MONITORING MANHATTAN'S TRAFFIC AT 5 INTERSECTIONS?

Siheng Chen[1,2]*, Yaoqing Yang*[1]*, Christos Faloutsos*[2]*, Jelena Kovačević*[1,3]

[1]Dept. of ECE, [2] Dept. of ML, [3]Dept. of BME,
Carnegie Mellon University, Pittsburgh, PA, USA

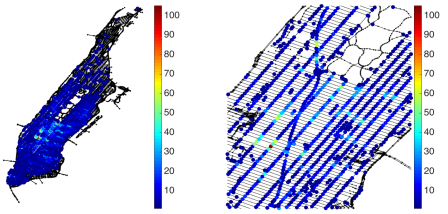
ABSTRACT

Is it possible to monitor the entire traffic in Manhattan at a few intersections? This paper proposes a series of sampling, recovery and representation techniques based on graph signal processing to handle complex, nonsmooth graph signals. We validate our proposed techniques on Manhattan's taxi pickups during the years of 2014 and 2015. We are able to approximately recover the taxi-pick activities in Manhattan by sampling at only 5 selected intersections. The same techniques can be applied to monitor other types of traffic data.

1. INTRODUCTION

Urban data records the behavior of urban ecosystem and analyzing those urban data potentially leads to improvements of the urban lives [1, 2]. As one of the most critical components of urban data, traffic data is a key to understand the mobility pattern and make cities more efficient; however, traffic data is usually sparse because a few sensors are installed to cover a limited number of intersections [3]. In this paper, we aim to recover entire traffic data in the entire city from sensors installed at a few intersections. To verify the feasibility of this idea, we focus on Manhattan's taxi pickups during the years of 2014 and 2015 because taxis are valuable sensors of city life [4]. We model taxi-pick activities as graph signals supported on a city street network where signal coefficient at a node reflects the number of taxi pickups at the corresponding intersection. The recovery of taxi-pick activities is nothing but graph signal sampling and recovery [5–7]; however, previous works only consider sampling and recovery of smooth graph signals. Real taxi-pick activities may not be smooth on a city street network; see Figure 1. To handle this problem, we propose a series novel techniques based on graph signal processing [8, 9] to learn traffic patterns from historical taxi-pick activities and design targeted sampling and recovery strategies. We are able to approximately recover the taxi-pick activities in entire Manhattan by taking samples at only 5 selected intersections. Here we focus on taxi-pick activity, but the same techniques can be applied to recover many other types of traffic data.

The authors gratefully acknowledge support from the NSF through awards 1130616 and 1017278, and CMU Carnegie Institute of Technology Infrastructure Award.



(a) Global view. (b) Zoom-in plot.

Fig. 1: Taxi-pickup distribution at 6 pm on January 1st, 2015. The data is not smooth on the Manhattan street network. We approximate this data by a piecewise-constant graph signal.

2. PROBLEM FORMULATION

We consider a city street network $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$ is the set of nodes (intersections), $\mathcal{E} = \{e_1, \ldots, e_M\}$ is the set of edges (streets). A *graph signal* models taxi pickups in a city that assigns the number of taxi pickups during a period of time $x_n \in \mathbb{R}$ to the node v_n; a vector form is $\mathbf{x} = \left[x_1, x_2, \ldots, x_N\right]^T \in \mathbb{R}^N$. Let $C \subseteq \mathcal{V}$ be a set of nodes (an area in a city). We can represent this set by using an indicator function, $\mathbf{1}_C \in \mathbb{R}^N$, where $(\mathbf{1}_C)_i = 1$ when $v_i \in C$, and 0 otherwise. The signal coefficients are ones in the node set C and zeros in the complement node set $\bar{C} = \mathcal{V}/C$. When the node set C forms a connected subgraph, we call C a *piece* and $\mathbf{1}_C$ a *one-piece graph signal*. A piecewise-constant graph signal is a linear combination of several one-piece graph signals $\mathbf{x} = \sum_{i=1}^{K} \mu_i \mathbf{1}_{C_i}$, where C_i is a piece, μ_i is a constant and K is the number of pieces.

Sampling & Recovery. We consider sampling the number of passing vehicles at several selected intersections and then recovering the number of passing vehicles at the rest intersections. Mathematically, we sample M coefficients at selected indices (intersections) in a graph signal $\mathbf{x} \in \mathbb{R}^N$ to produce a sampled signal $\mathbf{y} = \Psi\,\mathbf{x}$, where the sampling operator Ψ is a linear mapping from $\mathbb{R}^N$ to $\mathbb{R}^M$ with $\Psi_{i,j} = 1$ when we sample the jth node in the ith measurement, and 0, otherwise. Here we consider experimentally design sampling, which allows that sample indices are chosen beforehand. We then interpolate $\mathbf{y}$ to get a recovery $\widehat{\mathbf{x}} = \Phi\mathbf{y} \in \mathbb{R}^N$, where Φ is the interpolation operator designed based on Ψ.

3. PROPOSED METHOD

The proposed method involves two phases: learning phase and real-time processing phase. In the learning phase, we learn all the operators needed in the real-time processing phase from historical taxi-pickup activities. In the real-time processing phase, we sample the taxi-pickup activities at a few selected intersections and recover the rest by using the operators learned in the learning phase.

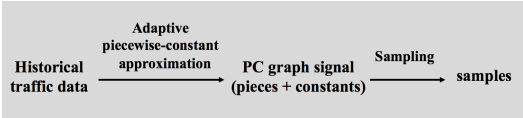


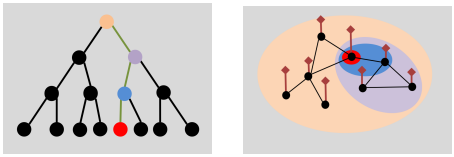
Fig. 2: Learning phase includes two main blocks: adaptive piecewise-constant approximation implemented by adaptively pruning a decomposition tree and sampling implemented by sampling bandlimited graph signals. In the learning phase, we decide which node to sample.

3.1. Learning Phase

The purpose of learning phase is to learn important patterns from historical traffic data and then decide which intersections we need to sample. A basic idea is to construct a graph that promotes smoothness for historical taxi-pickup activities and then use graph sampling techniques to design samples [10]. Does the original Manhattan street network promote smoothness for traffic data? Figure 1 shows the taxi-pickup distribution at 7 pm on January 1st, 2015. We see that many intersections have many more taxi pickups than their neighbors and the entire distribution is barely smooth. We thus need to learn a graph from traffic data. However, a city street network is usually huge and historical traffic data is limited. For example, Manhattan has $13,670$ intersections. It is thus inefficient and unrobust to construct a huge graph. To overcome this, we should reduce the size of graph. In real traffic data, we find that sometimes neighboring intersections have similar number of taxi pickups. We can significantly reduce the size of graph by exploring local information and grouping those neighboring intersections as one super-node. This is equivalent to approximate the original graph signal by using a piecewise-constant graph signal. Through approximation, the dimension reduces from the number of intersections to the number of pieces. We then construct a super-graph whose nodes are pieces and edges are the similarities between pieces. We then can use graph sampling to design samples. Figure 2 overviews the procedure of the learning phase. The two main modules are adaptive piecewise-constant approximation and graph sampling. We now elaborate them.

Adaptive piecewise-constant approximation. The goal is to adaptively find a piecewise-constant graph signal to approximate taxi-pickup activities. The key is to design a series of nonoverlapping pieces that captures the variation of an graph signal. There are usually two approaches to design such a series: predesigned approach and learning approach. In a predesigned approach, we design pieces before accessing any traffic data. We can simply use physical partitions, such as zipcodes and census blocks; however, these partitions may not be flexible enough to capture complex variations in traffic data; on the other hand, in a learning approach, we learn a series of pieces to fit traffic data; however, are multiple restrictions in the optimization: those pieces are connected, nonoverlapping and cover the vertex domain. It is inefficient and unrobust to solve a nonconvex optimization problem with multiple constraints and limited training data.

Here we consider combining the advantages of these two approaches. We first design a set of redundant pieces before having any data. Because of the redundancy, this set is able to capture various shapes and sizes. We then prune this set and selects the best series of nonoverlapping pieces according to historical taxi-pickup activities. This approach is both adaptive and efficient. The set of redundant pieces can be constructed beforehand and the bottleneck of the computational complexity is the pruning stage. By taking advantage of a tree-structure, the computational complexity is merely $O(N)$.



(a) Global view. (b) Zoom-in plot.

Fig. 3: Grow a binary tree in (a) is equivalent to decompose a graph in (b). The green path in (a) is a decomposition in (b), where the same color indicates the one-to-one mapping from a node in the decomposition tree to a piece in a graph.

A set of redundant pieces can be constructed via a binary tree decomposition as shown in [11]. This set provably represents arbitrary piecewise-constant graph signals. The main idea is to recursively partition a piece into two disjoint pieces until that all the pieces are individual nodes. Figure 3 shows an example. A node in (a) represents a piece in (b) and an edge represents a kinship where a parent node partitions into two children nodes. The top node (in orange) represents the entire vertex domain and the bottom nodes represents all the individual nodes. The green path in (a) is a decomposition in (b), where the same color indicates the one-to-one mapping from a node in a decomposition tree to a piece in a graph. We use the 2-means clustering to implement graph partitioning [11]. For each piece, we select two nodes with longest geodesic distance as the community centers and assign all the other nodes to their nearest community center based on the geodesic distance. We then recompute the community center for each community by minimizing the summation of the geodesic distances to all the other nodes in the community and assign node to its nearest community center again. We keep doing this until the community centers converge after a

few iterations. Please find more details in [11].

By using the binary tree decomposition, we obtain $(2N-1)$ pieces. which is redundant and captures various sizes and shapes of pieces. We then prune this set and selects the best series of nonoverlapping pieces according to historical traffic data. Let $\mathcal{C} = \{\mathbf{1}_{C_i}\}_{i=1}^{2N-1}$ be the set of constructed pieces. We aim to select a subset of pieces that minimizes the following optimization problem,

$$\begin{aligned} \widehat{\mathrm{D}}, \widehat{\mathrm{Z}} \;\; &= \;\; \arg\min_{\mathrm{D}_i \in \mathcal{C}, \mathrm{Z}} \|\mathrm{X} - \mathrm{D}\,\mathrm{Z}\|_F^2 + \lambda \dim(\mathrm{Z}), \qquad (1) \\ &\quad \text{subject to } \mathrm{D}\,\mathbf{1} \;=\; \mathbf{1}, \end{aligned}$$

where $\mathrm{X} \in \mathbb{R}^{N \times L}$ is a matrix representation of historical taxi-pick activities with L snapshots, $\mathrm{D} \in \mathbb{R}^{N \times K}$ is a matrix representation of constructed pieces with D_i being the ith column, λ is a tuning parameter and $\mathrm{Z} \in \mathbb{R}^{K \times L}$ stores the constants of all the pieces with $\dim(\mathrm{Z})$ the number of elements in Z. Note that K is variable during the optimization because we do not know how many pieces we need in advance.

The first term in the objective functions pushes the piecewise-constant approximation to fit the given data. The second term punishes a large size of the constant matrix Z and avoids overfitting; that is, when λ is large, we tend to select fewer pieces from $\mathcal{C}$ to fit data and when λ is small, we tend to select all the pieces in $\mathcal{C}$ to fit data. The constrain requires that all the selected pieces are nonoverlapping and covers the entire vertex domain. Since each column in D is a one-piece graph signal, the optimization problem (1) finds the best piecewise-constant approximation for given traffic data. Since the constructed pieces in $\mathcal{C}$ have a tree structure, we easily obtain the global optimum of (1) by pruning the tree, which is similar to [12, 13]. The main idea is to compare the representation based on a parent piece to the representation based on its two children pieces and see which representation minimizes the objective function (1). For example, C_1 is a parent piece and C_2, C_3 are its children pieces. Since the parent piece and two children pieces represent the same vertex domain ($C_1 = C_2 \cup C_3$), to satisfy the constraint, we either choose the parent piece or its two children pieces. The cost of using the parent piece is $\min_{\mathbf{z}} \left\|\mathrm{X} - \mathbf{1}_{C_1}\mathbf{z}^T\right\|_F^2 + \lambda L$, with optimum $\left\|\mathrm{X} - \mathbf{1}_{C_1}\mathbf{1}_{C_1}^T\,\mathrm{X}\right\|_F^2 + \lambda L$, and the cost of using the child pieces is $\min_{\mathrm{Z} \in \mathbb{R}^{2\times L}} \left\|\mathrm{X} - \begin{bmatrix}\mathbf{1}_{C_2} & \mathbf{1}_{C_3}\end{bmatrix}\mathrm{Z}\right\|_F^2 + 2\lambda L$, with optimum $\left\|\mathrm{X} - \begin{bmatrix}\mathbf{1}_{C_2} & \mathbf{1}_{C_3}\end{bmatrix}\begin{bmatrix}\mathbf{1}_{C_2} & \mathbf{1}_{C_3}\end{bmatrix}^T\mathrm{X}\right\|_F^2 + 2\lambda L$ Each time, we compare their costs and choose the one with a smaller cost to update the representation at the parent piece. The pruning process starts from the bottom level of the decomposition tree and move to an upper level iteratively until we reach the top level. Through the pruning process, we obtain the global optimum of (1).

Sampling. We next model each selected piece after pruning as a super-node and construct a super-graph. Since the selected pieces already capture the local similarities, the connection among super-nodes are not relevant to the geodesic distance any more. We need to learn a super-graph to promote smoothness for historical taxi-pickup activities and then design which super-nodes to sample. In graph sampling, we usually model a smooth graph signal as a bandlimited graph signal [6, 14, 15] whose sampling strategy is designed based on the corresponding graph Fourier basis. Thus, instead of constructing a full super-graph, we directly construct a graph Fourier basis. Recall that the bandlimited assumption requires that most energy of a graph signal is concentrated in the low-pass band; that is, we need to find a graph Fourier basis that pushes the energy to the subspace spanned by its first few basis vectors. Thus, all we need is the first few columns in the graph Fourier basis, which can be simply obtained by principal component analysis. Principal component analysis uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables [16], which exactly fits our requirement. Mathematically, let the constant matrix $\widehat{\mathrm{Z}} \in \mathbb{R}^{K \times L}$ be a matrix of graph signals on the super-graph, we obtain the first M graph Fourier basis vectors (principal components) by solving the following optimization problem,

$$\widehat{\mathrm{V}} = \arg\min_{\mathrm{V} \in \mathbb{R}^{K \times M}} \left\|\widehat{\mathrm{Z}} - \mathrm{V}\,\mathrm{V}^T\,\widehat{\mathrm{Z}}\right\|_F^2, \text{ subject to } \mathrm{V}^T\,\mathrm{V} = \mathrm{I}.$$

Note that we can obtain a truncated graph Fourier basis directly from X, which is equivalent to set λ be zero in (1); however, the computation is less efficient and the obtained principal components are learned from noisy and limited historical data and do not take advantage of the local grouping[1], which is explored by (1). We next design a sampling operator by using graph sampling techniques. For example, we solve $\widehat{\Psi} \;=\; \arg\max_{\Psi} \sigma_{\min}(\Psi\widehat{\mathrm{V}}) \in \mathbb{R}^{M \times K}$ by using a greedy method [6]. We sample super-nodes, instead of individual intersections. To directly operate on individual intersections, the sampling operator is $\widehat{\Psi}\widehat{\mathrm{D}}^T \in \mathbb{R}^{M \times N}$, meaning that $\widehat{\Psi}$ selects some pieces from $\widehat{\mathrm{D}}$ in (1). We need to sample all nodes in the selected pieces, or sample several nodes and estimate the average values. In the experiments, we find that the selected pieces happen to be individual nodes; that is, we only need to sample one intersection for a piece.

3.2. Real-time Processing Phase

In the learning phase, we obtain three operators from historical taxi-pickup activities: selected pieces $\widehat{\mathrm{D}}$, truncated graph Fourier basis $\widehat{\mathrm{V}}$, sampling operator $\widehat{\Psi}$. Given a real-time traffic data $\mathbf{x} \in \mathbb{R}^N$, we first take samples at the selected intersections, $\mathbf{y} = \widehat{\Psi}\widehat{\mathrm{D}}^T\,\mathbf{x}$. We then use the interpolation operator to recover all the constants, $\mathbf{z} = \widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}\mathbf{y}$. Finally, we obtain a piecewise-constant approximation to the real taxi pickups by

$$\widehat{\mathbf{x}} \;=\; \widehat{\mathrm{D}}\mathbf{z} \;=\; \widehat{\mathrm{D}}\widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}\mathbf{y} \;=\; \widehat{\mathrm{D}}\widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}\widehat{\Psi}\widehat{\mathrm{D}}^T\,\mathbf{x},$$

[1] Piecewise-constant approximation can be regarded as a denoising block. Many experiments indicate that reducing the dimension to $N/2$ provides the best recovery performance in the end, which is both better and faster than directly working with X.

where the interpolation operator is $\Phi = \widehat{\mathrm{D}}\widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}$. Figure 4 illustrates the procedure in real-time processing.

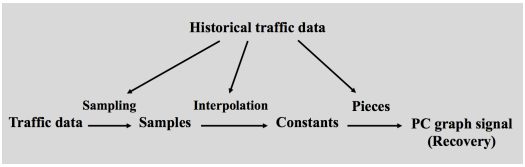
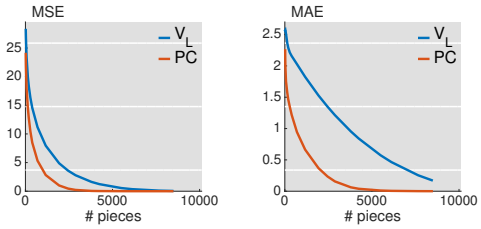


Fig. 4: In real-time processing, we sample the selected nodes, recover all the constants, and finally obtain the piecewise-constant estimation to the real-time traffic data.



(a) MSE. (b) MAE.

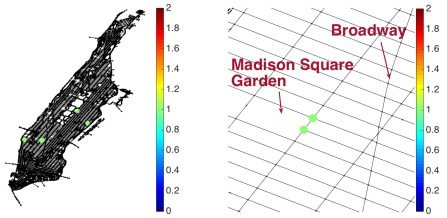
Fig. 5: Piecewise-constant approximation significantly outperforms smooth approximation.

4. VALIDATIONS

We validate the proposed method on a Manhattan's taxi-pickup dataset. We sample the number of taxi pickups at several intersections and recover the number of taxi pickups at the rest intersections.

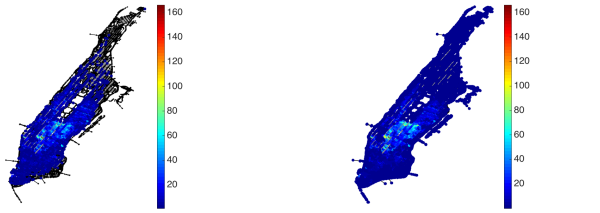
Dataset. We consider taxi pickups in Manhattan[2]. Here we use the dataset in the year of 2014 and 2015. We focus on rush hours during workdays (6 pm from Monday to Friday). We accumulate the taxi-pickup activities within a hour and project each taxi pickup to its closest intersection, obtaining 261 graph signals in the year of 2014 for learning and 261 graph signals in the year of 2015 for real-time processing.

Results. We first validate the proposed adaptive piecewise-constant approximation. We solve (1) based on 261 graph signals in 2014 by varying the regularization parameter λ. Two metrics are used to quantify the performance, including mean square error (MSE = $\frac{1}{261N}\sum_{i=1}^{261}\|\widehat{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$) and mean absolute error (MAE = $\frac{1}{261N}\sum_{i=1}^{261}\|\widehat{\mathbf{x}}_i - \mathbf{x}_i\|_1$), where $\widehat{\mathbf{x}}_i$ is the recovered taxi pickups in the ith day and $\mathbf{x}_i$ is the real taxi pickups in the ith day. Figure 5 compares the approximation errors between the graph Fourier basis based on the Laplacian matrix (V_L, in blue) and adaptive piecewise-constant approximation (PC, in red). We see that PC significantly outperforms V_L in terms of both metrics. We then set $\lambda = 1$ (corresponds to 3788 pieces) and obtain 5 samples provided by the optimal sampling operator, as shown in Figure 6. As discussed before, these 5 pieces happen to be individual nodes. Two adjacency intersections around Penn Station are sampled, indicating that Penn Station is the weathercock of Manhattan's traffic. We next validate those learned operators to the graph signals in 2015. Figure 7 shows the recovery of taxi-pick activity at 6 pm, Jan. 6th, 2015 by only using 5 samples. Even we just use 5 samples, the recovered taxi-pick distribution is very close to the real taxi-pick distribution. Finally, we show the daily recovery errors in Figure 8. The recovery errors are particularly large at Memorial day and Labor day (not surprise), but in general, the recovery error is small. For example, Figure 8 (b) shows that the average error at each intersection is merely 0.6 taxi pickups during the rush hour every weekday.



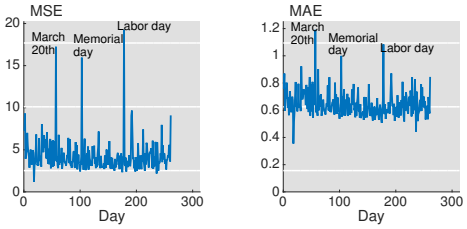
(a) Global view. (b) Zoom-in plot.

Fig. 6: Selected 5 intersections. Two adjacency intersections around Penn Station are sampled.



(a) Real taxi-pick distribution. (b) Recovered taxi-pick distribution.

Fig. 7: Recovered taxi pickups at 6 pm, Jan. 6th, 2015.



(a) MSE. (b) MAE.

Fig. 8: Daily recovery error in the year of 2015.

5. CONCLUSIONS

The goal is to monitor Manhattan's traffic from a few intersections. We show that an approximate recovery of the taxi-pick activities can be obtained by taking samples at only 5 selected intersections. The main techniques involves adaptive pieceiwise-constant approximation via decomposition tree pruning, super-graph Fourier basis construction via principal component analysis and sampling for bandlimited graph signals. The paper suggests that graph signal processing tools aid in urban computing.

[2]Data is downloaded from `http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml`

6. REFERENCES

[1] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva, "Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips," *IEEE Trans. on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2149–2158, Dec. 2013.

[2] H. Doraiswamy, N. Ferreira, T. Damoulas, J. Freire, and C. T. Silva, "Using topological analysis to support event-guided exploration in urban data," *IEEE Trans. on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2634–2643, 2014.

[3] J. Poco, H. Doraiswamy, H. T. Vo, J. L. D. Comba, J. Freire, and C. T. Silva, "Exploring traffic dynaics in urban environments using vector-valued functions," *Computer Graphics Forum (in proceedings of EuroVis 2015)*, vol. 34, no. 3, pp. 161–170, 2015.

[4] J. A. Deri and J. M. F. Moura, "Taxi data in New York City: A network perspective," in *Proc. Asilomar Conf. Signal, Syst. Comput.*, Pacific Grove, CA, Nov. 2015, pp. 1829–1833.

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

[6] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete signal processing on graphs: Sampling theory," *IEEE Trans. Signal Process.*, vol. 63, pp. 6510 – 6523, Aug. 2015.

[7] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal recovery on graphs: Fundamental limits of sampling strategies," *arXiv:1512.05405*, 2015.

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

[9] 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, pp. 83–98, May 2013.

[10] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, 2016, to appear.

[11] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal representations on graphs: Tools and applications," *arXiv:1512.05406*, 2015.

[12] R. R. Coifman and M. V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. Inf. Theory, sp. iss. Wavelet Transforms Multiresolution Signal Anal.*, vol. 38, no. 2, pp. 713–718, Mar. 1992.

[13] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Trans. Image Process.*, vol. 2, no. 2, pp. 160–175, Apr. 1993.

[14] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Trans. Signal Process.*, vol. 64, pp. 3775–3789, Jul. 2016.

[15] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, pp. 1832–1843, Apr. 2016.

[16] C. M. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. Springer, 2006.