Learning Shifting Communities Online in the Adversarial Block Model

Kiran Vodrahalli *

January 13, 2016

Abstract

The goal of this paper is to introduce regret-based analyses of the stochastic block model. We apply the framework of regret and a recent paper on the online learning of eigenvectors and extend the results to finding communities of a sequence of graphs, each graph drawn from a distribution where the community is exactly recoverable. We pay particular attention to the case where the communities for each graph shift over time and derive some properties of this setting in terms of regret bounds.

Contents

1 Introduction

As previous papers have tackled the fundamental information theoretic and computational limits for exact and partial recovery in the stochastic block model setting, we are interested in approaches for identifying the limits in the online setting: Namely,

*Department of Mathematics, Princeton University, USA, knv@princeton.edu

what kind of regret bounds can we get on the number of correctly classified nodes in the graph? This setting models graphs whose communities change over time, as is the case in a wide variety of settings, including social networks, neurons in the brain and others. Another important feature of the model is that we have to make a prediction **before** seeing the new graph, based on our history. Models with good regret ensure that average regret goes to zero, and the algorithm learns as it proceeds.

We are motivated also by the notion of robustness of algorithms. (MPW15) explains that "algorithms based on semidefinite programming are robust in ways that any algorithm meeting the information-theoretic threshold for community detection cannot be." Their analysis demonstrates that given a graph G, represented as a matrix B, which passes through an adversarial channel (this is known as the **semirandom block model**), the threshold for community detection changes from that of the stochastic block model. The adversary they consider is somewhat mild, a middle ground between average case and worst case analysis. We would like to approach adversarial guarantees on community recovery in graphs from the online regret point of view. In the most general case, with no restrictions on the kinds of graphs which may be seen, we designate this setting the **online adversarial block model**.

In this paper, we will impose various restrictions on the graphs which nature hands the learner, some of which are familiar from the stochastic block model and semirandom block model settings.

2 Background

2.1 Stochastic Block Model

How does one model clusters in a network? One famous and popular approach is the stochastic block model, which has been the subject of a lot of scrutiny recently.

Definition 2.1. The stochastic block model with 2 communities given by SBM(n, p, q) is a distribution on graphs where each of the n nodes is assigned a community 0 or 1, and where the probablity of edges existing between nodes in the same community is p and q across communities.

We will be concerned with the exact and partial recovery settings, as well as the setting where the parameters of the block model are not known. We take $k = 2$ communities for simplicity of exposition. The exact recovery setting deals with the case where complete reconstruction of the communities is information theoretically possible, and the partial recovery setting deals with the case where greater than 50% of the communities can be recovered. In a recent paper by (ABH14), the threshold for 2 communities for exact recovery was found. It was generalized in later works, namely (AS15a; AS15b).

2.2 Online Learning and Regret

Our goal in this paper is to detect communities in an online fashion, adversarially. Online learning introduces the notion of regret. There are a sequence of guesses the algorithm must make about the value of some quantity. After the algorithm makes a

guess, the true answer is revealed and the algorithm experiences some loss. Then the algorithm updates their guess for the next round. The goal is to perform as closely as possible to the optimal value to guess repeatedly. Formally,

Definition 2.2. Regret. Let $\{x_1, \cdots, x_T\}$ be a sequence of guesses that a learner makes. At each step, the learner experiences a loss $f_t(x_t)$. Let x^* be the optimal choice in hindsight if the learner were only able to play a single value. The regret of the sequence is defined to be

$$\text{Regret}(T) = \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x^*) \tag{1}$$

We often consider average regret, in which case divide the given quantity by T. A regret function is good if the average regret vanishes.

In the setting of our paper, we will define loss functions based on some notion of optimal separation. For instance, in the online learning of eigenvectors ((GHM15)): The "value" we guess at each iteration is an eigenvector x_t, and we experience a loss after a matrix A_t is revealed, according to the quadratic form $-x_t^T A_t x_t$. In particular, (GHM15) were able to prove the following theorem regarding the regret of learning an eigenvector online.

Definition 2.3. Eigenvector Regret.
For a sequence of matrices $\{A_t\}_{t=1}^T$, a learner plays a unit norm vector x_t before seeing the next matrix and experiences a loss. Then, the regret is

$$\begin{aligned} \text{Regret}(T) &= \max_{\|x\|_2=1} \sum_{t=1}^{T} x^T A_t x - \sum_{t=1}^{T} x_t^T A_t x_t \\ &= \lambda_{max}\left(\sum_{t=1}^{T} A_t\right) - \sum_{t=1}^{T} x_t^T A_t x_t \end{aligned} \tag{2}$$

The main theorem (cf. Theorem 3 in (GHM15)) is given by

Theorem 2.4. *(Garber, Hazan, Ma 2015).*
Let $c = \sqrt{\frac{T}{n}\max(1, \ln(T/n))}$. *Then for meta-algorithm* $\mathcal{A}$ *which attempts to learn eigenvectors, we bound the expected regret for eigenvectors with*

$$\mathbb{E}[\text{Regret}_{\mathcal{A}}(T)] = \mathcal{O}\left(\sqrt{nT\max(1, \ln(T/n))}\right) \tag{3}$$

The algorithm $\mathcal{A}$ *is well-known* `Follow the Perturbed Leader` *algorithm from the online learning literature, which requires the ability to sample low-rank matrices over symmetric* $n \times n$ *matrices, as well as an eigenvector oracle.*

We will later attempt to apply this to the problem of online community learning.

3 The Adversarial Block Model Setting

In this section we present several ways of adapting the online setting to community detection in graphs. Because we may relax the conditions on the graphs that we see, we denote this broad class of models the **Adversarial Block Model** (ABM).

Definition 3.1. Adversarial Block Model. (ABM)
Let $\{G_t\}_{t=1}^T$ be a sequence of graphs, each equipped with a partition on its vertices (A_t, B_t) where $A_t, B_t \subseteq V(G_t)$ and $A_t \cup B_t = V(G_t)$, $A_t \cap B_t = \emptyset$. We also fix $|V(G_t)| = n$. We also must define a loss function f_t such that $f_t(G_t)$ is some measure related to the communities.

We now proceed to lay out more detailed settings of this general setting, and relate this setting to the stochastic block model. There are four simple ways to select what remains invariant across the graphs.

1. Community Invariance: Suppose each $G_t \sim SBM(n, p, q)$ and that $(A_t, B_t) = (A_{t+1}, B_{t+1})$ for all $t \in [T]$. We can choose the parameters $p = \frac{\alpha \log(n)}{n}, q = \frac{\beta \log(n)}{n}$, $\alpha > \beta$, which allows for exact recovery in each individual G_t given the graph, as proven in (ABH14).
2. Parameter Invariance: We can choose many different parameters to remain invariant. We will give several examples later in the paper of some potential parameters of interest. For now, let us suppose that $G_t \sim SBM(n, p, q)$ (this implies fixed α, β) and that the sizes of the communities $|A_t|$ and $|B_t$ remain fixed over t, but that the actual set of nodes A_t, B_t do not remain fixed. This setting is possibly the simplest version of the ABM where something novel is gained by considering the online setting.
3. Phase Transition Invariance: This setting allows us to more loosely restrict α, β, but still remain in the setting where exact recovery is possible. In other words, we only require that for some function $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, we have that $f(\alpha, \beta)$ is restricted to certain subsets of $\mathbb{R}$.
4. Non-Stochastic Parameter Invariance: Here, we allow different kinds of restrictions on the graphs G_t, which are not necessarily drawn from a block model. One example is restricting the properties of the principal eigenvector of the adjacency matrix of the graph.

We could also consider a case where we do not even allow the regime of the block model parameters to remain the same, but that seems too hard.

The most important aspect of the online setting is the requirement on predicting what the next graph will look like **without seeing it beforehand**. In the case where the communites are all drawn from the same distribution, this aspect is trivial since after seeing the first graph (and perhaps losing on the first step), we can just run the $O(n^{1+\epsilon})$ algorithm given by (AS15a) and predict the same thing each time afterwards to get perfect accuracy (also note that this is highly computationally efficient).

However, in the case where the communites change at each step, perhaps adversarially though remaining in a setting where offline exact recovery is possible, we would like to be able to get a regret guarantee on the performance of our guess. Note that it is not possible to simply run the $O(n^{1+\epsilon})$ algorithm at each step, since we have to submit our guess for the communities before seeing the next graph.

Another reason online algorithms with regret analysis for the SBM are interesting is because the analysis can potentially hold for regimes of the stochastic block model where the parameters are not completely known. Regret based analysis could also help determine upper bounds on accuracy in the partial recovery setting.

4 Defining the Optimization Task

In order to apply the ABH setting to the problem of recovering communities in graphs, we need to specify the optimization problem we are trying to solve. We borrow the setting from (ABH14): Our algorithm attempts to find two communities such that (#between-community edges) − (#cross-community edges) is maximal. A community vector $x \in H = \frac{1}{\sqrt{n}}\{-1, 1\}^n$ denotes which communities each of the n nodes belongs to. Then, we define the matrix B_t for the graph G_t by

$$B_t^{(ij)} = \begin{cases} 1 & : (i,j) \in E(G_t) \\ 0 & : (i,i) \\ -1 & : (i,j) \notin E(G_t) \end{cases}$$

The optimization program is given by

$$\max_{x \in H} x^T B x \tag{4}$$

In the same manner as (ABH14), we also give an SDP relaxation

$$\max_{X \succeq 0, X_{ii}=1} Tr(BX) \tag{5}$$

They prove that this convex relaxation in fact exactly solves the problem, when certain conditions are imposed on the values of α, β:

Theorem 4.1. *(Abbe, Bandeira, Hall 2014).*
If $(\alpha - \beta)^2 > 8(\alpha + \beta) + \frac{8}{3}(\alpha - \beta)$, with high probability the SDP has a unique solution given by gg^T, where $g \in H$ whose entries correspond to the communities as defined above.

The previous two equations form the basis for our two attempts at analyzing online algorithms from the perspective of regret. Regret analysis using the first program will be denoted spectral optimization, and analysis using the second program will be denoted semidefinite optimization.

We can now describe two simple loss functions that could be used in online optimization: $f_{spec}(G_t) = -x^T B x$ and $f_{semi}(G_t) = -\mathrm{Tr}(BX)$.

5 Shifting Communities

In this section, we consider the parameter-invariant settings described before. Specifically, we consider two kinds of parameters: Restrictions upon α, β and restrictions upon the principal eigenvector of a matrix. Now, how can we approach solving the optimization problem posed in the previous section in the online ABM setting, such that we can apply previously-known theorems from the online optimization literature? We list some possibilities:

1. The community detection problem is actually convex. Here we use a result from (ABH14).

2. We might then think to examine the regime in which the maximal eigenvector is already in $\{-1, 1\}^n$, in which case we can directly apply the results from (GHM15), and merely provide an explanation of what partial recovery-regret means in terms of eigenvalues. For instance, the case where all non-diagonal vectors are 1 is a case where this holds. Note that this reflects the case where the graph is a complete graph, and thus there is only one community. However, this is the only case where the eigenvector is a solution, and is thus uninteresting.
3. We might try restricting some non-stochastic parameters: For instance, requiring the vector derived from taking the sign of each component of the maximal eigenvector to solve the optimization problem opens up a new door. Here, we additionally need to explain how to convert the notion of eigenvalue-regret into optimization regret, after which we connect to partial recovery-regret. Here, we run some numerical experiments to see what the degredation the signing of the maximal eigenvector causes to $x^T Ax$, and could try to prove something about the performance of this algorithm in the online setting. We try to establish an empirical difference in $x^T Ax$ as a function of n, the dimension of the space x lives in. We also try to prove a result about what the performance of the sign algorithm looks like on average.
4. Suppose the signed maximal eigenvector is not the perfect solution: We can still analyze the performance of this algorithm with respect to regret.
5. We may also want to try to take existing spectral algorithms ((YP14)) and use online eigenvector learning in order to bring them to the community detection setting.
6. We can consider relaxing solutions to the problem to be defined over the unit sphere, not $\{-1, 1\}^n$. Then, we can give an interpretation of the maximal eigenvector as a solution to the community detection problem.

5.1 Online Semidefinite Program Optimization

First we note that the semidefinite program given in (ABH14) can be extended to the online setting, in the same parameter regime given in the paper. We give explicit regret bounds for solving the SDP using online convex optimization, making use of convexity through Theorem 3 given in (ABH14) (previously stated above). We can simply use standard online gradient descent to solve this problem, provided that all graphs G_t are drawn from SBM(n, p, q) with $p = \frac{\alpha \log(n)}{n}, q = \frac{\beta \log(n)}{n}$ with $(\alpha - \beta)^2 > 8(\alpha + \beta) + \frac{8}{3}(\alpha - \beta)$ remaining invariant for all G_t. Recall that we define our loss function to be $f_t(X_t) = -\text{Tr}(B_t X_t)$, which is a linear function if we think of X_t as a vector. Therefore, the gradient is given by $\frac{\partial -\text{Tr}(B_t X_t)}{\partial X_t} = -B_t^T$ which can be seen by simple calculation. Therefore, note that the gradient matrix has a 0-diagonal as well, and thus the optimization does not affect the diagonal elements of X_t. Furthermore, since the diagonal of B_t is always 0, the diagonal elements of X_t are not used in the loss function. Therefore, we can satisfy the requirement $X_{ii} = 1$ without any penalty by setting them that way from the start.

The gradient update is given by

$$X_{t+1} = \prod_{\mathbb{S}_n^+} X_t - \eta_t \nabla f_t(X_t) \tag{6}$$

where we use projection onto the (convex) positive semidefinite cone $\mathbb{S}_n^+$ given by

$$\prod_{\mathbb{S}_n^+}(X) = U \begin{bmatrix} \max(0,\lambda_1) & 0 & \cdots & 0 \\ 0 & \max(0,\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \max(0,\lambda_n) \end{bmatrix} U^T$$

where we have the spectral decomposition $X = U\Sigma U^T$ and $\lambda_1 \geq \cdots \geq \lambda_n$ are the eigenvalues (see (HM11)). Then, since we know that this problem has a unique solution from Theorem 4.1, we can restrict our convex set even further so that it is clearly bounded. The only restriction we require is that all symmetric $n \times n$ matrices which have ± 1 on off-diagonals and 0-on diagonals are contained in the restricted convex set. Therefore, there is some upper bound on the diameter of the convex set we project onto D and since the gradient is linear, we also have that it is Lipschitz and there is some bound G on the magnitude of the gradient. Since $G = -B^T$, we can upper bound it with $G \leq n^2 - n$. Therefore, we can apply the regret of online gradient descent to this setting (Haz15):

Theorem 5.1. *Regret of Online Gradient Descent.*
With step sizes $\eta_t = \frac{D}{G\sqrt{t}}$ *for* $t \in [T]$*, for all* $T \geq 1$ *we have*

$$\text{Regret}(T) \leq \frac{3}{2} GD\sqrt{T} = \mathcal{O}(\sqrt{T}) \tag{7}$$

We remark that the analysis in this section is by no means optimal, and it is highly possible better algorithms and regret bounds can be obtained. We just used online gradient descent for simplicity, to illustrate the point.

5.2 Online Spectral Optimization

Remark 5.2. First, we note that the vein of this work strays from the recent approaches of Abbe and Sandon. The emphasis in the recent papers by those authors is on the fact that the analysis was information-theoretic and algorithm agnostic. However, in this paper we return to a more algorithmic-approach centric analysis in order to extend to online learning. Our whole approach is based on massaging the problem we have to the point where we can almost directly apply previous results from online learning and regret bounds to our problem. This approach is almost certainly sub-optimal, and should be approached from the point of view of giving new definitions and perhaps a new thresholding function which serves the same purpose as the CH-divergence in (AS15a).

Now we pivot away from the semidefinite optimization setting to the spectral setting. Here, we still assume that each of the graphs the learner sees are in the exact recovery setting, to ensure that the solution to the spectral optimization problem corresponds with communities.

Letting $H = \frac{1}{\sqrt{n}}\{-1,1\}^n$ be the Boolean hypercube, we define the pseudoeigenvalue of B_t to be

$$\rho = \max_{x \in H} x^T B_t x$$

and its accompanying vector x_ρ. Recall the definition of regret for eigenvectors from the Background section, we now extend it to pseudoeigenvectors:

Definition 5.3. Pseudoeigenvector Regret.
For a sequence of matrices $\{A_t\}_{t=1}^T$, a learner plays a vector $x_t \in H$ before seeing the next matrix and experiences a loss. Then, the regret is

$$\begin{aligned} \text{Regret}(T) &= \max_{x \in H} \sum_{t=1}^{T} x^T A_t x - \sum_{t=1}^{T} x_t^T A_t x_t \\ &= \rho \left(\sum_{t=1}^{T} A_t \right) - \sum_{t=1}^{T} x_t^T A_t x_t \end{aligned} \tag{8}$$

We now define an algorithm to approximate this quantity.

Definition 5.4. `Approximate-`ρ Algorithm.

1. Calculate $x^* = \text{argmax}_{\|x\|_2=1} x^T B_t x$ (calculate principal eigenvector).
2. Calculate $\text{sgn}(x^*)$, where sgn is the function sending real numbers to ± 1 and where sgn is applied elementwise.
3. Use $\text{sgn}(x^*)$ as an approximation of x_ρ.

Note that (YP14) perform a similar trick on the top two eigenvectors in their algorithm for community detection. We analyze this algorithm instead for simplicity.

The motivation behind this setup is essentially that we want to apply regret bounds from (GHM15) to online eigenvector computation (hence step 1 of the algorithm). If we can characterize the gap between $\text{sgn}(x^*)$ and x_ρ, then we can potentially carry over regret bounds on the eigenvalues to the quantity we want.

First, we empirically assess the approximate-ρ algorithm. Instead of comparing directly to ρ, we examine a worst-case situation. We must have that $\rho \leq \lambda_{max}$ since $H \subset S^n$, the unit sphere. Therefore, the ratio $\frac{\text{sgn}(x^*)^T \cdot A \cdot \text{sgn}(x^*)}{\lambda_{max}} \leq \frac{\text{sgn}(x^*)^T \cdot A \cdot \text{sgn}(x^*)}{\rho}$ where A has 0-diagonal and ± 1 elsewhere. Interestingly, Figure 1 appears to demonstrate an asymptote lower-bounding this ratio, potentially allowing for a constant factor conversion between the regrets:

$$c < \frac{\text{sgn}(x^*)^T \cdot A \cdot \text{sgn}(x^*)}{\lambda_{max}} \leq \frac{\text{sgn}(x^*)^T \cdot A \cdot \text{sgn}(x^*)}{\rho}$$

We generated the graph in Figure 1 by writing code in Matlab to generate random matrices and calculate their top eigenvectors and eigenvalues.

Therefore, we might suppose a lower bound for the principal eigenvector x^* of the form $\frac{\text{sgn}(x^*)^T \cdot A \cdot \text{sgn}(x^*)}{\lambda_{max}} \geq 1 - \hat{\epsilon}$ for some $\hat{\epsilon} \in [0, 1]$ (certainly this will be true if we fix the size of the matrix at $n \times n$, it is unclear whether the asymptotics described in Figure 1 continue as $n \to \infty$).

We might generalize this assumption to a stronger conjecture:

Conjecture 1. Quality of $\text{Sgn}(x)$.
For any $x \in S^n$, there exists some $\hat{\epsilon} > 0$ such that

$$\text{sgn}(x)^T \cdot A \cdot \text{sgn}(x) > x^T A x (1 - \hat{\epsilon}) \tag{9}$$

This equation amounts to a statement about $\text{sgn}(x)$: Namely that the "quality" of the vector x does not degrade by too much if fed through a sgn function. We can modify

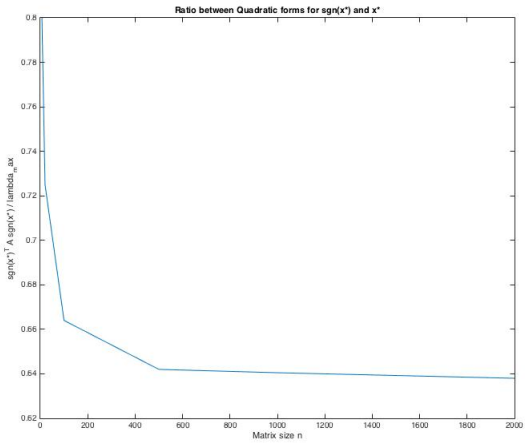

Figure 1: Empirical Average Ratio Between λ_{max} and $\text{sgn}(x^*)^T \cdot A \cdot \text{sgn}(x^*)$

the statement to be even stronger. If we sample $x \in S^n$ randomly (and do not just use the principal eigenvector x^*, as we did in our experiment), then

$$\mathbb{E}_x[\text{sgn}(x)^T \cdot A \cdot \text{sgn}(x)] = \mathbb{E}_x[x^T A x] \tag{10}$$

Assuming this conjecture, we can demonstrate an interesting statement about the "regret" involved in our optimization problem:

$$\begin{aligned}\mathbb{E}[\rho\left(\sum_{t=1}^{T} A_t\right) - \sum_{t=1}^{T} \text{sgn}(x_t)^T \cdot A_t \cdot \text{sgn}(x_t)] &< \mathbb{E}[\lambda_{max}\left(\sum_{t=1}^{T} A_t\right) - \sum_{t=1}^{T} \text{sgn}(x_t)^T \cdot A_t \cdot \text{sgn}(x_t)] \\ &< \mathbb{E}[\lambda_{max}\left(\sum_{t=1}^{T} A_t\right) - \sum_{t=1}^{T} x_t^T A_t x_t] \\ &= \mathcal{O}\left(\sqrt{nT \max(1, \ln(T/n))}\right) \text{ by Theorem 2.4}\end{aligned} \tag{11}$$

This demonstrates that given the conjecture, the simple `Approximate-`ρ algorithm applied to the modified `Follow the Perturbed Leader` algorithm from (GHM15) should be able to online learn communities in the shifting-community setting relatively well, in terms of regret.

Remark 5.5. Intuition behind the spectral regret bounds.
We have been giving regret bounds in terms of proxy-values instead the actual proportion of correctly recovered nodes (i.e., nodes assigned to the correct community, as in partial recovery). We take a few steps to remedy this dissonance in this note. First recognize that an optimal answer to the optimization problem $\max_{x_t \in H} x_t^T B_t x_t$ over the Boolean hypercube $H = \frac{1}{\sqrt{n}}\{-1, 1\}^n$ exists, since the problem is a combinatorial optimization problem and we are in the setting where the graph is recoverable. Suppose the optimal value is y, and we are attempting to construct y with our variable z. We can measure the difference between the vectors with the l_2 norm. Each incorrect choice of component for z costs us $\left(2\frac{1}{\sqrt{n}}\right)^2 = \frac{4}{n}$. Thus if there are k mistakes, we

have $\|y - z\|_2^2 = \frac{4k}{n}$. Then, if we consider z as the average of all the attempts of the algorithm,

$$\begin{aligned} \text{Regret}(T) &= y^T \left(\sum_{t=1}^{T} A_t \right) y - \sum_{t=1}^{T} z_t^T A_t z_t \\ &= \sum_{t=1}^{T} y^T A_t y - z_t^T A_t z_t = \sum_{t=1}^{T} \left(\|y\|_{A_t} - \|z_t\|_{A_t} \right) \\ &= \left(\sum_{t=1}^{T} \|y\|_{A_t} \right) - \left(\sum_{t=1}^{T} \|z_t\|_{A_t} \right) \end{aligned} \tag{12}$$

We also have $\|Ty - \sum_{t=1}^{T} z_t\|_2^2 = T^2 \|y - z\|_2^2 = T^2 \frac{4k}{n}$. It remains for future work to find a relationship expressing the difference of two induced norms with the l_2 norm of the difference.

References

[ABH14] E. Abbe, A. S. Bandeira, and G. Hall. Exact Recovery in the Stochastic Block Model. 2014, arXiv:1405.3267v4 [cs.SI].

[AS15a] E. Abbe and C. Sandon. Community detection in general stochastic block models: fundamental limits and efficient recovery algorithms. 2015, arXiv:1503.00609v2 [math.PR].

[AS15b] E. Abbe and C. Sandon. Recovering communities in the general stochastic block model without knowing the parameters. 2015, arXiv:1506.03729v1 [math.PR].

[GHM15] D. Garber, E. Hazan, and T. Ma. Online Learning of Eigenvectors. *Proceedings of The 32nd International Conference on Machine Learning*, 2015.

[Haz15] E. Hazan. *Introduction to Online Convex Optimization.* 2015, Preprint at: http://ocobook.cs.princeton.edu/OCObook.pdf.

[HM11] D. Henrion and J. Malick. Projection methods in conic optimization, 2011, At: http://www.optimization-online.org/DB_FILE/2011/03/2961.pdf.

[MPW15] A. Moitra, W. Perry, and A. S. Wein. How Robust are Reconstruction Thresholds for Community Detection? 2015, arXiv:1511.01473v1 [cs.DS].

[YP14] S. Yun and A. Proutiere. Community Detection via Random and Adaptive Sampling. 2014, arXiv:1402.3072v2 [cs.SI].