Martinez-Enriquez, E., Cid-Sueiro, J., Diaz-de-Maria, F., Ortega, A. (2018). Optimized Update/Prediction Assignment for Lifting Transforms on Graphs. *IEEE Transactions on Signal Processing,* vol.PP (issue 99)

DOI: 10.1109/TSP.2018.2802465

Optimized Update/Prediction Assignment for Lifting Transforms on Graphs

Eduardo Martínez-Enríquez, Jesús Cid-Sueiro, Fernando Díaz-de-María, *Member, IEEE*, and Antonio Ortega, *Fellow, IEEE*

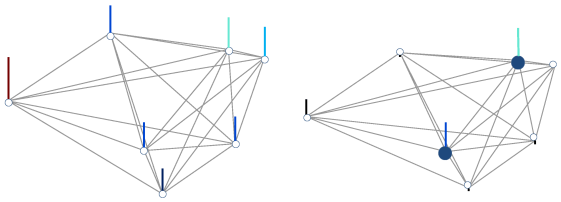
***Abstract*—Transformations on graphs can provide compact representations of signals with many applications in denoising, feature extraction or compression. In particular, lifting transforms have the advantage of being critically sampled and invertible by construction, but the efficiency of the transform depends on the choice of a good bipartition of the graph into update ($\mathcal{U}$) and prediction ($\mathcal{P}$) nodes. This is the update/prediction ($\mathcal{U}/\mathcal{P}$) assignment problem, which is the focus of this paper. We analyze this problem theoretically and derive an optimal $\mathcal{U}/\mathcal{P}$ assignment under assumptions about signal model and filters. Furthermore, we prove that the best $\mathcal{U}/\mathcal{P}$ partition is related to the correlation between nodes on the graph and is not the one that minimizes the number of conflicts (connections between nodes of same label) or maximizes the weight of the cut. We also provide experimental results in randomly generated graph signals and real data from image and video signals that validate our theoretical conclusions, demonstrating improved performance over state of the art solutions for this problem.**

***Index Terms*—Lifting transform, Graphs, $\mathcal{U}/\mathcal{P}$ Assignment, Splitting, Graph bipartition.**

I. Introduction

A. Motivation

Graphs are useful tools for describing signals defined on either regular or irregular domains such as sensor, community, transportation, or social networks. Every signal sample is associated with one node in the graph and weighted links (edges) between nodes reflect some relationships among samples (e.g., correlation, similarity, geometric distance or connectivity) [1]–[16]. Graph-based transforms have been recently developed to obtain a compact representation of graph signals, which is useful in many applications such as compression, denoising or feature extraction. A drawback of some of these transforms is that they are not critically sampled [1], [7]–[9], which leads to less compact representations. The lifting scheme [17], which has been extended to graphs [18]–[20], provides a general solution that allows to obtain critically sampled transforms on arbitrary undirected graphs. To perform lifting on graphs, the input graph signal should be split into update ($\mathcal{U}$) and prediction ($\mathcal{P}$) nodes, which is called $\mathcal{U}/\mathcal{P}$ assignment (UPA), and the update ($\mathbf{u}$) and prediction ($\mathbf{p}$) filters should be defined. In the prediction stage of the transform, $\mathcal{P}$ nodes are predicted from $\mathcal{U}$ neighbor nodes using $\mathbf{p}$ filters providing subsampled high-pass (detail coefficients) versions of the signal. Then, $\mathcal{U}$ nodes are updated from $\mathcal{P}$ detail coefficients using $\mathbf{u}$ filters giving rise to subsampled low-pass (smooth coefficients) versions of the signal. If detail coefficients (prediction residuals) are close to zero, most of the information is captured by the smooth coefficients, thus obtaining a more compact representation. Applying this process iteratively on the smooth coefficients leads to a multiresolution analysis [21] of the original signal. Figure 1 shows an example of a signal defined on a weighted graph, an UPA solution, and the prediction stage of the transform. Lifting transforms can operate on any graph with arbitrary disjoint UPAs and $\mathbf{p}$ and $\mathbf{u}$ filter designs, without compromising its perfect reconstruction and critically sampled properties [22]. Nevertheless, different UPA and filter choices may lead to different performance. In this paper we focus on how to optimize the UPA in order to achieve good energy compaction with the transform.



(a) *Original graph* (b) *UPA and prediction stage*

Fig. 1. (a) Signal defined on a weighted graph. Vertical bars represent the signal value at every node, with colder colors for lower values and vice versa, and weight values (not shown) are chosen inversely proportional to the geometrical distance between nodes. (b) The UPA process requires assigning a label ($\mathcal{U}$, dark-big nodes; or $\mathcal{P}$, white-small nodes) to each node of the graph. Then, in the prediction stage, $\mathcal{P}$ nodes are predicted from $\mathcal{U}$ neighbors, leading to detail coefficients (black bars).

B. Related Work

Figure 2 illustrates examples of UPAs proposed in the literature for different graph topologies and applications. In the figure, $\mathcal{P}$ nodes are white and $\mathcal{U}$ nodes are gray. Figure 2(a) shows the UPA proposed in [20], [22], [23] for graph trees: nodes at odd depth are assigned to $\mathcal{P}$, and nodes at even

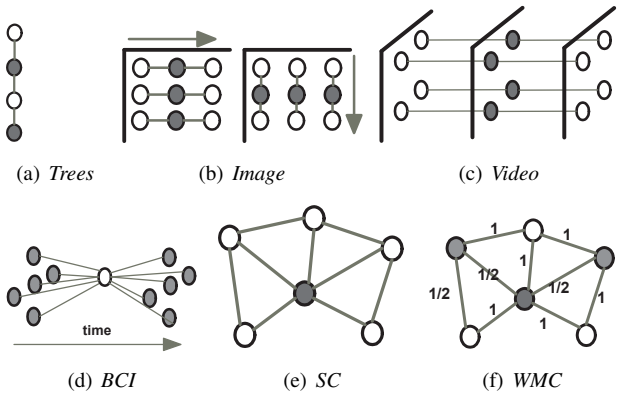
Eduardo Martínez-Enríquez is with the Instituto de Óptica, Consejo Superior de Investigaciones Científicas, Madrid, Spain (e-mail: emenriquez@tsc.uc3m.es).

Jesús Cid-Sueiro and Fernando Díaz-de-María are with the Department of Signal Theory and Communications, Carlos III University, Madrid, Spain (e-mail: jcid@tsc.uc3m.es, fdiaz@tsc.uc3m.es).

Antonio Ortega is with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089, (e-mail: antonio.ortega@sipi.usc.edu).

This work was supported in part by NSF under Grant CCF-1018977 and in part by the Spanish Ministry of Economy and Competitiveness under Grants TEC2014-53390-P, TEC2014-52289-R, TEC2016-81900-REDT/AEI and TEC2017-83838-R.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author. The material includes a graphical abstract and a README file describing the source code provided with the paper. This material is 2.13 MB in size.



(a) *Trees* (b) *Image* (c) *Video*

(d) *BCI* (e) *SC* (f) *WMC*

Fig. 2. Different UPA examples proposed in the literature. White nodes represent $\mathcal{P}$ nodes and gray nodes represent $\mathcal{U}$ nodes.

depth are assigned to $\mathcal{U}$. This even/odd assignment has been also employed for lifting-based compact image representation (Figure 2(b)): pixels along odd rows or columns are assigned to $\mathcal{P}$ [24]–[26]. Figure 2(c) shows a UPA proposed for lifting-based video coding, where even frames are used to predict the odd frames [27]–[29], and Figure 2(d) shows a UPA for a brain computer interface (BCI) application, where samples at odd time stamps are assigned to $\mathcal{P}$ [30], [31].

While straightforward UPAs have been proposed for these specific examples, optimizing the UPA for arbitrary weighted non-planar graphs of practical interest becomes a complex problem, and just a few solutions have been proposed in the literature. The UPA proposed in [32], [33] to minimize the total energy consumption in a wireless sensor network is equivalent to solving a *Set-Covering* (SC) problem, i.e., minimizing the number of $\mathcal{U}$ nodes while guaranteeing that every $\mathcal{P}$ node has at least one $\mathcal{U}$ neighbor, as in the example in Figure 2(e). [19] and [34] find techniques that minimize the number of discarded edges (i.e., the percentage of direct neighbors in the graph that have the same label), proving that the UPA that minimizes the error between the transform in the original graph and in the simplified graph (i.e., after edges are discarded) corresponds to the solution to the classical *Weighted Maximum-Cut* (WMC) problem (i.e., finding the UPA which maximizes the sum of weights over the edges between $\mathcal{U}$ and $\mathcal{P}$ sets, as in Figure 2(f)). In the same way, [3] proposed to decompose the graph in K bipartite subgraphs and to design two channel wavelet filterbanks in each bipartite graph. Nevertheless, none of these UPAs minimize the detail coefficient energy in order to improve the compaction ability of the transform.

In previous works we proposed UPAs that aim at minimizing the energy of detail coefficients in a lifting-based video encoder [35]–[37]. To this end, given a weighted graph that represents the video signal, we find the UPA that solves the WMC problem which, intuitively, maximizes the correlation between the $\mathcal{P}$ and the $\mathcal{U}$ sets (Figure 2(f)). The same idea was applied for image representation in [38] and for downsampling of signals on graphs [39]. In [40] we introduced a model-based UPA for a specific lifting-based video coding application. We considered a simplified model with only two different edge weights and did not prove optimality of the proposed UPA solution.

C. Contributions

The main contribution in this paper is to formalize the UPA problem in arbitrary weighted graphs and to propose general and novel solutions with a target to minimize the expected value of the detail coefficient energy (i.e., the expected value of the squared prediction error of $\mathcal{P}$ nodes from $\mathcal{U}$ nodes) for a given signal model.

Our approach to the UPA problem can be outlined as follows: consider a graph whose topology and edge weights are known[1] and represented by an adjacency matrix $\mathbf{W}$. We define a moving average (MA) signal model on the graph, which is characterized by a coefficient matrix $\mathbf{Q}$ that captures linear relationships between signal values at different nodes in the graph. Given that the edge weights capture some similarity or correlation measure between signal samples, we assume that simple relations exist between $\mathbf{W}$ and $\mathbf{Q}$, in such a way that the latter can be computed from the former. This is a reasonable assumption because we expect the graph to be informative about the signals observed. For a given UPA, we analyze the design of the $\mathbf{p}$ filters of the lifting transform as a linear estimation problem. As a consequence of this, the matrix of prediction coefficients $\mathbf{P}$ is also computed from $\mathbf{W}$. By analyzing the mean square prediction error of $\mathcal{P}$ nodes from $\mathcal{U}$ nodes, we obtain theoretical conclusions about the optimal UPA and propose a practical implementation using a greedy algorithm.

Note that other statistical models as the well known Gaussian Markov Random Fields (GMRF) [44], [45] can be used to optimize the UPA. Nevertheless, we use MA models because their simplicity facilitate the process of designing lifting transforms, allowing us to obtain the theoretical results presented in the paper.

The optimal UPA will depend on the signal statistics and the graph topology. Some interesting properties arise from the analysis of particular MA models. For instance, we show that, under simplifying assumptions, the best UPA consists of having, at every $\mathcal{P}$ node, a constant degree when counting only its direct $\mathcal{U}$ neighbors. This criterion is shown to be better than alternative ones such as minimizing the number of discarded edges (in unweighted graphs) or maximizing the weight of the cut (in weighted graphs).

Although this work is focused in the UPA in a lifting transform framework, the theoretical results we obtain can be useful in other settings that require bipartite graphs [3], [46], [47]. To the best of our knowledge, this is the first work to optimize the UPA problem on arbitrary graphs (in the sense of minimizing the detail coefficient energy).

The outline for the rest of the paper is as follows. In Section II we present the notation that will be used throughout the paper and overview lifting transforms on graphs. In Section III we formally define the UPA problem and describe our proposed general solution. In Section IV we analyze and

[1]The topology is given by the geometry and the physics of the problem at hand (e.g., sensor networks or community graphs) or selected based on signal characteristics [35]–[37], [40]–[43], and the weights capture similarity between samples and can be defined, for example, as the inverse of the distance, using a Gaussian kernel weighting function [2], or inferred from the data [36], [37].

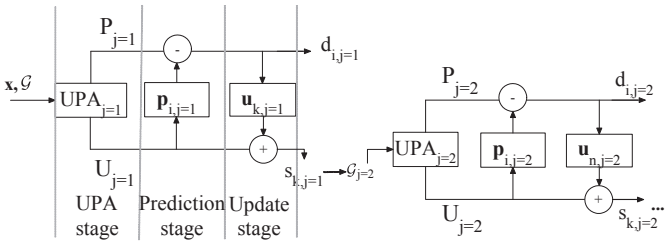


Fig. 3. Lifting scheme. Two levels of decomposition of the forward transform.

obtain optimal UPAs in some particular cases. In Section V we propose a greedy algorithm to obtain a practical solution to the proposed UPA. In Section VI we conduct experimental results that show the compaction ability of the strategy investigated in comparison with state of the art techniques in randomly generated graph signals and real data from image and video. Section VII concludes with some ideas for future work.

II. Preliminaries

A. Notation

A graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V} = \{1, \dots, N\}$ is a set of nodes, $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ a set of edges (or links) between nodes, and $\mathbf{W} = [w_{mn}]$ is the weighted adjacency matrix, with $w_{mn} \in \mathbb{R}^+$ representing the weight of the edge $mn \in \mathcal{E}$ ($w_{mn} = 0$ if $mn \notin \mathcal{E}$ and $w_{mn} = 1\ \forall mn \in \mathcal{E}$ for unweighted graphs). In the present work we consider arbitrary, undirected graphs. The order of the graph is the number of nodes, $N = |\mathcal{V}|$. $\mathcal{N}_m = \{n \in \mathcal{V} : mn \in \mathcal{E}\}$ is the set of neighbors of node m, and $\mathcal{N}_{[m]}$ is its closed neighborhood set ($\mathcal{N}_{[m]} = \mathcal{N}_m \cup \{m\}$). For any partition of $\mathcal{V}$ into two disjoint sets, $\mathcal{U}$ refers to the set of update nodes and $\mathcal{P}$ to the set of prediction nodes. The number of $\mathcal{U}$ neighbors of node $i \in \mathcal{P}$ is defined as $m_i = |\mathcal{N}_i \cap \mathcal{U}|$. The degree of a node m, D_m, is the sum of weights of all its incident edges (i.e., the number of neighbors if the graph is unweighted), $D_m = \sum_{n \in \mathcal{N}_m} w_{mn}$. A graph signal $\mathbf{x} = [x_1, x_2, \dots, x_m, \dots, x_N]$ is a signal defined on $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ so that x_m is the value associated to node $m \in \mathcal{V}$.

We use the index i for nodes in set $\mathcal{P}$, k (or h) for nodes in set $\mathcal{U}$, f and g for indexing different kinds of links, and j for indexing the level of the transform, while m and n are general indexes. Vectors and matrices are written in boldface letters, $\mathbf{I}$ is the identity matrix, $\mathbf{e}_i = (0, \dots, 0, 1, 0, \dots, 0)$ is the natural basis vector with a single 1 in position i, and $\mathbf{1}$ is a vector with all components equal to one. The dimension of these vectors will be clear from the context. For any sets of indexes $\mathcal{I}$ and $\mathcal{J}$ and any matrix $\mathbf{M}$, $\mathbf{M}_{\mathcal{I}}$ is the submatrix of $\mathbf{M}$ resulting from taking the rows in $\mathcal{I}$ and $\mathbf{M}_{\mathcal{I}\mathcal{J}}$ is the submatrix of $\mathbf{M}$ resulting from taking the rows in $\mathcal{I}$ and the columns in $\mathcal{J}$. When the index set is a singleton, $\{i\}$, we will write $\mathbf{M}_i$ instead of $\mathbf{M}_{\{i\}}$ for simplicity.

B. Lifting Transforms on Arbitrary Graphs

Lifting transforms on arbitrary graphs were initially proposed by [18], [19] and [20]. Given a graph signal $\mathbf{x}$ defined on $\mathcal{G}$, lifting is specified by three main stages (see Figure 3): (i) an UPA stage, which finds a bipartition of the graph so that the input node set at each specific level of decomposition j (s_{j-1}) is split into *prediction* ($\mathcal{P}_j$) and *update* ($\mathcal{U}_j$) sets; (ii) a *prediction stage*, where every sample $s_{i,j-1} \in \mathcal{P}_j$ is predicted from an arbitrary number of $\mathcal{U}_j$ neighbors using the $\mathbf{p}_{i,j}$ filter, yielding the detail coefficient $d_{i,j}$; and (iii) an *update stage*, where every sample $s_{k,j-1} \in \mathcal{U}_j$ is filtered with the $\mathbf{u}_{k,j}$ filter using an arbitrary number of $\mathcal{P}_j$ neighbor detail coefficients, giving rise to the smooth coefficient $s_{k,j}$. Mathematically, lifting on graphs can be written as:

$$\begin{aligned} d_{i,j} &= s_{i,j-1} - \sum_{k \in \mathcal{N}_{i,j} \cap \mathcal{U}_j} p_{i,k,j}\ s_{k,j-1} = s_{i,j-1} - \hat{s}_{i,j-1}, \\ s_{k,j} &= s_{k,j-1} + \sum_{i \in \mathcal{N}_{k,j} \cap \mathcal{P}_j} u_{k,i,j}\ d_{i,j}, \end{aligned} \tag{1}$$

where $p_{i,k,j}$ (resp. $u_{k,i,j}$) is the value of the $k{-}th$ (resp. $i{-}th$) position in the $\mathbf{p}_{i,j}$ (resp. $\mathbf{u}_{k,j}$) filter. Inverting the operations of the forward transform to obtain the inverse transform is straightforward from (1) as long as s_{j-1} can be recovered from $d_{i,j}$ and $s_{k,j}$ (because only connections between $\mathcal{U}$ and $\mathcal{P}$ nodes are used for filtering).

Without loss of generality, throughout this paper we focus the discussion on a specific level of the transform and we omit the subscript j for notational simplicity, so that $\mathbf{x} = \mathbf{s}_j$ is the graph signal at j and, for every $i \in \mathcal{P}_j$, detail coefficients (1) are written as

$$d_i = x_i - \sum_{k \in \mathcal{N}_i \cap \mathcal{U}} p_{i,k}\ x_k = x_i - \hat{x}_i. \tag{2}$$

III. Optimized UPA for Lifting Transforms on Graphs

A. Problem Formulation and Prediction Error

Given a weighted graph $\mathcal{G}$, a random signal $\mathbf{x}$ following a statistical model defined on $\mathcal{G}$, and a predictor $\hat{x}_i$ (2) for each node $i \in \mathcal{P}$, we define the total prediction error (E_{tot}) as the sum of the expected value of the squared prediction error (the detail coefficient energy) over the $\mathcal{P}$ nodes. Our goal is then to solve the following problem:

Problem III.1. ***UPA Problem:***
Find the UPA that minimizes the total prediction error

$$E_{tot} = \sum_{i \in \mathcal{P}} \mathbb{E}\{(x_i - \hat{x}_i)^2\} = \sum_{i \in \mathcal{P}} \mathbb{E}\{d_i^2\} \tag{3}$$

for a given number of $\mathcal{P}$ nodes ($|\mathcal{P}|$):

$$\arg\min_{\mathcal{U}/\mathcal{P}} E_{tot},\ \textit{subject to}\ |\mathcal{P}| = T. \tag{4}$$

Fixing $|\mathcal{P}|$ is important because E_{tot} is minimized by minimizing the size of $\mathcal{P}$. Thus, solving (4) is practical only if some constraint on the size of $\mathcal{P}$ is introduced. In practice, $|\mathcal{P}|$ should be found depending on the application and the metric used (e.g., finding the $|\mathcal{P}|$ optimal that minimizes the distortion for a given rate in a compression application).

The solution to the UPA problem will depend on the stochastic process generating the observed signal, $\mathbf{x}$. In the following, we will assume a moving average model.

Definition III.1. General Moving Average (MA) Model

The MA model for a signal $\mathbf{x}$ *defined on* $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ *is*

$$x_m = c + \sum_{n \in \mathcal{N}_{[m]}} q_{m,n}\epsilon_n + \eta_m \tag{5}$$

for some coefficients $q_{m,n}$*, where* c *is a constant, and* ϵ_n *and* η_m *are zero-mean independent random variables, with variances* σ_ϵ^2 *and* σ_η^2*, respectively, for any general node* $m \in \mathcal{V}$*.*

Equation (5) can be expressed in vector form as

$$\mathbf{x} = c\mathbf{1} + \mathbf{Q}\boldsymbol{\epsilon} + \boldsymbol{\eta}, \tag{6}$$

where $\mathbf{Q}$ is a matrix with components $q_{m,n}$ for $n \in \mathcal{N}_{[m]}$ and zero otherwise. Model coefficients $q_{m,n}$ can be computed from the weights of the graph $w_{m,n}$, as will be discussed in Section III-B. Next we define general linear predictors that will be driven by the graph in the sense that prediction $\hat{x}_i$ of x_i depends on the $\mathcal{U}$-neighbors of node i in $\mathcal{G}$.

Definition III.2. General Linear Predictor

Assume a given $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$*,* $\mathbf{x}$*, and UPA. General linear predictors are defined as:*

$$\hat{x}_i = \sum_{k \in \mathcal{N}_i \cap \mathcal{U}} p_{i,k} x_k. \tag{7}$$

Linear predictors can be expressed in vector form as:

$$\hat{\mathbf{x}} = \mathbf{P}\mathbf{x}, \tag{8}$$

where $\mathbf{P}$ is a matrix with components $p_{i,j}$ for $j \in \mathcal{N}_i \cap \mathcal{U}$ and zero otherwise, and $\mathcal{N}_i$ is the set of neighbors of node i in $\mathcal{G}$.

Lemma III.1. MA Prediction Error

For a node $i \in \mathcal{P}$*, let us assume a signal* $\mathbf{x}$ *defined on* $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$*, with* x_i *and* $\hat{x}_i$ *satisfying Definition III.1 and III.2 respectively. The total prediction error is given by*

$$E_{tot} = \sum_{i \in \mathcal{P}} \mathbb{E}\{(x_i - \hat{x}_i)^2\} = \sum_{i \in \mathcal{P}} E_{MA_i} \tag{9}$$

where

$$\begin{aligned} E_{MA_i} &= \mathbb{E}\{x_i^2\} - 2\mathbf{p}_i^\intercal \mathbf{K}_{\mathcal{U}_i i} + \mathbf{p}_i^\intercal \mathbf{K}_{\mathcal{U}_i\mathcal{U}_i}\mathbf{p}_i \\ &= c^2 + \mu_{i,i} + \sigma_\eta^2 - 2\mathbf{p}_i^\intercal(\mathbf{M}_{\mathcal{U}_i}\mathbf{e}_i + c^2\mathbf{1}) \\ &\quad + \mathbf{p}_i^\intercal(\mathbf{M}_{\mathcal{U}_i\mathcal{U}_i} + \sigma_\eta^2\mathbf{I} + c^2\mathbf{1}\mathbf{1}^\intercal)\mathbf{p}_i, \end{aligned} \tag{10}$$

with $\mathcal{U}_i = \mathcal{N}_i \cap \mathcal{U}$ *(i.e., the set of* $\mathcal{U}$*-neighbors of node i),* $\mathbf{K} = \mathbb{E}\{\mathbf{x}\mathbf{x}^\intercal\}$ *is the correlation matrix, column vector* $\mathbf{p}_i$ *has components* $p_{i,k}$*,* $k \in \mathcal{N}_i \cap \mathcal{U}$*, and*

$$\mathbf{M} = \sigma_\epsilon^2 \mathbf{Q}\mathbf{Q}^\intercal \tag{11}$$

is a matrix with components

$$\mu_{m,n} = \sigma_\epsilon^2 \sum_{\ell \in \mathcal{N}_{[m]} \cap \mathcal{N}_{[n]}} q_{m,\ell} q_{n,\ell}. \tag{12}$$

The proof is straightforward and is omitted.

Note that $\mu_{m,n}$ is the (m,n) element of $\mathbf{M}$, and can be interpreted as a measure of correlation (scaled by σ_ϵ^2) between nodes m and n. In particular, note that if nodes m and n are not neighbors and have no common neighbors, then $\mu_{m,n} = 0$.

The first conclusion that can be extracted from (10) is that the UPA minimizing E_{tot} depends on the correlation between nodes along the graph through $\mathbf{M}$. Note also that for a given i, E_{MA_i} is lower as the correlation between i and its $\mathcal{U}$ neighbors is higher, and as the correlation between $\mathcal{U}_i$ neighbors is lower, which is reasonable from the prediction theory point of view, as correlated $\mathcal{U}$ neighbors contribute with similar information to predict i.

For a fixed UPA, the optimal predictor for node i can be obtained as a function of the graph topology and model coefficients by minimizing E_{MA_i}:

$$\begin{aligned} \mathbf{p}_i^* =& \mathbf{K}_{\mathcal{U}_i\mathcal{U}_i}^{-1}\mathbf{K}_{\mathcal{U}_i i} \\ =& (\mathbf{M}_{\mathcal{U}_i\mathcal{U}_i} + \sigma_\eta^2\mathbf{I} + c^2\mathbf{1}\mathbf{1}^\intercal)^{-1}(\mathbf{M}_{\mathcal{U}_i}\mathbf{e}_i + c^2\mathbf{1}). \end{aligned} \tag{13}$$

Note that the optimal prediction coefficients are proportional to the correlation of node i with its $\mathcal{U}$ neighbors, and inversely proportional to the correlation between these neighbors.

B. Graph weights and model parameters

The MA model in (5) assumes that the graph supporting the signal model is exactly the same than the given weighted graph. Using a single graph is consistent with our practical approach to the UPA problem: we will assume that the model parameters $q_{m,n}$ are unknown and that they can be approximated from the graph weights $w_{m,n}$. In particular, if the graph weights are not normalized and can take any value in $\mathbb{R}^+$, we define the model coefficients from the weights as:

$$q_{m,n} = \frac{w_{m,n}}{\sum_{n' \in \mathcal{N}_{[m]}} w_{m,n'}}, \forall n \in \mathcal{N}_{[m]}. \tag{14}$$

If the graph has no self-loops (i.e., $w_{m,m} = 0$ for all m), which is frequent in practice, we define:

$$\begin{aligned} q_{m,m} &= \frac{\sum_{n \in \mathcal{N}_m} w_{m,n}}{|\mathcal{N}_m|} \frac{1}{\sum_{n' \in \mathcal{N}_m} w_{m,n'}\,(1 + 1/|\mathcal{N}_m|)}, \\ q_{m,n} &= w_{m,n} \frac{1}{\sum_{n' \in \mathcal{N}_m} w_{m,n'}\,(1 + 1/|\mathcal{N}_m|)}, \end{aligned} \tag{15}$$

so that coefficients $q_{m,m}$ average the weight over all neighbors. The normalization factors assure that $\sum_{n \in \mathcal{N}_{[m]}} q_{m,n} = 1$.

Note that this choice is up to some extent arbitrary and its effectiveness can be application dependent. Despite of this, if the weights of the graph capture some similarity measurement between nodes, they can be expected to have some correlation with the MA model coefficients. We will analyze this approach in an image/video coding scenario and in randomly generated graphs where the weights are inversely proportional to the distance between nodes.

IV. Case Studies

Even though (10) provides a closed form expression for the prediction error, it is not practical for extracting theoretical conclusions about the optimal UPA. In this section we analyze the selection of the $\mathcal{U}/\mathcal{P}$ partition under several specific scenarios derived from MA. Besides, for simplicity, we will define suboptimal unbiased predictors from $\mathcal{G}$ and the model

coefficients, but that have been proven to be near-optimal in the sense of minimal prediction error [37].

Definition IV.1. Unbiased Predictor

Given a weighted graph $\mathcal{G}$ and an UPA, an unbiased predictor for a MA model with coefficients $q_{m,n}$ is a general linear predictor (Definition III.2) with:

$$p_{i,k} = \frac{q_{i,k}}{\sum_{k' \in \mathcal{N}_i \cap \mathcal{U}} q_{i,k'}}. \tag{16}$$

Note that, for the weight-based model in (14) or in (15), predictions (7) based on (16) can be expressed as

$$\hat{x}_i = \frac{1}{\sum_{k \in \mathcal{N}_i \cap \mathcal{U}} w_{i,k}} \sum_{k \in \mathcal{N}_i \cap \mathcal{U}} w_{i,k} x_k. \tag{17}$$

According to the MA model in (5), $\mathbb{E}\{\hat{x}_i\} = \left(\sum_{k \in \mathcal{N}_i \cap \mathcal{U}} w_{i,k}\right)^{-1} \sum_{k \in \mathcal{N}_i \cap \mathcal{U}} w_{i,k} c = c$ and, thus, the predictor is statistically unbiased.

In particular, if graph $\mathcal{G}$ is unweighted, $p_{i,k} = 1/m_i$ (i.e., $\mathbf{p}_i = \frac{1}{m_i}\mathbf{1}$), with $m_i = |\mathcal{N}_i \cap \mathcal{U}|$, and, thus,

$$\hat{x}_i = \frac{1}{m_i} \sum_{k \in \mathcal{N}_i \cap \mathcal{U}} x_k. \tag{18}$$

A. Homogeneous Noise Model

First, we consider a very simplistic model in which the value of every node is a noisy version of some constant. Despite its simplicity, it will be useful to infer some interesting properties of a good UPA in some types of graphs.

Definition IV.2. Homogeneous Noise Model

A homogeneous noise model (HNM) is a general MA with $q_{m,n} = 0$, so that

$$x_m = c + \eta_m. \tag{19}$$

The HNM can give us an intuition about the optimal UPA when the samples of the graph signal are independent (i.e., x_m does not depend on neighbor nodes). In this situation, the links of the graph are only used to construct the predictors.

Proposition IV.1. Optimal UPA for HNM

Let us assume an unweighted $\mathcal{G}$ with x_i and $\hat{x}_i$ satisfying Definition IV.2 and Definition IV.1 respectively. For a fixed $|\mathcal{P}|$ and a fixed weight of the cut $W = \sum_{i \in \mathcal{P}} m_i$, any UPA satisfying that all $\mathcal{P}$ nodes have the same number of $\mathcal{U}$ neighbors (same degree with its $\mathcal{U}$ neighbors) is optimal in the sense of minimizing the total prediction error.

The proof is in Appendix A. Note that W is the number of links between $\mathcal{U}$ and $\mathcal{P}$, and Proposition IV.1 implies that, if the signal components are independent, the optimal UPA distributes these links equally amongst all the $i \in \mathcal{P}$ nodes, i.e., $m_i = W/|\mathcal{P}|$.

Although Proposition IV.1 is restricted to a particular model (HNM) and a specific filter, it unveils a property of the $\mathcal{U}/\mathcal{P}$ partition that could be useful for other scenarios. For instance, even though a constant (unless for the noise) model like the HNM may be unrealistic, it can be a useful local approximation for some graphs with links based on similarity measures between nodes (as, for instance, in image or video coding). This suggests an UPA algorithm that splits the graph in subgraphs and searches for $\mathcal{U}/\mathcal{P}$ partitions satisfying Proposition IV.1 inside each subgraph.

Also, it can be shown that the best UPA for the unbiased filter is also the best UPA for the optimal predictor in (13) for the HNM (the proof is omitted). This supports our seminal idea that reasonable choices of the prediction filters for the weighted graph may be useful to find good (if not optimal) solutions to the UPA problem.

B. Smooth Unweighted Model

In this section we propose a signal model for unweighted $\mathcal{G}$ that considers smooth data value variations between neighbors.

Definition IV.3. Smooth Unweighted Model

A smooth unweighted model (SUM) is a general MA given by (5) *with $c = 0$.* [2] *Furthermore, from* (15) *and given that $\mathcal{G}$ is unweighted, $q_{m,n} = q_{m,m} = \frac{1}{|\mathcal{N}_{[m]}|}$, so that*

$$x_m = \frac{1}{|\mathcal{N}_{[m]}|} \sum_{n \in \mathcal{N}_{[m]}} \epsilon_n + \eta_m. \tag{20}$$

From the unweighted $\mathcal{G}$, unbiased unweighted predictors can be defined as in (18). Note that $p_{i,k}$ and $q_{i,k}$ are related through the unweighted $\mathcal{G}$ to which they are linked, having that for node i, the difference between $q_{i,k}$ coefficients of the model and $p_{i,k}$ coefficients of the predictor is just the normalization factor ($1/\mathcal{N}_{[i]}$ and $1/m_i$ respectively).

Define the *clustering degree* of nodes m, n on graph $\mathcal{G}$ as

$$c(m,n) = \frac{|\mathcal{N}_{[m]} \cap \mathcal{N}_{[n]}|}{|\mathcal{N}_{[m]}||\mathcal{N}_{[n]}|}. \tag{21}$$

Intuitively, $c(m,n)$ is related to the proportion of neighbours shared by m and n, and coefficients $\mu_{m,n}$ in matrix $\mathbf{M}$ are given by $\mu_{m,n} = \sigma_\epsilon^2 c(m,n)$. Next, we calculate the expected value of the prediction error assuming the SUM (20) and using the unweighted predictor defined in (18).

Lemma IV.1. Smooth Unweighted Model Prediction Error

Consider the unbiased predictor (18) *for a signal $\mathbf{x}$ defined over an unweighted graph $\mathcal{G}$ and driven by the SUM (Definition IV.3). The prediction error at node $i \in \mathcal{P}$ is given by*

$$\begin{aligned} E_{SUM_i} &= \mathbb{E}\{(x_i - \hat{x}_i)^2\} \\ &= \mathbb{E}\{(x_i)^2\} + \mathbb{E}\{(\hat{x}_i)^2\} - 2\mathbb{E}\{x_i \hat{x}_i\} \\ &= \underbrace{\sigma_\eta^2 + \frac{\sigma_\epsilon^2}{|\mathcal{N}_{[i]}|}}_{\mathbf{A}} + \underbrace{\frac{\sigma_\eta^2}{m_i} + \frac{\sigma_\epsilon^2}{m_i^2} \sum_{k \in \mathcal{N}_i \cap \mathcal{U}} \sum_{h \in \mathcal{N}_i \cap \mathcal{U}} c(k,h)}_{\mathbf{B}} \\ &\quad \underbrace{- 2\,\frac{\sigma_\epsilon^2}{m_i} \sum_{k \in \mathcal{N}_i \cap \mathcal{U}} c(i,k)}_{\mathbf{C}}. \end{aligned} \tag{22}$$

The proof is in Appendix B. Component **A** in (22) represents the variance of the observation, $\mathbb{E}\{(x_i)^2\}$, component **B**

[2] Note that, considering normalized predictors, $\sum_{k \in \mathcal{N}_i \cap \mathcal{U}} p_{i,k} = 1$, and thus the constant c does not have influence on the prediction error, as can be derived from (10).

represents the variance of the predictor, $\mathbb{E}\{(\hat{x}_i)^2\}$, and component **C** represents the cross-correlation $\mathbb{E}\{x_i\hat{x}_i\}$. Interestingly, the variance of the predictor decreases as the number of $\mathcal{U}$ neighbors of node i (m_i) increases (i.e., the prediction is based on more variables) and as the clustering degrees $c(k,h)$ between $\mathcal{U}$ neighbors of node i decrease. Thus, it is of interest to have many uncorrelated $\mathcal{U}$ neighbors of i to perform its prediction. Note that, if correlation between node i and its $\mathcal{U}$ neighbors ($c(i,k)$) and between $\mathcal{U}$ neighbors of i ($c(k,h)$) are constants, the optimal UPA is the same as for the HNM. This will be formally stated in Corollary IV.1.

C. Discrete Number of Weight Values

A weighted graph $\mathcal{G}$ with discrete differentiated weights arises in some applications where the graph links may be of a few different types (e.g., in the video representation used in [35], every spatial (resp. temporal) neighbor is associated with a specific spatial (resp. temporal) $w_{m,n}$). In general, we consider F different kinds of neighbors of class $\mathcal{C}_f$ with weights w^f_{mn} (hereafter w_f) if $mn \in \mathcal{C}_f$, and with $\sum_f w_f = 1$. Note that $\bigcup_{f\in\{1,2,...,F\}} \mathcal{C}_f = \mathcal{E}$. In this situation, we can assume that the variance of every node m, the correlation between every pair of $\mathcal{C}_f$ neighbors of m, and the correlation between m and its $\mathcal{C}_f$ neighbors are constant:

$$\mathbb{E}\{(x_m)^2\} = \sigma^2, \tag{23}$$
$$\mathbb{E}\{(x_n x_l)\} = \begin{cases} \alpha^f, & \text{if } n,l \in \mathcal{N}_m; mn, ml \in \mathcal{C}_f \\ \alpha^{fg} & \text{if } n,l \in \mathcal{N}_m; mn \in \mathcal{C}_f, ml \in \mathcal{C}_g, \end{cases}$$
$$\mathbb{E}\{(x_m x_n)\} = \gamma^f, \text{if } n \in \mathcal{N}_m, mn \in \mathcal{C}_f.$$

For simplicity of notation, hereafter we refer $n \in \mathcal{N}^f_i$ to nodes that fulfill that $n \in \mathcal{N}_i$ and $ni \in \mathcal{C}_f$. In the discrete number of weights case, unbiased weighted predictors are defined by considering the linear prediction filters of Definition III.2 with:

$$p^f_{i,k} = \frac{w_f}{m^f_i}, \tag{24}$$

where $k \in \mathcal{N}^f_i \cap \mathcal{U}$ and $m^f_i = |\mathcal{N}^f_i \cap \mathcal{U}|$, so that

$$\hat{x}_i = \sum_f \frac{w_f}{m^f_i} \sum_{k\in\mathcal{N}^f_i\cap\mathcal{U}} x_k. \tag{25}$$

Proposition IV.2. Optimal UPA for MA, Assuming (23)

Let us assume a weighted $\mathcal{G}$ with x_i satisfying Definition III.1 (MA model) and assumptions in (23) *and $\hat{x}_i$ defined in* (25)*. For a fixed $|\mathcal{P}|$ and number of links between $\mathcal{P}$ and $\mathcal{U}$ sets $W_c = \sum_{i\in\mathcal{P}} m_i = \sum_{i\in\mathcal{P}}\sum_f m^f_i$, the optimal UPA is obtained when all the $\mathcal{P}$ nodes have the same proportion of $\mathcal{C}_f$ $\mathcal{U}$ neighbors (i.e., $m^{f^*}_i = K_f$, where K_f is a constant that does not depend on i). The optimal proportion depends on w_f (the higher w_f, the higher proportion of $\mathcal{C}_f$ $\mathcal{U}$ neighbors), and the correlation between $\mathcal{C}_f$ $\mathcal{U}$ neighbors of i. Specifically:*

$$m^{f^*}_i = K_f = \frac{W_c}{|\mathcal{P}|}\frac{w_f A^f}{\sum_g w_g A^g}, \tag{26}$$

where $g \in \{1,2,...,F\}$ and $A^g = \sqrt{\sigma^2 - \alpha^g}$.

The proof is in Appendix C. Next, we outline some conclusions derived from Proposition IV.2.

- If α^f is high, every $\mathcal{N}^f_i \cap \mathcal{U}$ provides similar information to predict i. Thus, the optimal number of m^f_i decreases with α^f. Furthermore, it increases with w_f. Note that given (23) and that $\mathbb{E}\{x_m x_n\} \leq \sqrt{\mathbb{E}\{(x_m)^2\}\mathbb{E}\{(x_n)^2\}}$, $\sigma^2 - \alpha^f \geq 0$.
- Equation (26) implies that to obtain the optimal UPA, every node i should have constant degree with its $\mathcal{U}$ neighbors, $D_i = \sum_f \sum_{k\in\mathcal{N}^f_i\cap\mathcal{U}} w_f = \sum_f K_f w_f = D$.
- WMC maximizes the cut between $\mathcal{U}$ and $\mathcal{P}$ sets, which could lead to some $\mathcal{P}$ nodes having a large number of $\mathcal{U}$ correlated neighbors, while other nodes may not have correlated neighbors, giving rise to good and bad predictions respectively. Under assumption (23), every node should have a balanced number of correlated neighbors, which globally improves the mean prediction error over all nodes in $\mathcal{P}$.

Corollary IV.1. Optimal UPA for SUM, Assuming (23)

Let us assume an unweighted $\mathcal{G}$ with x_i satisfying Definition IV.3 (SUM) and assumptions in (23) *and $\hat{x}_i$ satisfying Definition IV.1. For a fixed $|\mathcal{P}|$ and a fixed weight of the cut $W = \sum_{i\in\mathcal{P}} m_i$, the optimal UPA is obtained when all the $\mathcal{P}$ nodes have the same number of $\mathcal{U}$ neighbors.*

The proof is straightforward from Proposition IV.2.

V. Proposed Greedy Solution

Solving Problem III.1 implies minimizing E_{tot} for a given $|\mathcal{P}|$. A brute-force solution is infeasible because one should try every possible UPA and calculate the squared error over all the $\mathcal{P}$ nodes for each of these assignments. Thus, we design a greedy algorithm that locally minimizes E_{tot} (9) in each iteration[3]. Specifically, the algorithm starts with all graph nodes as $\mathcal{P}$ nodes and, in each iteration t, the algorithm changes to $\mathcal{U}$ the candidate node $c^* \in \mathcal{P}$ that minimizes E_{tot}.

Let us define $\Theta^{\{t,c\}}$ as the E_{tot} difference between iterations $t-1$ and t if node c is changed (i.e., $\Theta^{\{t,c\}} = E^{\{t-1\}}_{tot} - E^{\{t,c\}}_{tot}$). Note that, as E_{tot} decreases when we incorporate more $\mathcal{U}$ nodes, we have that $E^{\{t-1\}}_{tot} > E^{\{t,c\}}_{tot}$, and so $\Theta^{\{t,c\}} > 0$. Therefore:

$$c^* = \arg\min_{c\in\mathcal{P}} E^{\{t,c\}}_{tot} = \arg\max_{c\in\mathcal{P}} \Theta^{\{t,c\}}. \tag{27}$$

The advantage of maximizing $\Theta^{\{t,c\}}$ instead of minimizing $E^{\{t,c\}}_{tot}$ is that now we can find c^* just by observing how E_{tot} changes in the neighborhood of every candidate node c (specifically, looking at two-hop neighbors), and thus without having to evaluate E_{tot} summing over all nodes $i \in \mathcal{P}$ for every c. Note that $E^{\{t-1\}}_{tot}$ and $E^{\{t,c\}}_{tot}$ (and thus $\Theta^{\{t,c\}}$) can be calculated for every c with a few operations on small matrices (adjacency matrices of the two-hop neighborhood of c). Vector $\mathbf{\Theta}^{\{t,c\}} = [\Theta^{\{t,c=1\}}, \Theta^{\{t,c=2\}}, \ldots, \Theta^{\{t,c=|\mathcal{P}|^{\{t\}}\}}]$ is calculated sweeping through all candidate nodes, and the best node c^*

[3]The source code associated with this paper is available from **https://github.com/EduardoMartinezEnriquez**

is obtained searching the maximum of $\mathbf{\Theta}^{\{t,c\}}$. The complete greedy approach is summarized in Algorithm 1.

Algorithm 1 Proposed UPA Greedy Algorithm

Require: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and predictor definitions, $|\mathcal{P}|$ nodes, $\mathcal{U} = \{\varnothing\}$, $\mathcal{P} = \{\mathcal{E}\}$

Calculate $\mathbf{M}$ (11)
Obtain two-hop neighbors for every node $\in \mathcal{V}$
while $|\mathcal{P}|^{\{t\}} > |\mathcal{P}|$ **do**
  for $c = 1$ to $c = |\mathcal{P}|^{\{t\}}$ **do**
    Calculate $\Theta^{\{t,c\}}$ in the neighborhood of c
  end for
  Calculate $\mathbf{\Theta}^{\{t,c\}}$
  Select the node c^* with maximum $\Theta^{\{t,c\}}$, $c^* = \arg\max_{c \in \mathcal{P}} \mathbf{\Theta}^{\{t,c\}}$
  Let $\mathcal{U} \leftarrow \mathcal{U} \cup \{c^*\}$
  Let $\mathcal{P} \leftarrow \mathcal{P} \backslash \{c^*\}$
end while
return UPA

Note that the algorithm stops when the target number $|\mathcal{P}|$ of nodes in Problem III.1 is reached.

VI. Experiments

In this section we provide experimental results in randomly generated graph signals and in real data from image and video signals, comparing the compaction ability of the UPA technique investigated in Section III, the WMC solution, and a random UPA assignment. The WMC solution is obtained using the greedy approach described in [48], which basically starts with all graph nodes as $\mathcal{P}$ nodes, and in each iteration it moves the $\mathcal{P}$ node with largest degree to $\mathcal{U}$. The proposed MA solution is obtained using Algorithm 1. The random UPA starts with all nodes as $\mathcal{P}$ nodes, and in each iteration of the algorithm moves to $\mathcal{U}$ a $\mathcal{P}$ node randomly chosen.

To evaluate the performance of each approach, we measure the root mean square prediction error over all nodes in $\mathcal{P}$ as

$$E_{RMS} = \sqrt{\frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} (x_i - \hat{x}_i)^2} = \sqrt{\frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} (d_i)^2}, \quad (28)$$

where x_i is the actual signal value of the node i, and $\hat{x}_i$ the prediction. To obtain the prediction, we use the $\mathbf{p}$ filters defined in Definition IV.1 in all the methods. E_{RMS} is obtained as a function of the relation $|\mathcal{U}|/N$ selected by the different strategies, where different $|\mathcal{U}|/N$ are obtained by letting the restriction $|\mathcal{P}|$ in Problem III.1 and thus in the greedy algorithms vary. Finally, we compare the compaction ability of the proposed UPA with the filters employed in lifting based image and video encoders (JPEG 2000 and [29] respectively) in a multiresolution framework.

A. Randomly Generated Graph Signals

Figure 4 shows illustrative examples of three different graph topologies, generated with [49], used to test the algorithms, namely: Fig. 4(a) random sensor networks; Fig. 4(b) random community graphs; and Fig. 4(c) Minnesota road network.

In the sensor network and community graphs, for each realization of the experiment, a graph with an user-defined number of nodes N is generated, with random positions of the nodes and links between them. Weights are obtained with a Gaussian kernel of the euclidean distance between nodes. The graph signal is randomly generated following the MA in (5). ϵ_n and η_m are Gaussian, and different values for σ_ϵ^2, σ_η^2 and c are assessed. Coefficients of the model $q_{m,n}$ are obtained from graph weights using (15). In the graphs of Figure 4(a), (b) and (c), the color of the node represents the value of the sample associated to it. Minnesota road network is an unweighted graph with a deterministic structure of 2642 nodes.

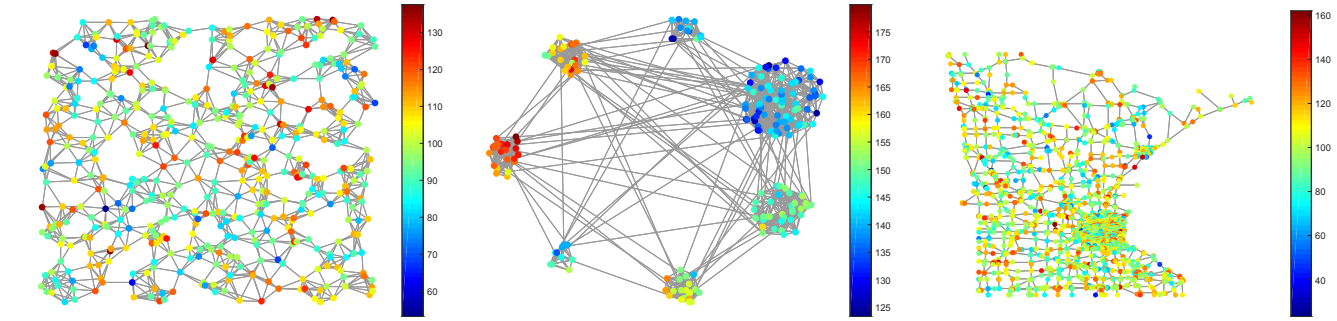
Figures 4 (d, e, f), show E_{RMS} as a function of $|\mathcal{U}|/N$ for the different UPA techniques, for the sensor network, community graph, and Minnesota road examples, respectively, using the values of N, σ_ϵ^2, σ_η^2 and c indicated in the figure caption. Experimental results are obtained averaging 100 realizations of the experiment.

As expected, E_{RMS} decreases as the proposed algorithms choose a higher number of $\mathcal{U}$ nodes, because the obtained predictions are better. The E_{RMS} obtained with the proposed method is lower than with the WMC and random assignments for any proportion of $|\mathcal{U}|$ nodes in the graph. This means that, for any $|\mathcal{P}|$ selected as a restriction in Problem III.1, the average energy of detail coefficients is lower with the proposed method, and thus the compaction ability of the transform is improved. This is especially important in non-dyadic transforms such as lifting on graphs. For example, choosing a 15% of nodes of the graph as $\mathcal{U}$ (and thus decorrelating the signal in the rest 85% of $\mathcal{P}$ nodes) with the proposed UPA algorithm, we obtain the same quality in the prediction than using more than the 30% of nodes of the graph as $\mathcal{U}$ selected with the WMC.

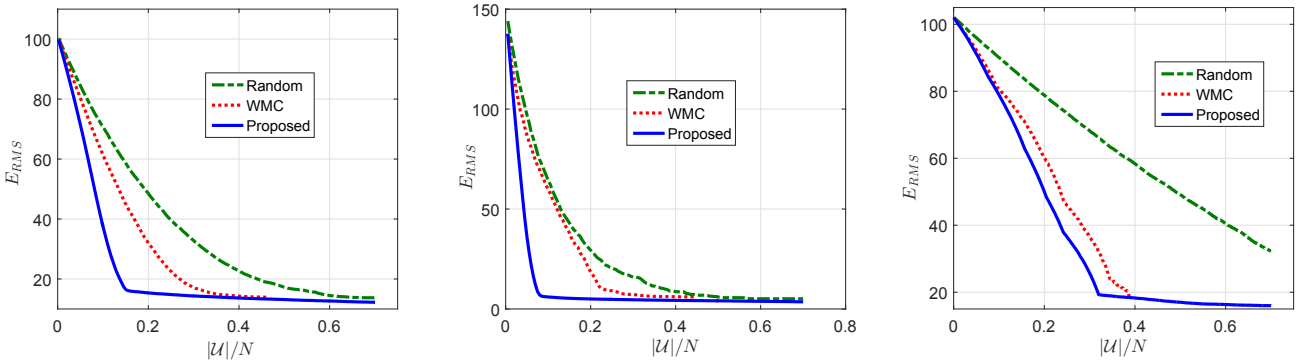
Figure 5 shows the trends for E_{RMS} in the sensor network example when parameters of the graph and the model are changed. Increasing σ_ϵ^2 and σ_η^2 the error curves move up (E_{RMS} increases) because the prediction of the node values becomes harder. This is more pronounced when increasing σ_η^2 than σ_ϵ^2, because σ_η^2 is related with the independent noise for every node. Increasing c implies that the error is higher for low $|\mathcal{U}|/N$, where not every $|\mathcal{P}|$ is predicted, but does not affect when every $|\mathcal{P}|$ has at least one $\mathcal{U}$ neighbor. Changing N does not have a significant influence in the E_{RMS}.

Figure 6 shows the complexity, measured as the time in seconds per $\mathcal{U}$ node selected, of the WMC and the proposed Algorithm 1 as a function of the number of nodes on the graph N. An exhaustive greedy approach with no simplifications, that for every iteration minimizes $E_{tot}^{\{t,c\}}$ instead of maximizing $\Theta^{\{t,c\}}$, is also shown to analyze the computational advantages of working just in the neighborhood of every candidate $\mathcal{P}$ node. Time is measured in a PC Intel Xeon CPU E3@3.5 GHz processor, 4 cores, 8GB RAM. As can be observed, computational time per node increases linearly with N in the proposed approach, and quadratically in the WMC, so that for low N, WMC is faster, but for $N > 8000$ nodes, the proposed approach outperforms WMC. Note also that the complexity of the greedy exhaustive algorithm increases very fast with N.

Finally, Figure 7 shows the mean degree per $\mathcal{P}$ node with its



(a) *Random sensor network with N=500, σ_ϵ^2=400, σ_η^2=100 and c=100.* (b) *Community graph with N=200, σ_ϵ^2=1600, σ_η^2=1 and c=150.* (c) *Minnesota road graph with N=2642, σ_ϵ^2=900, σ_η^2=100 and c=100.*



(d) *Average results for 100 realizations of (a).* (e) *Average results for 100 realizations of (b).* (f) *Results for 100 realizations of (c).*

Fig. 4. Randomly generated graph signals and E_{RMS} as a function of $|\mathcal{U}|/N$ for the proposed, WMC, and random UPAs.

$\mathcal{U}$ neighbors, and standard deviation, as a function of $|\mathcal{U}|/N$ when the UPA is performed using the proposed approach and the WMC solution, for (a) the sensor networks, (b) the community graph and (c) the Minnesota road. In all cases, the mean degree (μ_{Degree}) increases with $|\mathcal{U}|/N$, and is higher with WMC than with the proposed algorithm, which is reasonable because WMC moves to $\mathcal{U}$ the node that maximizes the degree with its $\mathcal{P}$ neighbors in each iteration. Nevertheless, the standard deviation (σ_{Degree}) is significantly lower with the proposed method. The reason is that, despite the optimal UPA depends on the graph topology and weights between nodes, the proposed algorithm makes that for every $\mathcal{P}$ node, the degree with its $\mathcal{U}$ neighbors is similar so that every $\mathcal{P}$ node is well predicted, tending to the conclusion of constant degree with $\mathcal{U}$ neighbors of Propositions IV.2 and IV.1.

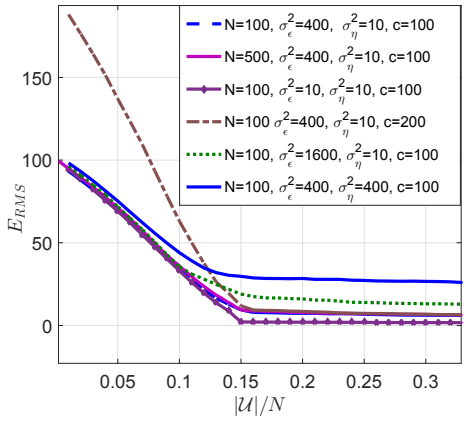


Fig. 5. E_{RMS} as a function of $|\mathcal{U}|/N$ for different graph and signal parameters.

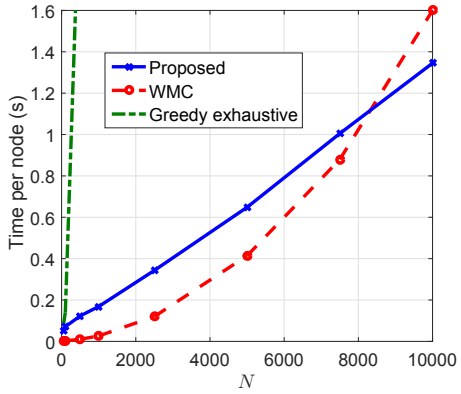


Fig. 6. Computational complexity (s) as a function of the order of the graph N of the greedy approach for the proposed UPA (Algorithm 1), a greedy exhaustive algorithm, and the greedy approach for the WMC proposed in [48].

B. Image and Video Signal

In this section we test the algorithms using real data from different standard image and video test sequences. In the image example, we use the graph representation proposed in [38], i.e., every pixel is linked to its 8 one-hop adjacent neighbors, with Gaussian weights $w_{i,j} = e^{-|x_i - x_j|^2/2\sigma^2}$, where x_i and x_j is the signal value at nodes i and j respectively, and σ the standard deviation estimated from the data. For the video example, we use the graph representation proposed in [35], [40], where every pixel is linked (i) to an arbitrary number of temporal neighbors (i.e., a pixel of frame in time instant t linked to a pixel in frame $t+1$) following a motion estimation process and (ii) to its 8 one-hop spatial neighbors (i.e., pixels of the same frame) that do not cross contours[4](we assume that pixels across contours will likely have very different luminance value). For every frame, weights have a different value for spatial and temporal links, and they are calculated following the process explained in [37], [40].

To reduce the computational time associated with the huge graphs resulting from image and video representations, images/frames are divided in M blocks of constant size, and results are obtained by averaging over the M disjoint subgraphs formed. Other low cost approaches [36], [37] could be used to reduce the computational time.

Figure 8 shows the E_{RMS} for the test images *Lena* (Figure 8(a)), *Cameraman* (Figure 8(b)), and *Barbara* (Figure 8(c)), and the test video sequences *Football* (Figure 8(d)), *Carphone* (Figure 8(e)), and *Tempete* (Figure 8(f)). As with the randomly generated graph signals, for any $|\mathcal{U}|$ the average energy of detail coefficients is lower with the proposed method than with the WMC and the random UPA solutions. To further analyze the implications of the improvement in energy compaction when using the proposed UPA technique, Figure 9 shows *Lena* and *Cameraman* test images reconstructed using the inverse lifting transform from a specific number of $\mathcal{U}$ nodes, setting all detail coefficients in $\mathcal{P}$ nodes to 0. Black pixels are $\mathcal{P}$ nodes that don't have $\mathcal{U}$ neighbors and thus they are not predicted, so that after applying the inverse transform they remain 0. First, second, and third columns show *Lena* reconstructed from $|\mathcal{U}| = 0.05N$, $|\mathcal{U}| = 0.2N$, and $|\mathcal{U}| = 0.4N$ respectively, using the random UPA (first row), WMC (second row), and the proposed UPA (third row). Fourth column shows *Cameraman* reconstructed from $|\mathcal{U}| = 0.3N$ using the random UPA, WMC, and the proposed UPA. Note that for $|\mathcal{U}| = 0.05N$, $|\mathcal{U}| = 0.2N$ and $|\mathcal{U}| = 0.3N$, WMC and random assignment have many non-predicted $\mathcal{P}$ nodes thus not decorrelated. With the WMC, this non-predicted nodes tend to be close to the contours of the image, where weights are low. Consequently, $\mathcal{U}$ nodes are not chosen, and $\mathcal{P}$ nodes remain without $\mathcal{U}$ neighbors. On the contrary, the proposed approach tend to distribute the $\mathcal{U}$ nodes so that more $\mathcal{P}$ nodes can be well predicted, accordingly with conclusion extracted from Figure 7. Interestingly, note that even in areas with all nodes predicted, the subjective quality obtained with the proposed approach significantly outperforms the WMC, especially close to the contours, where the prediction of $\mathcal{P}$ nodes is much better with the proposed method. This fact can be clearly seen comparing Figures 9(g) and (k).

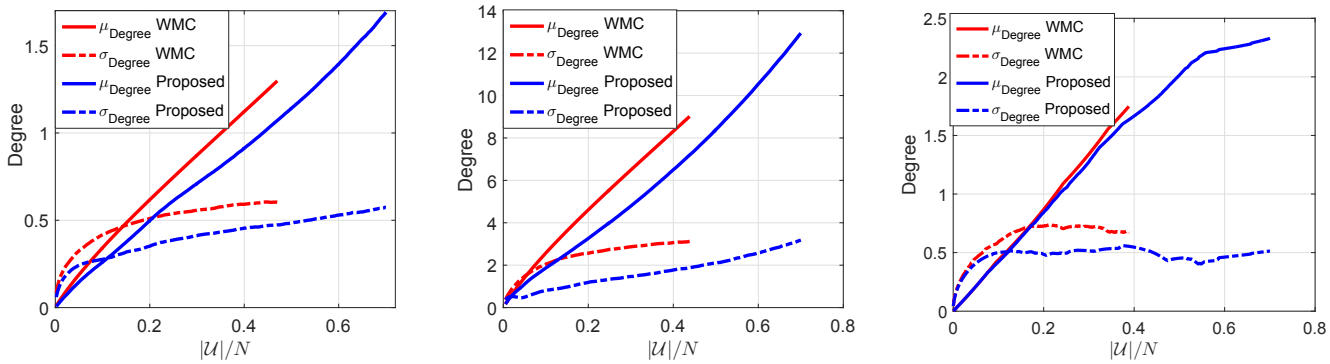
C. Multiresolution and Comparison with Lifting Based Image and Video Encoders

In this section we apply the proposed UPA algorithm at different levels of decomposition to obtain a multiresolution transform, and compare its compaction ability with the transform used in a lifting based image encoder (Daubechies 9/7 filters used in JPEG 2000 [50]) and video encoder (biorthogonal 5/3 filters proposed in [29]). To obtain the graph at transformation level j from the graph at level $j-1$, we use the algorithm explained in detail in [40], Algorithm 2. Basically, we connect those $\mathcal{U}$ nodes that are either directly connected or at two-hop of distance in the graph at level $j-1$, and the corresponding link weight is the product of the weights in the path between connected nodes at level $j-1$. For every graph at the different levels of the transform, the proposed UPA method is applied, fixing the restriction of $|\mathcal{P}|$ nodes in Algorithm 1 to have $|\mathcal{U}|/N = 1/2^j$ in order to obtain a dyadic transform. At every j, the prediction stage of the transform is performed (the update stage was not implemented in this example). In the experiments, four levels of decomposition are performed. Table I shows the E_{RMS} at different levels of decomposition for the test images *Lena*, *Cameraman*, and *Barbara*. Note that applying two levels of decomposition of the bidimensional separable transform used in JPEG 2000 is equivalent to four levels of decomposition of the proposed graph transform ($|\mathcal{U}|/N = 1/16$). Table II shows the E_{RMS} for the test video sequences *Football*, *Carphone*, and *Tempete*. As expected, the E_{RMS} is higher as j increases (i.e., as $|\mathcal{U}|$ is lower and more pixels are decorrelated). Furthermore, the E_{RMS} obtained with the proposed method is lower than with the lifting based JPEG 2000 and with [29] for the same $|\mathcal{U}|/N$ except for the video sequence *Tempete*.

D. Discussion

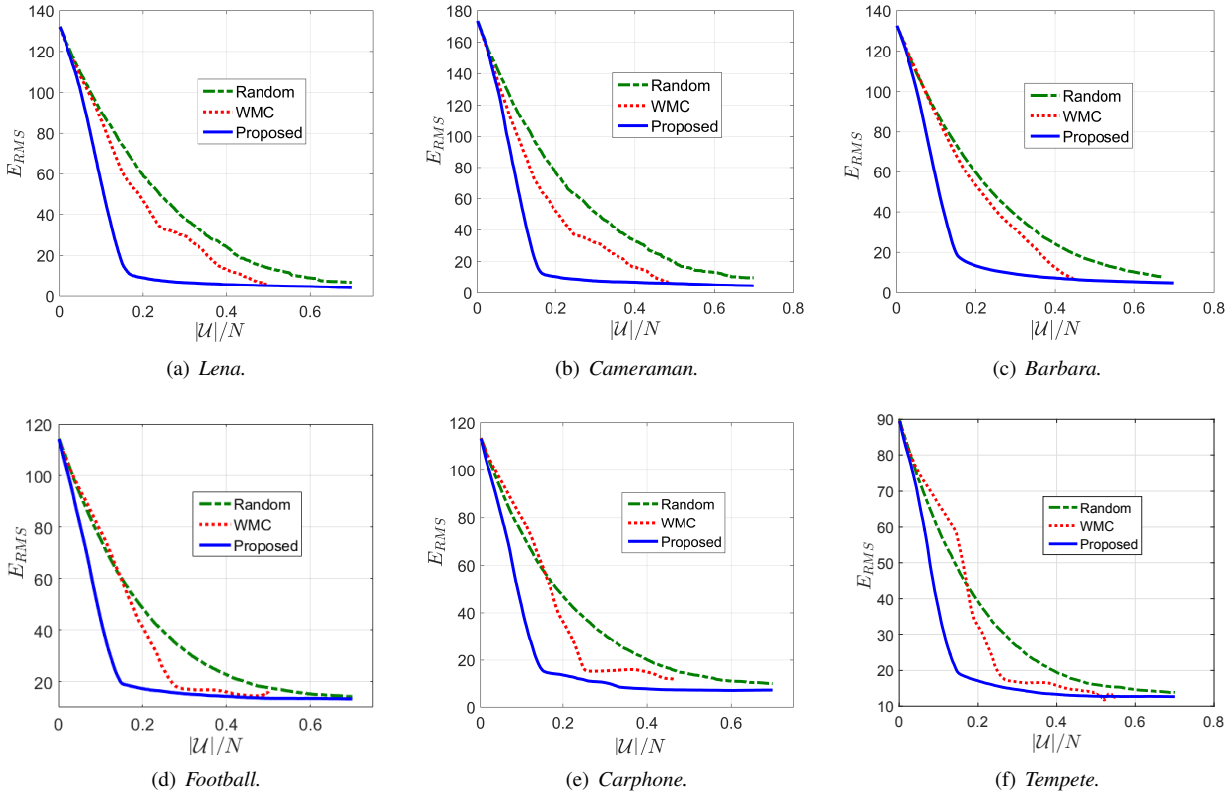
In the experiments with real data from image and video, $\mathbf{x}$ is a deterministic signal, while our solution has been obtained

[4]To avoid confusion we call image "contours" edges that appear in the image between sets of pixels of different intensities, while we reserve the term "edge" for the links between nodes in a graph.



(a) *Random sensor network with N=500,* σ_ϵ^2*=400,* σ_η^2*=100 and c=100.* (b) *Community graph with N=200,* σ_ϵ^2*=1600,* σ_η^2*=1 and c=150.* (c) *Minnesota road graph with N=2642,* σ_ϵ^2*=900,* σ_η^2*=100 and c=100.*

Fig. 7. μ_{Degree} and σ_{Degree} as a function of $|\mathcal{U}|/N$ for different graphs.



(a) *Lena.* (b) *Cameraman.* (c) *Barbara.*

(d) *Football.* (e) *Carphone.* (f) *Tempete.*

Fig. 8. E_{RMS} as a function of $|\mathcal{U}|/N$ in real data from image and video test sequences for the proposed, WMC, and random UPAs.

considering a random signal that follows a specific model. Even if the signal model is a crude approximation to the image or video generation process, the proposed UPA leads to a good energy compaction of the transform, much better than other UPAs as the WMC (Figure 8), so that the theoretical conclusions and the practical solution we propose are useful for different kinds of signals, and they are probably similar to the ones that would be obtained from more complex models as GMRFs. Furthermore, working with random instead of deterministic signals we obtain general solutions that do not depend on the signal itself, which is very useful in different applications such as coding, where an specific strategy is used in encoder and decoder and therefore the UPA is not needed to be sent to the decoder.

VII. Conclusions and Future Work

Lifting transforms on graphs are very useful to obtain an efficient invertible, critically sampled, compact representation of a given graph signal. Nevertheless, their performance depends on different optimization problems which solutions are not trivial in arbitrary graphs. The $\mathcal{U}/\mathcal{P}$ assignment process, which consists of assigning a label to every node on the graph, is one of the most relevant process of the transform. In this paper, we have introduced a theory in the optimization of the $\mathcal{U}/\mathcal{P}$ assignment, finding a general technique that minimizes the expected value of the squared prediction error. Given a weighted $\mathcal{G}$, and considering a moving average signal model and a linear predictor linked to $\mathcal{G}$, we have proven that the optimal assignment depends on the correlation between

(a) $|\mathcal{U}| = 0.05N$. *Random UPA.* (b) $|\mathcal{U}| = 0.2N$. *Random UPA.* (c) $|\mathcal{U}| = 0.4N$. *Random UPA.* (d) $|\mathcal{U}| = 0.3N$. *Random UPA.*

(e) $|\mathcal{U}| = 0.05N$. *WMC UPA.* (f) $|\mathcal{U}| = 0.2N$. *WMC UPA.* (g) $|\mathcal{U}| = 0.4N$. *WMC UPA.* (h) $|\mathcal{U}| = 0.3N$. *WMC UPA.*

(i) $|\mathcal{U}| = 0.05N$. *Proposed UPA.* (j) $|\mathcal{U}| = 0.2N$. *Proposed UPA.* (k) $|\mathcal{U}| = 0.4N$. *Proposed UPA.* (l) $|\mathcal{U}| = 0.3N$. *Proposed UPA.*

Fig. 9. Lena and Cameraman reconstructed using the inverse lifting transform from different $|\mathcal{U}|/N$ and UPA solutions.

nodes on the graph, and is not the one that minimize the edges discarding or that maximize the weight of the cut in unweighted and weighted graphs respectively.

Finally, we have experimentally validated this conclusion by evaluating the detail coefficient energy of the proposed UPA technique against state of the art methods in randomly generated graph signals and real data from image and video.

There are some interesting directions for future work. Moving average models have been used to derive the optimal $\mathcal{U}/\mathcal{P}$ assignments discussed in this paper. It would be interesting to use different models as GMRFs [45] and investigate optimal $\mathcal{U}/\mathcal{P}$ assignment methods assuming these models. This approach has been recently explored in [51]. Problem III.1 is formulated and Algorithm 1 is designed so that energy is minimized for a given $|\mathcal{P}|$ nodes. Therefore, in a practical application, one should fix $|\mathcal{P}|$ nodes. It would be interesting to select $|\mathcal{P}|$ as a function of the application and the signal of interest. Our current work is focused in the selection of $|\mathcal{P}|$ (and other parameters of the transform) by means of a Rate-Distortion optimization of the graph for image and video coding applications.

APPENDIX

A. Proof of Proposition IV.1

Proof:

According to the homogeneous noise model, we have $\mu_{m,n} = 0$, for all m, n. Thus, $\mathbf{M}_i = \mathbf{0}$ and the prediction error at node i reduces to

$$\begin{aligned} E_{\text{HNM}_i} &= c^2 + \sigma_\eta^2 - 2c^2\mathbf{p}_i^\intercal\mathbf{1} + \sigma_\eta^2\mathbf{p}_i^\intercal\mathbf{p}_i + c^2(\mathbf{p}_i^\intercal\mathbf{1})^2 \\ &= c^2\left(1 - \mathbf{p}_i^\intercal\mathbf{1}\right)^2 + \sigma_\eta^2\left(1 + \mathbf{p}_i^\intercal\mathbf{p}_i\right). \end{aligned} \tag{29}$$

Replacing $\mathbf{p}_i = \frac{1}{m_i}\mathbf{1}$ into (29), and summing over $\mathcal{P}$ nodes, we have

$$E_{tot_{\text{HNM}}} = \sigma_\eta^2\left(|\mathcal{P}| + \sum_{i\in\mathcal{P}}\left(\frac{1}{m_i}\right)\right). \tag{30}$$

TABLE I
E_{RMS} OF THE PROPOSED UPA AND JPEG 2000 AT DIFFERENT LEVELS OF THE TRANSFORM IN AN IMAGE CODING APPLICATION.

	JPEG 2000		Proposed			
	$j=1$ ($\lvert\mathcal{U}\rvert/N=1/4$)	$j=2$ ($\lvert\mathcal{U}\rvert/N=1/16$)	$j=1$ ($\lvert\mathcal{U}\rvert/N=1/2$)	$j=2$ ($\lvert\mathcal{U}\rvert/N=1/4$)	$j=3$ ($\lvert\mathcal{U}\rvert/N=1/8$)	$j=4$ ($\lvert\mathcal{U}\rvert/N=1/16$)
Lena	11.4	12.9	5.3	6.6	7.9	9.4
Cameraman	13.6	15.1	6.1	7.7	9.4	10.9
Barbara	15.6	16.0	6.2	11.0	12.2	13.4

TABLE II
E_{RMS} OF THE PROPOSED UPA AND [29] AT DIFFERENT LEVELS OF THE TRANSFORM IN A VIDEO CODING APPLICATION.

	[29]				Proposed			
	$j=1$ ($\lvert\mathcal{U}\rvert/N=1/2$)	$j=2$ ($\lvert\mathcal{U}\rvert/N=1/4$)	$j=3$ ($\lvert\mathcal{U}\rvert/N=1/8$)	$j=4$ ($\lvert\mathcal{U}\rvert/N=1/16$)	$j=1$ ($\lvert\mathcal{U}\rvert/N=1/2$)	$j=2$ ($\lvert\mathcal{U}\rvert/N=1/4$)	$j=3$ ($\lvert\mathcal{U}\rvert/N=1/8$)	$j=4$ ($\lvert\mathcal{U}\rvert/N=1/16$)
Football	19.4	20.4	21.2	21.9	13.5	14.9	15.7	16.2
Carphone	8.3	10.1	11.4	12.4	7.5	9.2	10.4	11.5
Tempete	9.2	11.2	12.7	14.7	12.8	14.3	15.3	16.2

Noting that the function $f(m)=1/m$ is convex in $0<m<\infty$, we can write

$$\sum_{i\in\mathcal{P}} f(m_i) \geq |\mathcal{P}| f\left(\frac{1}{|\mathcal{P}|}\sum_{i\in\mathcal{P}} m_i\right) \tag{31}$$

with equality for $m_i = W/|\mathcal{P}|$. Using (31) in (30) we can write

$$E_{tot_{\text{HNM}}} \geq \sigma_\eta^2\left(|\mathcal{P}| + \sum_{i\in\mathcal{P}} \frac{1}{\frac{W}{|\mathcal{P}|}}\right) \tag{32}$$

which shows that $m_i = W/|\mathcal{P}|$ is optimal.

B. Proof of Lemma IV.1

Proof:

Let x_i and $\hat{x}_i$ satisfy Definition IV.3 and Definition IV.1 respectively. Taking $p_{i,k} = 1/m_i$ in (10), we get

$$E_{\text{SUM}_i} = \mathbb{E}\{(x_i-\hat{x}_i)^2\} = \mu_{i,i} + \sigma_\eta^2 - 2\frac{1}{m_i}\sum_{k\in\mathcal{N}_i\cap\mathcal{U}}\mu_{i,k} + \frac{1}{m_i^2}\sum_{k\in\mathcal{N}_i\cap\mathcal{U}}\sum_{h\in\mathcal{N}_i\cap\mathcal{U}}\mu_{k,h} + \frac{\sigma_\eta^2}{m_i}, \tag{33}$$

and taking $q_{m,n} = 1/|\mathcal{N}_{[m]}|$ in (12) we get

$$\mu_{k,h} = \sigma_\epsilon^2 \sum_{\ell\in\mathcal{N}_{[k]}\cap\mathcal{N}_{[h]}} q_{k,\ell}q_{h,\ell} = \sigma_\epsilon^2 \frac{|\mathcal{N}_{[k]}\cap\mathcal{N}_{[h]}|}{|\mathcal{N}_{[k]}||\mathcal{N}_{[h]}|} = \sigma_\epsilon^2 c(k,h) \tag{34}$$

and, thus, $\mu_{k,k} = \sigma_\epsilon^2/|\mathcal{N}_{[h]}|$, for any k. Joining (33) and (34), we get

$$E_{\text{SUM}_i} = \sigma_\eta^2 + \frac{\sigma_\epsilon^2}{|\mathcal{N}_{[i]}|} + \frac{\sigma_\eta^2}{m_i} + \frac{\sigma_\epsilon^2}{m_i^2}\sum_{k\in\mathcal{N}_i\cap\mathcal{U}}\sum_{h\in\mathcal{N}_i\cap\mathcal{U}} c(k,h) - 2\frac{\sigma_\epsilon^2}{m_i}\sum_{k\in\mathcal{N}_i\cap\mathcal{U}} c(i,k). \tag{35}$$

C. Proof of Proposition IV.2

Proof:

E_{MA_i} can be expressed as:

$$E_{\text{MA}_i} = \mathbb{E}\{(x_i-\hat{x}_i)^2\} = \mathbb{E}\{(x_i)^2\} + \mathbb{E}\{(\hat{x}_i)^2\} - 2\mathbb{E}\{x_i\hat{x}_i\}. \tag{36}$$

Note that

$$\mathbb{E}\{(\hat{x}_i)^2\} = \sum_{k\in\mathcal{N}_i\cap\mathcal{U}}\sum_{h\in\mathcal{N}_i\cap\mathcal{U}} p_{i,k}p_{i,h}\mathbb{E}\{(x_kx_h)\} \tag{37}$$

and

$$\mathbb{E}\{x_i\hat{x}_i\} = \sum_{k\in\mathcal{N}_i\cap\mathcal{U}} p_{i,k}\mathbb{E}\{x_ix_k\}. \tag{38}$$

Therefore:

$$E_{\text{MA}_i} = \mathbb{E}\{(x_i)^2\} + \sum_{k\in\mathcal{N}_i\cap\mathcal{U}}\sum_{h\in\mathcal{N}_i\cap\mathcal{U}} p_{i,k}p_{i,h}\mathbb{E}\{(x_kx_h)\} - 2\sum_{k\in\mathcal{N}_i\cap\mathcal{U}} p_{i,k}\mathbb{E}\{x_ix_k\}. \tag{39}$$

Let us define E_{SAM_i} as E_{MA_i} under assumtions in (23). E_{SAM_i} can be expressed as:

$$E_{\text{SAM}_i} = \sigma^2 + \sum_f \left((p_{i,k}^f)^2 m_i^f\sigma^2 + (p_{i,k}^f)^2 m_i^f(m_i^f-1)\alpha^f\right) + \sum_{g\neq f} m_i^f m_i^g p_{i,k}^f p_{i,k}^g \alpha^{fg} - 2\sum_f m_i^f p_{i,k}^f \gamma^f, \tag{40}$$

where $f,g\in\{1,2,...,F\}$ are different classes of links.

Finally, using the weighted filters defined in (24) we get

$$E_{\text{SAM}_i} = \sigma^2 + \sum_f w_f^2\left(\frac{\sigma^2}{m_i^f} + \alpha^f(1-\frac{1}{m_i^f})\right) + \sum_f\sum_{g\neq f} w_fw_g\alpha^{fg} - 2\sum_f w_f\gamma^f. \tag{41}$$

Now, we want to seek the m_i^f, $i \in \mathcal{P}$, that minimizes E_{tot} for a fixed $|\mathcal{P}|$ and number of links between $\mathcal{P}$ and $\mathcal{U}$ sets $W_c = \sum_{i\in\mathcal{P}} m_i = \sum_{i\in\mathcal{P}} \sum_f m_i^f$:

$$\min_{m_i^f} \left\{ \sum_{i\in\mathcal{P}} E_{\text{SAM}_i} \right\}$$
$$\text{s. t.} \sum_{i\in\mathcal{P}} \sum_f m_i^f = W_c. \tag{42}$$

Minimizing the Lagrangian

$$\mathcal{L} = \sum_{i\in\mathcal{P}} E_{\text{SAM}_i} + \lambda \left(\sum_{i\in\mathcal{P}} \sum_f m_i^f - W_c \right). \tag{43}$$

with respect to each m_i^f, using (41):

$$\frac{\partial \mathcal{L}}{\partial m_i^f} = -w_f^2 \left(\frac{\sigma^2}{(m_i^f)^2} - \frac{\alpha^f}{(m_i^f)^2} \right) + \lambda = 0 \implies \tag{44}$$
$$m_i^{f^*} = w_f \sqrt{\frac{(\sigma^2 - \alpha^f)}{\lambda}} = K_f,$$

where K_f is a constant that does not depend on i.

Furthermore,

$$\sum_{i\in\mathcal{P}} \sum_g m_i^g = |\mathcal{P}| \sum_g K_g = W_c \implies \tag{45}$$
$$\sum_g K_g = \frac{W_c}{|\mathcal{P}|} = \sum_g w_g \sqrt{\frac{(\sigma^2 - \alpha^g)}{\lambda}} \implies$$
$$m_i^{f^*} = \frac{W_c}{|\mathcal{P}|} \frac{w_f A^f}{\sum_g w_g A^g},$$

where $g \in \{1, 2, ..., F\}$ and $A^g = \sqrt{\sigma^2 - \alpha^g}$.

ACKNOWLEDGMENT

E.M-E author would like to thank Javier Portilla for the useful discussions and suggestions.

REFERENCES

[1] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, Mar. 2011.

[2] 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," *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 83–98, 2013.

[3] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *Signal Processing, IEEE Transactions on*, vol. 60, no. 6, pp. 2786–2799, 2012.

[4] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *CoRR*, vol. abs/1210.4752, 2012.

[5] P. Milanfar, "A tour of modern image filtering: New insights and methods, both practical and theoretical." *Signal Processing Magazine, IEEE*, vol. 30, no. 1, pp. 106–128, 2013.

[6] W. Hu, G. Cheung, A. Ortega, and O. C. Au, "Multiresolution graph fourier transform for compression of piecewise smooth images," *Image Processing, IEEE Transactions on*, vol. 24, no. 1, pp. 419–433, Jan. 2015.

[7] M. Crovella and E. Kolaczyk, "Graph wavelets for spatial traffic analysis," in *IN IEEE INFOCOM*, 2002.

[8] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 53 – 94, 2006, special Issue: Diffusion Maps and Wavelets.

[9] W. Wang and K. Ramchandran, "Random multiresolution representations for arbitrary sensor network graphs," in *Acoustics, Speech and Signal Processing (ICASSP), 2006 IEEE International Conference on*, vol. 4, May 2006, pp. IV–IV.

[10] D. Liu and M. Flierl, "Video coding using multi-reference motion-adaptive transforms based on graphs," in *2016 IEEE 12th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, Jul. 2016, pp. 1–5.

[11] T. Maugey, A. Ortega, and P. Frossard, "Graph-based representation for multiview image geometry," *Image Processing, IEEE Transactions on*, vol. 24, no. 5, pp. 1573–1586, May 2015.

[12] D. Thanou, P. A. Chou, and P. Frossard, "Graph-based compression of dynamic 3D point cloud sequences," *Image Processing, IEEE Transactions on*, vol. 25, no. 4, pp. 1765–1778, Apr. 2016.

[13] W. M. Campbell, C. K. Dagli, and C. J. Weinstein, "Social network analysis with content and graphs," *Lincoln Laboratory Journal*, vol. 20, no. 1, pp. 61–81, 2013.

[14] X. Dong, D. Mavroeidis, F. Calabrese, and P. Frossard, "Multiscale event detection in social media," *Data Mining and Knowledge Discovery*, vol. 29, no. 5, pp. 1374–1405, 2015.

[15] G. Michau, P. Borgnat, N. Pustelnik, P. Abry, A. Nantes, and E. Chung, "Estimating link-dependent origin-destination matrices from sample trajectories and traffic counts," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 5480–5484.

[16] A. Sakiyama, Y. Tanaka, T. Tanaka, and A. Ortega, "Efficient sensor position selection using graph signal sampling theory," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 6225–6229.

[17] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," 1995, tech. report 1995:6, Industrial Math. Initiative, Dept. of Math., University of South Carolina, 1995.

[18] R. Wagner, H. Choi, and R. Baraniuk, "Distributed wavelet transform for irregular sensor network grids," in *IEEE Statistical Signal Processing (SSP) Workshop*, 2005.

[19] S. K. Narang and A. Ortega, "Lifting based wavelet transforms on graphs," in *APSIPA ASC 2009: Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference*, Oct. 2009.

[20] G. Shen and A. Ortega, "Optimized distributed 2D transforms for irregularly sampled sensor network grids using wavelet lifting," in *Acoustics, Speech and Signal Processing (ICASSP), 2008 IEEE International Conference on*, Mar. 2008, pp. 2513 –2516.

[21] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 11, no. 7, pp. 674–693, 1989.

[22] G. Shen and A. Ortega, "Transform-based distributed data gathering," *Signal Processing, IEEE Transactions on*, vol. 58, no. 7, pp. 3802–3815, 2010.

[23] ——, "Compact image representation using wavelet lifting along arbitrary trees," in *Image Processing (ICIP), 2008 IEEE International Conference on*, Oct. 2008, pp. 2808 –2811.

[24] A. Sánchez, G. Shen, and A. Ortega, "Edge-preserving depth-map coding using graph-based wavelets," in *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, 2009, pp. 578–582.

[25] R. L. Claypoole, G. M. Davis, W. Sweldens, and R. G. Baraniuk, "Nonlinear wavelet transforms for image coding via lifting," *Image Processing, IEEE Transactions on*, vol. 12, no. 12, pp. 1449–1459, Dec. 2003.

[26] R. Claypoole, G. Davis, W. Sweldens, and R. Baraniuk, "Nonlinear wavelet transforms for image coding," *Image Processing, IEEE Transactions on*, pp. 1449–1459, 1997.

[27] M. Flierl and B. Girod, "Video coding with motion-compensated lifted wavelet transforms." *Signal Processing: Image Communication*, vol. 19, no. 7, pp. 561–575, 2004.

[28] ——, "Investigation of motion-compensated lifted wavelet transforms," in *Proceedings of the Picture Coding Symposium*, 2003, pp. 59–62.

[29] A. Secker and D. Taubman, "Lifting-based invertible motion adaptive transform (LIMAT) framework for highly scalable video compression," *Image Processing, IEEE Transactions on*, vol. 12, no. 12, pp. 1530 – 1542, Dec. 2003.

[30] J. Asensio-Cubero, J. Q. Gan, and R. Palaniappan, "Multiresolution analysis over graphs for a motor imagery based online BCI game," *Computers in Biology and Medicine*, vol. 68, pp. 21 – 26, 2016.
[31] J. Asensio-Cubero, J. Gan, and R. Palaniappan, "Multiresolution analysis over simple graphs for brain computer interfaces," *Journal of neural engineering*, vol. 10, no. 4, p. 046014, 2013.
[32] S. Narang, G. Shen, and A. Ortega, "Unidirectional graph-based wavelet transforms for efficient data gathering in sensor networks," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, Mar. 2010, pp. 2902 –2905.
[33] J. Perez-Trufero, S. Narang, and A. Ortega, "Distributed transforms for efficient data gathering in arbitrary networks," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, Sept. 2011, pp. 1829–1832.
[34] S. Narang and A. Ortega, "Local two-channel critically sampled filter-banks on graphs," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, Sept. 2010, pp. 333 –336.
[35] E. Martínez-Enríquez and A. Ortega, "Lifting transforms on graphs for video coding," in *Data Compression Conference (DCC), 2011*, Mar. 2011, pp. 73 –82.
[36] E. Martínez-Enríquez, F. Díaz-de María, and A. Ortega, "Video encoder based on lifting transforms on graphs," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, Sept. 2011, pp. 3509 –3512.
[37] E. Martínez-Enríquez, F. Díaz-de María, J. Cid-Sueiro, and A. Ortega, "Filter optimization and complexity reduction for video coding using graph-based transforms," in *Image Processing (ICIP), 2013 IEEE International Conference on*, Sept. 2013, pp. 1948–1952.
[38] M. Hidane, O. Lezoray, and A. Elmoataz, "Lifting scheme on graphs with application to image representation," in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, Dec. 2013, pp. 431–434.
[39] H. Nguyen and M. Do, "Downsampling of signals on graphs via maximum spanning trees," *IEEE Transactions on Signal Processing*, vol. 63, no. 1, pp. 182–191, January 1, 2015.
[40] E. Martinez-Enriquez, J. Cid-Sueiro, F. Diaz-De-Maria, and A. Ortega, "Directional transforms for video coding based on lifting on graphs," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, no. 99, pp. 1–1, 2016.
[41] W.-S. Kim, S. Narang, and A. Ortega, "Graph based transforms for depth video coding," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 2012, pp. 813–816.
[42] G. Shen, W. S. Kim, S. K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *Picture Coding Symposium (PCS), 2010*, Dec 2010, pp. 566–569.
[43] W. Hu, G. Cheung, and A. Ortega, "Intra-prediction and generalized graph fourier transform for image coding," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1913–1917, Nov 2015.
[44] C. Zhang, D. Florencio, and P. Chou, "Graph signal processing - a probabilistic framework," Tech. Rep. MSR-TR-2015-31, April 2015.
[45] C. Zhang and D. A. F. Florêncio, "Analyzing the optimality of predictive transform coding using graph-based models." *Signal Processing Letters, IEEE*, vol. 20, no. 1, pp. 106–109, 2013.
[46] S. Narang and A. Ortega, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *Signal Processing, IEEE Transactions on*, vol. 61, no. 19, pp. 4673–4685, Oct 2013.
[47] J. Zeng, G. Cheung, and A. Ortega, "Bipartite subgraph decomposition for critically sampled wavelet filterbanks on arbitrary graphs," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 6210–6214.
[48] C.-P. Hsu, "Minimum-via topological routing," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 2, no. 4, pp. 235 – 246, Oct. 1983.
[49] N. Perraudin, J. Paratte, D. Shuman, V. Kalofolias, P. Vandergheynst, and D. K. Hammond., "Gspbox: A toolbox for signal processing on graphs," 2014, arXiv e-prints, Aug. 2014. http://arxiv.org/abs/1408.5781.
[50] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 36–58, Sep 2001.
[51] Y.-H. Chao, "Graph signal compression and the application to image and video coding," Ph.D. dissertation, University of Southern California, 2017.

Eduardo Martínez-Enríquez (SM'07-M'15) received the Telecommunications Engineering degree from Universidad Politécnica de Madrid, Madrid, Spain, in 2006, and Ph.D. degree from the Universidad Carlos III de Madrid, Spain, in 2013. In 2014 he joined the Instituto de Óptica at the Consejo Superior de Investigaciones Científicas, Madrid, Spain, where he is currently a postdoc researcher. His research interests include graph signal processing, lifting transforms on graphs, and their application to video coding. He received the Best Paper Award of ICIP 2011 for his paper on video coding based on lifting transform on graphs, co-authored with Fernando Díaz and Antonio Ortega.

Jesús Cid-Sueiro (M'95-SM'08) received the degree in Telecommunications Engineering from University of Vigo, Spain, and the Ph.D. degree from Polytechnic University of Madrid, Madrid, Spain, in 1990 and 1994, respectively.

He is currently a Professor with the Department of Signal Theory and Communications at Carlos III University of Madrid. His current research interests include machine learning, Bayesian methods, computational intelligence and their applications in sensor networks, big data and signal processing.

Fernando Díaz-de-María (M'97) received the Telecommunication Engineering degree and the Ph.D. degree from the Universidad Politécnica de Madrid, Spain, in 1991 and 1996, respectively. Since October 1996, he has been an Associate Professor in the Department of Signal Processing and Communications, Universidad Carlos III de Madrid, Madrid, Spain. His primary research interests include video coding, image and video analysis, and computer vision. He has led numerous projects and contracts in the fields mentioned. He is co-author of numerous papers in peer-reviewed international journals, two book chapters, and has presented a number of papers in national and international conferences.

Antonio Ortega (F'07) received the Telecommunications Engineering degree from the Universidad Politecnica de Madrid, Madrid, Spain in 1989 and the Ph.D. in Electrical Engineering from Columbia University, New York, NY in 1994. At Columbia he was supported by a Fulbright scholarship.

In 1994 he joined the Electrical Engineering department at the University of Southern California (USC), where he is currently a Professor. He has served as Associate Chair of EE-Systems and director of the Signal and Image Processing Institute at USC. He is a Fellow of the IEEE, and a member of ACM and APSIPA. He has been Chair of the Image and Multidimensional Signal Processing (IMDSP) technical committee, a member of the Board of Governors of the IEEE Signal Processing Society (SPS), and chair of the SPS Big Data Special Interest Group. He has been technical program co-chair of MMSP 1998, ICME 2002, ICIP 2008 and PCS 2013. He has been Associate Editor for the IEEE Transactions on Image Processing (IEEE TIP) and the IEEE Signal Processing Magazine, among others. He is the inaugural Editor-in-Chief of the APSIPA Transactions on Signal and Information Processing, an Associate Editor of IEEE T-SIPN and Senior Area Editor of IEEE TIP. He received the NSF CAREER award, the 1997 IEEE Communications Society Leonard G. Abraham Prize Paper Award, the IEEE Signal Processing Society 1999 Magazine Award, the 2006 EURASIP Journal of Advances in Signal Processing Best Paper Award, the ICIP 2011 best paper award, and a best paper award at Globecom 2012. He was a plenary speaker at ICIP 2013 and APSIPA ASC 2015.

His research interests are in the areas of signal compression, representation, communication and analysis. His recent work is focusing on distributed compression, multiview coding, error tolerant compression, information representation in wireless sensor networks and graph signal processing. Almost 40 PhD students have completed their PhD thesis under his supervision at USC and his work has led to over 300 publications in international conferences and journals, as well as several patents.