LEARNING GAUSSIAN GRAPHICAL MODELS USING DISCRIMINATED HUB GRAPHICAL LASSO

Zhen Li, Jingtian Bai, Weilian Zhou

Department of Statistics, North Carolina State University, Raleigh, NC 27695
zli34@ncsu.edu, jbai4@ncsu.edu, wzhou11@ncsu.edu

ABSTRACT

We develop a new method called Discriminated Hub Graphical Lasso (DHGL) based on Hub Graphical Lasso (HGL) by providing the prior information of hubs. We apply this new method in two situations: with known hubs and without known hubs. Then we compare DHGL with HGL using several measures of performance. When some hubs are known, we can always estimate the precision matrix better via DHGL than HGL. When no hubs are known, we use Graphical Lasso (GL) to provide information of hubs and find that the performance of DHGL will always be better than HGL if correct prior information is given, and will rarely degenerate when the prior information is incorrect.

Index Terms— Gaussian graphical model, precision matrix, graphical Lasso, discriminated hub graphical Lasso, prior information

1. INTRODUCTION

Graphical model has been widely used in a variety of fields. A graph consists of nodes, representing variables, and edges between nodes, representing the conditional dependency between two variables [1]. In order to get a sparse and interpretable estimate of the graph, many researches such as those from Yuan & Lin (2007) and Friedman et al. (2007) have considered the optimization problem in the form $\min_{\mathbf{\Theta}\in S}\{\ell(\mathbf{X},\mathbf{\Theta})+\lambda\left\|\mathbf{\Theta}-\text{diag}(\mathbf{\Theta})\right\|_1\}$, where $\mathbf{X}$ is an $n\times p$ data matrix, $\mathbf{\Theta}$ is a $p\times p$ symmetric matrix which contains the variables of interest, and $\ell(\mathbf{X},\mathbf{\Theta})$ is a loss function [2][3][4][5][6]. In many cases, we assume that $\ell(\mathbf{X},\mathbf{\Theta})$ is convex in $\mathbf{\Theta}$. There is a specific kind of nodes called hubs, which are connected to a large number of other nodes [7]. This is also an important characteristic in scale-free networks [8][9][10]. Tan et al. (2014) had proposed a method called Hub Graphical Lasso (HGL) to estimate the graph with hubs [5]. They decompose $\mathbf{\Theta}$ as $\mathbf{Z}+\mathbf{V}+\mathbf{V}^T$, where $\mathbf{Z}$ represents conditional dependency between non-hub nodes, and $\mathbf{V}$ represents conditional dependency between hub nodes and other nodes. Instead of the l_1 penalty in the above optimization problem, they used Hub Penalty Function which is the minimal of the term $\lambda_1\left\|\mathbf{Z}-\text{diag}(\mathbf{Z})\right\|_1+\lambda_2\left\|\mathbf{V}-\text{diag}(\mathbf{V})\right\|_1+\lambda_3\left\|(\mathbf{V}-\text{diag}(\mathbf{V}))_j\right\|_q$, with respect to $\mathbf{V},\mathbf{Z}$ such that $\mathbf{\Theta}=\mathbf{Z}+\mathbf{V}+\mathbf{V}^T$ [5]. Notice that $\|\cdot\|_1$ denotes the sum of the absolute values of all the matrix elements, while $\|(\cdot)_j\|_q$ denotes the q-norm of the jth column of the matrix.

Since HGL does not require prior information about hubs, and the number of hubs found by HGL is always not more than that of true hubs, we hope to refine HGL by providing the prior information about hubs if we have some beforehand. We introduce the Discriminated Hub Penalty Function $\text{P}(\mathbf{\Theta})$, which is the minimal of $\text{Q}(\mathbf{\Theta})$ with respect to $\mathbf{V},\mathbf{Z}$ such that $\mathbf{\Theta}=\mathbf{V}+\mathbf{V}^T+\mathbf{Z}$, where

$$\begin{aligned}\text{Q}(\mathbf{\Theta})=\lambda_1\|\mathbf{Z}-\text{diag}(\mathbf{Z})\|_1+\lambda_2\sum_{j\notin\mathcal{D}}\|(\mathbf{V}-\text{diag}(\mathbf{V}))_j\|_1\\+\lambda_3\sum_{j\notin\mathcal{D}}\|(\mathbf{V}-\text{diag}(\mathbf{V}))_j\|_q+\lambda_4\sum_{j\in\mathcal{D}}\|(\mathbf{V}-\text{diag}(\mathbf{V}))_j\|_1\\+\lambda_5\sum_{j\in\mathcal{D}}\|(\mathbf{V}-\text{diag}(\mathbf{V}))_j\|_q\end{aligned}\tag{1}$$

Here $\mathcal{D}\subset\{1,2,\cdots,p\}$ contains prior information of hubs and we impose constraints that $\lambda_4\le\lambda_2,\lambda_5\le\lambda_3$. In the penalty, we give "loose conditions" to nodes in $\mathcal{D}$ so that they tend to be estimated as hubs. We will show that DHGL always performs better than HGL and keeps stable even if incorrect prior information is provided.

2. METHODOLOGY

2.1. Discriminated Hub Graphical Lasso

2.1.1. Optimization Problem

We continue the definitions and notations in Introduction. Combining Discriminated Hub Penalty Function with the loss function, we have the convex optimization problem

$$\begin{aligned}\underset{\mathbf{\Theta}\in\mathcal{S},\mathbf{V},\mathbf{Z}}{\text{minimize}}\quad&\{\ell(\mathbf{X},\mathbf{\Theta})+\text{Q}(\mathbf{\Theta})\}\ ;\\\text{subject to}\quad&\mathbf{\Theta}=\mathbf{V}+\mathbf{V}^T+\mathbf{Z}\end{aligned}\tag{2}$$

where $\mathcal{S}$ depends on the loss function $\ell(\mathbf{X},\mathbf{\Theta})$, and $\text{Q}(\mathbf{\Theta})$ is defined in the Introduction.

Similar to that in Tan et al. (2014), we encourage the solution of $\mathbf{Z}$ to be a sparse symmetric matrix, and $\mathbf{V}$ to be a matrix with columns either entirely zero or almost entirely non-zero. Here $\lambda_1\ge 0$ controls the sparsity in $\mathbf{Z}$ [5]. For variables in $\mathcal{D}$, $\lambda_5\ge 0$ controls the hub nodes selection, and $\lambda_4\ge 0$ controls the sparsity of each hub's connections to other nodes. Similar for λ_2 and λ_3 for variables not in $\mathcal{D}$. Here we set $q=2$, same as Tan et al. (2014) [11][12][5].

As we can see, the "discrimination" involves using different tunning parameters controlling hub selection and hub sparsity for different columns in $\mathbf{V}$, or different variables (variables in $\mathcal{D}$ vs. not in $\mathcal{D}$). When $\mathcal{D}=\varnothing$, it reduces to the convex optimization problem corresponding to Hub Penalty Function in Tan et al. (2014). When $\lambda_2,\lambda_3,\lambda_4,\lambda_5\to\infty$, it reduces to the classical way to obtain a sparse graph estimate. Generally, $\mathcal{D}$ contains variables to be less penalized with $\lambda_4\le\lambda_2$ and $\lambda_5\le\lambda_3$.

Thanks to Eric Chi from Department of Statistics, NC State University for guidence and help.

When $\mathbf{x}_1, \ldots, \mathbf{x}_n \overset{\text{i.i.d.}}{\sim} N(\mathbf{0}, \mathbf{\Sigma})$, we simply set $\ell(\mathbf{X}, \mathbf{\Theta}) = -\log\det\mathbf{\Theta} + \text{trace}(\mathbf{S\Theta})$, where $\mathbf{S}$ is the empirical covariance matrix of $\mathbf{X}$ [3]. Then the Discriminated Hub Graphical Lasso (DHGL) optimization problem is

$$\underset{\mathbf{\Theta}\in\mathcal{S}}{\text{minimize}} \qquad \{-\log\det\mathbf{\Theta} + \text{trace}(\mathbf{S\Theta}) + \text{Q}(\mathbf{\Theta})\}. \qquad (3)$$

with $\mathcal{S} = \{\mathbf{\Theta} : \mathbf{\Theta} \succ 0 \text{ and } \mathbf{\Theta} = \mathbf{\Theta}^T\}$. Again, when $\mathcal{D} = \varnothing$, it reduces to Hub Graphical Lasso (HGL) in Tan et al. (2014), and when $\lambda_2, \lambda_3, \lambda_4, \lambda_5 \to \infty$, it reduces to the classical Graphical Lasso.

2.1.2. Computational Complexity of ADMM

We can use Alternating Direction Method of Multipliers (ADMM) to solve the convex optimization problems proposed above [13][14][15]. The algorithm details as well as derivations are very similar to those of Algorithm 1 in Tan et al. (2014), which is not provided here due to the page limit.

Also notice that the computational complexity is $\mathcal{O}(p^3)$ per iteration for solving DHGL, same in magnitude as HGL. We have also performed a simulation study to illustrate this. We have found that the run time per iteration scale the similar way with p regardless of other parameters, and the number of iterations increases as p increase. Moreover, HGL and DHGL perform similarly in terms of run time and number of iterations. The simulation setup and results are not shown here due to the page limit.

2.2. Tuning Parameter Selection

We can select tunning parameters $(\lambda_1, \lambda_2, \cdots, \lambda_5)$ by minimizing the same BIC-type quantity as proposed by Tan et al. (2014), which is

$$\text{BIC}(\hat{\mathbf{\Theta}}, \hat{\mathbf{V}}, \hat{\mathbf{Z}}) = -n \cdot \log\det(\hat{\mathbf{\Theta}}) + n \cdot \text{trace}(\mathbf{S}\hat{\mathbf{\Theta}}) + \log(n) \cdot |\hat{\mathbf{Z}}| + \log(n) \cdot \Big(\nu + c \cdot [|\hat{\mathbf{V}}| - \nu]\Big) \qquad (4)$$

where the number of estimated hub nodes $\nu = \sum_{j=1}^{p} 1_{\{|\hat{\mathbf{V}}_j| > 1\}}$, $|\cdot|$ denotes the number of unique non-zero matrix elements, and $c \in (0, 1)$ is a constant [5]. Same as Tan et al. (2014), here we choose $c = 0.2$ for our simulation experiments [5].

In order for both optimal solutions for $\mathbf{V}$ and $\mathbf{Z}$, i.e., $\mathbf{V}^\star$ and $\mathbf{Z}^\star$ to be non-diagonal, we can select tuning parameters such that $\lambda_4 \leq \lambda_2, \lambda_5 \leq \lambda_3$ and [5]

$$\frac{\lambda_2}{2} + \frac{\lambda_3}{2(p-1)^{\frac{1}{s}}} \leq \lambda_1 \leq \frac{\lambda_2 + \lambda_3}{2}, \text{where } \frac{1}{s} + \frac{1}{q} = 1$$

Detailed proof is not provided here due to the page limit.

2.3. Algorithms in the Application of DHGL

2.3.1. DHGL Applied with Known hubs

In Tan et al. (2014), HGL is proposed assuming they do not know beforehand which nodes are hubs. However, before estimating the dependency structures among different nodes, we sometimes have domain knowledge of some dependency. In this case, we can use DHGL to utilize the prior information of hubs to estimate $\mathbf{\Theta}$ more accurately. In this section, we give Algorithm 1 using DHGL to estimate $\mathbf{\Theta}$ when some hubs are known.

Notice that here $\mathcal{D}$ contains the prior information, and we propose $\lambda_4 \leq \lambda_2$ and $\lambda_5 \leq \lambda_3$. In HGL, when λ_2 and λ_3 get smaller, more edges and hubs tend to be etimated since entries in $\hat{\mathbf{\Theta}}_{\text{HGL}}$ tend to be larger due to less restrictions, so we can find more correct edges or hubs. In this way, however, the number of incorrect edges or hubs found may also increase. If some hub nodes or prior information about which nodes are likely to be hubs are given, it is encouraged to give"loose conditions" such that only the entries in the corresponding columns of the estimated precision matrix tend to be larger, which is what DHGL does.

Algorithm 1 DHGL Applied with Known Hubs

1. Use HGL to get the estimated hubs $\hat{\mathcal{H}}_{\text{HGL}}$.
2. Set $\mathcal{D} = \mathcal{K} \setminus \hat{\mathcal{H}}_{\text{HGL}}$, where $\mathcal{K}$ is a set of known hubs.
3. If $\mathcal{D} \neq \varnothing$, use DHGL to estimate Θ and get the estimated hubs $\hat{\mathcal{H}}_{\text{DHGL}}$, where λ_1, λ_2, λ_3 remain the same values as those in HGL and λ_4, λ_5 are selected using the BIC-type quantity. Then, set $\hat{\mathcal{H}} = \hat{\mathcal{H}}_{\text{HGL}} \bigcup \hat{\mathcal{H}}_{\text{DHGL}}$ as the set of estimated hubs. If $\mathcal{D} = \varnothing$, use the estimation in HGL directly.

Also notice that we set $\mathcal{D} = \mathcal{K} \setminus \hat{\mathcal{H}}_{\text{HGL}}$. If a known hub node in $\mathcal{K}$ fails to be estimated as a hub node by HGL with too many zeros in the corresponding column of $\hat{\mathbf{\Theta}}_{\text{HGL}}$, we are encouraged to give "loose conditions" to it in order to get estimated $\hat{\mathbf{\Theta}}_{\text{DHGL}}$ with some of the corresponding entries larger. In this way we can find more correct edges. However, if a node has been estimated as a hub node by HGL with enough nonzeros in the corresponding column of $\hat{\mathbf{\Theta}}_{\text{HGL}}$ but we still give "loose conditions" to it, the corresponding column of $\hat{\mathbf{\Theta}}_{\text{DHGL}}$ may be overamplified, making the performance of DHGL even worse than HGL. So we exclude $\hat{\mathcal{H}}_{\text{HGL}}$ from $\mathcal{K}$.

Finally, we combine the hub nodes found by HGL and DHGL together to prevent losing useful information about hubs. While DHGL finds more correct edges of the nodes in $\mathcal{D}$, a few correct edges found earlier by HGL may disappear, making some hub nodes found by HGL not estimated as hub nodes by DHGL any more.

2.3.2. DHGL Applied without Known Hubs

In the previous section, we discuss the application of DHGL when some hubs are known. However, we often do not have any prior information about hubs. In this section, we give Algorithm 2 using the Graphical Lasso (GL) to provide prior information and DHGL to estimate $\mathbf{\Theta}$. Notice that $|\cdot|$ is the cardinality of the set.

Algorithm 2 DHGL Applied without Known Hubs

1. Use HGL to get the estimated hubs $\hat{\mathcal{H}}_{\text{HGL}}$.
2. (Prior Information Screening) Adjust regularization parameter λ of GL from large to small until $|\hat{\mathcal{H}}_{\text{GL},\lambda} \setminus \hat{\mathcal{H}}_{\text{HGL}}| > 0$ and $|\hat{\mathcal{H}}_{\text{GL},\lambda} \bigcup \hat{\mathcal{H}}_{\text{HGL}}| \leq \max\{|\hat{\mathcal{H}}_{\text{HGL}}| + a, b|\hat{\mathcal{H}}_{\text{HGL}}|\}$ where $a \in \boldsymbol{N}_+$,$b > 1$ but $b \approx 1$ and $\hat{\mathcal{H}}_{\text{GL},\lambda}$ is the set of estimated hubs by GL with the parameter λ.
3. Set $\mathcal{D} = \hat{\mathcal{H}}_{\text{GL},\lambda} \setminus \hat{\mathcal{H}}_{\text{HGL}}$ which is non-empty.
4. Use DHGL to estimate $\mathbf{\Theta}$, where λ_1, λ_2, λ_3, λ_4 remain the same values as those in HGL and λ_5 is selected using the BIC-type quantity.

The BIC-type quantity can help us select the tuning parameters to get a relatively accurate estimate of $\mathbf{\Theta}$. However, $|\hat{\mathcal{H}}_{\text{HGL}}|$ is always

not more than the number of true hub nodes $|\mathcal{H}|$ due to the penalty on the number of estimated hub nodes in the BIC, especially when $|\mathcal{H}|$ is relatively large. Therefore, we are encouraged to use some tools to find extra prior information of the hubs not found by HGL.

Since GL is fast (less than $\mathcal{O}(p^3)$ for sparse $\mathbf{\Theta}$ [3]), we can adjust the regularization parameter of GL from large to small quickly to get nodes that belong to $\hat{\mathcal{H}}_{\text{GL},\lambda}$ but not belong to $\hat{\mathcal{H}}_{\text{HGL}}$. These extra nodes are considered likely to be hubs if $|\hat{\mathcal{H}}_{\text{HGL}}| < |\mathcal{H}|$, although there is no guarantee for this. We set the extra nodes as discriminated nodes, i.e., nodes in $\mathcal{D}$.

Since the prior information provided by GL may not be correct, we keep $\lambda_4 = \lambda_2$ to be conservative and only select λ_5 using the BIC-type quantity in DHGL. If the extra nodes are true hubs, λ_5 tends to be selected small to make BIC-type quantity small so that more true edges and hub nodes will be found. If the extra nodes are not true hubs, λ_5 tends to be selected the same as λ_3 and the estimated $\hat{\mathbf{\Theta}}$ tends to be the same as that in HGL. As we can see, the algorithm tends to be open to correct prior information to make improvements, and immune to incorrect information to keep stable.

3. RESULT AND ANALYSIS

3.1. Measurement of Performance

In the simulation studies, the precision matrix $\mathbf{\Theta}$ and the set of indices of hub nodes $\mathcal{H}$ are given, and we estimate $\mathbf{\Theta}$ using our approaches displayed above. For $j \neq j'$, we know that if $|\Theta_{jj'}| \neq 0$, a true edge between j and j' exists. If further $j \in \mathcal{H}$, this is also a hub edge. Then we set a tolerance value t (in the simulations, we use $t = 0.005$). Then if $|\hat{\mathbf{\Theta}}_{jj'}| > t$, an edge between j and j' is estimated. The estimated hubs $\hat{\mathcal{H}}_r$ are defined by the set of nodes that have at least r edges. Then we can define the number of correctly estimated edges, the proportion of correctly estimated hub edges, the proportion of correctly estimated hub nodes, the sum of squared errors between $\mathbf{\Theta}$ and $\hat{\mathbf{\Theta}}$ similar to those defined in Tan et al. (2014) [5]. We have also defined the effective accuracy rate of hub nodes by the proportion of all p nodes that are correctly estimated as hubs or correctly estimated as non-hub nodes (excluding the known hubs).

3.2. DHGL Applied with Known Hubs

In this experiment, firstly, we use the simulation set-up I in Tan et al. (2014) with $p = 150$, $n = 50$, $r = 30$, $|\mathcal{H}| = 5$ and suppose we know two true hubs randomly [5]. We fix $\lambda_1 = 0.4$ and consider two cases where $\lambda_3 = 1$ and $\lambda_3 = 1.5$. In each case, λ_2 ranges from 0.1 to 0.7. In DHGL, for simplicity, we do not use the BIC-type quantity to select λ_4 and λ_5, but set λ_4 ralatively smaller than λ_2 and $\lambda_5 = 0.1$. For every set of tuning parameters, we conduct 50 simulations and average the results. Figure 1(a) shows the comparison of measures of performance between HGL and DHGL.

From the figure, we see that for every set of tuning parameters, the number of correctly estimated edges, the proportion of correctly estimated hub edges and the sum of squared errors perform better in DHGL than those in HGL. For the effective proportion of correctly estimated hub nodes and the effective accuracy rate of hub nodes, HGL performs better than DHGL, reasonably because the discriminated nodes (known hubs) are not considered in the calculation of the two measures. In order not to lose useful information, we combine the hub nodes found by HGL and DHGL so that we can always get better information of hub nodes using our method.

Besides, we consider three other cases for $(p, n) \in \{(200, 50), (200, 100), (300, 100)\}$, where $\lambda_2 = 0.4$, $\lambda_3 = 1$, $\lambda_4 = 0.2, r = p/5$ and other parameters are set to the same. We give the Figure 1(b) of performances of 50 simulations when $(p, n) = (300, 100)$. All of the three results are similar to the case when $(p, n) = (150, 50)$.

3.3. DHGL Applied without Known Hubs

In the experiment, we use the simulation set-up I in Tan et al. (2014) with $p = 150$, $n = 50$, $r = 30$ and $|\mathcal{H}| \in \{5, 10\}$ [5]. We fix $\lambda_1 = 0.4$, $\lambda_3 = 1$ and select λ_2 from $[0.05, 0.15]$ using the BIC-type quantity. For prior information screening, we set $a = 2$ and $b = 1.1$. In DHGL, we select λ_5 from $[0.5, 1]$ using the BIC-type quantity. Figure 1(c) and 1(d) display the comparison of 50 simulations between HGL and DHGL when $|\mathcal{H}| = 10$ and 5, respectively.

From Figure 1(c), we see that when $|\mathcal{H}| = 10$ which is relatively large, all of the five measures in DHGL outperform those in HGL. Notice that there are 4 out of 50 simulations where the accuracy rates of estimated hub nodes in HGL equal to 1, which means the nodes in $\mathcal{D}$ are not true hubs, but all the performances in DHGL do not degenerate.

From Figure 1(d), we see that when $|\mathcal{H}| = 5$ which is relatively small, there are 31 out of 50 simulations where the accuracy rates of estimated hub nodes in HGL equal to 1. Among them, only 5 simulations have accuracy rates of estimated hub nodes degenerated in DHGL with other four measures of performances degenerated slightly. However, for the rest 19 simulations where the accuracy rates of estimated hub nodes in HGL are less than 1, which suggests $|\hat{\mathcal{H}}_{\text{HGL}}| < |\mathcal{H}|$, 14 of them have five measures of performances in DHGL outperforming those in HGL obviously.

These two results display both the advantage and robustness of DHGL applied without known hubs. When $|\hat{\mathcal{H}}_{\text{HGL}}| < |\mathcal{H}|$, DHGL always performs better than HGL. On the other hand, when $|\hat{\mathcal{H}}_{\text{HGL}}| = |\mathcal{H}|$ which is not always the case, the performance of DHGL rarely degenerates and always keeps the same as HGL.

4. DISCUSSION

Based on Hub Graphical Lasso in Tan et al. (2014), we propose Discriminated Hub Graphical Lasso to estimate the precision matrix when some prior information of hub nodes is known. For the cases where some hub nodes are known or no hub node is known, we construct corresponding algorithms to make improvements on the measures of performance. The improvements essentially result from the increase in the number of tuning parameters when utilizing the prior information. In this case, the BIC-type quantity tends to be smaller than that in HGL when the optimal tuning parameters are selected. The computational complexity for DHGL is the same as that in HGL, but the algorithms generally implement both HGL and DHGL once, which is more time-consuming but the running time remains in the same scale.

In both algorithms, only the prior information of hubs not found by HGL are set in $\mathcal{D}$, so sometimes we even cannot make use of the prior information, especially when $\mathcal{D} = \varnothing$. Thus, it remains an open question how to use DHGL only to make improvements and make use of all prior information of hubs.

As for Algorithm 2, we select relatively small a and b when screening prior information using Graphical Lasso, which is conservative and may limit the performance of the algorithm. In the future, we will try to figure out how to select the optimal a and b. In other words, the difference between the number of estimated hub nodes in HGL and the number of true hub nodes will be studied in depth. Moreover, other approaches to detect prior information of hubs apart from Graphical Lasso are still to be studied.

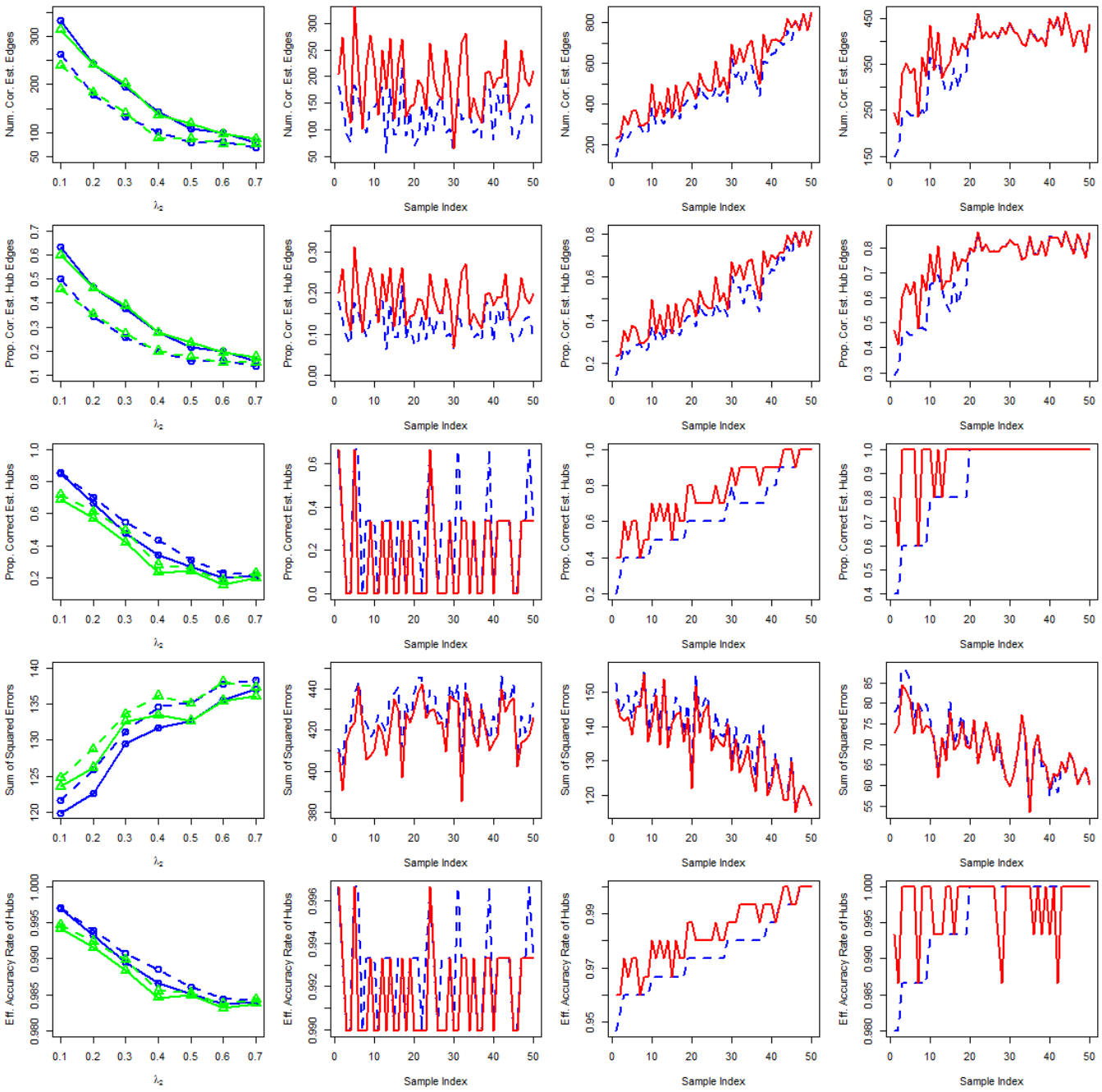

Fig. 1. Measures of performances including number of correctly estimated edge (row 1), proportion of correctly estimated hub edges (row 2), proportion of correctly estimated hub nodes (row 3), sum of squared errors (row 4), effective accuracy rate of hubs (row 5) of HGL and DHGL. Column (a) and (b) show the cases when some hubs are known, while column (c) and (d) show the cases when no hub is known. For the first two columns, row 3 and 5 are calculated not considering known hubs. For column (a), the dark blue lines with round points and light green lines with triangular points correspond to the cases when $\lambda_3 = 1$ and $\lambda_3 = 1.5$, respectively, while the solid lines and dashed lines correspond to DHGL and HGL, respectively. Notice that the horizontal axes suggest changes in λ_2, and 50 simulations are conducted for each set of parameters with average calculated. The rest three columns show the measures of performances in each of the 50 simulations, where the 50 simulations on the horizontal axes are sorted based on the accuracy rates of hub nodes. Here the red solid lines and blue dashed lines correspond to DHGL and HGL, respectively. Column (b) shows the cases when $p = 300$, $n = 100$ and $(\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5) = (0.4, 0.4, 1, 0.2, 0.1)$ with known hubs. Column (c) and (d) show the cases when $p = 150$, $n = 50$ without known hubs, with $|\mathcal{H}| = 10$ in (c) and $|\mathcal{H}| = 5$ in (d).

5. REFERENCES

[1] M. Drton and M.H. Maathuis, "Structure learning in graphical modeling," *Annual Review of Statistics and Its Application*, vol. 4, pp. 365–393, 2017.

[2] M. Yuan and Y. Lin, "Model selection and estimation in the gaussian graphical model," *Biometrika*, vol. 94, no. 1, pp. 19–35, 2007.

[3] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[4] O. Banerjee, L.E. Ghaoui, and A. dAspremont, "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data," *Journal of Machine Learning Research*, vol. 9, pp. 485–516, 2008.

[5] K.M. Tan, P. London, K. Mohan, S. Lee, M. Fazel, and D. Witten, "Learning graphical models with hubs," *Journal of Machine Learning Research*, vol. 15, pp. 3297–3331, 2014.

[6] K. Hirose, H. Fujisawa, and J. Sese, "Robust sparse gaussian graphical modeling," *Journal of Multivariate Analysis*, vol. 161, pp. 172–190, 2017.

[7] D. Hao, C. Ren, and C. Li, "Revisiting the variation of clustering coefficient of biological networks suggests new modular structure," *BMC Systems Biology*, vol. 6, no. 34, pp. 1–10, 2012.

[8] Q. Liu and A. Ihler, "Learning scale-free networks by reweighted l1 regularization," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. PMLR, 2011, vol. 15, pp. 40–48.

[9] A. Defazio and T.S. Caetano, "A convex formulation for learning scale-free networks via submodular relaxation," in *Advances in Neural Information Processing Systems 25*, pp. 1250–1258. 2012.

[10] Q. Tang, S. Sun, and J. Xu, "Learning scale-free networks by dynamic node-specific degree prior," in *Proceedings of the 32nd International Conference on Machine Learning*. PMLR, 2015, vol. 37, pp. 2247–2255.

[11] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society, Series B*, vol. 68, pp. 49–67, 2006.

[12] N. Simon, J.H. Friedman, T. Hastie, and R. Tibshirani, "A sparse-group lasso," *Journal of Computational and Graphical Statistics*, vol. 22, no. 2, pp. 231–245, 2013.

[13] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the admm," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[14] J. Eckstein and D. Bertsekas, "On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, pp. 293–318, 1992.

[15] J. Eckstein, "Augmented lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results," *RUTCOR Research Reports*, vol. 32, pp. 1–34, 2012.