An analysis of spectral transformation techniques on graphs

Igor Djurović[a], Ervin Sejdić[b], Nikola Bulatović*[a], Marko Simeunović[a]
[a]Dept. of Computer Engineering, Faculty of Electrical Engineering, Univ. of Montenegro, Džordža Vašingtona bb, Podgorica, MNE; [b]Dept. of Electrical and Computer Engineering, Swanson School of Engineering, Univ. of Pittsburgh, Pittsburgh, USA

ABSTRACT

Emerging methods for the spectral analysis of graphs are analyzed in this paper, as graphs are currently used to study interactions in many fields from neuroscience to social networks. There are two main approaches related to the spectral transformation of graphs. The first approach is based on the Laplacian matrix. The graph Fourier transform is defined as an expansion of a graph signal in terms of eigenfunctions of the graph Laplacian. The calculated eigenvalues carry the notion of frequency of graph signals. The second approach is based on the graph weighted adjacency matrix, as it expands the graph signal into a basis of eigenvectors of the adjacency matrix instead of the graph Laplacian. Here, the notion of frequency is then obtained from the eigenvalues of the adjacency matrix or its Jordan decomposition. In this paper, advantages and drawbacks of both approaches are examined. Potential challenges and improvements to graph spectral processing methods are considered as well as the generalization of graph processing techniques in the spectral domain. Its generalization to the time-frequency domain and other potential extensions of classical signal processing concepts to graph datasets are also considered. Lastly, it is given an overview of the compressive sensing on graphs concepts.

Keywords: graph theory, spectral analysis, eigenvalues and eigenvectors, graph filtering, compressed sensing

1. INTRODUCTION

Large amounts of data are collected daily. Collecting, analyzing and processing such data is an important and essential task in order to extract valuable information to be used in decision making processes. Today's big data systems are ubiquitous in numerous domains from engineering to energy and neuroscience. The volume and complex structure of data makes it difficult for current information systems to process it efficiently. New data processing algorithms and methods have to be developed in order to pursue the pace of the current technological growth.

One of the most effective ways to represent diverse types of high dimensional data is a graph structure. Processing methods for such data are mainly correlated to classical signal processing concepts which have to be adapted to data residing at graph vertices. Usual data processing tasks include the spectrum analysis, filtering, denoising and compressing.

There are several approaches that cover the fundamental signal processing concepts on graphs. One of them[1] is focused on the Laplacian matrix as a base for graph signal processing methods that include a graph Fourier transform, graph representations in vertex and spectral domains, filtering, convolution, translation, modulation and dilation. The other approach[2] is based on the graph weighted adjacency matrix which is used to describe the connections between the nodes. Both the Laplacian and the adjacency matrices carry information about the type of connections between the graph vertices, which can be a physical distance, node similarity, dependency or some other factors, depending on the nature of the problem to be resolved. Both approaches are similar in some aspects but use different processing techniques and are applicable to different graph types.

Very effective methods for processing billion scale graphs are being constantly developed and one such technique[2] is successfully applied to Twitter social network (56 Gb, 2 billion edges) and the "YahooWeb" dataset, one of the largest graphs (120 Gb, 1.4 billion nodes, 6.6 billion edges). Representation of the graph wavelets[3,4] is also a popular research topic, as it can be used to obtain a scale-dependent analysis of communities in numerous types of networks.

*nbulatovic@ac.me; phone 00382 67 418-264; www.etf.ac.me

Compressive Sensing IV, edited by Fauzia Ahmad, Proc. of SPIE Vol. 9484
94840G · © 2015 SPIE · CCC code: 0277-786X/15/$18
doi: 10.1117/12.2177188

Fundamental signal processing on graph concepts such as the graph model, graph signals and a graph adjacency matrix are covered. Special attention is devoted to comparison between the approaches[1,5], with both advantages and drawbacks considered. Some disadvantages of these approaches, including the computational and memory requirements are addressed. Graph filtering methods are also discussed, including the linear and nonlinear filtering. The introduction of Dijkstra matrix as a graph signals' processing base is reviewed alongside with its differences and advantages over the Laplacian and adjacency matrices. The paper also covers the graph local neighborhood analysis with interesting features that can be extracted and used in graph processing. Lastly, the basic concept of compressed sensing is considered with respect to a graph structure.

2. PROCESSING METHODS OF THE GRAPH SIGNALS

2.1 Graph model and graph signals

Graph are used to represent high dimensional data in a manner that can be efficiently used for data processing. The graph $\mathrm{G}=\{\mathrm{V},\mathrm{A}\}$ is comprised of N vertices $\mathrm{V}=\{v_0,...,v_{N-1}\}$ connected by edges, and a weighted adjacency matrix A. The edge weights are usually defined by an application and are stored in a graph weighted adjacency matrix A. If there is an edge connecting the vertices v_i and v_j its weight is defined by $\mathrm{A}_{i,j}\neq 0$, otherwise $\mathrm{A}_{i,j}=0$.

A data set residing at the vertices of the graph forms a graph signal:

$$f:\mathrm{V}\to\mathrm{R},\ v_n\to f_n \tag{1}$$

which can be interpreted as a vector. The *i*th component $f(i)$ of the graph signal's vector represents the data value of the *i*th vertex in V. An example of a graph signal is shown in Figure 1.

2.2 Fundamental signal processing on graphs approaches

There are different base operators used for processing graph signals. The Laplacian-based method[1] focuses on calculating the graph eigenvectors and corresponding eigenvalues from the graph Laplacian matrix. The graph Laplacian is defined as:

$$\mathrm{L}=\mathrm{D}-\mathrm{A} \tag{2}$$

where D is a degree diagonal matrix with a value at the *i*th row equal to the sum of the weights of all the edges connected to vertex *i*. The graph Laplacian as a symmetric matrix has a complete set of real and ordered eigenvalues which are computed by the eigendecomposition of the Laplacian matrix. Eigenvalues carry the notion of graph signal's frequency. This method can be applied only to undirected graphs with real and nonnegative weights due to the properties of the Laplacian matrix.

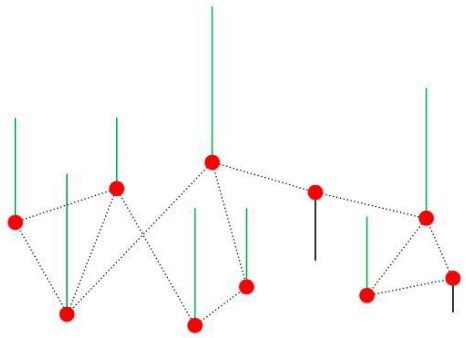

Figure 1. Representation of the graph signal residing on the vertices of the random graph. Green and black colored bars are the values of data components at associated vertices, with positive (green) and negative (black) values.

The second approach[5,6,7] focuses on a weighted adjacency matrix which is used as a shift operator on graphs. The shift operator is understood as an elementary filter in algebraic signal processing theory[8] and this concept is adapted to graphs. The graph frequency spectrum is defined by the eigenvalues of the eigendecomposed weighted adjacency matrix, instead of the graph Laplacian. The adjacency matrix must meet a condition to be diagonalizable in order to compute its eigendecomposition. If it is not the case, Jordan decomposition into generalized eigenvectors is used[5].

2.3 Graph Fourier transform

The concept of Fourier transform applied to graph signals is based on the expansion of a graph signal in terms of eigenfunctions of the applied operator.

In one case, the operator of interest is the Laplacian operator and the graph Fourier transform is defined as the expansion of the graph signal in terms of the eigenvectors of the graph Laplacian matrix:

$$\hat{f}(\lambda_l) = \sum_{i=1}^{N} f(i) u_l^*(i) \tag{3}$$

where λ_l is the eigenvalue of the corresponding eigenvector u_l and N is the number of graph vertices. Similarly, the inverse graph Fourier transform is defined as:

$$f(i) = \sum_{l=0}^{N-1} \hat{f}(\lambda_l) u_l(i) \tag{4}$$

In the approach based on weighted adjacency matrix, the operators of interest are graph filters. The weighted adjacency matrix is used as a graph shift matrix, where the graph shift is understood as an elementary graph filter. Then, the graph Fourier transform is defined as the expansion of the graph signal in terms of eigenvectors of the graph weighted adjacency matrix.

The expansion of a graph signal into eigenbasis is achieved via the multiplication with the inverse eigenvector matrix[10] which is defined as:

$$\hat{f} = \mathrm{V}^{-1} f \tag{5}$$

Then, the Fourier transformation matrix is represented as an inverse eigenvector matrix of the graph Laplacian or the graph weighted adjacency matrix, respectively for the mentioned approaches. The eigenvalues represent the oscillating frequency of the corresponding eigenvectors. The higher the order of the eigenvalue is, more rapidly the corresponding eigenvector' values vary across the graph.

2.4 The neighborhood impact on the graph signal

It is known from the matrix theory that the adjacency matrix A is diagonalizable if there exists an invertible matrix V which satisfies:

$$\mathrm{A}=\mathrm{V}\Lambda\mathrm{V}^{-1} \tag{6}$$

This is the eigendecomposition of the adjacency matrix. Columns of the matrix V represent eigenvectors of the adjacency matrix while Λ is diagonal matrix with corresponding eigenvalues $\lambda_0,...,\lambda_{N-1}$ of A.

Some interesting features of the graph signals could be found analyzing the relationship between the graph signal's value at *i*th vertex and the graph signal's values at neighbor vertices, in different time. The neighborhood of *i*th vertex is defined as $\mathrm{N}_i = \left\{ j \mid \mathrm{A}_{i,j} \neq 0 \right\}$. Mathematically speaking the relation can be formulated as follows:

$$f\left(i,t\right) = \alpha f\left(i,t-\Delta t\right) + \beta \sum_{j \in N_i} \mathrm{A}_{i,j} f\left(j,t-\Delta t\right) \tag{7}$$

where $f\left(i,t\right)$ is the signal value in moment *t* and α, β are the appropriate coefficients. The neighborhood impact is not necessarily linear like in (7). Depending on the problem, it can be formulated as a function $g\left(\mathrm{A}\right) f\left(j,t-\Delta t\right)$, where $g\left(\mathrm{A}\right)$ can be quadratic or any general function and $f\left(j,t-\Delta t\right)$ represents the values of the graph signal at *j*th

vertex in some previous moment $t-\Delta t$. The representation of matrix A through its eigendecomposition permits relatively easy way to compute any power of it because it only involves the powers of the diagonal matrix Λ. According to that, the power n of adjacency matrix is defined as $\mathrm{A}^n = \mathrm{V}\Lambda^n\mathrm{V}^{-1}$. This approach can be generalized to other matrix functions.

The expansion of the matrix function $g(\mathrm{A})$ into the Taylor series can be formulated as:

$$g(\mathrm{A}) = a_0\mathrm{I}+a_1\frac{\mathrm{A}^1}{1!}+a_2\frac{\mathrm{A}^2}{2!}+\ldots=\sum_{n=0}^{\infty}a_{\mathrm{n}}\frac{\mathrm{A}^n}{n!} \tag{8}$$

where $a_0, a_1, ..., a_n$ are the expansion coefficients. According to the previous considerations, the neighborhood impact for the general matrix function is:

$$g(\mathrm{A})f=\sum_{n=0}^{\infty}a_n\frac{\mathrm{V}\Lambda^n\mathrm{V}^{-1}}{n!}f=\mathrm{V}\sum_{n=0}^{\infty}a_n\frac{\Lambda^n}{n!}\mathrm{V}^{-1}f \tag{9}$$

where the term $\mathrm{V}^{-1}f$ is the Fourier transform of the graph signal f. It can be seen that the neighborhood impact on the graph signal is comprised of the variable and constant part. The resulted Fourier transform term $\mathrm{V}^{-1}f$ is the main reason for analyzing the expansion of the adjacency matrix function in Taylor series because it is constant and is independent of the matrix function type. It can be concluded that the analysis related to modelling the neighborhood impact on the graph signal is directly dependent on the graph signal's frequency spectrum.

3. ANALYSIS OF THE PROPOSED APPROACHES

3.1 Advantages and drawbacks

In order to process big data efficiently there are several factors that have to be taken into consideration. Some of the most important are the memory demand and the speed of the processing algorithms. The number of graph vertices is usually measured in hundreds, thousands or millions. If the number of graph vertices is N then the size of the graph base matrix (Laplacian or adjacency) is NxN. The complexity of computing the graph Fourier transform is $O(N^2)$ arithmetic operations (additions and multiplications) where the necessary eigendecomposition of the base matrix adds another $O(N^3)$ operations. Additionally, both computations require the access to the entire signal in memory which is slower than arithmetic operations and leads to performance decreasing. When processing a graph with a large set of vertices and edges, then the processing speed and memory capacity becomes a very demanding issue. One of the important drawbacks of proposed methods is that the eigendecomposition of the base matrix has to be computed every time the graph structure or graph signal's values have changed. It additionally increases the complexity of the processing algorithms.

The eigendecompositions of the graph Laplacian and weighted adjacency matrices have very similar definitions where the eigendecomposition of the graph Laplacian is computed as follows:

$$\mathrm{L}=\mathrm{D}-\mathrm{A}=\mathrm{V}(\mathrm{D}-\Lambda)\mathrm{V}^{-1} \tag{10}$$

If the graph is d-regular then the degree matrix becomes $\mathrm{D}=d\mathrm{I}$ and the eigenvectors of the graph Laplacian are equal to the corresponding eigenvectors of the graph adjacency matrix. Having in mind that there is a connection between the matrices L and A then it can be concluded that there is high similarity or equality in their corresponding eigenvalues.

4. GRAPH FILTERING

4.1 Graph filtering approaches

Graph filtering represents an extension of signal filtering in classical signal processing theory. Most common filters found in DSP are linear shift-invariant (LSI) ones.

The classical filters can be generalized to graph filters[5] in the following form:

$$h(\mathrm{A}) = \sum_{l=0}^{L-1} h_l \mathrm{A}^l \tag{11}$$

where the coefficients $\mathrm{h}_0, \mathrm{h}_1, \ldots, \mathrm{h}_1$ are called filter taps. The output of the graph filter is computed as:

$$\tilde{f} = h(\mathrm{A}) f \tag{12}$$

If the eigendecomposition of the adjacency matrix is computed, the graph filter's output can be written in the following form:

$$\tilde{f} = h(\mathrm{A}) f = h\left(\mathrm{F}^{-1}\Lambda\mathrm{F}\right) f = \mathrm{F}^{-1} h(\Lambda)\mathrm{F} f \tag{13}$$

where $\mathrm{F}=\mathrm{V}^{-1}$ is the graph Fourier transform matrix and $h(\Lambda)$ is a diagonal matrix with values $h(\lambda_n) = \sum_{l=0}^{L-1} h_l \lambda_n^l$ on the diagonal. The equation (13) can be formulated as the graph convolution theorem:

$$\mathrm{F}\tilde{f} = h(\Lambda)\hat{f} \tag{14}$$

with $h(\lambda_n)$ values representing the graph frequency response of the graph filter. From (14) it can be concluded that filtering the graph signal in the vertex domain is equivalent to multiplication of the graph signal's Fourier transform and the graph filter's frequency response.

In the Laplacian based approach, the graph spectral filtering is defined as follows:

$$\hat{f}_{out}(\lambda_l) = \hat{f}_{in}(\lambda_l)\hat{h}(\lambda_l) \tag{15}$$

Taking an inverse graph Fourier transform, the previous equation can be written as:

$$f_{out}(i) = \sum_{l=0}^{N-1} \hat{f}_{in}(\lambda_l)\hat{h}(\lambda_l) u_l(i) \tag{16}$$

Now if the equation (13) is considered and compared to the previously defined one (16), it can be concluded that the mentioned approaches have very similar filtering formulations. The equality is obvious if the following identities are taken into consideration:

$$\hat{f}_{in}(\lambda_l) \Leftrightarrow \hat{f}\,,\ \hat{h}(\lambda_l) \Leftrightarrow h(\Lambda)\,,\ u_l(i) \Leftrightarrow \mathrm{F}^{-1}$$

4.2 Graph signal denoising methods

One of the main applications of signal processing on graphs include graph signal filtering and denoising. Denoising methods from classical signal processing can be efficiently applied to graph datasets. Two cases have to be taken into consideration depending on the noise type which affected the graph signal, and these are linear and nonlinear filtering methods.

Linear filtering methods showed the best results in case the graph signal is affected by additive Gaussian noise:

$$\tilde{f} = f + \varepsilon \tag{17}$$

Graph spectral filtering (15) can be used as a base for implementation of classical signal processing filtering techniques such as Gaussian smoothing, bilateral filtering, total variation filtering, anisotropic diffusion, and nonlocal means filtering. Many of these filtering methods are widely used in image denoising[11]. Graph signal denoising is usually formulated as the optimization problem and solved using regularization methods[12,13].

Like in classical signal processing, when filtering the signal that is affected by the noise with impulsive characteristics, the median filters can be efficiently used for noise removal[14]. The concept of median filtering is based on running the filtering window over the input discrete sequence.

The output for the selected sequence of the input signal x is generally computed as:

$$y(n) = \text{MEDIAN}\left[x(n-N_L),...,\text{x}(n),...,\text{x}(n+N_R)\right] \tag{18}$$

where N_L and N_R are the nonnegative integers and $N=N_L+N_R+1$ is the window size. The samples of the input signal are sorted and the middle is taken as the output. The filtering equation (16) can be understood as a type of weighted sum and in case the filtering operator is applied, the resulted signal can be written in the following form:

$$f_{out}(i) = \text{T}\langle \hat{f}_{in}(\lambda_l) u_l(i) \Diamond \hat{h}(\lambda_l) \rangle, l \in [0, N-1] \tag{19}$$

where the operators of interest could be different types of weighted median filters. The operator $\Diamond$ can be a replication or any other operator used in filtering.

These are the fundamental filtering techniques, usually derived from the classical signal processing, which can be applied to graph signals with the aim of noise removal.

5. FURTHER STEPS IN GRAPH PROCESSING

The existing fundamental approaches in signal processing on graphs, were overviewed in previous sections. Advantages and drawbacks of the Laplacian and adjacency matrix based methods were analyzed. However, these are not the only operators that can be used. The properties of Dijkstra matrix, as a potential operator for graph signal processing, are considered in the first part of this section. Also, the local neighborhood analysis is conducted in the second part.

5.1 Application of the Dijkstra matrix in graph processing

The approach is based on introducing the Dijkstra matrix as a graph processing base and taking advantage of its features. The Dijkstra algorithm is generally used for finding the shortest paths between the vertices of the graph. The adjacency matrix values, used to describe connections between the graph vertices, depend on the nature of the problem. In order to apply Dijkstra algorithm to graph structure, some conditions have to be satisfied. The graph has to be connected and all the edges must have nonnegative weights assigned. These weights form a weighted adjacency matrix with zeros on the main diagonal and infinite (or large) values if there is no connection between the two graph vertices. After the weighted matrix is formed, the shortest paths are calculated using the Dijkstra algorithm. The procedure consists of finding the shortest path between two vertices going through the k vertex, connected with both. The path is calculated as a minimum between the direct path and the path through the k vertex which is formulated as:

$$\min(\text{A}(i,j), \text{A}(i,k) + \text{A}(k,j)) \tag{20}$$

Going through all the vertices and finding all the possible k paths, results in the output matrix W which consists of the shortest paths between all the graph vertices. If there are edges connecting the vertices v_i and v_j then the shortest path between them is defined by $\text{W}_{i,j} \neq 0$, otherwise $\text{W}_{i,j} = 0$.

The reason for introducing Dijkstra matrix instead of Laplacian or adjacency matrices is mainly based on decreasing rate of its eigenvalues. Considering that the eigenvalues represent graph signal's frequency content, the decreasing rate of the frequency components can play an important role in signal processing on graphs. Eigenvalues' decreasing rates for the adjacency and Dijkstra matrices are analyzed on the graph structure with 1000 vertices and are presented in Figure 2. The curves represent the logarithm of the normalized, ordered and interpolated eigenvalues. It can be seen that the Dijkstra matrix eigenvalues decrease more rapidly. The calculated decreasing coefficients of the beginning linear parts of the curves (colored in green) clearly determine the mentioned rate.

The characteristics of the eigenvalues determine the properties of the graph frequency spectrum. The faster the eigenvalues decrease there are fewer graph frequency components, which leads to many interesting applications. If the graph frequency spectrum is concentrated in fewer components it can be used as an important feature in graph signal's filtering and compression of the graph signal. Having in mind the considered features, the method based on the Dijkstra matrix could represent a promising concept in graph processing.

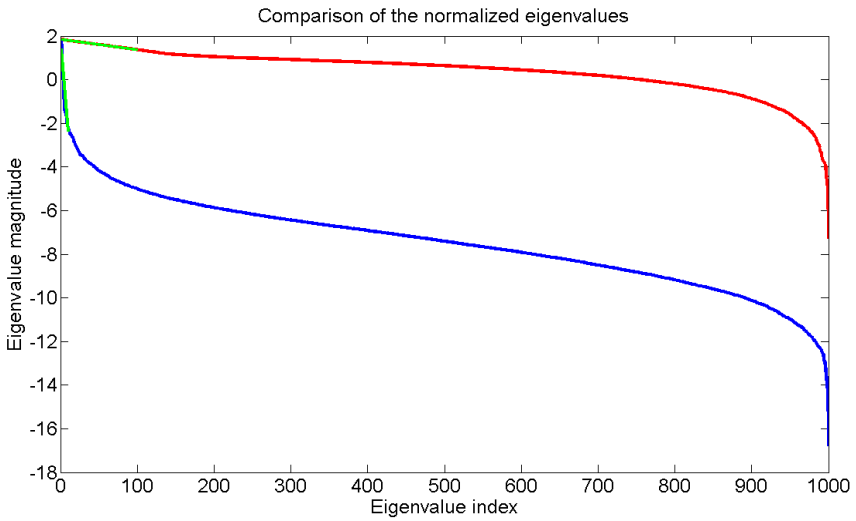

Figure 2. Decreasing curves of the normalized, ordered and interpolated logarithm of the eigenvalues of the weighted adjacency matrix (red) and Dijkstra matrix (blue). The beginning linear parts of the curves are colored in green.

5.2 Local neighborhood analysis

Some of the time-frequency analysis concepts in classical signal processing can be generalized and applied to graph signals in a vertex-frequency analysis. The windowed Fourier transform applied to graphs has been recently proposed[15]. Our research is based on extracting interesting features from local neighborhood analysis of the graph and finding possible applications.

The analysis is based on finding the local neighborhood of the graph vertices, for the specified distance expressed in number of edges from the central vertex in the neighborhood. The neighborhood size is determined by the number of vertices and edges included, which is proportional to a specified distance. The analysis is applied to well-connected graphs where every three adjacent vertices form the triangle of edges. When the neighborhood is found, the eigendecomposition of the local adjacency matrix can be computed. The resulted eigenvalues represent the local frequency content of the graph signal. Sometimes it is only important to determine the number of nonzero local eigenvalues. Instead of computing the eigendecomposition of the local adjacency matrix for every specified neighborhood, the number of nonzero eigenvalues can be found computing the local adjacency matrix's rank. The experiments showed that the rank of the adjacency matrix is equivalent to the number of its nonzero eigenvalues so it can save computational time and increase the speed of algorithms dealing with this kind of problem.

Increasing the distance of the included neighborhood vertices, the neighborhood subgraph more resembles the original graph so the graph is covered more rapidly. The results were analyzed for different neighborhood distances required to cover the whole graph, and are shown in Table 1.

Table 1. Average neighborhood size and rank of the local adjacency matrix for different graph sizes and specified distances.

Graph size (number of vertices)	Neighborhood distance	Average neighborhood size (number of vertices)	Average adjacency matrix rank
100	7	>95	>85
200	9	>190	>175
300	11	>290	>275
400	12	>390	>375
500	13	>480	>450
1000	16	>970	>940

6. GRAPH COMPRESSED SENSING

The compression of the graph signals and matrices is a concept recently proposed in signal processing on graphs. Different fundamental methods for graph signal processing were mentioned in the paper. These concepts can be used and applied to signal compression. In previous sections it was explained how the graph signal can be transformed and manipulated in the spectral domain. The graph Fourier transform can be used to store only a few signals' spectral coefficients and maintain its original properties. If the signal is smooth then its Fourier coefficients decay rapidly. It means that the original signal values can be reconstructed from the few spectral coefficients using the inverse Fourier transform[1,5-8].

One of the examples how it can be done in wireless sensor networks is considered in recent papers[16,17]. It is possible to collect and transmit measurements from a random subset of nodes (graph vertices) and after the interpolation, the measured data can be retrieved in the original form. Transmitting only the random subset of signal values leads to energy savings, increased processing speed and lower memory demand. It as also analyzed how the signal smoothness and the underlying graph structure can be used for better compressibility of the graph signals.

Very popular type of graph structures, called the expander graphs made a remarkable results in the field of compressed sensing. The expander-graph-based methods give better results, lower the complexity and improve the performance of the compressed sensing algorithms[18-20].

Graph structures are widely used in studying telecommunication networks and one such application is demonstrated in recent paper[21], which studied the network tomography from the compressive sensing on graphs perspective. The main focus was on recovering sparse vectors which represent the properties of the graph edges.

As it was shown, the concept of compressive sensing on graphs is in constant development and the methods proposed in this paper could be used in further research improvements related to this field.

ACKNOWLEDGEMENT

This research is supported in part by Ministry of science of Montenegro through national project "Intelligent search techniques for parametric estimation in communication and power engineering," FP7 Project Foremont, Project for establishment of pilot Montenegrin Centre of Excellence in Bioinformatics - BIO-ICT.

REFERENCES

[1] Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P., "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," IEEE Signal Processing Magazine 30(3), 83-98 (2013).
[2] Kang, U., Meeder, B., Papalexakis, E. E., and Faloutsos, C., "Heigen: Spectral analysis for billion-scale graphs," IEEE Transactions on Knowledge and Data Engineering 26(2), 350-362 (2014).
[3] Tremblay, N., and Borgnat, P., "Multiscale community mining in networks using spectral graph wavelets," Proc. EUSIPCO, 1-5 (2013).
[4] Tremblay, N., Borgnat, P., and Flandrin, P., "Graph empirical mode decomposition," Proc. EUSIPCO, 2350-2354 (2014).
[5] Sandryhaila, A., and Moura, J. M., "Discrete signal processing on graphs," IEEE Transactions on Signal Processing 61(7), 1644-1656 (2013).
[6] Sandryhaila, A., and Moura, J. M., "Discrete signal processing on graphs: Frequency analysis," IEEE Transactions on Signal Processing 62(12), 3042-3054 (2014).
[7] Sandryhaila, A., and Moura, J. M., "Discrete signal processing on graphs: Graph Fourier transform," Proc. ICASSP, 6167-6170 (2013).
[8] Puschel, M., and Moura, J. M., "Algebraic signal processing theory: Cooley–Tukey type algorithms for DCTs and DSTs," IEEE Transactions on Signal Processing 56(4), 1502-1521 (2008).
[9] Chen, S., Sandryhaila, A., Moura, J. M., and Kovacevic, J., "Signal Denoising on Graphs via Graph Filtering," Proc. GlobalSIP, 872-876 (2014).

[10] Lancaster, P., and Tismenetsky, M., [The theory of matrices: with applications], Academic press, (1985).
[11] Buades, A., Coll, B., and Morel, J. M., "A review of image denoising algorithms, with a new one," SIAM Multiscale Modeling and Simulation 4(2), 490-530 (2005).
[12] Rudin, L. I., Osher, S., and Fatemi, "E. Nonlinear total variation based noise removal algorithms," Physica D: Nonlinear Phenomena 60(1), 259-268 (1992).
[13] Elmoataz, A., Lezoray, O., and Bougleux, S., "Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing," IEEE Transactions on Image Processing 17(7), 1047-1060 (2008).
[14] Bovik, A. C., Huang, T. S., and Munson Jr, D. C., "A generalization of median filtering using linear combinations of order statistics," Proc. ICASSP, 1342-1350 (1983).
[15] Shuman, D. I., Ricaud, B., and Vandergheynst, P., "Vertex-frequency analysis on graphs," arXiv preprint, 1307-5708 (2013).
[16] Zhu, X., and Rabbat, M., "Graph spectral compressed sensing for sensor networks," Proc. ICASSP, 2865-2868 (2012).
[17] Zhu, X., and Rabbat, M., "Approximating signals supported on graphs," Proc. ICASSP, 3921-3924 (2012).
[18] Xu, W., and Hassibi, B., "Efficient compressive sensing with deterministic guarantees using expander graphs," IEEE Information Theory Workshop, 414-419 (2007).
[19] Xu, W., & Hassibi, B., "Further results on performance analysis for compressive sensing using expander graphs," Proc. ACSSC, 621-625 (2007).
[20] Jafarpour, S., Xu, W., Hassibi, B., and Calderbank, R., "Efficient and robust compressed sensing using optimized expander graphs," IEEE Transactions on Information Theory 55(9), 4299-4308 (2009).
[21] Xu, W., Mallada, E., and Tang, A., "Compressive sensing over graphs," Proc. INFOCOM, 2087-2095 (2011).