad \forall i. \end{aligned} \tag{8}$$

which could be solved using our previously proposed GRSC algorithm [6, Algorithm 2] when setting $D = \Phi A$.

The second step is updating the dictionary atoms given the sparse representations in X. Note that unlike DGRDL, our structural constraint is here imposed directly on A, which is additionally required to preserve column-wise sparsity. This necessitates major modifications of the dictionary update procedure.

The dictionary update is performed one atom at a time, optimizing the target function for each atom individually while keeping the remaining atoms fixed. To devise the update rule for the j-th atom, let

$$\|Y - \Phi A X\|_F^2 = \left\| Y - \sum_i \Phi a_i x_i^T \right\|_F^2 = \|E_j - \Phi a_j x_j^T\|_F^2, \tag{9}$$

where x_j^T denotes the j-th row of X and we have defined the error matrix without the j-th atom as $E_j = Y - \sum_{i \neq j} \Phi a_i x_i^T$.

Along with the j-th atom we update its corresponding row of coefficients x_j^T. To preserve the representation sparsity constraints, this update uses only the subset of signals in Y whose sparse representations use the current atom. Denote by Ω_j the indices of the subset of signals using the j-th atom.

For notation simplicity, let us denote by E, g^T, L_c^R the restricted versions of E_j, x_j^T, L_c (respectively) limited to the subset Ω_j, and let $a = a_j$. The target function to be minimized for updating the j-th atom with its corresponding coefficients is therefore

$$\begin{aligned} \arg\min_{a, g^T} \quad & \|E - \Phi a g^T\|_2^2 + \alpha a^T \Phi^T L \Phi a + \beta g^T L_c^R g \\ \text{s.t.} \quad & \|a\|_0 \leq P, \ \|\Phi a\|_2 = 1. \end{aligned} \tag{10}$$

Note that the atom update in [18] may be seen as a special case of Equation (10) obtained for $\alpha = \beta = 0$. Therefore, we shall utilize similar derivation steps to those employed in [18], while generalizing the results to integrate the additional graph constraints.

To optimize over the atom a, one needs to solve

$$\begin{aligned} \arg\min_a \quad & \|E - \Phi a g^T\|_2^2 + \alpha a^T \Phi^T L \Phi a \\ \text{s.t.} \quad & \|a\|_0 \le P, \quad \|\Phi a\|_2 = 1. \end{aligned} \tag{11}$$

To approximate the solution of this problem, we solve (11) without the norm constraint on Φa, followed by a post-processing step that transfers energy between a and g to achieve $\|\Phi a\|_2 = 1$ while keeping the product ag^T fixed. This choice is justified if the regularization coefficient α is small such that the chosen support of a is not impacted by the normalization.

According to Lemma 1 in [18], $\|E - \Phi a g^T\|_2^2 = \|Eg - \Phi a\|_2^2$ as long as $g^T g = 1$. Following a similar derivation, if $\|g\|_2 = 1$ Equation (11) is equivalent to

$$\arg\min_a \|Eg - \Phi a\|_2^2 + \alpha a^T \Phi^T L \Phi a \quad \text{s.t.} \quad \|a\|_0 \le P. \tag{12}$$

By applying a preprocessing step of normalizing g to unit length, we can therefore further simplify the problem. This step is valid as it will only result in a scaled version of a, which is afterwards re-normalized by balancing between a and g.

B. Dictionary Update via OMP-Like Algorithm

One possible solution of (12) leverages the orthogonality of Φ, by which the problem is equivalent to

$$\arg\min_a \|\Phi^T Eg - a\|_2^2 + \alpha a^T M a \quad \text{s.t.} \quad \|a\|_0 \le P \tag{13}$$

where to simplify notation, we have denoted $M = \Phi^T L \Phi$.

We can devise a greedy atom pursuit algorithm for this problem, similar to the Orthogonal Matching Pursuit (OMP) [24]. The energy to minimize for each element in the vector a will here include a penalty for its correlation with all previously selected elements as reflected through the matrix M.

At the k-th iteration, we have $\|a\|_0 = k - 1$ and we seek the k-th entry to be added. The current residual is $r = \Phi^T Eg - a$. The cost of choosing to add the j-th vector entry (assuming it was not yet included) with coefficient value z_j is

$$\epsilon_j = \|r - e_j z_j\|_2^2 + \alpha(a + e_j z_j)^T M (a + e_j z_j), \tag{14}$$

where e_j denotes the j-th canonical vector. Note that the j-th entry in both a and r is assumed nulled.

If this entry is chosen, the optimal coefficient value would be

$$z_j^* = \arg\min_{z_j} \epsilon_j = \frac{r_j - \alpha a^T M_j}{1 + \alpha M_{jj}} \tag{15}$$

where $M_{jj} = e_j^T M e_j$ is the j-th diagonal entry of M, and $M_j = M e_j$ is the j-th column of M.

Reorganizing ϵ_j and plugging in z_j^*, we obtain

$$\epsilon_j^* = -\frac{(r_j - \alpha a^T M_j)^2}{1 + \alpha M_{jj}} + \|r\|_2^2 + \alpha a^T M a. \tag{16}$$

The minimum over j is attained when the term $\frac{(r_j - \alpha a^T M_j)^2}{1 + \alpha M_{jj}}$ is maximal,[1] and the corresponding j^*-th entry will be added to the vector a with entry value z_j^*. Repeating the above described process for P iterations, the complete sparse atom a is assembled.

[1]As a sanity check, notice that for $\alpha = 0$ this term is simply r_j^2 hence the maximum is reached when $j^* = \arg\max_j \; |r_j|$, in consistency with the classic OMP.

The result could be further improved by adding an orthogonalization step, in which the determined support is kept fixed and the coefficient values z_j^* are replaced globally using least-squares. Explicitly, denote by a^R, M^R, Ψ^R the versions of a, M and $\Psi = \Phi^T Eg$ restricted to the subset of entries Ω chosen by the greedy process. Then solving

$$\arg\min_{a^R} \|\Psi^R - a^R\|_2^2 + \alpha (a^R)^T M^R a^R \tag{17}$$

leads to the optimized entries $a^R = (I + \alpha M^R)^{-1} \Psi^R$ at the support Ω, composing the final atom a.

C. Dictionary Update via ADMM

While OMP is equipped with an efficient implementation that significantly reduces runtime, a better result can be obtained by seeking a different solution for (12). The approach we take here relies on the alternating direction method of multipliers (ADMM) [25], and is similar in spirit to the GRSC pursuit algorithm developed for DGRDL [6].

In this approach, we split the non-convex sparsity constraint to an auxiliary variable b, and Equation (12) is reformulated as

$$\begin{aligned} \arg\min_{a,b} \quad & \|Eg - \Phi a\|_2^2 + \alpha a^T M a \\ \text{s.t.} \quad & a = b, \; \|b\|_0 \le T, \end{aligned} \tag{18}$$

where we have again denoted $M = \Phi^T L \Phi$.

The augmented Lagrangian is then given by

$$\mathcal{L}_\rho(a, b, u) = f(a) + g(b) + \rho\|a - b + u\|_2^2 \tag{19}$$

where $f(a) = \|Eg - \Phi a\|_2^2 + \alpha a^T M a$, $g(b) = \mathcal{I}(\|b\|_0 \le P)$ for an indicator function $\mathcal{I}()$, and u is the scaled dual form variable.

The iterative solution consists of sequential optimization steps over each of the variables. Namely, in the k-th iteration

$$\begin{cases} a^{(k)} = \arg\min_a \|Eg - \Phi a\|_F^2 + \alpha a^T M a \\ \qquad\qquad + \rho\|a - b^{(k-1)} + u^{(k-1)}\|_2^2, \\ b^{(k)} = \arg\min_b \mathcal{I}(\|b\|_0 \le P) + \rho\|a^{(k)} - b + u^{(k-1)}\|_2^2, \\ u^{(k)} = u^{(k-1)} + a^{(k)} - b^{(k)}. \end{cases} \tag{20}$$

Substituting the sub-optimization problems with their closed-form solutions results in

$$\begin{cases} a^{(k)} = \left(\Phi^T \Phi + \alpha M + \rho I\right)^{-1} \left(\Phi^T Eg + \rho(b^{(k-1)} - u^{(k-1)})\right) \\ b^{(k)} = \mathcal{S}_P\left(a^{(k)} + u^{(k-1)}\right) \\ u^{(k)} = u^{(k-1)} + a^{(k)} - b^{(k)} \end{cases} \tag{21}$$

where $\mathcal{S}_P$ is a hard-thresholding operator, keeping only the P largest magnitude entries of its argument vector.

After a few iterations, the process converges to the desired sparse atom $a = b^{(k)}$. Though the ADMM solution is more

Algorithm 1: Graph-Enhanced Multi-Scale Dictionary Learning (GEMS).

```
Inputs: signal set $Y$, base dictionary $\Phi$, initial dictionary representation $A$, target atom sparsity $P$, target signal sparsity $T$, graph Laplacians $L$ and $L_c$
for $k = 1, 2, ...$ do
  • Sparse Coding: apply GRSC [6] to solve (8) for $X$
  • Dictionary Update:
    for $j = 1, 2, \dots, K$ do
      – Identify the samples using the $j$-th atom,
            $\Omega_j = \left\{ i \mid 1 \le i \le M \, , \, X_{(k)}[j, i] \ne 0 \right\}$
      – Define the operator $P_j$ restricting to columns to the subset $\Omega_j$
      – $E_j = Y - \sum_{i \ne j} \Phi a_i x_i^T$
      – Set the restricted variables $E \triangleq E_j P_j$, $g^T \triangleq x_j^T P_j$ and $L_c^R \triangleq P_j^T L_c P_j$
      – Normalize $g = \frac{g}{\|g\|_2}$
      – Solve (12) for $a$ (using one of the proposed methods)
      – Normalize $a = \frac{a}{\|\Phi a\|_2}$
      – $g = \left( I + \beta L_c^R \right)^{-1} E^T \Phi a$
      – Plug the results $A_j = a$, $X_{(j, \Omega_j)} = g^T$
    end for
end for
Outputs: $A, X$
```

time consuming, it usually leads to better performance in practice compared with the greedy approach. Consequently, we have used this variant throughout the experiments described in Section V.

We should note that this algorithm comes with no convergence or optimality guarantees, since the original problem is not convex. This can be easily changed if the ℓ_0 sparsity constraint is relaxed by an ℓ_1 norm, thus replacing the hard-thresholding step with a soft-thresholding one.

D. Updating the Coefficients

So far, we have presented two alternative techniques for optimizing each atom of the sparse dictionary. Finally, having updated the atom a, we should update its corresponding coefficients by solving

$$\arg\min_{g} \|E - \Phi a g^T\|_F^2 + \beta g^T L_c^R g \tag{22}$$

which yields

$$g = \left(I + \beta L_c^R \right)^{-1} E^T \Phi a. \tag{23}$$

Combining the pieces, the final atom update process consists of the following steps: (1) normalize g to unit length; (2) solve (12) using either the ADMM atom update algorithm or the OMP-like greedy pursuit proposed above; (3) normalize a to fulfill $\|\Phi a\|_2 = 1$; and (4) update g. The complete algorithm is detailed in Algorithm 1.

E. Computational Complexity

In the following, we discuss some computational aspects of the proposed algorithm. In particular, we analyze the complexity of building the graph-Haar basis Φ, and of computing the matrix inversions involved in the dictionary learning algorithm.

1) Building Φ: The graph-wavelet base dictionary construction, as detailed in Section II-C, consists of recursive graph bisection based on the Fiedler vector. The number of partitions scales linearly with the number of graph nodes N, and is upper bounded by $N - 1$, with the bound obtained when the partitioning process reaches individual nodes and not stopped earlier due to a unipolar Fiedler vector.

Estimating the Fiedler vector for an n-dimensional subgraph requires $\mathcal{O}(n^2)$ operations due to matrix-vector multiplications involved in the power-method iterations. However, the graph is commonly constructed via a thresholded kernel or as a nearest-neighbor graph. As a result, the Laplacian matrix is usually sparse with q non-zeros per row on average. In such case, the complexity is reduced to $\mathcal{O}(qn)$. In terms of the number of graph edges $|E|$, having q non-zeros per row of L implies that $|E| = N(q-1)/2$, therefore we deduce that the complexity of constructing Φ scales linearly with the number of edges $\mathcal{O}(|E|)$.

Finally, note that for finer hierarchy levels, the lower complexity of the power-method iteration compensates for the larger number of partitions, thus the total complexity of building Φ is $\mathcal{O}(N \log N)$.

2) Matrix Inversions: One of the computational costly components in our algorithm is the involved matrix inversions. However, exploiting the sparsity of both the representations and the atoms, and restricting the operations to the given support, the matrices to be inverted are in fact typically small.

Concerning the coefficient update in Equation (23), the matrix to be inverted is restricted to the subset of indices corresponding to the signals choosing a specific atom. Given the typical redundancy of the dictionary, this number is much smaller than the total number of signals M.

Considering Equation (17), since the atom a has only P non-zero entries and only the values on the determined support are updated, this amounts to inverting a $P \times P$ matrix, followed by multiplication with a P-dimensional vector. The complexity is thus $\mathcal{O}(P^3)$, which is independent of N.

Alternatively, using the ADMM-based atom update of Equation (21) requires inverting an $N \times N$ matrix for optimizing a, implying a complexity of $\mathcal{O}(N^3)$. However, as the matrix to be inverted is symmetric positive-definite, this cost can be significantly reduced. Following the analysis in [6], we could precompute the eigendecomposition of the positive semi-definite Laplacian, $L = V \Lambda V^T$, where Λ is a diagonal matrix and V is orthogonal. By construction, Φ is also an orthogonal matrix. Therefore,

$$\begin{aligned} \left(\Phi^T \Phi + \alpha M + \rho I \right)^{-1} &= \left(\alpha \Phi^T L \Phi + (\rho + 1) I \right)^{-1} \\ &= \Phi^T V \left(\alpha \Lambda + (\rho + 1) I \right)^{-1} V^T \Phi. \end{aligned} \tag{24}$$

The complexity of inverting a diagonal matrix is $\mathcal{O}(N)$, and combined with the matrix multiplications the total complexity is $\mathcal{O}(N^2)$, which is far lower than a direct matrix inversion. While the eigendecomposition of L still necessitates $\mathcal{O}(N^3)$ operations, it can be carried out once and should not be repeated for every atom at every dictionary update iteration.

IV. Adaptive Base Dictionary

In cases where the true underlying graph is unknown, it can be constructed or inferred from the data. Several attempts have recently been made to learn the underlying graph from data observations [26]–[30]. In this work, similarly to the approach we proposed in [6], we could leverage the trained dictionary, that already processed the input and captured its essence, to adapt and improve the graph Laplacian estimation. That is, the graph is learned jointly with the dictionary rather than being learned directly from the observed signals.

The extension of the proposed GEMS algorithm for this case is straightforward. When the graph Laplacian is unknown, we initialize it from the training data Y using some common construction (such as a Gaussian kernel). Based on this initial L, we construct the base dictionary Φ as described in Section II-C and run a few iterations of Algorithm 1 without reaching full convergence. Having at hand an updated sparse matrix A, and therefore an updated effective dictionary D, we could optimize the graph Laplacian L such that it leads to smoother atoms over the graph. Adding some requirements to normalize L and make it a valid graph Laplacian matrix, the resulting optimization problem is

$$\begin{aligned} \arg\min_{L} \quad & \alpha Tr(A^T\Phi^T L\Phi A) + \mu\|L\|_F^2 \\ \text{s.t.} \quad & L_{ij} = L_{ji} \leq 0 \ (i \neq j),\ L\underline{1} = \underline{0},\ Tr(L) = N. \end{aligned} \tag{25}$$

Note that this is in fact the same problem defined in [6] for the setting $D = \Phi A$.

By vectorizing L, Equation (25) can be cast as a quadratic optimization problem with linear constraints, which could be solved using existing convex optimization tools. As the computational complexity scales quadratically with the number of nodes N, for very large graphs an approximate solution may be sought based on splitting methods or using iterative approaches.

The complete GEMS scheme can now be assembled by alternating between the dictionary learning of Algorithm 1 and the Laplacian learning of Equation (25).

An important consequence of the graph optimization is that given an updated L, the base dictionary Φ could now be refined as well. By doing so, we effectively replace the fixed graph-wavelet basis with an adaptive one, which is iteratively tuned along with the dictionary learning process, thus adding yet another level of flexibility to the proposed scheme. Having reconstructed Φ, the dictionary update algorithm can be resumed for several more iterations. This process of updating L, refining Φ and then training A and X can be repeated until converging to a desired output.

It should be emphasized that the Laplacian optimization may be applied to the manifold Laplacian L_c as well in a similar manner.

Before diving into the experimental section, we briefly discuss a special setting of the proposed GEMS scheme obtained by omitting the explicit regularizations, i.e. setting $\alpha = \beta = 0$. This choice alleviates the additional complexity of updating the atoms and so further improves the scalability of this method and enables treatment of very large graphs. The optimization problem for this setting reduces to

$$\begin{aligned} \arg\min_{A,X} \quad & \|Y - \Phi AX\|_F^2 \\ \text{s.t.} \quad & \|x_i\|_0 \leq T \quad \forall i, \\ & \|a_j\|_0 \leq P \quad \forall j, \quad \|\Phi a_j\|_2 = 1 \quad \forall j. \end{aligned} \tag{26}$$

Nevertheless, the graph Laplacian L is still accounted for implicitly through the construction of Φ. Therefore, an optimized Laplacian could still enhance our method even in this setting: by gradually refining the base dictionary along the training process, it instigates an adaptive graph-Haar wavelet dictionary. In that sense, this configuration can be seen as an adaptive version of SDL [19], in which the base dictionary Φ is updated along the training process. We shall henceforth refer to this high-dimensional setting as GEMS-HD.

V. Experiments and Applications

In this section, we demonstrate the effectiveness of the proposed GEMS algorithm on synthetic examples of piecewise-smooth nature and on real network data, and show its potential use in various data processing and analysis applications.

A. Synthetic Experiment

We first carry out synthetic experiments, similar to the ones described in [6], [19]. However, to corroborate the applicability of GEMS to a broader class of graph signals, the generated data here complies with a piecewise-smooth model rather than the global-smooth one used in [6], [19].

Initially, we generated a random graph consisting of N randomly distributed nodes. The edge weights between each pair of nodes were determined based on the Euclidean distances between them $d(i,j)$ and using the Gaussian Radial Basis Function (RBF) $w_{ij} = \exp(\frac{-d^2(i,j)}{2\sigma^2})$ with $\sigma = 0.5$.

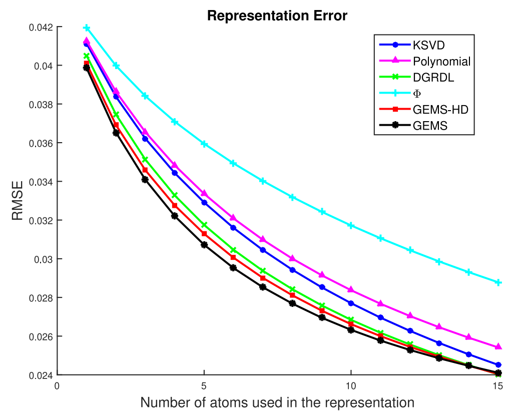
For the data generation, we started by simulating two sets of globally-smooth graph signals. Each such set was created by randomly drawing an initial matrix $Y_0 \in \mathbb{R}^{N\times 10N}$ and solving

$$\arg\min_{Y} \|Y - Y_0\|_F^2 + \lambda Tr(Y^T LY), \tag{27}$$

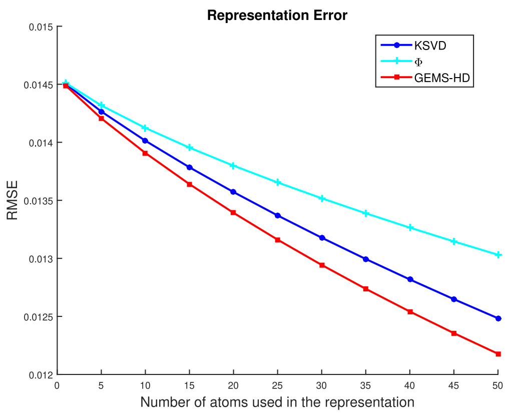
which yields smoothed signals $Y = (I + \lambda L)^{-1}Y_0$.

Given two such data matrices Y_1 and Y_2, we combined them to generate piecewise-smooth graph signals. For that purpose, a local neighborhood was randomly chosen for each signal, and its measurements in that region were taken from Y_2 while the rest were taken from Y_1. Consequently, each signal was normalized to have unit norm. A subset of 40% of the generated signals was used for training, leaving the rest for testing.

Using this training data, several dictionaries were learned including the K-SVD [8], the graph polynomial dictionary [5], DGRDL [6], and the proposed GEMS with a graph-Haar

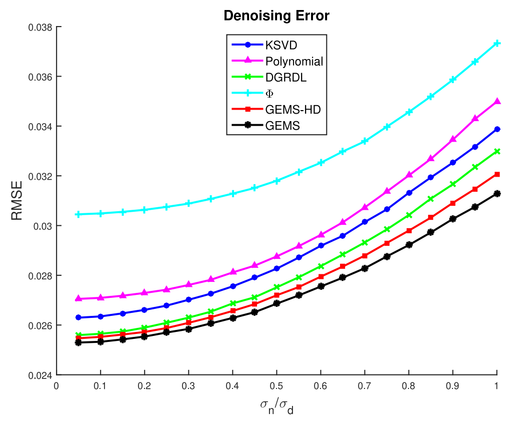


(a) Representation error for $N = 256$

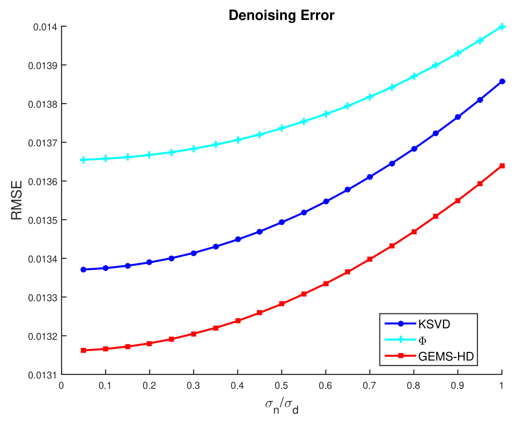


(b) Representation error for $N = 4096$

Fig. 2. Comparison of the learned dictionaries in terms of normalized RMSE for representing synthetic data of different dimensions with various sparsity levels.



(a) Denoising error for $N = 256$



(b) Denoising error for $N = 4096$

Fig. 3. Comparison of the learned dictionaries in terms of normalized RMSE for the task of synthetic data denoising with different noise levels σ_n with respect to the data standard deviation σ_d.

base dictionary constructed as in Section II-C. Additionally, we trained the high-dimensional mode GEMS-HD, for setting $\alpha = \beta = 0$. For a fair comparison, all these dictionaries are of the same size, $N \times 2N$. We also evaluated a direct use of the constructed graph wavelet basis Φ, whose size is $N \times N$.

Two setups were tested: the first with a moderate size graph of $N = 256$ nodes, and the second with a high-dimensional graph containing $N = 4096$ nodes. The dictionaries were trained with a fixed number of non-zeros in the sparse coding stage ($T = 12$ and $T = 25$, respectively). For the presented variants of GEMS, the respective sparsity levels of the dictionary A were set to $P = 12$ for the medium graph and $P = 40$ for the large one.

The dictionaries were first compared by their ability to obtain the best m-term approximation of the test data (for different sparsity levels, both smaller and larger than the number of non-zeros used during training), and performance was measured in terms of the normalized Root Mean Squared Error (RMSE), $\frac{1}{\sqrt{NM}}\|Y - DX\|_F$.

The representation errors presented in Figure 2a show that for a moderate size graph, the proposed GEMS yields lower errors compared with K-SVD, DGRDL, the polynomial method and the graph-Haar wavelet basis Φ, for all evaluated sparsity levels. Furthermore, the complete GEMS scheme offers an additional improvement over GEMS-HD, that only accounts for the graph implicitly.

The representation errors obtained for a large graph setting are presented in Figure 2b. For this data dimension, the polynomial dictionary and DGRDL can no longer train in reasonable time, and were therefore omitted from the comparison. For computational reasons, GEMS was also trained only in the GEMS-HD mode. Nevertheless, it still outperforms K-SVD and the graph wavelet base dictionary Φ, demonstrating the scalability of the proposed method to high dimensional data.

Next, the performance of the trained dictionaries was evaluated for the common task of signal denoising, by adding Gaussian noise of different levels σ_n to the test signals and comparing recovery using each of the dictionaries in terms of the normalized RMSE. Assuming a noisy test signal is modeled as $y_i = Dx_i + n_i$ where n_i denotes the added noise, its denoised version $\hat{y}_i = D\hat{x}_i$ is obtained by seeking the sparse approximation of y_i (denoted $\hat{x}_i$) over each dictionary D with a known sparsity level T, using OMP.

The results of this experiment are depicted in Figure 3. Similarly to the previous experiment, these results show that GEMS outperforms the other dictionary models for all the tested noise

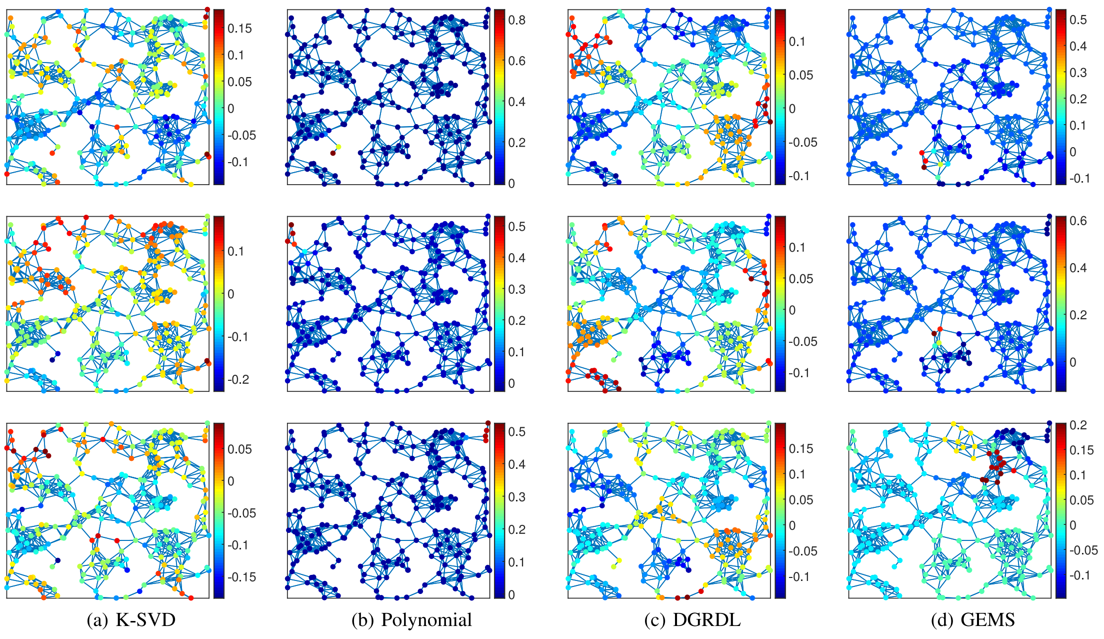


Fig. 4. Demonstrating the top 3 atoms used of each evaluated dictionary. Each column refers to a different dictionary (from left to right): K-SVD [8], Polynomial [5], DGRDL [6] and GEMS. It can be observed that GEMS yields atoms that obey a piecewise-smooth model as desired.

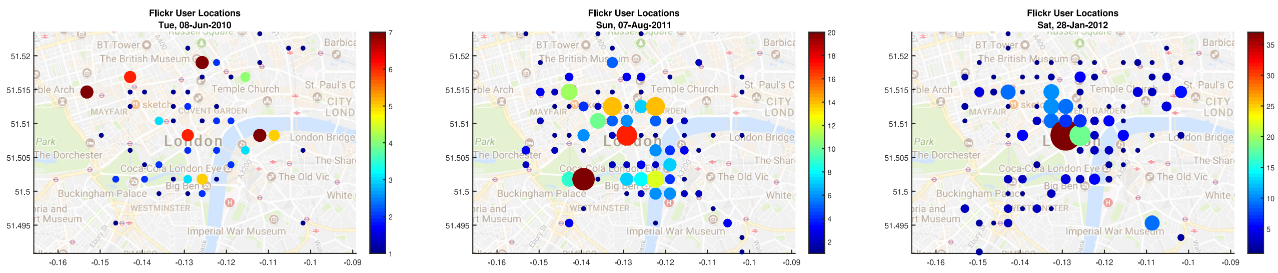


Fig. 5. Characteristic graph signals demonstrating the daily number of distinct Flickr users that have taken photos at different locations in London. The size and color of each circle indicate the signal value at that graph node.

levels, and offers a performance boost for graph signal denoising even in high dimensions.

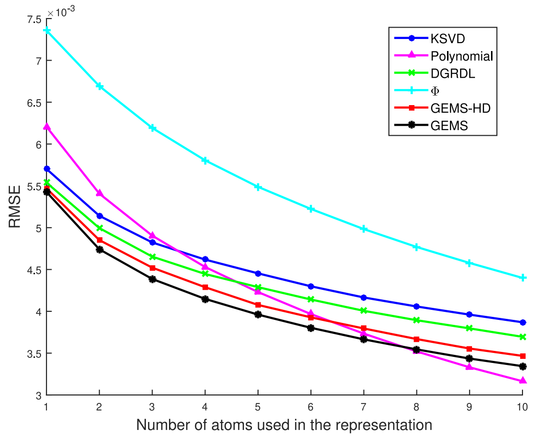
Additionally, we verify that the proposed dictionary indeed results in more localized atoms by visualizing the top 3 used atoms of each of the trained dictionaries. As can be observed in Figure 4, although all dictionaries were trained from piecewise-smooth graph signals, the atoms learned by K-SVD are unstructured and possess a random appearance, the polynomial atoms are extremely sparse and localized, and the DGRDL atoms vary more gradually than their K-SVD counterparts, yet they span the support of the entire graph. The atoms learned by GEMS are more localized and structured compared with those learned by K-SVD and DGRDL, though not as localized as the polynomial atoms. GEMS thus offers a balance between the localization and smoothness properties, yielding atoms that have the desired piecewise-smooth nature.

B. Flickr Data

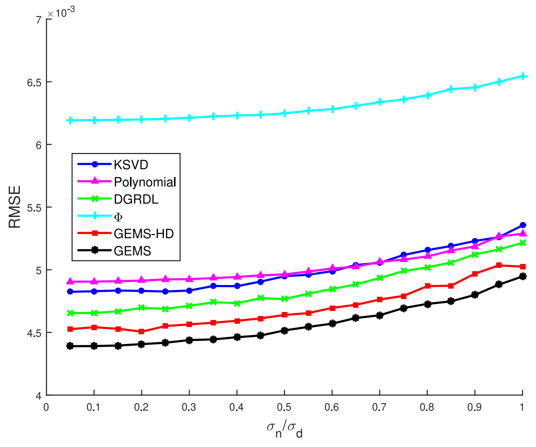
In the sequel, the proposed method was evaluated on real network data from the Flickr dataset. The dataset consists of 913 signals, representing the daily number of distinct Flickr users that have taken photos at different geographical locations around Trafalgar Square in London, between January 2010 and June 2012. An area of approximately 6×6 km was covered by a grid of size 16×16, to a total of $N = 256$ nodes. The initial graph Laplacian L was designed by connecting each node and its 8 nearest neighbors, setting the edge weights to be inversely proportional to the Euclidean distance between the nodes.

Each photo acquisition was allocated to its nearest grid point, so that the graph signals represent the spatially aggregated daily number of users taking photos near each grid location. A random subset of 700 signals constitutes the training set, and the rest were used for testing. All signals were normalized with respect to the one having the maximal energy. Some typical signals from the Flickr dataset are illustrated in Figure 5.

For this dataset, the proposed GEMS dictionary was again compared with K-SVD [8], the graph polynomial dictionary [5] and DGRDL [6], as well as with the constructed graph-Haar wavelet basis Φ. All evaluated dictionaries are of the same size



(a)



(b)

Fig. 6. Comparison of the learned dictionaries in terms of normalized RMSE for different applications tested on the Flickr dataset: (a) representation error for different sparsity levels, (b) denoising error for different noise levels σ_n with respect to the data standard deviation σ_d.

of $N \times 2N$ (with the exception of the orthogonal basis Φ whose dimensions are $N \times N$) and sparsity thresholds of $T = 3$ and $P = 10$ were used for training.

Similarly to the synthetic experiment, the different dictionaries were evaluated on two tasks: their ability to represent the test set data with different sparsity levels (number of used atoms), and their performance in signal denoising with different noise levels.

The representation errors for this dataset are presented in Figure 6a, and the corresponding denoising errors in Figure 6b.

It can be observed that in both tasks, for all sparsity levels and all noise levels tested, GEMS yields significantly lower errors compared with K-SVD, the polynomial graph dictionary, and DGRDL. These results coincide with those obtained for the synthetic experiment. The only exception is the approximation error using a larger number of atoms ($T > 8$), for which the polynomial dictionary achieves slightly better results than GEMS. Recall, however, that the polynomial dictionary training is much more complex and its runtime is substantially longer, making its use impractical for larger dimensions.

It should also be emphasized that the performance of GEMS is expected to further improve as the training set becomes scarce.

Moreover, the results could be improved by re-training the dictionaries for every sparsity level. Instead, training was performed once for a fixed T and the generalization ability of the dictionaries was challenged by evaluating them using different (both smaller and larger) sparsity levels. Nevertheless, as the experimental results demonstrate, the trained GEMS model fits the data very well even in this setting.

In terms of runtime, for this experimental setup GEMS-HD was 4.2 times faster than DGRDL and 5.8 times faster than the polynomial method.

C. Uber Pickups in New York City

Next, we consider a larger real network dataset of Uber pickups in New York City [31]. This dataset contains information on over 4.5 million Uber pickups in New York City from April to September 2014, with each trip listed by date and pickup time, as well as GPS coordinates.

To create a graph from this raw data, we sampled the New York City region on a grid of 150×150 points and assigned each pickup to its nearest grid point, accumulating the number of pickups in each grid location. Similarly, pickups were aggregated over time intervals of one hour each, such that the total number of pickups in a specific hour is a graph signal. To enrich the graph structure, we selected only the subset of grid points for which the overall number of pickups exceeded 500, keeping a total of $N = 2442$ nodes. The weight of the edge between the nodes i and j was set to $w_{ij} = \exp(\frac{-d^2(i,j)}{2\sigma^2})$, where $d(i,j)$ is the Euclidean distance between their respective coordinates and σ is a scaling factor proportional to the median distance. Exemplar signals of this dataset are illustrated in Figure 7.

Following the previous experiments, we compared GEMS-HD with K-SVD and DGRDL, which were the leading competitors. The different dictionaries were again trained and evaluated on the tasks of signal approximation and denoising. Sparsity thresholds of $T = 20$ and $P = 30$ were used for training, and all signals were normalized with respect to the one having the maximal energy. The results are depicted in Figure 8, establishing again the advantage of GEMS-HD over the other compared methods. In terms of runtime, for this experimental setup GEMS-HD was about 8 times faster than DGRDL, demonstrating its better scalability to high data dimensions.

D. Discussion

Just before we conclude this section, we would like to discuss an additional side benefit of the proposed GEMS algorithm. The multi-scale nature of the GEMS dictionary may serve data analysis tasks and be used for capturing important phenomena in the data. For instance, one might characterize and distinguish

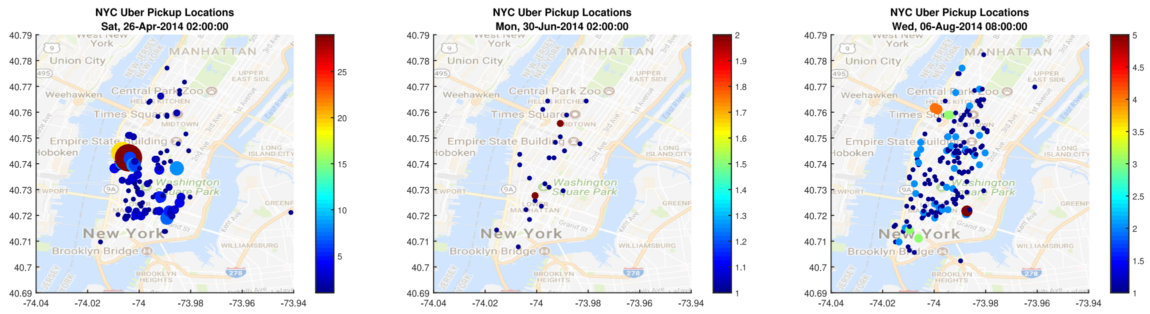
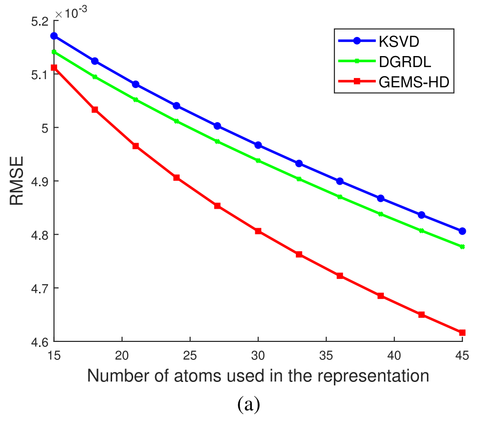

NYC Uber Pickup Locations
Sat, 26-Apr-2014 02:00:00
NYC Uber Pickup Locations
Mon, 30-Jun-2014 02:00:00
NYC Uber Pickup Locations
Wed, 06-Aug-2014 08:00:00

Fig. 7. Characteristic graph signals demonstrating the hourly number of Uber pickups at different locations in New York City. The size and color of each circle indicate the signal value at that graph node.



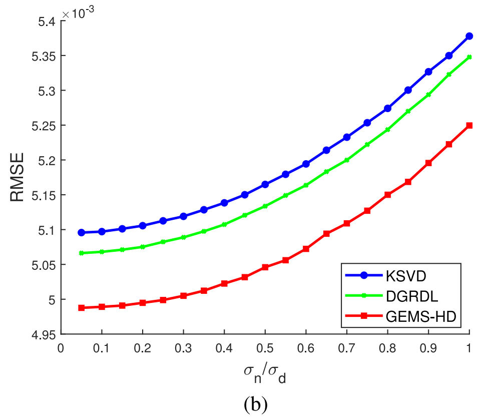


Fig. 8. Comparison of the learned dictionaries in terms of normalized RMSE for different applications tested on the Uber NYC pickups dataset: (a) representation error for different sparsity levels, (b) denoising error for different noise levels σ_n with respect to the data standard deviation σ_d.

between different signals based solely on the dictionary atoms chosen for their approximation. Put differently, some of the learned patterns may be associated with a specific day of the week, or a specific time of day.

To demonstrate this idea, we consider the signals measuring the number of Uber pickups during the times 7AM-8AM. By sparse coding of these signals over the trained dictionary and analyzing the chosen atoms statistics, we can distinguish between regions that are more active on weekdays and others that are more active on weekends. Repeating the experiment for other signal groups reveals the pattern variability between different hours of the day, as illustrated in Figure 9.

Naturally, different days and times present with different characteristic patterns. For instance, business district areas tend to be more active on weekdays and working hours, compared with touristic areas that are more active at nights and on weekends. Since different time slots were found to be represented by different dictionary atoms, as demonstrated in Figure 9, this indicates to the richness of the learned dictionary that captures all these different patterns, as well as hints to a potential use of this dictionary for data analysis tasks.

As mentioned earlier, another essential property that the proposed dictionary structure introduces is locality and piecewise-smooth behavior. As advocated in [5], for example, similar local patterns may appear at various locations across the network, and thus learning localized atoms may benefit the processing of high-dimensional graph signals.

Indeed, graph signals emerging in various real-life applications are only piecewise-smooth (and not globally-smooth) over the graph. For instance, while each community in a social network may have a relatively homogeneous behavior, some variability could be expected between communities, exhibiting delicate differences that the graph Laplacian cannot encode. Similarly, traffic patterns may be different in rural areas compared with urban regions and city centers, with sharper transitions occurring near city boundaries. Such phenomena are ill-represented by the graph Laplacian, even when inferred from the data. Since the Laplacian matrix models the common underlying structure of the given signals, it is often unable to account for the local nature of different network regions. In these cases, relying on a global smoothness is insufficient, and an alternative local (piecewise) regularity assumption may better fit such signals.

To highlight this property, the data in all the experiments presented above has a localized, clustered, or piecewise-smooth nature. As demonstrated throughout all the experiments, the global regularity assumption of DGRDL [6] evidently makes it suboptimal for representing such signals. However, by relaxing this assumption and infusing a multi-scale structure to the learned dictionary, GEMS better applies to this broader class of graph signals.

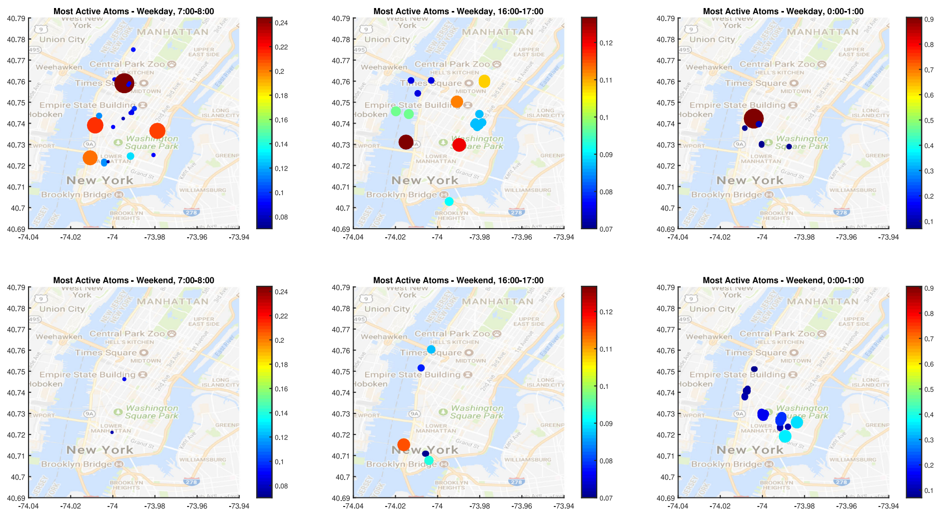

Most Active Atoms - Weekday, 7:00-8:00
Most Active Atoms - Weekday, 16:00-17:00
Most Active Atoms - Weekday, 0:00-1:00
Most Active Atoms - Weekend, 7:00-8:00
Most Active Atoms - Weekend, 16:00-17:00
Most Active Atoms - Weekend, 0:00-1:00

Fig. 9. Comparing the most active GEMS atoms in representing Uber pickup counts on weekdays (top) and weekends (bottom), at different hours of the day (from left to right): 7-8AM, 4-5PM, 0-1AM.

VI. Conclusions

In this paper, we introduced a new dictionary learning algorithm for graph signals that mitigates the global regularity assumption and applies to a broader class of graph signals, while enabling treatment of higher dimensional data compared with previous methods.

The core concept of the proposed GEMS method lies in combining a simple and efficient graph-Haar wavelet basis, that brings a multi-scale nature we deem vital for representing large signals, with a learned sparse component, that makes it adaptive to the given data.

The underlying graph topology was incorporated in two manners. The first is implicit, by modeling the learned dictionary atoms as sparse combinations of graph wavelet functions, thus practically designing an adaptable multi-scale dictionary. The second is explicit, by adding direct graph constraints to preserve the local geometry and promote smoothness in both the feature and manifold domains.

Furthermore, the complete optimization scheme offers the ability to refine the graph Laplacian L, as well as the graph-wavelet basis Φ, as an integral part of the dictionary learning process.

The effectiveness of the proposed algorithm was demonstrated through experiments on both synthetic data and real network data, showing that it achieves superior performance to other tested methods in data processing and analysis applications of different nature and different dimensions. While the imposed dictionary structure already reduces the number of trainable parameters, the GEMS-HD version that emerges as a special configuration of GEMS pushes it to accommodate even higher dimensional graph data, in the order of thousands, or even tens-of-thousand of nodes.

We believe that graph dictionary learning methods could scale beyond these dimensions by developing online training schemes, learning in batches, employing a convolutional model etc. However at the moment, supporting graphs containing millions of nodes, which may occur in some realistic scenarios, remains an open challenge for the graph signal processing community.

References

[1] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[2] 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.

[3] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Berlin, Germany: Springer, 2010.

[4] X. Zhang, X. Dong, and P. Frossard, "Learning of structured graph dictionaries," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2012, pp. 3373–3376.

[5] 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.

[6] 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.

[7] K. Engan, S. O. Aase, and J. H. Husoy, "Method of optimal directions for frame design," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 1999, vol. 5, pp. 2443–2446.

[8] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[9] Y. Yankelevsky and M. Elad, "Structure-aware classification using supervised dictionary learning," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 2017, pp. 4421–4425.

[10] Y. Yankelevsky and M. Elad, "Graph-constrained supervised dictionary learning for multi-label classification," in *Proc. IEEE Int. Conf. Sci. Elect. Eng.*, Nov. 2016, pp. 1–5.

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

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

[13] S. K. Narang and A. Ortega, "Lifting based wavelet transforms on graphs," in *Proc. APSIPA ASC Asia-Pacific Signal Inf. Process. Assoc.*, Oct. 2009, pp. 441–444.

[14] M. Gavish, B. Nadler, and R. R. Coifman, “Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning,” in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 367–374.
[15] I. Ram, M. Elad, and I. Cohen, “Redundant wavelets on graphs and high dimensional data clouds,” *IEEE Signal Process. Lett.*, vol. 19, no. 5, pp. 291–294, May 2012.
[16] R. Rustamov and L. J. Guibas, “Wavelets on graphs via deep learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 998–1006.
[17] 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, Apr. 2016.
[18] R. Rubinstein, M. Zibulevsky, and M. Elad, “Double sparsity: Learning sparse dictionaries for sparse signal approximation,” *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1553–1564, Mar. 2010.
[19] Y. Yankelevsky and M. Elad, “Dictionary learning for high dimensional graph signals,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 2018, pp. 4669–4673.
[20] D. Zhou and B. Schölkopf, “A regularization framework for learning from graph data,” in *Proc. ICML Workshop Statistical Relational Learn. Connections Other Fields*, 2004, pp. 132–137.
[21] M. Fiedler, “A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory,” *Czechoslovak Math. J.*, vol. 25, no. 4, pp. 619–633, 1975.
[22] H. D. Simon, “Partitioning of unstructured problems for parallel processing,” *Comput. Syst. Eng.*, vol. 2, no. 2-3, pp. 135–148, 1991.
[23] B. N. Parlett, H. Simon, and L. M. Stringer, “On estimating the largest eigenvalue with the Lanczos algorithm,” *Math. Comput.*, vol. 38, no. 157, pp. 153–165, 1982.
[24] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, “Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition,” in *Proc. 27th Asilomar Conf. Signals, Syst., Comput.*, Nov. 1993, pp. 40–44.
[25] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.
[26] E. Pavez and A. Ortega, “Generalized Laplacian precision matrix estimation for graph signal processing,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 6350–6354.
[27] 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.
[28] V. Kalofolias, “How to learn a graph from smooth signals,” in *Proc. Int. Conf. Artif. Intell. Statist.*, 2016, pp. 920–929.
[29] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, “Network topology inference from spectral templates,” *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 467–483, Sep. 2017.
[30] H. E. Egilmez, E. Pavez, and A. Ortega, “Graph learning from data under Laplacian and structural constraints,” *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 825–841, Sep. 2017.
[31] Kaggle, “Uber pickups in New York City,” 2015. [Online]. Available: https://www.kaggle.com/fivethirtyeight/uber-pickups-in-new-york-city

Yael Yankelevsky (S’16) received the B.Sc. and M.Sc. degrees from the Department of Electrical Engineering, Technion-Israel Institute of Technology, Haifa, Israel, in 2006 and 2013, respectively. She is currently working toward the Ph.D. degree in the Department of Computer Science, Technion-Israel Institute of Technology. Her research interests include signal and image processing, sparse representations, inverse problems, and graphical models.

Michael Elad (F’12) received the B.Sc., M.Sc., and D.Sc. degrees from the Department of Electrical Engineering, Technion-Israel Institute of Technology, Haifa, Israel, in 1986, 1988, and 1997, respectively. Since 2003, he has been a Faculty Member with the Department of Computer Science, Technion-Israel Institute of Technology, and since 2010, he has been a Full Professor. He works in the field of signal and image processing, specializing in particular on inverse problems and sparse representations. He has been the Editor-in-Chief for the *SIAM Journal on Imaging Sciences* since 2016. He received numerous teaching awards, such as the 2008 and 2015 Henri Taub Prizes for academic excellence, and the 2010 Hershel-Rich Prize for innovation.