GRAPH SIGNAL SAMPLING VIA REINFORCEMENT LEARNING

Oleksii Abramenko, Alexander Jung

Department of Computer Science, Aalto University, Finland; firstname.lastname@aalto.fi

ABSTRACT

We model the sampling and recovery of clustered graph signals as a reinforcement learning (RL) problem. The signal sampling is carried out by an agent which crawls over the graph and selects the most relevant graph nodes to sample. The goal of the agent is to select signal samples which allow for the most accurate recovery. The sample selection is formulated as a multi-armed bandit (MAB) problem, which lends naturally to learning efficient sampling strategies using the well-known gradient MAB algorithm. In a nutshell, the sampling strategy is represented as a probability distribution over the individual arms of the MAB and optimized using gradient ascent. Some illustrative numerical experiments indicate that the sampling strategies obtained from the gradient MAB algorithm outperform existing sampling methods.

***Index Terms*—** machine learning, reinforcement learning, multi-armed bandit, graph signal processing, total variation.

1. INTRODUCTION

Modern information processing systems generate massive datasets which are typically heterogeneous partially labeled mixtures of different data types (audio, video, text). A successful approach to machine learning problems involving such datasets is based on exploiting their intrinsic network structure. In particular, we represent datasets by graph signals defined over an undirected graph, which reflects similarities between individual data points. The graph signal values encode label information which often conforms to a clustering hypothesis, i.e., the signal values (labels) of close-by nodes (similar data points) are similar. This graph signal representation allows using tools from graph signal processing (GSP) to data science or machine learning problems.

Two core problems within GSP are (i) how to efficiently sample graph signals, i.e., which signal values provide the most information about the entire dataset, and (ii) how to recover the entire graph signal from few signal values (samples). These problems have been studied in [1–6] which discussed convex optimization methods for recovering a graph signal from a small number of signal values observed on the nodes belonging to a given (small) sampling set. Sufficient conditions on the sampling set and clustering structure such that these convex methods are successful have been discussed in [4, 7]. In addition to this, the study [8] investigates performance of several graph sampling algorithms, such as uniform random sampling and experimentally designed sampling. Another research [9] addresses sampling in the frequency domain using discrete Fourier transform of a graph. Greedy sampling algorithm [10] tries to find near-optimal global solution by following the sequence of locally optimal decisions.

Contribution. In contrast to currently existing sampling methods, we propose a novel adaptive approach to the graph signal sampling by interpreting it as RL problem. In particular, we interpret online sampling algorithm as an artificial intelligence agent which chooses the nodes to be sampled on-the-fly. The behavior of the sampling agent is represented by a probability distribution ("policy") over a discrete set of different actions which are at the disposal of the sampling agent in order to choose the next node at which the graph signal is sampled. The ultimate goal is to learn a sampling policy which chooses signal samples that allow for a small reconstruction error.

Notation. The vector with all entries equal to zero is denoted $\mathbf{0}$. Given a vector $\mathbf{x}$ with non-negative entries, we denote by $\sqrt{\mathbf{x}}$ the vector whose entries are the square roots of the entries of $\mathbf{x}$. Similarly, we denote the element-wise square of the vector as $\mathbf{x}^2$.

2. PROBLEM FORMULATION

We consider networked datasets which are represented by an empirical graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The empirical graph is a simple undirected graph with nodes $\mathcal{V} = \{1, \ldots, N\}$, which are connected by edges $\{i, j\} \in \mathcal{E}$. Without loss of generality we consider datasets whose empirical graph is connected. However, our results can be easily extended to datasets whose empirical graph consists of several components. Each node $i \in \mathcal{V}$ represents an individual data point and an edge $\{i, j\} \in \mathcal{E}$ connects nodes representing similar data points. The distance $\mathrm{dist}(i, j)$ between nodes $i, j \in \mathcal{V}$ is defined as the length of the shortest path between them. The neighbourhood of node i is

$$\mathcal{N}(i) := \{j \in \mathcal{V} : \{i, j\} \in \mathcal{E}\}.$$

It will be handy to generalize the notion of neighbourhood and define, for some $r \in \mathbb{N}$, the r-step neighbourhood of a node $i \in \mathcal{V}$ as $\mathcal{N}(i, r) := \{j \in \mathcal{V} : \mathrm{dist}(i, j) = r\}$. The

1-step neighbourhood coincides with the neighbourhood of a node: $\mathcal{N}(i,1)=\mathcal{N}(i)$.

Besides the network structure, encoded in the empirical graph, datasets typically contain additional information in the form of labels $x[i] \in \mathbb{R}$ assigned to each data point $i \in \mathcal{V}$. These labels induce a graph signal $\mathbf{x}: \mathcal{V} \to \mathbb{R}$ defined over the empirical graph $\mathcal{G}$. We aim at recovering a graph signal $\mathbf{x}$ based on observing its values $x[i]$ only for nodes i belonging to a sampling set

$$\mathcal{M} := \{i_1, \ldots, i_M\} \subseteq \mathcal{V}.$$

Since acquiring signal values (labelling data points) is often expensive (requiring manual labor), the sampling set is typically much smaller than the overall dataset, i.e., $M = |\mathcal{M}| \ll N$. For a fixed sampling set size (sampling budget) M we want to choose the sampling set such that the signal samples $\{x[i]\}_{i \in \mathcal{M}}$ carry maximal information about the entire graph signal.

The recovery of the entire graph signal from (few) signal samples $\{x[i]\}_{i \in \mathcal{M}}$ is possible for clustered graph signals which do not vary too much over well-connected subsets of nodes (clusters) [4, 11]. We quantify how well a graph signal conforms with the cluster structure of $\mathcal{G}$ using the total variation (TV)

$$\|\mathbf{x}\|_{\rm TV} := \sum_{\{i,j\} \in \mathcal{E}} |x[j]-x[i]|.$$

Recovering a clustered graph signal from the signal samples $x[i]$, for $i \in \mathcal{M}$, can be accomplished by solving

$$\hat{\mathbf{x}}^{\mathcal{M}} \in \arg\min_{\tilde{\mathbf{x}}} \|\tilde{\mathbf{x}}\|_{\rm TV} \text{ s.t. } \tilde{x}[i]=x[i] \text{ for all } i \in \mathcal{M}. \quad (1)$$

This is a convex optimization problem with a non-differentiable objective function which precludes the use of simple gradient descent methods. However, the problem (1) has a particular structure which allows us to apply an efficient primal-dual optimization method [11, 12].

A simple but useful model for clustered graph signals is:

$$\mathbf{x} = \sum_{\mathcal{C} \in \mathcal{F}} a_{\mathcal{C}} \mathbf{t}_{\mathcal{C}}, \quad (2)$$

with the cluster indicator

$$t_{\mathcal{C}}[i] = \begin{cases} 1, \text{ if } i \in \mathcal{C} \\ 0 \text{ else.} \end{cases}$$

The partition $\mathcal{F}$ underlying the signal model (2) can be chosen arbitrarily in principle. However, our methods are expected to be most useful for clustered signals of the form (2) with a partition that matches intrinsic cluster structure of the empirical graph $\mathcal{G}$. In particular, if the underlying partition $\mathcal{F} = \{\mathcal{C}_1, \ldots, \mathcal{C}_{|\mathcal{F}|}\}$ consists of disjoint clusters $\mathcal{C}_l$ with small cut-sizes graph signals of the form (2) will have a small TV $\|\mathbf{x}\|_{\rm TV}$ which is favorable for the recovery method (1).

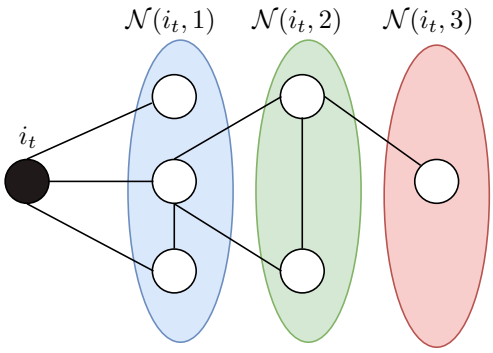

Fig. 1: The filled node represents the current location i_t of the sampling agent at time t. We also indicate the 1-, 2- and 3-step neighbourhoods.

3. SIGNAL SAMPLING AS REINFORCEMENT LEARNING

We consider the selection of the nodes to be sampled being carried out by an "agent" which crawls over the empirical graph $\mathcal{G}$. At each time step the agent can choose one of a finite number of H actions which we collect in the set $\mathcal{A} = \{1, ..., H\}$.

At a given time step t, the sampling agent chooses an action $a \in \mathcal{A}$ which refers to the number of hops the sampling agent performs starting at the current node i_t to reach a new node i_{t+1}, which will be added to the sampling set, i.e., $\mathcal{M} := \mathcal{M} \cup \{i_{t+1}\}$. In particular, the new node i_{t+1} is selected uniformly at random among the nodes which belong to its a-step neighbourhood $\mathcal{N}(i_t, a_t)$ (see Figure 1).

The problem of optimally selecting actions a_t can be formulated as a MAB problem. Action $a \in \mathcal{A}$ corresponds to one arm of a hypothetical MAB machine. In our setup, a sampling strategy (or policy) amounts to specifying a probability distribution over the individual actions $a \in \mathcal{A}$. We parametrize this probability distribution with a weight vector $\mathbf{w} = (w_1, \ldots, w_H)^T \in \mathbb{R}^H$ using the softmax rule:

$$\pi^{(\mathbf{w})}(a) = \frac{e^{w_a}}{\sum_{b \in \mathcal{A}} e^{w_b}} \quad (3)$$

The weight vector $\mathbf{w}$ is tuned in the episodic manner with each episode amounting to selecting sampling set $\mathcal{M}$ based on the policy $\pi^{(\mathbf{w})}$. At each timestep t the agent randomly draws an action a_t according to the distribution $\pi^{(\mathbf{w})}$ and performs transition to the next node i_{t+1} which is selected uniformly at random from the a_t-step neighbourhood $\mathcal{N}(i_t, a_t)$. As mentioned earlier, the node t_{t+1} is added to the sampling set, i.e., $\mathcal{M} := \mathcal{M} \cup \{i_{t+1}\}$. We also record the action a_t and add it to the action list, i.e., $\mathcal{L} := \mathcal{L} \cup \{a_t\}$. The process continues until we obtain a sampling set $\mathcal{M}$ with the prescribed size (sampling budget) M.

We aim at learning a policy $\pi^{(\mathbf{w})}$ that selects a sampling set $\mathcal{M}$ which allows for accurate recovery of the entire clus-

tered graph signal. The quality of the sampling set is measured by the mean squared error (MSE) obtained when recovering the graph signal using (1):

$$MSE := (1/N) \sum_{j \in \mathcal{V}} (x[j] - \hat{x}^{\mathcal{M}}[j])^2$$

Consequently, the reward R of our RL agent can be expressed using MSE as follows:

$$R := -MSE \tag{4}$$

In (4) we note that the reward is larger when MSE is smaller, which encourages the agent to select a sampling set minimizing MSE of graph signal recovery.

The obtained reward is associated with all actions/arms which contributed to picking samples into sampling set during the episode. For example, if the sampling set has been obtained by pulling arms 1, 2 and 5, the obtained reward will be associated with all these arms, because we do not know what is the exact contribution of the specific arm to the finally obtained reward.

The key idea behind gradient MAB is to adjust the weights $\mathbf{w}$ in the parametrization of $\pi^{(\mathbf{w})}$ (see (3)) so that the actions yielding higher rewards become more probable under $\pi^{(\mathbf{w})}$ [13, Chapter 2.8]. This is accomplished using a gradient ascent algorithm:

$$w_a := \begin{cases} w_a + \alpha R(1 - \pi(a)), a = a_k \\ w_a - \alpha R \pi(a), \forall a \neq a_k \end{cases} \tag{5}$$
$$\text{for } k = 1..M-1, a \in \mathcal{A}, a_k \in \mathcal{L}$$

The main difference between update rule (5) and [13, Eq. 2.12] is that in our case weights update is performed in the end of each episode and not after an arm pull. That is because we do not know reward immediately after pulling an arm and should wait until the whole sampling set is collected and reward is observed. The intuition behind the update equation (5) is that for each arm which has participated in picking a node into sampling set ($a = a_k$), the weight is increased, whereas weights w_a of the remaining arms ($\forall a \neq a_k$) are decreased. In both cases, the amount of weight increase/decrease is scaled by the reward obtained with help of this arm as well as by the learning rate α. For faster convergence we use mini-batch gradient ascent in combination with RMSprop technique [14] (see Algorithm 1 for implementation details). Obtained probability distribution $\pi^{(\mathbf{w})}$ represents sampling strategy which incurs the minimum reconstruction MSE when using the convex recovery method (1).

Algorithm 1 Online sampling and reconstruction

Input: graph $\mathcal{G}$, sampling budget M, batch size B, α
Initialize: $\mathbf{w} := \mathbf{0}, \nabla\mathbf{w} = \mathbf{0}, \mathbf{g} = \mathbf{0}, ep = 0$
repeat
  select starting node $i \in \mathcal{V}$ randomly
  $\mathcal{M} := \{i\}$
  $\mathcal{L} = \{\emptyset\}$
  for $t := 1; t < M$ **do**
    draw action a from the distribution $\pi^{(\mathbf{w})}$
    draw i_{next} from a-step neighbourhood $\mathcal{N}(i, a)$
    $\mathcal{M} := \mathcal{M} \cup \{i_{next}\}$
    $\mathcal{L} := \mathcal{L} \cup \{a\}$
    $i := i_{next}$
  end for
  $\hat{\mathbf{x}} \in \arg\min_{\tilde{\mathbf{x}}} \|\tilde{\mathbf{x}}\|_{\mathrm{TV}}$, s.t. $\tilde{x}[i] = x[i]$ for all $i \in \mathcal{M}$
  $R := -(1/N) \sum_{j \in \mathcal{V}} (x[j] - \hat{x}[j])^2$
  for $k := 1; k < M$ **do**
    for $a := 1; a \leqslant H$ **do**
      $\nabla w_a := \begin{cases} \nabla w_a + R(1 - \pi(a)), \text{if } a = \mathcal{L}[k] \\ \nabla w_a - R\pi(a), \text{otherwise} \end{cases}$
    end for
  end for
  $ep := ep + 1$
  if $ep \bmod B = 0$ **then**
    $\mathbf{g} := 0.9\mathbf{g} + 0.1(\nabla\mathbf{w})^2$
    $\mathbf{w} := \mathbf{w} + \alpha\nabla\mathbf{w}/\sqrt{\mathbf{g}}$
    $\nabla\mathbf{w} := \mathbf{0}$
  end if
until convergence is reached
Output: $\pi^{(\mathbf{w})}$

4. NUMERICAL RESULTS

Using synthetic networked data, we compare the performance of Algorithm 1 with two other existing approaches, i.e., random walk sampling (RWS) [15] and uniform random sampling (URS) [16, Section 2.3]. We generate an empirical graph using a stochastic block model [17] with 10 clusters. The cluster sizes are drawn from a geometric distribution with success probability $8/100$. The intra- and inter-cluster connection probabilities are chosen as $p = 7/10$ and $q = 1/100$. We then generate a clustered graph signal according to (2) with the signal coefficients $a_{\mathcal{C}_l} = l$ for $l = 1, 2, ..., 10$. In Figure 2, we depict a single realization of the specified above probabilistic model for the empirical graph $\mathcal{G}$.

We would like the agent to learn a policy which would allow sampling arbitrary graph instance conforming with the probabilistic model of the graph $\mathcal{G}$. However, if the agent learns a policy based only on one graph realization, the obtained policy may be biased towards this particular training instance and will not be able to generalize sampling strategy for the whole probabilistic model. In order to deal with this overfitting problem we generate $K = 500$ random graph realizations and for each one find policy $\pi_k^{(\mathbf{w})}$ by running Algorithm 1 for 10000 episodes, which is sufficient to reach convergence (see Figure 3). We then average the policies $\pi_k^{(\mathbf{w})}$

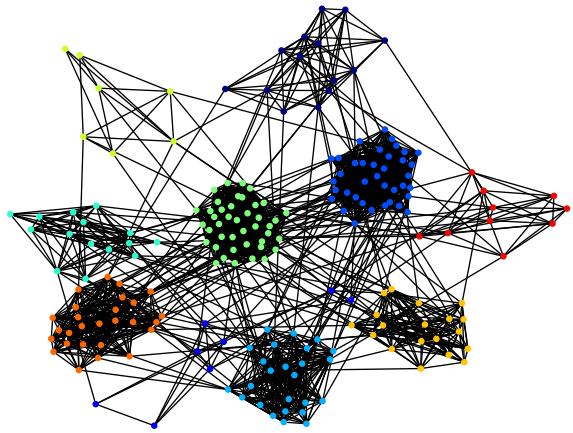

Fig. 2: Empirical graph obtained from the stochastic block model with $p = 7/10$ and $q = 1/100$.

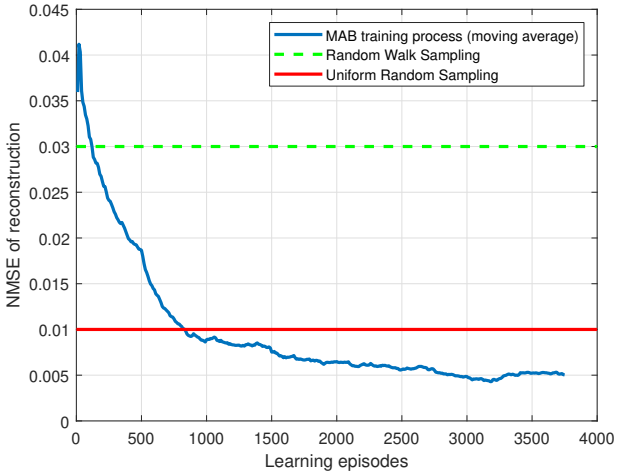

Fig. 3: Convergence of gradient MAB for one realization $\mathcal{G}^{(k)}$ (showing first 3700 episodes).

obtained for the individual realizations $\mathcal{G}^{(k)}$:

$$\pi^{(\mathbf{w})} := (1/K) \sum_{k=1}^{K} \pi_k^{(\mathbf{w})} \tag{6}$$

The finally obtained policy (6) (see Figure 4) now generalizes sampling strategy for the probabilistic model of graph $\mathcal{G}$ and can be used to sample arbitrary graph realization which conforms to this probabilistic model. We evaluate (6) by applying it to 500 new i.i.d. realizations of the empirical graph, yielding the sampling sets $\mathcal{M}^{(i)}$, $i = 1, \ldots, 500$, and measuring the normalized mean squared error (NMSE) incurred by graph signal recovery from those sampling sets:

$$NMSE_{\mathcal{G}_i} := \frac{\|\hat{\mathbf{x}}^{(i)} - \mathbf{x}^{(i)}\|_2^2}{\|\mathbf{x}^{(i)}\|_2^2}$$

$$NMSE := (1/500) \sum_{i=1}^{500} NMSE_{\mathcal{G}_i}$$

We perform similar measurements of the NMSE for random walk and random sampling algorithms under different

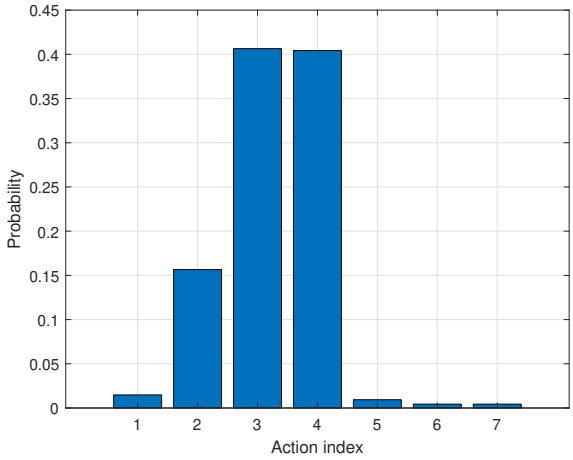

Fig. 4: Average policy for the probabilistic model of the empirical graph $\mathcal{G}$.

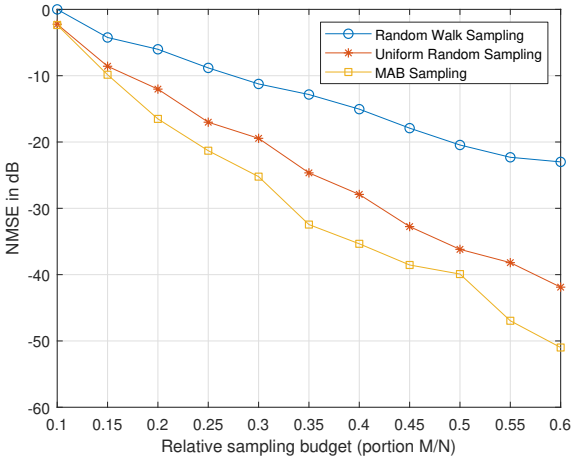

Fig. 5: Test set error obtained from graph signal recovery based on different sampling strategies.

sampling budgets and convert results to the logarithmic scale. From Figure 5 we obtain that for relative sampling budget 0.2 improvement in NMSE amounts to 5 dB in comparison to random sampling and 10 dB in comparison to random walk approach. This gap increases even more for the sampling budget 0.4, to 8 dB and 20 dB respectively.

5. CONCLUSIONS

We have proposed a novel approach to graph signal processing based on interpreting graph signal sampling and recovery as a RL problem. Using this interpretation lends naturally to an online sampling strategy which is based on determining an optimal policy which minimizes MSE of graph signal recovery. The proposed approach has been tested on a synthetic dataset generated in accordance to the stochastic block model. Obtained experimental results have confirmed effectiveness of the proposed sampling algorithm in the stochastic settings and demonstrated its advantages over existing approaches.

6. REFERENCES

[1] G. B. Eslamlou, A. Jung, N. Goertz, and M. Fereydoon, "Graph signal recovery from incomplete and noisy information using approximate message passing," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 6170–6174.

[2] Alexander Jung, Peter Berger, Gabor Hannak, and Gerald Matz, "Scalable graph signal recovery for big data over networks," in *Signal Processing Advances in Wireless Communications (SPAWC), 2016 IEEE 17th International Workshop on*. IEEE, 2016, pp. 1–6.

[3] Alexander Jung, Ayelet Heimowitz, and Yonina C Eldar, "The network nullspace property for compressed sensing over networks," in *Sampling Theory and Applications (SampTA), 2017 International Conference on*. IEEE, 2017, pp. 644–648.

[4] Alexander Jung, Nguyen Tran, and Alexandru Mara, "When is network lasso accurate?," *Frontiers in Applied Mathematics and Statistics*, vol. 3, pp. 28, 2018.

[5] James Sharpnack, Aarti Singh, and Alessandro Rinaldo, "Sparsistency of the edge lasso over graphs," in *Artificial Intelligence and Statistics*, 2012, pp. 1028–1036.

[6] A. Mara and A. Jung, "Recovery conditions and sampling strategies for network lasso," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*, Oct 2017, pp. 405–409.

[7] Alexander Jung and Madelon Hulsebos, "The network nullspace property for compressed sensing of big data over networks," *Frontiers in Applied Mathematics and Statistics*, vol. 4, pp. 9, 2018.

[8] Siheng Chen, Rohan Varma, Aarti Singh, and Jelena Kovačević, "Signal recovery on graphs: Fundamental limits of sampling strategies," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 539–554, 2016.

[9] Yuichi Tanaka, "Spectral domain sampling of graph signals," *IEEE Transactions on Signal Processing*, vol. 66, no. 14, pp. 3752–3767, July 2018.

[10] Luiz FO Chamon and Alejandro Ribeiro, "Greedy sampling of graph signals," *IEEE Transactions on Signal Processing*, vol. 66, no. 1, pp. 34–47, 2018.

[11] Alexander Jung, Alfred O Hero III, Alexandru Mara, and Sabeur Aridhi, "Scalable semi-supervised learning over networks using nonsmooth convex optimization," *arXiv preprint arXiv:1611.00714*, 2016.

[12] Antonin Chambolle and Thomas Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of mathematical imaging and vision*, vol. 40, no. 1, pp. 120–145, 2011.

[13] Richard S. Sutton and Andrew G. Barto, *Reinforcement learning: An introduction*, MIT Press, 2018.

[14] Tijmen Tieleman and Geoffrey Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[15] Saeed Basirian and Alexander Jung, "Random walk sampling for big data over networks," *2017 International Conference on Sampling Theory and Applications (SampTA)*, pp. 427–431, 2017.

[16] Gilles Puy, Nicolas Tremblay, Rmi Gribonval, and Pierre Vandergheynst, "Random sampling of bandlimited signals on graphs," *Applied and Computational Harmonic Analysis*, vol. 44, no. 2, pp. 446 – 475, 2018.

[17] Elchanan Mossel, Joe Neeman, and Allan Sly, "Stochastic block models and reconstruction," *arXiv preprint arXiv:1202.1499*, 2012.