Learning Gaussian Graphical Models via Multiplicative Weights

Anamay Chaturvedi
Khoury College of Computer Sciences
Northeastern University
chaturvedi.a@northeastern.edu

Jonathan Scarlett
Depts. Computer Science & Mathematics
National University of Singapore
scarlett@comp.nus.edu.sg

Abstract

Graphical model selection in Markov random fields is a fundamental problem in statistics and machine learning. Two particularly prominent models, the Ising model and Gaussian model, have largely developed in parallel using different (though often related) techniques, and several practical algorithms with rigorous sample complexity bounds have been established for each. In this paper, we adapt a recently proposed algorithm of Klivans and Meka (FOCS, 2017), based on the method of multiplicative weight updates, from the Ising model to the Gaussian model, via non-trivial modifications to both the algorithm and its analysis. The algorithm enjoys a sample complexity bound that is qualitatively similar to others in the literature, has a low runtime $O(mp^2)$ in the case of m samples and p nodes, and can trivially be implemented in an online manner.

1 Introduction

Graphical models are a widely-used tool for providing compact representations of the conditional independence relations between random variables, and arise in areas such as image processing [Geman and Geman, 1984], statistical physics [Glauber, 1963], computational biology [Durbin et al., 1998], natural language processing [Manning and Schütze, 1999], and social network analysis [Wasserman and Faust, 1994]. The problem of *graphical model selection* consists of recovering the graph structure given a number of independent samples from the underlying distribution. Two particularly prominent models considered for this problem are (generalized) Ising models and Gaussian models, and our focus is on the latter.

In the Gaussian setting, the support of the sparse inverse covariance matrix directly corresponds to the graph under which the Markov property holds [Wainwright and Jordan, 2008]: Each node in the graph corresponds to a variable, and any two variables are independent conditioned on a separating subset of nodes.

In this paper, we present an algorithm for Gaussian graphical model selection that builds on the *multiplicative weights* approach recently proposed for (discrete-valued) Ising models [Klivans and Meka, 2017]. This extension comes with new challenges due to the continuous and unbounded nature of the problem, prohibiting the use of several parts of the analysis in [Klivans and Meka, 2017] (as discussed more throughout the paper). Under suitable assumptions on the (inverse) covariance matrix, we provide formal recovery guarantees of a similar form to other algorithms in the literature; see Section 1.2 and Theorem 9.

1.1 Related Work

Learning Gaussian graphical models. The problem of learning Gaussian graphical models (and the related problem of inverse covariance matrix estimation) has been studied using a variety of techniques and assumptions; our overview is necessarily brief, with a focus on those most relevant to the present paper.

Information-theoretic considerations lead to the following algorithm-independent lower bound on the number of samples m [Wang et al., 2010]:

$$m = \Omega\bigg(\max\bigg\{ \frac{\log p}{\kappa^2}, \frac{d \log p}{\log(1+\kappa d)} \bigg\} \bigg), \qquad (1)$$

where p is the number of nodes, d the maximal degree of the graph, and κ the minimum normalized edge strength (see (3) below). Ideally, algorithmic upper

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

bounds would also depend only on (p, d, κ) with no further assumptions, though as we describe below, this is rarely the case thus far (including in our own results).

Early algorithms such as SGS and PC [Kalisch and Bühlmann, 2007, Spirtes et al., 2000, van de Geer et al., 2013] adopted conditional independence testing methods, and made assumptions such as strong faithfulness. A popular line of works studied the Graphical Lasso and related ℓ_1-based methods [d'Aspremont et al., 2008, Hsieh et al., 2013, Meinshausen et al., 2006, Ravikumar et al., 2011, Yuan and Lin, 2007, Zhou et al., 2011], typically attaining low sample complexities (e.g., $(d^2 + \kappa^{-2}) \log p$ [Ravikumar et al., 2011]), but only under somewhat strong coherence-based assumptions. More recently, sample complexity bounds were given under walk-summability assumptions [Anandkumar et al., 2012, Kelner et al., 2019] and eigenvalue (e.g., condition number) assumptions [Cai et al., 2011, 2016, Wang et al., 2016]. Another line of works has adopted a Bayesian approach to learning Gaussian graphical models [Leppä-Aho et al., 2017, Mohammadi and Wit, 2015], but to our knowledge, these have not come with sample complexity bounds.

Misra *et al.* [Misra et al., 2017] provide an algorithm that succeeds with $m = O\big(\frac{d \log p}{\kappa^2}\big)$ without further assumptions, thus coming fairly close to the lower bound (1). However, this is yet to be done efficiently, as the time complexity of $p^{O(d)}$ in [Misra et al., 2017] (see also [Kelner et al., 2019, Thm. 11]) is prohibitively large unless d is small. Very recently, efficient algorithms were proposed for handling general graphs under the additional assumption of attractivity (i.e., only having non-positive off-diagonal terms in the inverse covariance matrix) [Kelner et al., 2019].

Learning (generalized) Ising models. Since our focus is on Gaussian models, we only briefly describe the related literature on Ising models, other than a particular algorithm that we directly build upon.

Early works on Ising models relied on assumptions that prohibit long-range correlations [Anandkumar et al., 2012, Bento and Montanari, 2009, Bresler et al., 2008, Jalali et al., 2011, Ravikumar et al., 2010], and this hurdle was overcome in a series of works pioneered by Bresler *et al.* [Bresler, 2015, Bresler et al., 2014, Hamilton et al., 2017]. Recent developments have brought the sample complexity upper bounds increasingly close to the information-theoretic lower bounds [Santhanam and Wainwright, 2012], using techniques such as interaction screening [Vuffray et al., 2016], multiplicative weights [Klivans and Meka, 2017], and sparse logistic regression [Wu et al., 2019].

The present paper is particularly motivated by [Klivans and Meka, 2017], in which an algorithm was developed for learning (generalized) Ising models based on the method of multiplicative weights. More specifically, the algorithm constructs the underlying graph with a nearly optimal sample complexity and a low time complexity by using a weighted majority voting scheme to learn neighborhoods variable-by-variable, and updating the weights using Freund and Schapire's classic Hedge algorithm [Freund and Schapire, 1997]. The proof of correctness uses the regret bound for the Hedge algorithm, as well as showing that approximating the distribution well according to a certain prediction metric ensures accurately learning the associated weight vector (and hence the neighborhood).

1.2 Contributions

In this paper, we adapt the approach of [Klivans and Meka, 2017] to Gaussian graphical models, and show that the resulting algorithm efficiently learns the graph structure with rigorous bounds on the number of samples required, and a low runtime of $O(mp^2)$ when there are m samples and p nodes. As we highlight throughout the paper, each step of our analysis requires nontrivial modifications compared to [Klivans and Meka, 2017] to account for the continuous and unbounded nature of the Gaussian distribution.

While we do not claim that our sample complexity bound improves on the state-of-the-art, it exhibits similar assumptions and dependencies to existing works that adopt condition-number-type assumptions (e.g., ACLIME [Cai et al., 2016]; see the discussion following Theorem 9). In addition, as highlighted in [Klivans and Meka, 2017], the multiplicative weights approach enjoys the property of directly applying in the online setting (i.e., samples arrive one-by-one and must be processed, but not stored, before the next sample).

In Appendix A, we discuss the runtimes of a variety of the algorithms mentioned in Section 1.1, highlighting the fact that our $O(mp^2)$ runtime is very attractive.

2 Problem Statement

Given a Gaussian random vector $X \in \mathbb{R}^p$ taking values in $\mathbb{R}^p$ with zero mean,[1] covariance matrix $\Sigma \in \mathbb{R}^{p \times p}$, and inverse covariance matrix $\Theta \in \mathbb{R}^{p \times p}; \Theta = [\theta_{ij}]_{i,j \in [p]}$ (where $[p] = \{1, \ldots, p\}$), we are interested in recovering the graph $G = (V, E)$ (with $V = [p]$)

[1] Our techniques can also handle the non-zero mean setting, but we find the zero-mean case to more concisely convey all of the relevant concepts.

whose adjacency matrix coincides with the support of Θ. That is, we are interested in learning which entries of Θ are non-zero.

The graph learning is done using m independent samples $(X^1, \dots, X^m)$ from $\mathcal{N}(0, \Sigma)$. Given these samples, the estimation algorithm forms an estimate $\hat{G}$ of the graph, or equivalently, an estimate $\hat{E}$ of the edge set, and the error probability is given by

$$\mathbb{P}(\text{error}) = \mathbb{P}(\hat{G} \neq G). \tag{2}$$

We are interested in characterizing the worst-case error probability over all graphs within some class (described below). Since our approach is based on neighborhood estimation, and each node has $p-1$ candidate neighbors, it will be convenient to let $n = p - 1$.

Definitions and assumptions. Similarly to existing works such as [Misra et al., 2017, Wang et al., 2010], our results depend on the minimum normalized edge strength, defined as

$$\kappa = \min_{(i,j)\in E} \left| \frac{\theta_{ij}}{\sqrt{\theta_{ii}\theta_{jj}}} \right|. \tag{3}$$

Intuitively, the sample complexity must depend on κ because weaker edges require more samples to detect.

We introduce some assumptions that are similar to those appearing in some existing works. First, for each $i = 1, \dots, p$, we introduce the quantity

$$\lambda_i = \sum_{j\neq i} \left| \frac{\theta_{ij}}{\theta_{ii}} \right|, \tag{4}$$

and we assume that $\max_{i\in[p]} \lambda_i$ is upper bounded by some known value λ. As we discuss following our main result (Theorem 9), this is closely related to an assumption made in [Cai et al., 2011, 2016], and as discussed in [Kelner et al., 2019], the latter can be viewed as a type of condition number assumption, though eigenvalues do not explicitly appear.

In addition, we define an upper bound $\theta_{\max}$ on the absolute values of the entries in Θ, and an upper bound $\nu_{\max}$ on the variance of any marginal variable:

$$\theta_{\max} = \max_{i,j} |\theta_{i,j}| = \max_i \theta_{i,i} \tag{5}$$

$$\nu_{\max} = \max_i \text{Var}[X_i]. \tag{6}$$

A $(\theta_{\max}\nu_{\max})^2$ term appears in our final sample complexity bound (Theorem 9). This can again be viewed as a type of condition number assumption, since matrices with a high condition number may have large $\theta_{\max}\nu_{\max}$; e.g., see the example of [Misra et al., 2017].

We will sometimes refer to the maximal degree d of the graph in our discussions, but our analysis and final result will not depend on d. Rather, one can think of λ as implicitly capturing the dependence on d.

For the purpose of simplifying our final expression for the sample complexity, we make some mild assumptions on the scaling laws of the above parameters:

- We assume that $\lambda = \Omega(1)$. This is mild since one can verify that $\lambda = \Omega(\kappa d)$, and the typical regimes considered in existing works are $\kappa d = \Theta(1)$ and $\kappa d \to \infty$ (e.g., see [Wang et al., 2010]).
- We assume that λ, κ, $\nu_{\max}$, and $\theta_{\max}$ are in between $\frac{1}{\text{poly}(p)}$ and $\text{poly}(p)$. This is mild since these may be high-degree polynomials, e.g., p^{10} or p^{100}.

3 Overview of the Algorithm

To recover the graph structure, we are first interested in estimating the inverse covariance matrix $\Theta \in \mathbb{R}^{p\times p}$ of the multivariate Gaussian distribution.

For a zero-mean Gaussian random vector X, we have the following well-known result for any index i (see Lemma 2 below):

$$\mathbb{E}[X_i | X_{\bar{i}}] = \sum_{j\neq i} \frac{-\theta_{ij}}{\theta_{ii}} X_j = w^i \cdot X_{\bar{i}}, \tag{7}$$

where $w^i = \left(\frac{-\theta_{ij}}{\theta_{ii}}\right)_{j\neq i}$, $X_{\bar{i}} = (X_j)_{j\neq i}$, and $a \cdot b$ denotes the dot product. In the related setting of learning Ising models and generalized linear models, the authors of [Klivans and Meka, 2017] used an analogous relation to turn the 'unsupervised' problem of learning the inverse covariance matrix to a 'supervised' problem of learning weight vectors given samples (x^t, y^t), where the x^t are n-dimensional tuples consisting of the values of $X_{\bar{i}}$, and y^t are the values of X_i. In particular, under the standard Ising model, the relationship analogous to (7) follows a logistic (rather than linear) relation.

In [Klivans and Meka, 2017], the Hedge algorithm of [Freund and Schapire, 1997] is adapted to the problem of estimating the coefficients of w^i. This is achieved for sparse generalized linear models (with bounded Lipschitz transfer functions) by first finding a vector v that approximately minimizes an expected risk quantity with high probability, which we define analogously for our setting. Their algorithm, referred to as Sparsitron, is shown in Algorithm 1. Note that both $(\tilde{x}^t, \tilde{y}^t)$ will represent suitably-normalized samples (x^t, y^t) to be described below, and (a^t, b^t) will represent further samples with the same distribution as $(\tilde{x}^t, \tilde{y}^t)$.

Data: $T+M$ normalized samples $\{(\tilde{x}^t, \tilde{y}^t)\}_{t=1}^T, \{(a^j, b^j)\}_{j=1}^M$; ℓ_1-norm parameter λ; update parameter β (default value $\frac{1}{1+\sqrt{\frac{\ln n}{T}}}$)
Result: Estimate of weight vector in $\mathbb{R}^n$
Initialize $v^0 = 1/n$
for *t=1,...,* T **do**
- Let $p^t = \frac{v^{t-1}}{\|v^{t-1}\|_1}$
- Define $l^t \in \mathbb{R}^n$ by $l^t = (1/2)(\mathbf{1} + (\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t)\tilde{x}^t)$
- Update the weight vectors: For each $i \in [n]$, set $v_i^t = v_i^{t-1} \cdot \beta^{l_i^t}$

end
for *t=1,...,M* **do**
- Compute the empirical risk for each t:

$$\hat{\varepsilon}(\lambda p^t) = \frac{\sum_{j=1}^M (\lambda p^t \cdot a^j - b^j)^2}{M} \quad (9)$$

end
return λp^{t^*} *for* $t^* = \arg\min_{t\in[T]} \hat{\varepsilon}(\lambda p^t)$

Algorithm 1: Sparsitron algorithm for estimating a weight vector $w \in \mathbb{R}^n$. It is assumed here that the true weight vector has only positive weights and ℓ_1-norm exactly λ (see Footnote 2).

Data: $T+M$ samples, tuple $(\nu_{\max}, \theta_{\max}, \lambda, \kappa)$, target error probability δ
Result: Estimate of the graph
for $i = 1, \cdots, p$ **do**
- Normalize the T samples as $(\tilde{x}^t, \tilde{y}^t) := \frac{1}{B\sqrt{\nu_{\max}(\lambda+1)}}(x^t, y^t)$ with $B = \sqrt{2\log\frac{2pT}{\delta}}$, and similarly normalize the final M samples to obtain $\{(a^j, b^j)\}_{j=1}^M$
- Run Sparsitron on the normalized samples to obtain an estimate v^i of the weight vector $w^i = \left(\frac{-\theta_{ij}}{\theta_{ii}}\right)_{j\neq i}$ of node i

end
For every pair i and j, identify an edge between them if $\max\{|v_j^i|, |v_i^j|\} \geq 2\kappa/3$

Algorithm 2: Overview of the algorithm for Gaussian graphical model selection.

Definition 1. *The* expected risk *of a candidate* $v \in \mathbb{R}^n$ *for the neighborhood weight vector* w^i *of a marginal variable* X_i *is*

$$\varepsilon(v) := \mathbb{E}_X\left[\left(v \cdot X_{\bar{i}} - w^i \cdot X_{\bar{i}}\right)^2\right]. \quad (8)$$

The Sparsitron algorithm uses what can be seen as a simple majority weighted voting scheme. For every possible member X_j of the neighborhood of node i, the algorithm maintains a weight v_j (which we think of as seeking to approximate the j-th entry of w^i), and updates the weight vector via multiplicative updates as in the Hedge algorithm. After T such consecutive estimates, the algorithm uses an additional M samples to estimate the expected risk for each of the T candidates empirically, and then returns the candidate with the smallest empirical risk.

As in [Klivans and Meka, 2017], we assume without loss of generality that $w_i \geq 0$ for all i; for if not, we can map our samples (x, y) to $([x, -x], y)$ and adjust the weight vector accordingly. We can also assume that $\|w\|_1$ equals its upper bound λ, since otherwise we can introduce a new coefficient and map our samples to $([x, -x, 0], y)$.[2] If the true norm were $\lambda' < \lambda$, then the modified weight vector would have a value of $\lambda - \lambda'$ corresponding to the 0 coefficient.

Once the neighborhood weight vectors have been estimated, we recover the graph structure using thresholding, as outlined in Algorithm 2. Here, T and M must satisfy certain upper bounds that we derive later (see Theorem 9). The overall sample complexity is $m = T + M$, and as we discuss following Theorem 9, the runtime is $O(mp^2)$. This runtime is compared to the runtimes of various existing algorithms for learning Gaussian graphical models in Appendix A.

4 Analysis and Sample Complexity

Our analysis proceeds in several steps, given in the following subsections.

4.1 Preliminary Results

4.1.1 Properties of Multivariate Gaussians

We first recall some results regarding multivariate Gaussian random variables that we will need throughout the analysis.

[2] This step is omitted in Algorithm 1, so that we can lighten notation and work with vectors in $\mathbb{R}^n$ rather than $\mathbb{R}^{2n+1}$. Formally, it can be inserted as an initial step, and then the resulting length $2n+1$ weight vector can be mapped back to a length-n weight vector by taking the first n entries and subtracting the second n entries, while ignoring the final entry. The initial part of our analysis considering Sparsitron can be viewed as corresponding to the case where the weights are already positive and the ℓ_1-norm bound λ already holds with equality, but it goes through essentially unchanged in the general case.

Lemma 2. *Given a zero-mean multivariate Gaussian $X = (X_1, \dots, X_p)$ with inverse covariance matrix $\Theta = [\theta_{ij}]$, and given T independent samples $(X^1, \dots, X^T)$ with the same distribution as X, we have the following:*

1. *For any $i \in [p]$, we have $X_i = \eta_i + \sum_{j \neq i} \big(- \frac{\theta_{ij}}{\theta_{ii}} \big) X_j$, where η_i is a Gaussian random variable with variance $\frac{1}{\theta_{ii}}$, independent of all X_j for $j \neq i$.*
2. *$\mathbb{E}[X_i | X_{\bar{i}}] = \sum_{j \neq i} \big(\frac{-\theta_{ij}}{\theta_{ii}} \big) X_j = w^i \cdot X_{\bar{i}}$, where $w^i = \big(\frac{-\theta_{ij}}{\theta_{ii}} \big)_{j \neq i} \in \mathbb{R}^n$ (with $n = p - 1$).*
3. *Let λ and $\nu_{\max}$ be defined as in (4) and (6), set $B := \sqrt{2 \log \frac{2pT}{\delta}}$, and define $(\tilde{x}^t, \tilde{y}^t) := \frac{1}{B\sqrt{\nu_{\max}(\lambda+1)}} (x^t, y^t)$, where $(x^t, y^t) = (X^t_{\bar{i}}, X^t_i)$ for an arbitrary fixed coordinate i. Then, with probability at least $1 - \delta$, $\tilde{y}^t$ and all entries of $\tilde{x}^t$ $(t = 1, \dots, T)$ have absolute value at most $\frac{1}{\sqrt{\lambda+1}}$.*

Proof. These properties are all standard and/or use standard arguments; see Appendix B for details. □

4.1.2 Loss Guarantee for Sparsitron

Recall that $n = p - 1$. In the proof of [Klivans and Meka, 2017, Theorem 3.1], it is observed that the Hedge regret guarantee implies the following.

Lemma 3. ([Klivans and Meka, 2017]) *For any sequence of loss vectors $l^t \in [0,1]^n$ for $t = 1, \dots T$, the Sparsitron algorithm guarantees that*

$$\sum_{t=1}^{T} p^t \cdot l^t \leq \min_{i \in [n]} \sum_{t=1}^{T} l_i^t + O(\sqrt{T \log n} + \log n). \quad (10)$$

To run the Sparsitron algorithm, we need to define an appropriate sequence of loss vectors in $[0,1]^n$. Let

$$l^t = (1/2)(\mathbf{1} + (\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t)\tilde{x}^t), \quad (11)$$

where $\mathbf{1}$ is the vector of ones, and λp^t is Sparsitron's estimate at the beginning of the t-th iteration, formed using samples $1, \dots, t-1$. To account for the fact that the Hedge algorithm requires bounded losses for its regret guarantee, we use the high probability scaling in the third part of Lemma 2: Since $p^t \in [0,1]^n$ and $\sum_{t=1}^n p_t = 1$, we have that $|\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t| < \sqrt{\lambda + 1}$, and that consequently $(\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t)\tilde{x}^t \in [-1, 1]^n$. It then follows that l^t, as defined in (11), lies in $[0,1]^n$. Hence, Lemma 3 applies with probability at least $1 - \delta$ when we use $(\tilde{x}^t, \tilde{y}^t) := \frac{1}{B\sqrt{\nu_{\max}(\lambda+1)}} (x^t, y^t)$.

4.1.3 Concentration Bound for Martingales

Unlike the analysis of Ising (and related) models in [Klivans and Meka, 2017], here we do not have the liberty of assuming bounded losses. In the previous subsection, we circumvented this issue by noting that the losses are bounded with high probability, and such an approach is sufficient for that step due to the fact that the Hedge regret guarantee applies for arbitrary (possibly adversarially chosen) bounded losses. However, while such a "truncation" approach was sufficient above, it will be insufficient (or at least challenging to make use of) in later parts of the analysis that rely on the Gaussianity of the samples.

In this subsection, we present a concentration bound that helps to overcome this difficulty, and serves as a replacement for the Azuma-Hoeffding martingale concentration bound used in [Klivans and Meka, 2017].

Specifically, we use [van de Geer, 1995, Lemma 2.2], which states that given a martingale M_t, if we can establish Bernstein-like inequalities on the 'sums of drifts' of certain higher order processes, then we can establish a concentration bound on the main process. Here we state a simplified version for discrete-time martingales that suffices for our purposes (in [van de Geer, 1995], continuous-time martingales are also permitted). This reduction from [van de Geer, 1995, Lemma 2.2] is outlined in Appendix C.

Lemma 4. ([van de Geer, 1995]) *Let M_t be a discrete-time martingale with respect to a filtration $\mathcal{F}_t$ such that $\mathbb{E}[M_t^2] < \infty$ for all t, and define $\Delta M_t = M_t - M_{t-1}$ and $V_{m,t} = \sum_{j=1}^t \mathbb{E}[|\Delta M_j|^m \,|\, \mathcal{F}_{j-1}]$. Suppose that for all t and some $0 < K < \infty$, we have*

$$V_{m,t} \leq \frac{m!}{2} K^{m-2} R_t, \qquad m = 2, 3, \dots \quad (12)$$

for some process R_t that is measurable with respect to $\mathcal{F}_{t-1}$. Then, for any $a, b > 0$, we have

$$\mathbb{P}(M_t \geq a \text{ and } R_t \leq b^2 \text{ for some } t) \leq \exp\left(-\frac{a^2}{2aK + b^2} \right). \quad (13)$$

4.2 Bounding the Expected Risk

For compactness, we subsequently write w as a shorthand for the weight vector $w^i \in \mathbb{R}^n$ of the node i whose neighborhood is being estimated. We recall the choice of l^t in (11), and make use of the following definitions from [Klivans and Meka, 2017]:

$$Q^t := (p^t - w/\lambda) \cdot l^t \quad (14)$$

$$Z^t := Q^t - \mathbb{E}_{t-1}[Q^t], \quad (15)$$

where here and subsequently, we use the notation $\mathbb{E}_t[\cdot] := \mathbb{E}[\cdot|(x^1, y^1), \ldots, (x^t, y^t)]$ to denote conditioning on the samples up to time t. The analysis proceeds by showing that $\sum_{j=1}^T Z^j$ is concentrated around zero, upper bounding the expected risk in terms of $\mathbb{E}_{t-1}[Q^t]$, and applying Sparsitron's guarantee from Lemma 3.

We first use Lemma 4 to obtain the following result.

Lemma 5. $|\sum_{j=1}^T Z^j| = O\left(\sqrt{T \log \frac{1}{\delta}}\right)$ *with probability at least* $1 - \delta$.

Proof. The proof essentially just requires substitutions in Lemma 4. The martingale process is $M_t = \sum_{j \le t} Z^j$, and we obtain $\Delta M_t = Z^t$, along with

$$V_{m,t} = \sum_{j=1}^{t} \mathbb{E}_{j-1}[|Z^j|^m]. \tag{16}$$

The rest of the proof entails unpacking the definitions and using standard properties of Gaussian random variables to show that the Bernstein-like requirements are satisfied for the concentration bound in Lemma 4. The details are provided in Appendix D. □

Lemma 6. *If Sparsitron is run with* $T \ge \log n$, *then*

$$\min_{t \in [T]} \varepsilon(\lambda p^t) = O\left(\frac{\lambda(\lambda+1)\nu_{\max} \log \frac{nT}{\delta} \left(\sqrt{T \log \frac{n}{\delta}}\right)}{T}\right) \tag{17}$$

with probability at least $1 - \delta$.

Proof. From the definition of Q^t in (14), we have that

$$\begin{aligned}
&\mathbb{E}_{t-1}[Q^t] \\
&= \mathbb{E}_{t-1}[(p^t - (1/\lambda)w) \cdot l^t] && (18)\\
&= \mathbb{E}_{t-1}[(p^t - (1/\lambda)w) \cdot (1/2)(\mathbf{1} + (\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t)\tilde{x}^t)] && (19)\\
&= \frac{1}{2}\mathbb{E}_{t-1}\Bigg[\left(\sum_i p_i^t - \sum_i \frac{w_i}{\lambda}\right) \\
&\qquad + (p^t - (1/\lambda)w) \cdot \tilde{x}^t(\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t)\Bigg] && (20)\\
&= \frac{1}{2}\mathbb{E}_{t-1}\Bigg[\left(1 - \frac{\lambda}{\lambda}\right) \\
&\qquad + (p^t \cdot \tilde{x}^t - (1/\lambda)w \cdot \tilde{x}^t)(\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t)\Bigg] && (21)\\
&= \frac{1}{2}\mathbb{E}_{t-1}\left[(p^t \cdot \tilde{x}^t - (1/\lambda)w \cdot \tilde{x}^t)(\lambda p^t \cdot \tilde{x}^t - \tilde{y}^t)\right] && (22)\\
&= \frac{1}{2\lambda}\mathbb{E}_{t-1}[(\lambda p^t \cdot \tilde{x}^t - w \cdot \tilde{x}^t)^2] && (23)\\
&\ge \frac{1}{2\lambda(\lambda+1)\nu_{\max}B^2}\varepsilon(\lambda p^t), && (24)
\end{aligned}$$

where (19) uses the definition of l^t in (11), (21) uses $\|w\|_1 = \lambda$ (see Footnote 2) and $\sum_i p_i^t = 1$, (23) follows by noting that p^t is a function of $\{\tilde{x}^i, \tilde{y}^i\}_{i=1}^{t-1}$) and computing the expectation over $\tilde{y}^t$ first (using the second part of Lemma 2), and (24) uses the definition of expected risk in (8), along with $\tilde{x}^t = \frac{1}{B\sqrt{\nu_{\max}(\lambda+1)}}x^t$.

Summing both sides above over $t = 1, \ldots, T$, we have the following with probability $1 - O(\delta)$:[3]

$$\begin{aligned}
&\frac{1}{2\lambda(\lambda+1)\nu_{\max}B^2}\sum_{t=1}^T \varepsilon(\lambda p^t) \\
&\le \sum_{t=1}^T \mathbb{E}_{t-1}[Q^t] && (25)\\
&= \sum_{t=1}^T (Q^t - Z^t) && (26)\\
&\le \sum_{t=1}^T Q^t + O\left(\sqrt{T \log \frac{1}{\delta}}\right) && (27)\\
&\le \sum_{t=1}^T (p^t - w/\lambda) \cdot l^t + O\left(\sqrt{T \log \frac{1}{\delta}}\right) && (28)\\
&\le \min_{i \in [n]} \sum_{t=1}^T l_i^t - \sum_{t=1}^T (w/\lambda) \cdot l^t + O(\sqrt{T \log n} + \log n) \\
&\qquad + O\left(\sqrt{T \log \frac{1}{\delta}}\right), && (29)
\end{aligned}$$

where (26) follows from the the definition of Z^t in (15), (27) uses Lemma 5, (28) follows from the definition of Q^t in (14), and (29) follows from Lemma 3.

Since $\|w\|_1 = \lambda$ (see Footnote 2), $\min_{i \in [n]} \sum_{t=1}^T l_i^t - \sum_{t=1}^T (w/\lambda) \cdot l^t \le 0$. It follows from (29) that

$$\begin{aligned}
&\frac{1}{2\lambda(\lambda+1)\nu_{\max}B^2}\sum_{j=1}^T \varepsilon(\lambda p^t) \\
&\qquad = O\left(\sqrt{T \log n} + \log n + \sqrt{T \log \frac{1}{\delta}}\right), && (30)
\end{aligned}$$

and substituting $B = \sqrt{2 \log \frac{2(n+1)T}{\delta}}$ gives

$$\begin{aligned}
\min_{t \in [T]} \varepsilon(\lambda p^t) = O\Bigg(&\frac{\lambda(\lambda+1)\nu_{\max} \log \frac{nT}{\delta}}{T} \\
&\times \left(\sqrt{T \log n} + \log n + \sqrt{T \log \frac{1}{\delta}}\right)\Bigg), && (31)
\end{aligned}$$

[3] In the analysis, we apply multiple results that each hold with probability at least $1-\delta$. More precisely, δ should be replaced by δ/L when applying a union bound over L events, but since L is finite, this only amounts to a change in the constant of the $O(\cdot)$ notation in (17).

where we also lower bounded $\sum_{j=1}^{T} \varepsilon(\lambda p^t)$ by T times the minimum value. When $T \geq \log n$, the above bound simplifies to (17), as desired. □

Having ensured that that the minimal expected risk is small, we need the algorithm to identify a candidate whose expected risk is also sufficiently close to that minimum. Sparsitron does this by using an additional M samples to estimate the expected risk empirically.

Lemma 7. *For $\gamma > 0$, $\rho \in (0, 1]$, and fixed $v \in \mathbb{R}^n$ satisfying $\|v\|_1 \leq \lambda$, there is some $M = O\big((\lambda+1)\frac{\log(1/\rho)}{\gamma}\big)$ such that*

$$\mathbb{P}\left(\left|\frac{1}{M}\sum_{j=1}^{M}\Big((v \cdot a^j - b^j)^2 - \Xi\Big) - \varepsilon(v)\right| \geq \gamma\right) \leq \rho, \tag{32}$$

where $\{(a^j, b^j)\}_{j=1}^{M}$ are the normalized samples defined in Algorithm 2, and $\Xi = \mathbb{E}\big[\mathrm{Var}[b^j \mid a^j]\big]$.[4]

Proof. The high-level steps of the proof are to first establish the equality

$$\mathbb{E}[(v \cdot a^j - b^j)^2] = \varepsilon(v) + \Xi, \tag{33}$$

and then use Bernstein's inequality to bound the deviation of $\sum_{j=1}^{M}\big((v \cdot a^j - b^j)^2 - \Xi\big)$ from its mean value $\varepsilon(v)$. The details are given in Appendix E. □

4.3 Graph Recovery and Sample Complexity

We complete the analysis of our algorithm in a sequence of three steps, given as follows.

An ℓ_∞ bound. We show that if our estimate v approximates the true weight vector $w \in \mathbb{R}^n$ well in terms of the expected risk, then it also approximates it in the ℓ_∞ norm. In [Klivans and Meka, 2017], this was done using a property termed the 'δ-unbiased condition', whose definition relies on the underlying random variables being binary. Hence, we require a different approach, given as follows.[5]

Lemma 8. *Under the preceding setup, if we have $\varepsilon(v) \leq \epsilon$, then we also have $\|v - w\|_\infty \leq \sqrt{\epsilon\theta_{\max}}$, where $\theta_{\max}$ is a uniform upper bound on the diagonal entries of Θ.*

Proof. The proof uses a direct calculation to establish that $\mathrm{Var}((v-w)\cdot X_{\bar{i}}) \geq |v_{i^*} - w_{i^*}|^2 \mathrm{Var}(\eta_{i^*})$ for a fixed index i^*; the details are given in Appendix F. □

[4] This quantity is the same for all values of j.

[5] See also [Kelner et al., 2019, Thm. 17] for similar considerations under a different set of assumptions.

Suppose that we would like to recover the true weight vector with a maximum deviation of ϵ' in any coordinate with probability at least $1 - \delta$. By Lemma 8, we require ϵ to be no more than $(\epsilon')^2/\theta_{\max}$. We know from Lemma 6 that

$$\min_{t\in[T]} \varepsilon(\lambda p^t) = O\left(\frac{\lambda(\lambda+1)\nu_{\max}\log\frac{nT}{\delta}\left(\sqrt{T\log\frac{n}{\delta}}\right)}{T}\right), \tag{34}$$

from which we have that with $T = O\left(\frac{\lambda^2(\lambda+1)^2\nu_{\max}^2}{\epsilon^2}\log^3\frac{n}{\delta}\right)$,[6] the minimum expected risk is less than $\epsilon/2$ with probability at least $1 - \delta/2$.

From Lemma 7 with $\rho = \delta/(2T)$ and $\gamma = \epsilon/2$, we observe that we can choose M satisfying

$$M \leq O\left((\lambda+1)\frac{\log(T/\delta)}{\epsilon}\right) \tag{35}$$

$$\leq O\left((\lambda+1)\log\frac{\lambda(\lambda+1)\nu_{\max}\log^{3/2}\frac{n}{\delta}}{\epsilon\delta}\right) \tag{36}$$

and estimate $\varepsilon(\lambda p^t) + \Xi$ (note that the second term doesn't affect the $\arg\min$) of the T candidates λp^t within $\epsilon/4$ with probability at least $1 - \frac{\delta}{2T}$. By the union bound (which blows the $\frac{\delta}{2T}$ up to $\frac{\delta}{2}$), the same follows for all T candidates simultaneously. We then have that the candidate with the lowest estimate has expected risk within $\epsilon/2$ of the candidate with the lowest expected risk, and that the latter candidate's expected risk is less than $\epsilon/2$, so in sum the vector returned by the candidate has an expected risk less than ϵ with probability at least $1-\delta$. Moreover, the sample complexity is

$$\begin{aligned} &T + M \\ &= O\left(\frac{\lambda^2(\lambda+1)^2\nu_{\max}^2}{\epsilon^2}\log^3\frac{n}{\delta}\right) \\ &\quad + O\left((\lambda+1)\log\frac{\lambda(\lambda+1)\nu_{\max}\log^{3/2}\frac{n}{\delta}}{\epsilon\delta}\right) \qquad (37) \\ &= O\left(\frac{\lambda^4\nu_{\max}^2}{\epsilon^2}\log^3\frac{n}{\delta}\right), \qquad (38) \end{aligned}$$

where the simplification comes by recalling from Section 2 that $\lambda = \Omega(1)$ and all parameters are polynomially bounded with respect to n. While the sample complexity (38) corresponds to probability at least $1-\delta$ for the algorithm of only a single $i \in [p]$, we can replace δ by δ/p and apply a union bound to conclude the same for all $i \in [p]$; since $p = n + 1$, this only amounts to a chance in the constant of the $O(\cdot)$ notation.

[6] The removal of T in the logarithm $\log\frac{nT}{\delta}$ can be justified by the assumption that all parameters are polynomially bounded with respect to p (see Section 2).

Recovering the graph. Recall from Lemma 8 that an expected risk of at most ϵ translates to a coordinate-wise deviation of at most $\epsilon' = \sqrt{\epsilon\theta_{\max}}$. We set $\epsilon = \frac{\kappa^2}{9\theta_{\max}}$, so that $\epsilon' = \frac{\kappa}{3}$.

We observe that if X_i and X_j are neighbors, then (3) yields the following lower bound:

$$\frac{\theta_{ij}^2}{\theta_{ii}\theta_{jj}} \geq \kappa^2 \tag{39}$$

This ensures that at least one of the two values $|\theta_{ij}/\theta_{ii}|$ and $|\theta_{ij}/\theta_{jj}|$ must be greater than or equal to κ. On the other hand, if they are not neighbors, then the true value of both of these terms must be 0. Since we have estimated all weights to within $\kappa/3$, it follows that any estimate of at least $2\kappa/3$ must arise from a true neighborhood relation (with high probability). Conversely, if there is a neighborhood relation, then at least one of the two factors θ_{ij}/θ_{ii} and θ_{ij}/θ_{jj} must have been found to be at least $2\kappa/3$.

The method for recovering the graph structure is then as follows: For each possible edge, the weight estimates v_j^i and v_i^j are calculated; if either of them is found to be greater than $2\kappa/3$, then the edge is declared to lie in the graph, and otherwise it is not.

Substituting $\epsilon = \frac{\kappa^2}{9\theta_{\max}}$ into (38), and recalling our notation $n = p - 1$, we deduce the final sample complexity, stated as follows.

Theorem 9. *For learning graphs on p nodes with minimum normalized edge strength κ, under the additional assumptions stated in Section 2 with parameters $(\lambda, \nu_{\max}, \theta_{\max})$, the algorithm described above attains $\mathbb{P}(\text{error}) \leq \delta$ with a sample complexity of at most*

$$m = O\left(\frac{\lambda^4\nu_{\max}^2\theta_{\max}^2}{\kappa^4}\log^3\frac{p}{\delta}\right). \tag{40}$$

We can compare this guarantee with those of existing algorithms: As discussed in [Kelner et al., 2019, Remark 8], the ℓ_1-based ACLIME algorithm [Cai et al., 2016] can be used for graph recovery with $m = O\big(\frac{\tilde{\lambda}^2\log\frac{p}{\delta}}{\kappa^4}\big)$ samples, where $\tilde{\lambda}$ is an upper bound on the ℓ_1 norm of any row of Θ. An algorithm termed HybridMB in [Kelner et al., 2019] achieves the same guarantee, and a greedy pruning method in the same paper attains a weaker $m = O\big(\frac{\tilde{\lambda}^4\log\frac{p}{\delta}}{\kappa^6}\big)$ bound.

The quantities λ and $\tilde{\lambda}$ are closely related; for instance, in the case that $\theta_{ii} = 1$ for all i, we have $\tilde{\lambda} = 1 + \lambda$. More generally, if $\nu_{\max}$ and $\theta_{\max}$ behave as $\Theta(1)$, then our bound can be written as $O\big(\frac{\tilde{\lambda}^4}{\kappa^4}\log^3\frac{p}{\delta}\big)$, which is qualitatively similar to the bounds of [Cai et al., 2016, Kelner et al., 2019] but with an extra $\big(\lambda\log\frac{p}{\delta}\big)^2$ term.

We again highlight that our main goal is not to attain a state-of-the-art sample complexity, but rather to introduce a new algorithmic approach to Gaussian graphical model selection. The advantages of this approach, as highlighted in [Klivans and Meka, 2017], are low runtime and direct applicability to the online setting. In addition, as we discuss in the following section, we expect that there are parts of our analysis that could be refined to bring the sample complexity down further.

Runtime. The algorithm enjoys a low runtime similar to the case of Ising models [Klivans and Meka, 2017]: Sparsitron performs $m = T + M$ iterations that each require time $O(n) = O(p)$, for an overall runtime of $O(mp)$. Since this is done separately for each $i = 1, \ldots, p$, the overall runtime is $O(mp^2)$.

5 Conclusion

We have introduced a novel adaptation of the multiplicative weights approach to graphical model selection [Klivans and Meka, 2017] to the Gaussian setting, and established a resulting sample complexity bound under suitable assumptions on the covariance matrix and its inverse. The algorithm enjoys a low runtime compared to existing methods, and can directly be applied in the online setting.

The most immediate direction for further work is to seek refinements of our algorithm and analysis that can further reduce the sample complexity and/or weaken the assumptions made. For instance, we normalized the samples to ensure a loss function in $[0, 1]$ with high probability, and this is potentially more crude then necessary (and ultimately yields the $\log^3 p$ dependence). One may therefore consider using an alternative to Hedge that is more suited to unbounded rewards. In addition, various steps in our analysis introduced $\theta_{\max}$ and $\nu_{\max}$, and the individual estimation of diagonals of Σ and/or Θ (e.g., as done in [Cai et al., 2016]) may help to avoid this.

Acknowledgment

This work was supported by the Singapore National Research Foundation (NRF) under grant number R-252-000-A74-281.

Bibliography

A. Anandkumar, V. Y. F. Tan, F. Huang, and A. S. Willsky. High-dimensional Gaussian graphical model selection: Walk summability and local separation criterion. *J. Mach. Learn. Res.*, 13:2293–2337, 2012.

J. Bento and A. Montanari. Which graphical models are difficult to learn? In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*. 2009.

S. Boucheron, G. Lugosi, and P. Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

G. Bresler. Efficiently learning Ising models on arbitrary graphs. In *ACM Symp. Theory Comp. (STOC)*, 2015.

G. Bresler, E. Mossel, and A. Sly. Reconstruction of Markov random fields from samples: Some observations and algorithms. In *Appr., Rand. and Comb. Opt. Algorithms and Techniques*, pages 343–356. Springer Berlin Heidelberg, 2008.

G. Bresler, D. Gamarnik, and D. Shah. Structure learning of antiferromagnetic Ising models. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*. 2014.

T. Cai, W. Liu, and X. Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *J. Amer. Stat. Assoc.*, 106(494):594–607, 2011.

T. T. Cai, W. Liu, and H. H. Zhou. Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation. *Ann. Stats.*, 44(2):455–488, 04 2016.

A. d'Aspremont, O. Banerjee, and L. El Ghaoui. First-order methods for sparse covariance selection. *SIAM J. Matrix Analysis and Apps.*, 30(1):56–66, 2008.

R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge Univ. Press, 1998.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comp. Sys. Sci.*, 55(1):119–139, 1997.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Patt. Analysis and Mach. Intel.*, (6):721–741, 1984.

R. J. Glauber. Time-dependent statistics of the Ising model. *J. Math. Phys.*, 4(2):294–307, 1963.

L. Hamilton, F. Koehler, and A. Moitra. Information theoretic properties of Markov random fields, and their algorithmic applications. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*, pages 2463–2472, 2017.

C.-J. Hsieh, M. A. Sustik, I. S. Dhillon, P. K. Ravikumar, and R. Poldrack. BIG & QUIC: Sparse inverse covariance estimation for a million variables. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*, pages 3165–3173, 2013.

A. Jalali, C. C. Johnson, and P. K. Ravikumar. On learning discrete graphical models using greedy methods. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*, 2011.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *J. Mach. Learn. Research*, 8(Mar):613–636, 2007.

J. Kelner, F. Koehler, R. Meka, and A. Moitra. Learning some popular Gaussian graphical models without condition number bounds. *https://arxiv.org/abs/1905.01282*, 2019.

A. Klivans and R. Meka. Learning graphical models using multiplicative weights. In *Symp. Found. Comp. Sci. (FOCS)*, pages 343–354. IEEE, 2017.

J. Leppä-Aho, J. Pensar, T. Roos, and J. Corander. Learning Gaussian graphical models with fractional marginal pseudo-likelihood. *Int. J. Approximate Reasoning*, 83:21–42, 2017.

R. Liptser and A. N. Shiryayev. *Theory of martingales*, volume 49. Kluwer, Dordrecht, 1989.

C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

N. Meinshausen, P. Bühlmann, et al. High-dimensional graphs and variable selection with the lasso. *Ann. Stats.*, 34(3):1436–1462, 2006.

S. Misra, M. Vuffray, and A. Y. Lokhov. Information theoretic optimal learning of Gaussian graphical models. https://arxiv.org/abs/1703.04886, 2017.

A. Mohammadi and E. C. Wit. Bayesian structure learning in sparse Gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.

P. Ravikumar, M. J. Wainwright, J. D. Lafferty, and B. Yu. High-dimensional Ising model selection using ℓ_1-regularized logistic regression. *Ann. Stats.*, 38(3): 1287–1319, 2010.

P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Elec. J. Stats.*, 5:935–980, 2011.

N. Santhanam and M. Wainwright. Information-theoretic limits of selecting binary graphical models in high dimensions. *IEEE Trans. Inf. Theory*, 58(7): 4117–4134, July 2012.

P. Spirtes, C. N. Glymour, R. Scheines, D. Heckerman, C. Meek, G. Cooper, and T. Richardson. *Causation, prediction, and search*. MIT press, 2000.

S. van de Geer. Exponential inequalities for martingales, with application to maximum likelihood estimation for counting processes. *Ann. Stats.*, pages 1779–1801, 1995.

S. van de Geer, P. Bühlmann, et al. ℓ_0-penalized maximum likelihood for sparse directed acyclic graphs. *Ann. Stats.*, 41(2):536–567, 2013.

M. Vuffray, S. Misra, A. Lokhov, and M. Chertkov. Interaction screening: Efficient and sample-optimal learning of Ising models. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*, pages 2595–2603, 2016.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trend. Mach. Learn.*, 2008.

L. Wang, X. Ren, and Q. Gu. Precision matrix estimation in high dimensional Gaussian graphical models with faster rates. In *Int. Conf. Art. Intel. Stats. (AISTATS)*, 2016.

W. Wang, M. Wainwright, and K. Ramchandran. Information-theoretic bounds on model selection for Gaussian Markov random fields. In *IEEE Int. Symp. Inf. Theory (ISIT)*, 2010.

S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge Univ. Press, 1994.

S. Wu, S. Sanghavi, and A. G. Dimakis. Sparse logistic regression learns all discrete pairwise graphical models. In *Conf. Neur. Inf. Proc. Sys. (NeurIPS)*, 2019.

M. Yuan and Y. Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1): 19–35, 2007.

S. Zhou, P. Rütimann, M. Xu, and P. Bühlmann. High-dimensional covariance estimation based on Gaussian graphical models. *J. Mach. Learn. Res.*, 12:2975–3026, Nov. 2011.