Active Learning Algorithms for Graphical Model Selection

Gautam Dasarathy[†], **Aarti Singh**[†], **Maria F. Balcan**[†,*], **and Jong H. Park**[*]

[†]Machine Learning Department
[*]Computer Science Department
Carnegie Mellon University

Abstract

The problem of learning the structure of a high dimensional graphical model from data has received considerable attention in recent years. In many applications such as sensor networks and proteomics it is often expensive to obtain samples from all the variables involved simultaneously. For instance, this might involve the synchronization of a large number of sensors or the tagging of a large number of proteins. To address this important issue, we initiate the study of a novel graphical model selection problem, where the goal is to optimize the total number of scalar samples obtained by allowing the collection of samples from only subsets of the variables. We propose a general paradigm for graphical model selection where feedback is used to guide the sampling to high degree vertices, while obtaining only few samples from the ones with the low degrees. We instantiate this framework with two specific active learning algorithms, one of which makes mild assumptions but is computationally expensive, while the other is computationally more efficient but requires stronger (nevertheless standard) assumptions. Whereas the sample complexity of passive algorithms is typically a function of the maximum degree of the graph, we show that the sample complexity of our algorithms is provably smaller and that it depends on a novel local complexity measure that is akin to the average degree of the graph. We finally demonstrate the efficacy of our framework via simulations.

1 INTRODUCTION

Probabilistic graphical models provide a powerful formalism for expressing the relationships among a large family of random variables. They are finding applications in increasingly complex scenarios from computer vision and natural language processing to computational biology and statistical physics. One important problem associated with graphical models is that of learning the structure of dependencies between the variables described by such a model from data. This is useful as it not only allows a succinct representation of a potentially complex multivariate distribution, but it might in fact reveal fundamental relationships among the underlying variables. Unfortunately, the problem of learning the structure of graphical models is known to be hard in the high dimensional setting (where the number of observations is typically smaller than the total number of variables) since many natural sufficient statistics such as the sample covariance matrix are poorly behaved (see e.g., [1,2]). An exciting line of work has explored many conditions under which this problem becomes tractable (e.g., [3, 4, 5, 6]). Various authors have discovered that by constraining both the structure of the graph and the parameters of the probabilistic model, there are interesting situations where given $\mathcal{O}(\log p)$ samples from the underlying distribution, one can learn the structure and sometimes the parameters of the underlying graphical model.

In this paper, we will look at this problem in a different light. In a wide variety of situations, it might be costly to obtain many samples across all the variables in the underlying system. For instance, in a sensor network, obtaining a sample across all the sensors is equivalent to obtaining a synchronized measurement from all the sensors. Similarly in many other applications in neuroscience [7, 8, 9] and proteomics [10], it might be much easier to obtain (marginalized) samples from small subsets of variables as opposed to full snapshots.

We propose to handle this problem by what we call *active marginalization*. That is, we present a general

Appearing in Proceedings of the 19^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2016, Cadiz, Spain. JMLR: W&CP volume 51. Copyright 2016 by the authors.

paradigm for "activizing" graphical model structure learning algorithms. In the spirit of active learning, the algorithms we propose decide which vertices to marginalize out (and therefore which vertices to sample further from) based on the samples previously obtained. This general framework is laid out in Algorithm 1, which may be considered a meta algorithm that can be used to make any vertex based graph structure learning algorithm active. The algorithm principally uses two subroutines, one which selects candidate neighborhoods given a vertex and the other one which verifies if this candidate is a good one. We then instantiate this in two different ways for structure learning of Gaussian graphical models. Algorithm 2, which we call AdPaCT (for Adaptive Partial Correlation Testing) is an exhaustive search based algorithm that shows improved sample complexity over passive algorithms. However, like other algorithms based on exhaustive search, the computational complexity of this algorithm is often prohibitive. We then propose Algorithm 3, which we call AMPL (for Adaptive Marginalization based Parallel Lasso). Algorithm 3 performs the neighborhood selection far more efficiently using the Lasso (as in the seminal work of [11]). At the cost of more restrictive, but standard assumptions, this gives us a much more computationally efficient algorithm whose total sample complexity is similar to that of Algorithm 2, and therefore significantly better than its passive counterparts.

While the sample complexity, i.e., the total number of samples needed for exact structure recovery with high probability, for typical passive graph learning algorithms scale as a function of the maximum degree $d_{\max}$ of the graph, the sample complexity of the algorithms we propose scale as a function of a more local property of the graph. For instance, in the Gaussian setting, the neighborhood selection algorithm of [11] is guaranteed to reconstruct the graph if the number of (scalar) measurements it obtains scales as $\mathcal{O}(d_{\max} p \log p)$ (i.e., $\mathcal{O}(d_{\max} \log p)$ samples each in p dimensions). On the other hand, AdPaCT (Algorithm 2) and AMPL(Algorithm 3) are guaranteed to work with $\mathcal{O}(\bar{d}_{\max} p \log p)$ samples, where $\bar{d}_{\max}$ is the average maximum degree across all the neighborhoods in the graph. We will define this quantity formally in Section 2 and as we shall see at the end of Section 2, $\bar{d}_{\max}$ can be significantly smaller than $d_{\max}$.

2 PROBLEM STATEMENT

Let $G = ([p], E)$ denote an undirected graph on the vertex set $[p] = \{1, 2, \ldots, p\}$ with edge set $E \subseteq \binom{p}{2}$. To each vertex of this graph, we associate the components of a $0-$mean Gaussian[1] random vector $X \in \mathbb{R}^p$ with covariance matrix $\Sigma \in \mathbb{R}^{p \times p}$. The density of X is then given by

$$f_X(x_1, \ldots x_p) = \frac{1}{\sqrt{(2\pi)^p |K|}} \exp\left\{-\frac{1}{2} x^T K x\right\}, \quad (1)$$

where $K = \Sigma^{-1}$ is the $p-$dimensional inverse covariance (concentration) matrix. We will abbreviate this density as $\mathcal{N}(0, \Sigma)$ in the sequel.

X is said to be Markov with respect to the graph G if for any pair of vertices i and j, $\{i, j\} \notin E$ implies that $X_i \perp\!\!\!\perp X_j \mid X_{[p]\setminus\{i,j\}}$. By the Hammersley-Clifford theorem [12], we know that for all $\{i, j\} \notin E$, $K_{ij} = 0$.

In this paper, we study the problem of recovering the structure of the graph G given samples from the distribution f_X, that is, we would like to construct an estimate $\widehat{E}$ of the edge set of the underlying graph. As mentioned in Section 1, we are interested in the setting where our estimators are allowed to *actively marginalize* the components of X and only observe samples from the desired components of the random vector X. More formally, operating in L stages, the estimator or algorithm produces a sequence $\{(S_k, n_k)\}_{k\in[L]}$, where $S_k \subseteq [p]$ and $n_k \in \mathbb{N}$. For each $k \in [L]$, the algorithm receives n_k samples of the marginalized random vector $X_{S_k} \in \mathbb{R}^{|S_k|}$. Notice that X_{S_k} is distributed according to $\mathcal{N}(0, \Sigma(S_k))$, where $\Sigma(S_k)$ denotes the sub-block of the Σ corresponding to the indices in S_k. The algorithm is also allowed to be *adaptive* in that the choice of (S_k, n_k) is allowed to depend on all the previous samples obtained. Notice that the "passive" algorithms (e.g., [11], [13], [4]) can be thought of as operating in a single stage with $S_1 = [p]$.

We will now define a natural metric for evaluating the performance of a graphical model selection algorithm that both allows us to compare the algorithms proposed here with their passive counterparts, and reflects the penalty for obtaining samples from large subsets of variables. Towards this end, first observe that the total number of scalar samples obtained by an algorithm as defined in the previous paragraph is given by $\sum_{k=1}^{L} |S_k| \, n_k$. Then, we may define the following.

Definition 1 (Total Sample Complexity)**.** *Fix $\delta \in (0, 1)$. Suppose that an algorithm returns an estimate $\widehat{E}_n$ given a budget of n total scalar samples. We will say that its total sample complexity at confidence level δ is n_0 if for all $n \geq n_0$, $\mathbb{P}\left[\widehat{E}_n \neq E\right] \leq 1 - \delta$.*

Our primary objective will be to produce active marginalization algorithms and demonstrate sufficient

[1] Our general framework applies much more broadly. We assume Gaussian distributions for ease of presentation.

conditions on their total sample complexity (at some given level δ). These sufficient conditions are expressed as a function of the graph size p, the magnitudes of the entries in the covariance matrix Σ, and an interesting structural property of the graph. To define this, we require a few more definitions. For a vertex $i \in [p]$, we will define its neighborhood $N_G(i) \triangleq \{j \in [p] : \{i, j\} \in E\}$ and the closure of its neighborhood $\overline{N}_G(i) \triangleq N_G(i) \cup \{i\}$. The size of the set $N(i)$ will be referred to as the degree $d(i)$ of this vertex[2] and we define the *maximum degree* $d_{\max} \triangleq \max_{i\in[p]} d(i)$. In addition to these standard definitions, we also define the *local maximum degree* of a vertex as $d^i_{\max} \triangleq \max_{j\in\overline{N}(i)} d(j)$, that is, the maximum degree in i's closed neighborhood. We also define the following quantity

$$\bar{d}_{\max} \triangleq \frac{1}{p}\sum_{i=1}^{p} d^i_{\max}, \tag{2}$$

which is the average of the local maximum degrees. As we will see in the sequel, $\bar{d}_{\max}$, which is a more local notion of the complexity of the graph structure, will play a central role in the statement of our results concerning the total sample complexity of active learning algorithms for graphical model selection.

First, we observe that in the passive setting, the total sample complexity (TSC) at some (constant) confidence level δ is typically shown to be $\mathcal{O}(d_{\max} p \log p)$. For instance, using the results of [14], we can deduce that the Lasso based neighborhood selection method of [11] has a TSC which is no more than $\mathcal{O}(d_{\max} p \log p)$. Similarly, the PC algorithm [15], which is a smart exhaustive search algorithm, has a TSC that scales like $\mathcal{O}(d_{\max} p \log p)$ (see [13]).

On the other hand, in Sections 4 and 5 we will demonstrate algorithms whose total sample complexity (TSC) is bounded from above by $\mathcal{O}(\bar{d}_{\max} p \log p)$. By definition, $\bar{d}_{\max} \leq d_{\max}$, and it is not hard to see that $\bar{d}_{\max}$ can be significantly smaller than $d_{\max}$ when the graph has a heterogenous degree sequence, which is quite typical in many practical applications. Therefore the total sample complexity of our algorithms scales at a much better rate than their passive counterparts, which also results in a significant practical advantage as shown by our preliminary simulation study in Section 6.

In the wake of the significant progress made in the high-dimensional statistics literature, we can reason about these results intuitively. To discover the neighborhood of a vertex $i \in [p]$, it should suffice to sample i and (at least) its neighbors $\mathcal{O}(d_i \log p)$. This implies that the neighbor of i with maximum degree in $\bar{N}(i)$ will force i to be sampled $\mathcal{O}(d^i_{\max} \log p)$ times. Therefore, a total of $\mathcal{O}(\bar{d}_{\max} p \log p)$ samples should suffice, if the algorithm is able to focus its samples on the right subsets of vertices. We demonstrate in Sections 4 and 5 exactly how this can be achieved. It is also curious to note that $\bar{d}_{\max}$ appears as the natural notion of local complexity of the graph rather than, say, the average degree. Understanding whether this quantity is fundamental to active learning of graphical models, via a minimax lower bound, for instance, is an extremely interesting avenue for future work.

Formal lower bound arguments for passive algorithms for recovering the structure of Gaussian graphical models exist in [16], which establishes that for the class of graphs with maximum degree $d_{\max}$, if the smallest partial correlation coefficient between any pair of nodes conditioned on any subset of nodes is bounded from below by a constant, then the passive algorithms require $\Omega(d_{\max} p \log p)$ scalar observations. The class of graphs with average maximum degree $\bar{d}_{\max}$ is a subset of the class of graphs with maximum degree $d_{\max}$, implying that the lower bound may not apply to the former. However, a careful investigation of the lower bound construction in [16] reveals that the lower bound is based on hardest examples from the class of graphs with a single clique of size $d_{\max}$ and all other nodes disconnected, for which $\bar{d}_{\max} = d^2_{\max}/p \leq d_{\max}$. This establishes that, for the class of graphs whose $d_{\max}$ and $\bar{d}_{\max}$ are such that $\bar{d}_{\max} \leq d^2_{\max}/p$, all passive algorithms are much worse than the active algorithms proposed in this paper.

$\underline{\bar{\mathbf{d}}_{\max}\ \textbf{vs.}\ \mathbf{d}_{\max}}$. As mentioned earlier, the total sample complexity of our algorithms scale like $\mathcal{O}(\bar{d}_{\max} p \log p)$ as opposed to the typical $\mathcal{O}(d_{\max} p \log p)$ scaling of passive algorithms. By definition, we can see that $\bar{d}_{\max} \leq d_{\max}$. There are many situations where $\bar{d}_{\max}$ can be significantly smaller than $d_{\max}$, potentially allowing the active algorithms we propose to yield steep savings in terms of the total sample complexity by running the active algorithm. As is often the case in real-world graphs, if the degree distribution is non-uniform, it is likely the case that $\bar{d}_{\max}$ is significantly smaller than $d_{\max}$. For instance, consider a graph on p vertices where there are $\Theta(p)$ low (say $\Theta(1)$) degree vertices, and there are very few (say $\Theta(1)$) vertices with degree d. For such a graph, observe that $\bar{d}_{\max}$ scales like d^2/p, while $d_{\max}$ scales like d.

Another interesting case is that of graphs with small *correlation dimension* [17], which can be thought of as a global version of the doubling dimension (see, e.g., [18] for more on doubling dimension). κ is said to be the correlation dimension of $G = ([p], E)$ if κ is the smallest constant such that for all $r \in [p]$: $\sum_{i\in[p]} |B(i, 2r)| \leq 2^\kappa \sum_{i\in[p]} |B(i, r)|$, where $B(i, r)$ is

[2] We will suppress the dependence on G when it is clear from the context

the set of all vertices that are at most r away (in shortest path distance) from i. Such graphs are of interest since they, like graphs with small doubling dimension, are amenable to more efficient graph processing algorithms (see [17] and references there in). Suppose a graph has a correlation dimension of κ, then it is not hard to see that $\bar{d}_{\max} \leq 2^{\kappa} \sum_i d_i/p$, i.e., $\bar{d}_{\max}$ is controlled by the average degree of the graph, which can be significantly smaller than $d_{\max}$.

3 A GENERAL FRAMEWORK

In this section, we will first describe our general framework for building an active algorithm for graphical model selection; this is described in Algorithm 1. We note here that the ideas and results here are not predicated upon the assumption that X is Gaussian.

Algorithm 1 accepts a natural number B which is a budget for the total number of scalar samples allowed. In our theoretical analysis, we will identify a sufficient condition for the budget in terms of natural parameters of the graphs we wish to learn. Algorithm 1 also accepts "sample complexity functions" $g, f : \mathbb{N} \to \mathbb{N}$. Finally, the algorithm depends on two subroutines: **nbdSelect**() and **nbdVerify**(). The former subroutine takes as input (the index of) a vertex i, a candidate neighborhood size ℓ, and samples from a subset S of the variables. It then return a subset of S of size no more than ℓ as its estimate $\widehat{N}(i)$ of the neighborhood of i. **nbdVerify()** accepts a vertex i, a candidate neighborhood $\widehat{N}(i)$, and samples from the variables in $\textsc{settled}^c = [p] \backslash \textsc{settled}$. It then checks if $\widehat{N}(i)$ is indeed a potential neighborhood of i. We will refer to $g()$ and $h()$ as the sample complexity functions of the subroutines **nbdSelect** and **nbdVerify** respectively. At this point, we will let these subroutines and their sample complexity functions remain abstract and focus on the structural details of our active graphical model selection framework. Sections 4 and 5 will demonstrate two explicit instantiations of this framework when X is Gaussian, and the application to more general distributions would simply involve establishing appropriate instantiations of **nbdSelect**(), **nbdVerify**(), $g()$, and $h()$.

We will now describe Algorithm 1. We start with an empty graph on $[p]$, i.e., $\widehat{N}(i) = \emptyset$ for all $i \in [p]$. And initialize the counter ℓ to 1, and the variable nSamples to 0. We also initialize the sets NBDFOUND, SETTLED to $\emptyset$. As the names suggest, NBDFOUND will be used to keep track of the vertices whose neighborhood estimates the algorithm is confident about and SETTLED keeps track of the vertices that no longer need to be sampled from. Notice that the faster SETTLED is populated, the better the performance is of Algorithm 1 in terms of the total sample complexity, since in successive stages only the vertices in $\textsc{settled}^c$ are sampled. The algorithm then loops over ℓ (by doubling it) until one of the following holds: $\ell > 2p$, $\textsc{nbdFound} = [p]$ or nSamples exceeds the budget. At each iteration, the algorithm obtains $g(\ell) + h(\ell)$ new samples from variables in the set $\textsc{settled}^c$ and stores these in S_1 and S_2 as in Steps 3 and 4. Next, in Steps 6-11, for each vertex $i \notin \textsc{settled}$ the algorithm uses the subroutine **nbdSelect**() to estimate a neighborhood of i of size at most ℓ. And, then if this neighborhood passes the check of the subroutine **nbdVerify**, the algorithm adds i to the set **nbdFound**. Finally, in Steps 12 - 15, the set SETTLED gets updated. Any i in NBDFOUND whose entire estimated neighborhood is in NBDFOUND gets enrolled in SETTLED and does not get sampled henceforth. That is, the algorithm "settles" a vertex $i \in [p]$ if it is both confident about the vertex's neighborhood and about the neighborhood of i's neighbors. It is this step that gives our algorithm its improved total sample complexity.

We are now ready to state the following result that characterizes the performance of Algorithm 1. We will postpone the proof, which formalizes the intuition of the above marginalization argument, to the supplementary material.

Theorem 1. *Fix $\delta \in (0,1)$. For each $\ell \leq d_{\max}$, assume that the subroutines* **nbdSelect** *and* **nbdVerify** *satisfy:*

(C1) *For any vertex $i \in [p]$ and subset $F \subseteq [p]$ that are such that $|N(i)| = d_i \leq \ell$ and $\overline{N}(i) \subseteq F$, the following holds. Given $g(\ell)$ samples from X_F,* **nbdSelect**$(i, \ell, \left\{X_F^{(j)}\right\}_{j\in S_1})$ *returns the true neighborhood of i with probability greater than $1 - \delta/2pd_{\max}$.*

(C2) *For any vertex $i \in [p]$ and subsets $F, H \subseteq [p]$ that are such that $|N(i)| = d_i \leq \ell$, $\overline{N}(i) \subseteq F$, and $H \subseteq F$, the following holds. Given $h(|H|)$ samples from X_F,* **nbdVerify** $(i, H, \left\{X_F^{(j)}\right\})$ *returns* **true** *if and only if $N(i) \subseteq H$ with probability greater than $1 - \delta/2pd_{\max}$.*

Then, with probability no less than $1 - \delta$, Algorithm 1 returns the correct graph. Furthermore, it suffices if $B \geq \sum_{i\in[p]} \sum_{0\leq k \leq \lceil \log_2 d^i_{\max} \rceil} g(2^k) + h(2^k))$. That is, Algorithm 1 has a total sample complexity of $\sum_{i\in[p]} \sum_{0\leq k \leq \lceil \log_2 d^i_{\max} \rceil} g(2^k) + h(2^k)$ at confidence level $1 - \delta$.

Theorem 1 tells us that if the subroutines **nbdSelect** (resp. **nbdVerify**) satisfies condition (C1) (resp. (C2)) with probability exceeding $1 - \delta/2pd_{\max}$ for a fixed ℓ with $g(\ell)$ (resp. $h(\ell)$) samples, then Algorithm 1 has a TSC of $\sum_{i\in[p]} \sum_{k \leq \lceil \log_2 d^i_{\max} \rceil} g(2^k) +$

Algorithm 1 Active Neighborhood Selection

```
Require: budget $B \in \mathbb{N}$, sample complexity functions $g(), h()$, subroutines nbdSelect, nbdVerify.
Set $\ell = 1$, nSamples $= 0$, and initialize $\widehat{N}(i)$, $\forall i \in [p]$, NBDFOUND, SETTLED, $S_1$, $S_2$ to $\emptyset$ (the empty set).
    /* Obtain new samples from SETTLED$^c$ */
repeat
    Obtain $g(\ell)$ independent samples $X^{(j)}_{\text{SETTLED}^c}, j = 1, \ldots, g(\ell)$; add to $S_1$
    Obtain $h(\ell)$ independent samples $X^{(j)}_{\text{SETTLED}^c}$, $j = 1, \ldots, h(\ell)$; add to $S_2$
    Increment nSamples by $(p - |\text{SETTLED}|) \times (g(\ell) + h(\ell))$.
    /* Generate and verify candidate neighborhoods */
    for $i \in \text{NBDFOUND}^c$ do
        $\widehat{N}(i) =$ nbdSelect$(i, \ell, \{X^{(j)}_{\text{SETTLED}^c}\}_{j \in S_1})$        /* $|\widehat{N}(i)|$ is at most $\ell$ */
        if (nbdVerify$(i, \widehat{N}(i), \{X^{(j)}_{\text{SETTLED}^c}\}_{j \in S_2})$ = true) then
            NBDFOUND = NBDFOUND $\cup \{i\}$
        end if
    end for
    /* Settle vertices */
    for $i \in$ NBDFOUND do
        if $\widehat{N}(i) \subseteq$ NBDFOUND then SETTLED = SETTLED $\cup \{i\}$
        end if
    end for
    Set $\ell = 2 \times \ell$, $S_1, S_2 = \emptyset$.
until $\ell \geq 2p$ or NBDFOUND $= [p]$ or nSamples $> B$
return Graph $\widehat{G}$ such that $\{i,j\} \in \widehat{G} \Leftrightarrow i \in \widehat{N}(j)$ or $j \in \widehat{N}(i)$
```

$h(2^k)$ at a confidence level $1 - \delta$. In what follows, we will show two specializations of Algorithm 1. Our strategy to establish the TSC of these algorithms will be to first estimate the probability that **nbdSelect** and **nbdFound** fail to satisfy (C1) and (C2). This will suggest a choice for the functions $g()$ and $h()$, which can then be used to identify the total sample complexity.

4 The AdPaCT Algorithm

We call our first instantiation of Algorithm 1, the AdPaCT algorithm, which stands for Adaptive Partial Correlation Testing. In this section, for the sake of simplicity and concreteness, we will assume that X follows the 0 mean multivariate normal distribution with a covariance matrix $\Sigma \in \mathbb{R}^{p \times p}$.

First, we observe that since $X \sim \mathcal{N}(0, \Sigma)$, the partial correlation coefficient contains all the conditional independence information of the distribution. In particular, for a pair of vertices $i, j \in [p]$, and for a subset $S \subseteq [p]$, the partial correlation coefficient $\rho_{ij|S} = 0$ if and only if $X_i \perp\!\!\!\perp X_j \mid X_S$ (except for a pathological, measure 0 set of covariance matrices). As the name suggests, the AdPaCT algorithm uses estimates of these partial correlation coefficients in order to learn the graph represented by Σ. Recall that the partial correlation coefficient $\rho_{ij|S}$ satisfies the following recursive relationship which holds for any $k \in S$:

$$\rho_{i,j|S} = \frac{\rho_{i,j|S\setminus\{k\}} - \rho_{i,k|S\setminus\{k\}}\rho_{j,k|S\setminus\{k\}}}{\sqrt{\left(1 - \rho^2_{i,k|S\setminus\{k\}}\right)\left(1 - \rho^2_{j,k|S\setminus\{k\}}\right)}}. \tag{3}$$

In order to compute empirical estimates of these quantities one can begin with the natural empirical estimates of the correlation coefficients and substitute recursively in the above formula, or equivalently, one might invert the relevant sub-matrices of the empirical covariance matrix of the observed data. In what follows, we will write $\widehat{\rho}_{ij|S}$ to mean either of these estimates; our theoretical results hold for both.

We will now describe Algorithm 2. The framework provided by Algorithm 1 will allow us to do this by prescribing choices for the subroutines **nbdSelect** and **nbdVerify** and the functions $g()$ and $h()$.

We choose the sample complexity functions $g(\ell) = \lceil c\ell \log p \rceil$ and $h(\ell) = 0$. The **nbdSelect** subroutine exhaustively searches over all subsets $S \subseteq F$ (note that F will be $[p]\backslash$SETTLED when **nbdSelect** is called inside of Algorithm 1) of cardinality between $\ell/2 + 1$ and ℓ to find the smallest set $\widehat{S}$ which is such that $\max_{j \notin \widehat{S}} \left|\widehat{\rho}_{i,j|\widehat{S}}\right| \leq \xi$. If such a set is not found, then the subroutine returns the empty set.

Observe that **nbdSelect** on its own performs conditional independence tests to ensure that it is returning the right neighborhood. Therefore, **nbdVerify** simply returns **true** unless $S = \emptyset$. It is worth observing here that the passive counterpart of this algorithm is a natural algorithm for Gaussian graphical model selection and indeed serves as the foundation for various algorithms in literature like the CCT algorithm of [6] and the PC algorithm (see e.g., [15]).

In order to theoretically characterize the performance of Algorithm 2, we need the following assumptions.

(A1) The distribution of X is faithful to G.

(A2) For each $i, j \in [p]$ and $S \subseteq [p] \setminus \{i, j\}$, $\left|\rho_{i,j|S}\right| \leq M$. And, for each $i, j \in [p]$ and $S \subseteq [p]$, if $X_i \not\perp X_j \mid X_S$, then $\left|\rho_{i,j|S}\right| \geq m$.

Assumption (A1) is a standard assumption in the graphical model selection literature and is violated only on a set of measure 0 (see e.g., [15,19]). Assumption (A2) has appeared in the literature (e.g., [13]) as a way of strengthening the faithfulness assumption. While the upper bound in assumption (A2) is a mild regularity condition, the lower bound of (A2) may be hard to verify in practice. However, under certain parametric and structural conditions, one can obtain a handle on m. For example, the authors in [6] show that if the underlying graph has small local separators and if the concentration matrix is *walk-summable*, then m in (A2) can be replaced essentially by the smallest non-zero entry of the concentration matrix.

We can now state the following theorem about the performance of the AdPaCT algorithm.

Theorem 2. *Fix $\delta \in (0, 1)$ and suppose that assumptions (A1) and (A2) hold. Then, there exists a constant $c = c(m, M, \delta)$ such that if we set $g(\ell) = \lceil c\ell \log p \rceil$ and $\xi = m/2$, then with probability no less than $1 - \delta$, the following hold:*

1. The AdPaCT algorithm successfully recovers the graph G.

2. The computational complexity of the AdPaCT algorithm is no worse than $\mathcal{O}(p^{d_{\max}+2})$

This implies that the total sample complexity of the AdPaCT algorithm at confidence level $1 - \delta$ is bounded by $2c\bar{d}_{\max} p \log p$.

To prove this theorem, as mentioned earlier, we show that the choice for the subroutines **nbdSelect** and **nbdVerify** satisfy the conditions (C1) and (C2) of Theorem 1 with high probability. We bound the event that **nbdSelect** fails in some iteration ℓ in terms of concentration inequalities for the partial correlation coefficient. This gives us a corresponding choice of $g(\ell)$ that determines the TSC of the AdPaCT algorithm. Please refer to the supplementary material for the details of the proof.

It is clear that this procedure is advantageous over passive algorithms in situations where $d_{\max}$ is large compared to $\bar{d}_{\max}$. Unfortunately, in these settings which lend themselves to improved sample complexity, the computational complexity of the AdPaCT algorithm could be prohibitively large. In the next section, we will propose a different instantiation of Algorithm 1 based on the Lasso [20] which achieves sample complexity savings similar to Algorithm 2 whilst also enjoying vastly lower computational complexity.

5 The AMPL algorithm

In this section, we will discuss a computationally efficient active marginalization algorithm for learning graphs. As alluded to in Section 4, this algorithm uses Lasso as an efficient means for neighborhood selection and hence can be thought of as an active version of the seminal work of [11]. We call this algorithm AMPL for Adaptive Marginalization based Parallel Lasso.

As in Section 4, we will describe the algorithm by prescribing choices for the subroutines **nbdSelect** and **nbdVerify** and the functions $g()$ and $h()$.

Algorithm 2 AdPaCT: Adaptive Partial Correlation Testing

```
Require: Budget $B \in \mathbb{N}$, a constant $c > 0$, and threshold $\xi > 0$.
Sample Complexity Functions
 1: $g(\ell) = \lceil c\ell \log p \rceil$, $h(\ell) = 0$.
nbdSelect($i, \ell, \{X_F^{(j)}\}_{j \in S_1}$)
 2: for $k = \ell/2 + 1, \ell/2 + 2, \ldots, \ell$ do
 3:     $\mathcal{S} = \{S \subseteq F : |S| = k, \max_{j \notin S} |\widehat{\rho}_{i,j|S}| \leq \xi\}$
 4:     if $\mathcal{S} = \emptyset$ then continue   (i.e., go to Step 7)
 5:     else return $\widehat{S} = \arg\min_{S \in \mathcal{S}} \max_{j \notin S} |\widehat{\rho}_{i,j|S}|$
 6:     end if
 7: end for
 8: return $\emptyset$
nbdVerify($i, S, \{X_F^{(j)}\}_{j \in S_2}$)
 9: if  $S \neq \emptyset$  then return true
10: else return false
11: end if
```

We choose the sample complexity functions to be $g(\ell) = h(\ell) = c\ell \log p$. **nbdSelect** operates as follows. Let y denote the vector of samples corresponding to the random variable X_i and let $\mathbf{X}$ denote the corresponding matrix of samples from the the random variables $X_{F \setminus \{i\}}$. The subroutine solves the following optimization program

Algorithm 3 AMPL: Adaptive Marginalization based Parallel Lasso

```
Require: Budget $B \in \mathbb{N}$, a constant $c > 0$, and threshold $\xi > 0$.
Sample Complexity Functions
  $g(\ell) = h(\ell) = \lceil \ell c \log p \rceil$.
nbdSelect$(i, \ell, \{X_F^{(j)}\}_{j \in S_1})$
  Let $y \in \mathbb{R}^{\lceil \ell c \log p \rceil}$ be the vector of samples from $X_i$ in $\mathcal{S}_1$.
  Let $\mathbf{X} \in \mathbb{R}^{(\lceil \ell c \log p \rceil) \times (p - |\text{SETTLED}| - 1)}$ be the corresponding matrix of samples from $X_{[p] \setminus \{\text{SETTLED} \cup \{i\}\}}$.
  $\widehat{\beta} \leftarrow$ Lasso$(y, \mathbf{X})$
  if $\left|\text{supp}(\widehat{\beta})\right| > \ell$ then return top $\ell$ coordinates of $\left|\widehat{\beta}\right|$
  else return $\widehat{\beta}$
  end if
nbdVerify$(i, S, \{X_F^{(j)}\}_{j \in S_2})$
  if for each $j \in [p] \setminus F \cup S \cup \{i\}$, $\left|\widehat{\rho}_{i,j|S}\right| \leq \xi$ then return true
  else return false
  end if
```

$$\widehat{\beta} = \underset{\beta \in \mathbb{R}^{|F|-1}}{\arg\min} \frac{1}{2n_{i,\ell}} \|y - \mathbf{X}\beta\|_2^2 + \lambda_{\ell,i} \|\beta\|_1 , \tag{4}$$

where the choice of $\lambda_{\ell,i}$ is stated in Theorem 3 and $n_{i,\ell}$ is the number of samples (i.e., dimension of y). If the size of the support of $\widehat{\beta}$ is greater than ℓ, then the algorithm returns the ℓ largest coordinates of $\left|\widehat{\beta}\right|$, else the algorithm returns the support of $\widehat{\beta}$. **nbdVerify** returns **true** if $\left|\widehat{\rho}_{i,j|\widehat{N}(i)}\right| \leq \xi$ for every $j \in [p] \setminus (\text{SETTLED} \cup \widehat{N}(i) \cup \{i\})$, else it returns **false**.

Before we can state our theoretical results on the performance of the AMPL algorithm, we need to make some assumptions. For each $i \in [p]$, let $\Sigma^{\setminus i}$ denote the covariance matrix with the $i-$the row and column removed and set $S_i \triangleq N(i)$. We will assume the following conditions.

(A3) There exists a constant $\gamma \in (0, 1]$ such that for all $i \in [p]$, $\widetilde{\Sigma}^i$ satisfies the following:

$$\left|\left|\left| \widetilde{\Sigma}^i_{S_i^c S_i} \left(\widetilde{\Sigma}^i_{S_i S_i} \right)^{-1} \right|\right|\right|_\infty \leq 1 - \gamma$$

(A4) There exist constants $0 < C_{\min} \leq C_{\max} < \infty$ such that for all $i \in [p]$, the covariance matrix $\widetilde{\Sigma}^i$ also satisfies the following:

$$C_{\min} \leq \Lambda_{\min} \left(\widetilde{\Sigma}^i_{S_i S_i} \right) \leq \Lambda_{\max} \left(\widetilde{\Sigma}^i_{S_i S_i} \right) \leq C_{\max}$$

Assumption (A3) is a kind of incoherence assumption often dubbed the *irrepresentability condition*; similar assumptions have appeared in the literature for graphical model selection [3,4,11] and in the analysis of the Lasso [21]. Intuitively speaking, this restricts the influence that non-edge pairs of vertices have on the pairs of vertices that are edges.

Assumption (A4) is a commonly imposed regularity condition on the covariance matrix.

We can now state the following theorem that characterizes the performance of the AMPL algorithm.

Theorem 3. *Fix $\delta > 0$. Suppose that assumptions (A1)-(A4) hold. There exists constants C_1, C_2, C_3 which depend on Σ, m, δ such that if we set $c = C_1$ (i.e., $g(\ell) = c\ell \log p$), $\xi = m/2$, $\lambda_\ell = \sqrt{\frac{2C_2 \|\Sigma\|_\infty}{C_1 \gamma^2}}$, and budget $B = 2c\bar{d}_{\max} p \log p$, then with probability at least $1 - \delta$, the following hold*

1. *the AMPL algorithm successfully recovers the graph G,*

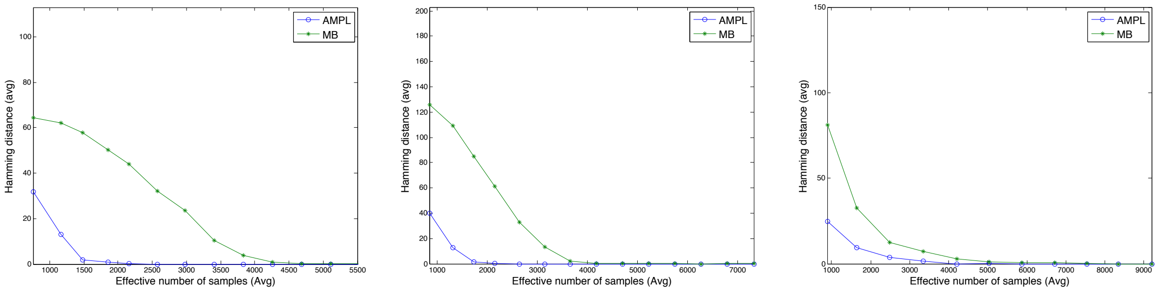
2. *the computational complexity is bounded from above by $\mathrm{d}_{\max} p \mathfrak{C}$, where $\mathfrak{C}$ is the computational cost of solving a single instance of Lasso,*

provided $m \geq \left(\frac{C_{\min}}{C_{\max}} + \frac{C_{\max}}{C_{\min}} + 2 \right) \times \frac{1}{4 \min_i |\Sigma_{ii}|} \left[C_3 \sqrt{\frac{2C_1 \|\Sigma\|_\infty}{C_2 \gamma^2}} \max_i \left|\left|\left| \left(\widetilde{\Sigma}^i_{N(i),N(i)} \right)^{-1/2} \right|\right|\right|_\infty^2 + 20 \sqrt{\frac{\|\Sigma\|_\infty}{C_{\min} C_2}} \right].$

Again, we prove this theorem by showing that our choice for the subroutines **nbdSelect** and **nbdVerify** satisfy the conditions (C1) and (C2). The proof characterizing the behavior of **nbdVerify** is very similar to the proof of Theorem 2. On the other hand, to characterize the behavior of **nbdSelect**, one part of our reasoning is similar to the argument in [14, Theorem 3]. The rest of the proof follows from a strengthening of the argument in [14] for the case when the degree is $o(\log p)$. We also needed to be cognizant of the fact that our adaptive marginalization approach results in samples in different stages having different distributions. We refer the interested reader to the supplementary material for the details.

6 SIMULATIONS

We will now describe some preliminary experimental results. In particular, we will focus on the computationally efficient AMPL algorithm (Algorithm 3)



Single Clique Multiple Cliques Power Law

Figure 1: Plot of Effective number of Samples (total number of scalars/p) -v- Hamming Error

Table 1: Comparison of ESC (10 trials)

Graph	ESC (0.9)		ESC(1)	
	AMPL	MB	AMPL	MB
Single Clique	1202.1	3361.8	1202	3361.9
Mult. Cliques	1154.3	2943.8	2649.5	6216.1
Power Law	1280.2	2300.4	4212.8	8004.7

and compare it to its natural passive counterpart, the neighborhood regression algorithm from [11] (henceforth, MB). We implemented each algorithm using the Glmnet package [22], where we tuned the parameters so that each algorithm achieved the best model selection performance against the true model.

We evaluated these algorithms on the following graphs:

(a) **Single Clique :** a graph on $p = 60$ vertices composed of a clique on 12 vertices and a chain graph connecting the rest. Observe that here $d_{\max} = 11$ and $\bar{d}_{\max} = 3.8$.

(b) **Multiple Cliques:** a graph on $p = 100$ vertices with 4 (disconnected) cliques of sizes $5, 8, 10, 11$ respectively; the rest of the graph is again connected with a chain. Here, $d_{\max} = 10$ and $\bar{d}_{\max} = 4.08$

(c) **Power Law:** a (random) power law graph on $p = 60$ vertices generated according to the Barabási-Albert model with a randomly generate sparse seed graph on 5 vertices. These graphs are such that the number of vertices of degree x in the graph behaves like $a{*}x^{-b}$ for constants $a, b > 0$. See [23] for more on these graphs. For a typical graph we generated, we onserved that $d_{\max} = 13$ and $\bar{d}_{\max} = 3.68$.

In Table 1, we report the comparison between AMPL and MB. To run AMPL, we choose the constant c (cf. Section 5) and allow the algorithm to run until it reports all vertices are settled. We then record (i) the total number of scalar samples consumed by the algorithm divided by p; we call this the effective sample complexity, and (ii) its model selection performance in terms of the hamming distance between the estimated and the true adjacency matrices. We then ran MB for various sample sizes and recorded its hamming error performance. In Table 1, we report two numbers for each algorithm and each graph: (a) ESC(0.9): the average (over 10 trials) number of effective samples required to get atleast 90% of the edges, and (b) ESC(1): the average number of effective samples required to get all the edges right. Notice that ESC(1)$\times p$ is the total number of scalar samples consumed.

As we can see from Table 1, the AMPL algorithm clearly outperforms the neighborhood regression algorithm of Meinshausen and Buhlmann in the situations where the degree distribution of the true graphs is non-uniform.

We also compare the "average" hamming error performance of these two algorithms in Figure 1. Since the total sample complexity of the AMPL algorithm is a random number, we adopt the following protocol to generate these plots: We first choose a value for the constant c (cf. Section 5) and allow AMPL to run to completion. We then compute the effective number of samples by dividing the total number of scalars by the size of the graph (p). These many independent samples are then fed to MB and its hamming error performance is recorded. For each value of c, we repeat this 10 times and record the average effective number of samples and the average hamming error for both AMPL and MB. These results are shown in Figure 1.

Acknowledgements

A. S. was supported in part by NSF grants IIS-1247658, CAREER IIS-1252412, and AFOSR YIP FA9550-14-1-0285. M. F. B was supported in part by grants NSF CCF-145117, NSF CCF-1422910, NSF CCF-1535967, and a Sloan Fellowship.

References

[1] V. A. Marčenko and L. A. Pastur, "Distribution of eigenvalues for some sets of random matrices," *Sbornik: Mathematics*, vol. 1, no. 4, pp. 457–483, 1967.

[2] I. M. Johnstone, "On the distribution of the largest eigenvalue in principal components analysis," *Annals of statistics*, pp. 295–327, 2001.

[3] P. Ravikumar, M. J. Wainwright, and J. D. Lafferty, "High-dimensional ising model selection using 1-regularized logistic regression," *The Annals of Statistics*, vol. 38, no. 3, pp. 1287–1319, 2010.

[4] P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu, "High-dimensional covariance estimation by minimizing 1-penalized log-determinant divergence," *Electronic Journal of Statistics*, vol. 5, pp. 935–980, 2011.

[5] A. Anandkumar, V. Y. Tan, F. Huang, A. S. Willsky, *et al.*, "High-dimensional structure estimation in ising models: Local separation criterion," *The Annals of Statistics*, vol. 40, no. 3, pp. 1346–1375, 2012.

[6] A. Anandkumar, V. Y. Tan, F. Huang, and A. S. Willsky, "High-dimensional gaussian graphical model selection: Walk summability and local separation criterion," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 2293–2337, 2012.

[7] S. Keshri, E. Pnevmatikakis, A. Pakman, B. Shababo, and L. Paninski, "A shotgun sampling solution for the common input problem in neural connectivity inference," *arXiv preprint arXiv:1309.3724*, 2013.

[8] S. Turaga, L. Buesing, A. M. Packer, H. Dalgleish, N. Pettit, M. Hausser, and J. Macke, "Inferring neural population dynamics from multiple partial recordings of the same neural circuit," in *Advances in Neural Information Processing Systems*, pp. 539–547, 2013.

[9] W. E. Bishop and M. Y. Byron, "Deterministic symmetric positive semidefinite matrix completion," in *Advances in Neural Information Processing Systems*, pp. 2762–2770, 2014.

[10] K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan, "Causal protein-signaling networks derived from multiparameter single-cell data," *Science*, vol. 308, no. 5721, pp. 523–529, 2005.

[11] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the lasso," *The Annals of Statistics*, pp. 1436–1462, 2006.

[12] S. L. Lauritzen, *Graphical models*. Oxford University Press, 1996.

[13] M. Kalisch and P. Bühlmann, "Estimating high-dimensional directed acyclic graphs with the pc-algorithm," *The Journal of Machine Learning Research*, vol. 8, pp. 613–636, 2007.

[14] M. J. Wainwright, "Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (lasso)," *Information Theory, IEEE Transactions on*, vol. 55, no. 5, pp. 2183–2202, 2009.

[15] P. Spirtes, C. N. Glymour, and R. Scheines, *Causation, prediction, and search*, vol. 81. MIT press, 2000.

[16] W. Wang, M. J. Wainwright, and K. Ramchandran, "Information-theoretic bounds on model selection for gaussian markov random fields," in *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pp. 1373–1377, IEEE, 2010.

[17] T.-H. H. Chan, *Approximation algorithms for bounded dimensional metric spaces*. ProQuest, 2007.

[18] A. Slivkins, "Distance estimation and object location via rings of neighbors," *Distributed Computing*, vol. 19, no. 4, pp. 313–333, 2007.

[19] J. Pearl, *Causality: models, reasoning and inference*, vol. 29. Cambridge Univ Press, 2000.

[20] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[21] S. A. Van De Geer, P. Bühlmann, *et al.*, "On the conditions used to prove oracle results for the lasso," *Electronic Journal of Statistics*, vol. 3, pp. 1360–1392, 2009.

[22] J. Qian, T. Hastie, J. Friedman, R. Tibshirani, and N. Simon, "Glmnet for matlab (2013),"

[23] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of modern physics*, vol. 74, no. 1, p. 47, 2002.