Computational-Statistical Tradeoffs in learning Graphical models

Abrar Zahin
Arizona State University

Akarshan Sajja
Arizona State University

Anirudh Rayas
Arizona State University

Mujahed Syed
Arizona State University

Vishnu Vaidya
Arizona State University

Abstract

In this review we explore the computational statistical tradeoffs in structure learning of graphical models. Towards this end we begin with a survey of an algorithm for learning the Ising model, namely we look at (1) which talks about a combinatorial greedy approach which runs in $\tilde{O}(n^2)$ but requires doubly exponential number of samples from the distribution. Concurrently we will also look at (2) which talks about learning Gaussian graphical models in an online set-up, the paper claims to have a low runtime compared to other existing works however there is a complimentary paper (3) to this which achieves nearly optimal sample complexity but runtime is large. We ask the following question, is there a computational lower bound for learning Gaussian graphical models with nearly optimal sample complexity. Therefore as a first step towards formulating the tradeoff for Gaussian graphical models we review (4) which characterizes the computational lower bounds for inferring combinatorial structures such as clique detection and nearest neighbor for gaussian graphical models.

1 Introduction

A typical learning algorithm has two components to it, one is the statistical component which talks about the number of training samples the algorithm would require in order to learn the true hypothesis with a low risk and the second is the computational aspect of the algorithm which describes how efficient the learning algorithm is. With the advent of complex machine learning tasks, there is a growing need for our learning algorithms to be both statistically optimal and computationally efficient. While classic statistical theory addresses the issue of optimality, it however largely ignores the computational aspect of it. Recently there has been a huge interest in studying these two aspects under one general framework, however it turns out that there exists a tension between the optimal number of samples required and run-time of an algorithm which is generally called the Computational-Statistical trade-off. Therefore a natural question to ask is, if whether or not it is possible for efficient algorithms to achieve information theoretic limits, which is the main theme of our project[1]. We intend to explore the following question, what is the minimum computational complexity to achieve nearly optimal limits for learning graphical models.

More broadly speaking, we live in the era of big data where complex machine learning tasks require massive datasets which creates a fundamental problem that lies at the intersection of

[1]Code for implementations can be found at `https://github.com/swish-coder/SML_Project.git`

computational and statistical sciences since our algorithms need to make statistical inferences with low error but at the same time the algorithm is constrained by computational budget such as time and space. This fundamental dichotomy arises due to the fact that computational complexity theory views the increase in the size of the dataset as a source of complexity that can be tamed via algorithms or hardware, however classic statistical theory views the same as a source of simplicity because classic asymptotic results can be invoked for large samples. However classical statistical theory gives no guidance as to how to design algorithms such that a certain level of inferential accuracy is achieved when one imposes limited time budget constraints. There are two main themes of research in this area, the notion of algorithmic weakness (5) and the idea of Coresets.(6)

Roughly speaking algorithmic weakening is a procedure where as data accumulates one would desire to switch to simpler more computationally efficient algorithmic strategies to achieve the same desired risk. However in (6) another parameter is added ie. space s. Informally speaking the idea of coresets is to compress a large dataset to a small one by keeping only the most representative elements and throwing out the other points. The paper claims that it is possible to do this without incurring much computational cost. This paper talks about how trade-offs between time and data behave for a fixed $\epsilon(p)$ by tuning the space parameter s.

1.1 Formally stating Time-Data Tradeoffs

To illustrate the time-data trade-off more formally, consider the class of parameter estimation problems, we say that an inference procedure belongs to a class $\mathbb{TD}(t(p), n(p), \epsilon(p))$ if a p dimensional parameter underlying the unknown population can be estimated with the help of n iid samples with a risk of $\epsilon(p)$ by an algorithm with run-time of $t(p)$. In this formalism, classical estimation theory emphasizes the trade-offs between the second and third parameters (ie. data and risk). However, the main focus of this topic is to fix $\epsilon(p)$ to a desired level of accuracy and then investigate the trade-offs between the first two parameters namely run-time and data size.

For the sake of qualitative comparison of inference algorithms, consider the graph in fig (1) where the x-axis represents the sample complexity and the y-axis denotes the time complexity. For example, procedure A achieves the minimax lower bound ie. the minimum number of samples required to achieve the desired level of $\epsilon(p)$ with no constraints on the run-time, determining such fundamental limits has been the focus of classical estimation theory, however determining similar computational lower bounds corresponding to horizontal lines in the graph has been an open problem in computational complexity theory.

2 An Efficient Algorithm for Learning Ising Models

Problem. In this section, we briefly summarize the contents of this paper (1). The author considers the problem of learning a specific form of discrete graphical model. He considers an Ising model on a graph $G = (V, E)$ with $|V| = p$. ∂i stands for the neighbourhood of a node i in the graph. Each of the configurations $x \in \{-1, +1\}^V$ on the nodes of the graph, is assigned a probability according to

$$\mathbb{P}(x) = \exp\Big(\sum_{\{i,j\} \in \mathcal{E}} \theta_{ij} x_i x_j + \sum_{i \in V} \theta_i x_i - \Phi(\theta) \Big) \tag{1}$$

Here $\Phi(\theta)$ is a log-partition function or a normalizing constant. Therefore, the model is parameterized by $\{\theta_{ij}\}_{\{i,j\} \in E} \cup \{\theta_i\}_{i \in V} \in \mathbb{R}^{E \cup V}$. They assume that $\alpha \leq |\theta_{ij}| \leq \beta$, for all θ_{ij}, for some constants α, β such that $0 < \alpha \leq \beta$, and that $|\theta_i| \leq h$, for all θ_i, for some h.

Structure learning. A structure learning algorithm for a graphical model takes in a set of samples $X^{(1)}, \ldots, X^{(n)}$ sampled from the distribution and returns the graph G that describes the structure of the model. The performance of the algorithm is characterized by the reconstruction error of the algorithm

$$P_\theta(\phi(X^{1:n}) \neq G) \tag{2}$$

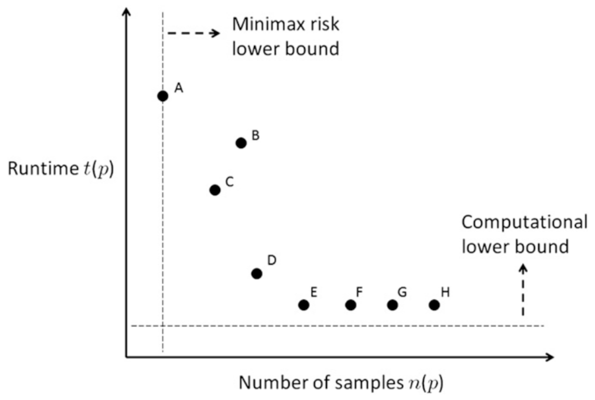

Figure 1: The plot represents the computational statistical trade-offs for a given estimation problem and for a desired level of accuracy. The points A-H represents different inference procedures and the horizontal and vertical lines represents lower bounds for sample and time complexity

where $\phi(X^{1:n})$ is the graph returned by the algorithm on being given n samples $X^{1:n}$.

Influence of a variable. The proposed algorithm uses a certain *conditional influence* of one variable on another. For nodes $u, i \in V$, subset of nodes $S \in V \setminus \{i, j\}$, and a certain configuration $x_S \in \{-1, +1\}^S$, the conditional influence is defined as

$$\nu_{u|i;x_S} := \mathbb{P}(X_u = 1|X_i = 1, X_S = x_s) - \mathbb{P}(X_u = -1|X_i = 1, X_S = x_s) \tag{3}$$

The average version of the above quantity is defined as a weighted average

$$\nu^{\text{avg}}_{u|i;x_S} := E(\lambda_i(X_S)|\nu_{u|i;X_S}|). \tag{4}$$

The *empirical conditional influence* $\hat{\nu}^{\text{avg}}_{u|i;x_S}$ is defined suitably in terms of the empirical versions of the constituent quantities: $\hat{\lambda}_i(X_S)$ and $\hat{\nu}_{u|i;x_S}$, which are computed from an estimated probability distribution $\hat{P}$.

The LearnNbhd algorithm. The proposed algorithm is described. The algorithm returns the neighbourhood of the input node u. τ is an important parameter of the algorithm that we need to choose.

Algorithm 1 LearnNbhd$(X^{(1)}, \ldots, X^{(n)}, \tau, u)$

```
Pseudo-neighbourhood:
    1. Let $S = \varnothing$
    2. Let $(i^*, \eta^*) = (\arg\max_i \hat{\nu}_{u|i;x_S}, \max_i \hat{\nu}_{u|i;x_S})$
    3. If $\eta^* \geq \tau$, then add $i^*$ to $S$
    4.   Else go to Step 6
    5. Repeat Steps 2 to 4
Pruning:
    6. For each $i \in S$ if $\hat{\nu}_{u|i;x_S} \leq \tau$ remove $i$
    7. Output $S$
```

The algorithm starts with an empty set for the pseudo-neighbourhood for u. As long as there exists a node with influence on i greater than or equal to τ, it keeps adding the node that has maximum influence on u to the pseudo-neighbourhood. The authors prove that the pseudo-neighbourhood S so constructed contains fully the neighbourhood of u if a certain condition $\mathcal{A}(l^*, \epsilon^*)$ is satisfied, where l^* and ϵ^* are parameters that depend on the τ that we set. They then prove that the *pruning* stage of the algorithm removes no neighbours when $\mathcal{A}(l^*, \epsilon^*)$ is satisfied. This implies that the S returned is the neighbourhood of u. They prove that for suitable values of l^* and ϵ^*, ensured by the right choice of τ, the condition $\mathcal{A}(l^*, \epsilon^*)$ is met with probability $\mathbb{P}(\mathcal{A}(l^*, \epsilon^*)) > 1 - \delta$ for arbitrarily small $\delta > 0$ if the necessary statistical requirement is met.

Statistical performance. If the number of samples observed n is such that

$$n \geq \exp\{c\alpha^{-c'} e^{c'' d(\beta d + h)}\} \cdot \log(\frac{p}{\delta}) \tag{5}$$

where c, c', c'' are numerical constants, and d is the maximum degree of the graph G, LearnNbhd returns the correct neighbourhood for all u with probability greater that $1 - \delta$.

Computational performance. The runtime for computation on a graph G of p nodes is $\mathcal{O}(p \log p)$.

3 Information Theoretic Optimal Learning of Gaussian Graphical Models

3.1 Setup:

Let $G = (V, E)$ (where $V = [p]$) is an undirected graph on the vertex set $[p] = \{1, 2 \dots p\}$ with edge set $E \subseteq \binom{p}{2}$. Each vertex of the graph is associated with the components of zero mean Gaussian Random vector $\underline{X} \in \mathbb{R}^p$, covariance matrix $\Sigma \in \mathbb{R}^{p \times p}$ and Inverse covariance matrix $\Theta \triangleq \Sigma^{-1}$. $\underline{X}$ is said to be Markov w.r.t to the Graph G if for any pair of vertices i, j, $\{i, j\} \notin E$ implies X_i and X_j are conditionally independent given $X_{[p] \setminus \{i,j\}}$. $\forall \{i, j\} \notin E$, $\Theta_{ij} = 0$ and thus our goal is to learn the non-zero entries of Θ. In Gaussian graphical model reconstruction task, we are interested in algorithms which can output an accurate estimate $\widehat{G}$ of G, i.e, $\mathbb{P}\left(G \neq \widehat{G}\right) > 1 - \delta$ for a given confidence $\delta > 0$. This paper (3) first points out the information-theoretic lower bound (IT) lower bound for minimum number of samples n^* required for reconstructing a sparse graph, which is given as follows

$$n^* > \max\left\{\frac{\log\binom{p-d}{2} - 1}{4\kappa^2}, \frac{2\left(\log\binom{p}{d} - 1\right)}{\log\left(1 + \frac{d\kappa}{1-\kappa}\right), \frac{d\kappa}{1+(d-1)\kappa}}\right\} \tag{6}$$

where $\kappa \triangleq \min\limits_{(ij \in E)} \frac{|\Theta_{ij}|}{\sqrt{\Theta_{ii}\Theta_{jj}}}$ and denotes the minimum normalized edge strength.

Note that bound in eq. (6) depends only on the parameters of the underlying graph, not on any additional assumptions. Then they proposed two algorithms named *Sparse Least-squares Inverse Covariance Estimator* (Slice) and *Degree- constrained Inverse Covariance Estimator* (Dice), and their sample complexity scales like $d + \frac{32}{\kappa^2}\log\left(\frac{4p^{d+1}}{\delta}\right)$ and $2d + \frac{192}{\kappa^2} d \log p + \frac{64}{\kappa^2}\log\left(\frac{4d}{\delta}\right)$ respectively. Both Dice and Slice consist of three steps and a brief overview of the steps are given as follows.

3.2 DICE

3.2.1 Estimating conditional variances:

In this step, Dice obtains an estimate of the conditional variances $\widehat{\Theta_{ii}}$ of each variable $i \in [p]$ conditioned on all the neighbors of i by solving the following optimization problem

$$\frac{1}{\widehat{\Theta_{ii}}} = \min_{\widehat{\beta} \in \mathbb{R}^{p-1}} L_i\left(\widehat{\beta}, \widehat{\Sigma}\right) = \frac{1}{n}\sum_{k=1}^{n}\left(x_i^k + \sum_{j \neq i} \beta_{ij} x_j\right)^2 \text{, such that } ||\widehat{\beta}||_0 \leq d \tag{7}$$

3.2.2 Iterative Support Testing:

Fix $i \in V$ and this subroutine will test all obtained candidate neighborhoods for i. Consider a candidate neighborhood $B_1 \subset V \backslash \{i\}$ with $|B_1| = d$. The objective of this subroutine is to obtain the true neighborhood $B_i \subseteq B_1$. This testing proceeds as follows.

1. Fix some adversarial neighborhoods $B_2 \subset V \backslash \{\{i\} \cup B_1\}$ with cardinality d.
2. Let $B_iB_j = B_i \cup B_j$ and compute regression coefficients $\widehat{\beta}_{iB_1B_2} = -\widehat{\Sigma}^{-1}_{B_1B_2,B_1B_2}\widehat{\Sigma}^{-1}_{B_1B_2,i}$ and B_1 is a neighborhood if $\forall B_2$,

$$\max_{j \in B_2} \widehat{\kappa}_{ij} \triangleq |\widehat{\beta_{ij}}| \sqrt{\frac{\widehat{\Theta_{ii}}}{\widehat{\Theta_{ij}}}} < \frac{\kappa}{2} \tag{8}$$

   There are two key things to notice in eq. (8)

   - Assume that $\widehat{\beta}_{iB_1B_2}$ is correct, then $\forall B_2$ and $\forall j \in B_2$, estimate $\widehat{\kappa}_{ij} \approx \kappa_{ij}$ (where κ_{ij}=0)
   - If $\exists j$ such that $j \in B_i \backslash B_1$, then condition in eq. (8) will fail make B_1 fail the criterion.

3.2.3 Eliminate non-edges:

Given that we have a candidate neighborhood B_1 such that $|B_1| = d$ and $B_i \subseteq B_1$, this subroutine will eliminate all extra edges from B_1 by computing the estimated edge strength $\widehat{\kappa}_{ij}$ $\forall j \in B_1$ and discarding any $j \in B_1$ as an edge if $\widehat{\kappa}_{ij} < \frac{\kappa}{2}$

3.3 SLICE

Their second proposed algorithm named Slice offers better computational complexity by trading off in sample complexity which exploits mixed integer programming formulation in Phase 3 from Dice. Key steps of slice are as follows

3.3.1 Least Squares with l_0- constraint

$$\widehat{\beta}_i = \min_{\widehat{\beta} \in \mathbb{R}^{p-1}} L_i\left(\widehat{\beta}, \widehat{\Sigma}\right) = \frac{1}{n}\sum_{k=1}^{n}\left(x_i^k + \sum_{j \neq i} \beta_{ij} x_j\right)^2, \quad \text{s.t } ||\widehat{\beta}||_0 \leq d \tag{9}$$

By comparing eq. (7) and eq. (9) we can see that the purpose of slice is to estimate regression coefficients rather than estimating the conditional variances as it was in dice.

3.3.2 Estimate the support

After estimating $\widehat{\beta}_i$ $\forall i \in V$, the estimate of edge-set $\widehat{E}$ can be obtained by following thresholding procedure

$$\widehat{E} = \left\{ (i,j) \in V \times V : \sqrt{|\widehat{\beta}_{ij} \times \widehat{\beta}_{ji}|} > \frac{\kappa}{2} \right\} \tag{10}$$

3.3.3 Implementation as a mixed integer quadratic program

In order to prevent an exhaustive search over all possible size d neighborhood of each vertex $i \in V$, when d is big enough, this phase of slice algorithm formulates the problem as a significantly faster mixed integer quadratic program. In the following formulation L and U denote upper and lower bounds on the regression. variables.

$$\begin{aligned}
\min_{\widehat{\beta}\in\mathbb{R}^{p-1}} \quad & \beta_i^T \widehat{\Sigma}_{\bar{i}\bar{i}} + 2\widehat{\Sigma}_{\bar{i}i} + \widehat{\Sigma}_{ii} && \text{(11a)}\\
\text{such that} \quad & s_{ij}L \leq \beta_{ij} < s_{ij}U, \ \ \forall j \neq i && \text{(11b)}\\
& \textstyle\sum_{j\neq i} s_{ij} = d && \text{(11c)}\\
& s_{ij} \in \{0,1\} \ \ \forall j \neq i && \text{(11d)}
\end{aligned}$$

3.4 Condition Number dependence

One important aspect of this paper is that their proposed algorithms are not sensitive to condition number parameter absent in IT lower bound (eq. (6)). They illustrated this fact by sketching a sequence of matrices which has a growing condition number whereas sample complexity of DICE and SLICE are not scaled accordingly.

4 Learning Gaussian Graphical Models via Multiplicative Weights

Consider the preceding setup for GGM reconstruction. We will now provide a brief overview of the online algorithm proposed in (2). Let X_i and $\underline{X}_{\bar{i}}$ denote i^{th} coordinate and all other coordinates except i of $\underline{X}$ respectively.We can express $\mathbb{E}\left[X_i|\underline{X}_{\bar{i}}\right]$ by a linear combination of measurements from other $p-1$ nodes as follows,

$$\mathbb{E}\left[X_i|\underline{X}_{\bar{i}}\right] = \sum_{j\neq i}\left(-\frac{\Theta_{ij}}{\Theta_{ii}}\right)X_j = \underline{w}_i \cdot \underline{X}_{\bar{i}}$$

Where $\underline{w}_i \in \mathbb{R}^{p-1}$ represents corresponding weights of samples from other $p-1$ nodes estimate X_i. In what follows is a brief listing of the core parts the online algorithm. For a detailed description of different parameters see the paper.

Algorithm 2 Learning weight vector for a node i

```
 1: Input: $N$ samples of $X$, $\underline{v}^{(0)}$ as $\left(\frac{1}{p},\frac{1}{p},\ldots,\frac{1}{p}\right)\in\mathbb{R}^n$, learning rate, $\beta$ distribution vector
 2: $\underline{\rho}$, $\lambda_i = \sum_{i\neq j}\left|\frac{\theta_{ij}}{\theta_{ii}}\right|$ and $\max_i \lambda_i \leq \lambda$.
 3: Output: A “good” approximation $\underline{v}^*$ of $\underline{w}_i$.
 4: for $n = 1$ to $N$ do
 5:     $\underline{\rho}^{(n)} = \frac{\underline{v}^{(n-1)}}{||\underline{v}^{(n)}||_1}$    // $\underline{v}^{(n-1)}, \underline{\rho}^{(n)} \in \mathbb{R}^n$
 6:     $\underline{l}^{(n)} = (1/2)(\underline{1} + \underbrace{(\lambda\underline{\rho}^{(n)}\cdot\underline{x}^{(n)} - y^{(n)})}_{\text{prediction error}}\underline{x}^{(n)})$
 7:     $\forall i \in [p]$, $\underline{v}_i^{(n)} = \underline{v}_i^{(n-1)}\underline{\beta}^{l_i^{(n)}}$
 8: end for
 9: Get $N$ candidate weight vectors.
10: Further use $M$ samples to see which candidate vector exhibits smallest empirical risk.
11: Return $\underline{v}_i^* = \lambda\underline{\rho}^n$, where $n$ denotes the candidate weight vector with smallest empirical
    risk.
```

Although the algorithm presented in (2) is the adoption of SPARSITRON algorithm proposed in (7) for Gaussian case, (2) tackle some challenges due to the continuous and unbounded nature of the problem which prohibits the use of several parts of the analysis in (7). Their sample complexity scales like $\mathcal{O}\left(\left(\frac{\lambda}{\kappa}\right)^4 \log^3\frac{p}{\delta}\right)$

5 Computational-Statistical trade-offs in Combinatorial Inference

As a first step towards formulating the computational-statistical trade-offs for structure learning in undirected gaussian graphical models which has not been addressed to the best of our knowledge we take a look at (4) which talks about trade-offs for combinatorial inference in gaussian graphical models. The fundamental question is to quantify the minimum computational complexity to achieve information theoretic limits in inferring combinatorial structures like clique detection, nearest neighbor graph and distinguishing graphs with large clique against small clique using an oracle computational model. More precisely, the paper defines two topological properties called the weak and strong edge densities, μ and μ' respectively which characterizes the trade-off,in particular their main result shows that if μ and μ' are of different orders then the information theoretic limit is not achievable by any tractable algorithm.

Let $X = (X_1, \dots, X_d)^T \in \mathbb{R}^d$ be a d-dimensional random vector which follows a multivariate normal distribution $\mathcal{N}(0, \Theta^{-1})$, let $G(\Theta) = (E, V)$ be the corresponding undirected graphical model. Given n independent observations $x_1, \dots, x_n$ we are interested in the hypothesis testing problem

$$H_0 : G \in \mathcal{G}_0 \quad versus \quad H_1 : G \in \mathcal{G}_1$$

where $\mathcal{G}_0$ and $\mathcal{G}_1$ are two disjoints sets of graphs and graphs in $\mathcal{G}_1$ share the combinatorial structure of interest. We will now look at a few definition and state the main result.

Definition 5.1. Let M be an oracle computational model, an algorithm $\mathcal{A}$ is defined as a tuple $M(\mathcal{Q}, T, q_{\text{init}}, \delta_{t \in [T]})$, where $\mathcal{Q}$ is the query space, T is the maximum number of rounds the algorithm is allowed to query the oracle, q_{init} is the initial query and δ_t decides the $(t+1)^{th}$ query based on the previois queries and their returns, in addition if δ_t returns $HALT$ then the algorithm terminates. Let $\mathcal{Q}_{\mathcal{A}}$ be the set of all queries that $\mathcal{A}$ queries the oracle. We define the computational complexity as $|\mathcal{Q}_{\mathcal{A}}|$

Definition 5.2. Let $\mathcal{C}_0$ and $\mathcal{C}_1$ be the set of precision matrices corresponding to $G(\Theta)$ in $\mathcal{G}_0$ and $\mathcal{G}_1$, we define the minimax risk of testing $\mathcal{C}_0$ against $\mathcal{C}_1$ under M with a a statistical query oracle r as

$$\mathcal{R}_n(\mathcal{C}_0, \mathcal{C}_1, \mathcal{A}, r) = \inf_{\psi \in \mathcal{H}(\mathcal{A}, r)} \left[\sup_{\theta \in \mathcal{C}_0} \mathbb{P}_\theta(\psi = 1) + \sup_{\theta \in \mathcal{C}_1} \mathbb{P}_\theta(\psi = 0) \right] \tag{12}$$

where $\mathcal{H}(\mathcal{A}, r)$ is the set of all test functions.

Definition 5.3. If there exists an oracle r such that

$$\liminf_{n \to \infty} \mathcal{R}_n(\mathcal{C}_0, \mathcal{C}_1, \mathcal{A}, r) = 1 \tag{13}$$

then any hypothesis test computed by an algorithm based on at most T queries under the computational model is asymptotically powerless.

Definition 5.4 (Null alternative separator)**.** Let $G_0 = (V, E_0) \in \mathcal{G}_0$ be some graph under the null hypothesis. We call a collection of edge sets $\mathcal{E}$ a null alternative separator with null base G_0 if for all edge sets $S \in \mathcal{E}$, we have $S \cap E_0 = \emptyset$ and $(V, E_0 \cup S) \in \mathcal{G}_1$

The fundamental hardness of the testing problem depends on the parameter spaces $\{\Theta_0\}$ and $\{\Theta_S\}_{S \in \mathcal{E}}$, thus the computational hardness of the problem depends on the structure of $\mathcal{E}$. The paper defines two quantities namely the weak edge density and vertex cut ratio which captures the notion of computational hardness.

Definition 5.5 (Weak edge density)**.** For a null alternative separator $\mathcal{E}$, we define it's weak edge density as

$$\mu = \max_{S, S' \in \mathcal{E}} \frac{|S \cap S'|}{|V(S \cap S')|^2} \tag{14}$$

Intuitively, the weak edge density measures the concentration of critical edges that changes a graph from $\mathcal{G}_0$ to $\mathcal{G}_1$. Therefore larger the value of μ easier it is to distinguish between $\mathcal{G}_0$ and $\mathcal{G}_1$. Similarly, we define the notion of strong edge density whose role would be evident in the main result. It is defined as

$$\mu' = \max_{S, S' \in \mathcal{E}} \frac{|S \cap S'|}{|V(S \cap S')|} \tag{15}$$

It's called the strong edge density since μ' always dominates μ.

Theorem 5.1. Suppose that we have a null alternative separator $\mathcal{E}$ with the null base $\mathcal{G}_0$. Under the oracle computational model M, if we require the number of queries $T \leq d^{\nu}$ for some constant $\nu > 0$, then for some sufficiently small constants κ_1 and κ_2 we have

- **Information theoretic bound:** If $\theta \leq \kappa_1/\sqrt{\mu' n}$, any hypothesis test is asymptotically powerless.
- **Computationally-Efficient bound:** If $\theta \leq \kappa_2/\sqrt{\mu n}$, any hypothesis test computed by a polynomial time algorithm is asymptotically powerless

Therefore, computational-statistical trade-offs appear when $\mu << \mu'$.

References

[1] G. Bresler, "Efficiently learning ising models on high degree graphs," *CoRR*, 2014.

[2] A. Chaturvedi and J. Scarlett, "Learning gaussian graphical models via multiplicative weights," *arXiv preprint arXiv:2002.08663*, 2020.

[3] S. Misra, M. Vuffray, and A. Y. Lokhov, "Information theoretic optimal learning of gaussian graphical models," in *Conference on Learning Theory.* PMLR, 2020, pp. 2888–2909.

[4] H. Lu, Y. Cao, Z. Yang, J. Lu, H. Liu, and Z. Wang, "The edge density barrier: Computational-statistical tradeoffs in combinatorial inference," in *International Conference on Machine Learning*, 2018, pp. 3247–3256.

[5] V. Chandrasekaran and M. I. Jordan, "Computational and statistical tradeoffs via convex relaxation," *Proceedings of the National Academy of Sciences*, vol. 110, no. 13, pp. E1181–E1190, 2013.

[6] M. Lucic, M. Ohannessian, A. Karbasi, and A. Krause, "Tradeoffs for space, time, data and risk in unsupervised learning," in *Artificial Intelligence and Statistics.* PMLR, 2015, pp. 663–671.

[7] A. Klivans and R. Meka, "Learning graphical models using multiplicative weights," in *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS).* IEEE, 2017, pp. 343–354.