Learning Coefficient Heterogeneity over Networks: A Distributed Tree-Based Fused-Lasso Approach

Xin Zhang
Department of Statistics
Iowa State University
Ames, IA, 50011
xinzhang@iastate.edu

Jia Liu
Department of Computer Science
Iowa State University
Ames, IA, 50011
jialiu@iastate.edu

Zhengyuan Zhu
Department of Statistics
Iowa State University
Ames, IA, 50011
zhuz@iastate.edu

Elizabeth S. Bentley
Air Force Research Laboratory
Information Directorate
Rome, NY, 13441
elizabeth.bentley.3@us.af.mil

Abstract

We propose an adaptive fused-lasso based coefficient subgroup approach for decentralized network systems. The major goal is to improve the model estimation efficiency by aggregating the neighbors' information as well as identifying the cluster membership for each node in the network. In particular, a tree-based ℓ_1 penalty is proposed to reduce the computation and communication costs. We show that our proposed estimator guarantees model selection consistency and asymptotic normality. Also, we design a decentralized generalized alternating direction method of multiplier algorithm for solving the objective function in parallel and establish its linear convergence speed. We conduct thorough numerical experiments to verify our theoretical results, which show that our approach outperforms existing works in terms of estimation accuracy, computation speed and communication costs.

1 Introduction

In this paper, we consider a fundamental *distributed* linear model clustering problem over networks (also sometimes referred to as subgroup analysis in the statistics literature): Suppose there are K nodes in the network, each of which holds a dataset that is denoted as $D_k = \{(\mathbf{x}_{k,i}, y_{k,i})\}_{i=1}^{n}$, where $\mathbf{x}_{k,i} \in \mathbb{R}^d$ and $y_{k,i} \in \mathbb{R}$ $(k = 1, \dots, K)$ represent the i-th covariate vector and response in the k-th dataset, respectively; and n denotes the size of the dataset. For ease of exposition, the size of each dataset is assumed to be balanced (i.e., all nodes have n samples)[1]. Hence, the total sample size in the network is $N = Kn$. We assume that there exist S underlying clusters of the nodes, and the data pair $(y, \mathbf{x})$ from the s-th cluster follows a common linear model:

$$y = \mathbf{w}_s^\top \mathbf{x} + \varepsilon, \tag{1}$$

where $\mathbf{w}_s = [\mathbf{w}_{s,1}, \cdots, \mathbf{w}_{s,d}]^\top$ is a d-dimensional coefficient vector for the s-th cluster, the independent error ε has a zero mean and a known variance σ^2. The linear model in (1) varies across the underlying clusters, i.e., the datasets in the same cluster s share the same coefficient $\mathbf{w}_s$ and vice versa. Our goal is to identify the cluster membership of each node and their corresponding coefficient. However, due to communication limitation or privacy restrictions, one cannot merge these datasets to

[1] Our algorithms and results in this paper can easily be extended to cases with datasets of unbalanced sizes.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

a single location. Thus, the main challenge of this problem is to perform clustering and estimate the coefficients of each cluster in the network in a *distributed* fashion.

The above problem naturally arises in many machine learning applications. For example, a wireless sensor network is deployed in a large spatial domain to collect and learn the relationship between the soil temperature y and air temperature $\mathbf{x}$ [14]. The domain can be divided into several subregions due to the landcover types, such as forest and grassland, and temperature relationships may vary geographically: sensors in the same subregion may share the same regression relationship, and the coefficients vary across different subregions. Similar scenarios could also emerge in other applications, such as meta-analysis on medical data [22], federated learning on the speech analysis [13], to name just a few.

Unfortunately, distributively clustering nodes based on regression model over networks is challenging as it includes two non-trivial *inter-dependent* and *conflicting* subtasks: i) statistical estimator design and ii) distributed optimization under the proposed estimator. In the literature, there exists tree-based centralized estimator designs that achieve strong statistical performance guarantee with $\Theta(K)$ computational complexity[2] (e.g., [22, 15], see Section 2 for detailed discussions). However, the tree-based penalty architectures make it difficult to design distributed optimization algorithms. On the other hand, there exist efficient distributed algorithms for solving related clustering problems over networks (e.g., [11, 24, 9], see Section 2 for details). However, it is unclear whether they could provide statistical performance guarantees, such as the selection consistency and estimation normality. Moreover, they all suffer $O(K^2)$ computational and communication costs. In light of the limitations of these existing work, in this paper, we ask the following fundamental question: *Could we develop a new distributed approach to achieve both strong statistical performance guarantees and* $\Theta(K)$ *computation and communication costs?* In other words, could we achieve the best of both worlds of the existing methods in the literature?

In this paper, we show that the answer to the above question is *affirmative*. The main contribution of this paper is that, for the first time, we develop a new minimum spanning tree (MST) based fused-lasso approach for solving the network clustering problem. Our approach enjoys oracle statistical performance and enables low-complexity distributed optimization algorithm design with *linear* convergence rate. The main results of this paper are summarized as follows:

- *Low-Complexity Estimator Design:* We propose a new MST-based penalty function for the clustering problem with $\Theta(K)$ complexity. Specifically, by comparing the coefficient similarities between the nodes, we construct a minimum spanning tree from the original network graph and only the edges in the tree are considered in the penalty function. Under this approach, the terms in the penalty function is reduced to $K-1$ (hence $\Theta(K)$ as opposed to $O(K^2)$).
- *Statistical Performance Guarantee:* Based on the MST structure, we propose the use of adaptive lasso to penalize the linear model coefficient differences. We show that our proposed estimator enjoys elegant oracle properties (cf. [7]), which means that our method can identify the nodes' cluster memberships almost surely (i.e., with probability one) as the size of datasets n increases and the estimators achieve asymptotic normality.
- *Distributed Optimization Algorithm Design:* Due to the restrictions imposed by the tree-based estimator design, traditional gradient- or ADMM-type (alternating direction method of multipliers) distributed methods cannot be applied to solve the objective function and find the nodes' cluster memberships distributively. In this paper, we develop a novel decentralized generalized ADMM algorithm for solving the tree-based fused-lasso problem. Moreover, we show that our algorithm has a simple *node-based* structure that is easy to implement and also enjoys the *linear* convergence.

Collectively, our results in this paper contribute to the theories of low-complexity model inference/clustering over networks and distributed optimization. Due to space limitation, we relegate most of the proof details to supplementary material.

2 Related work

In the literature, many approaches have been developed to cluster the heterogeneous data, such as the mixture model methods [10, 20, 3], the spectral clustering methods[19], etc. However, most of

[2] A function $f(x)$ is said to have growth rate $\Theta(g(x))$ if there exist two constants $0 < C_1 \leq C_2 < \infty$ such that $C_1 g(x) \leq f(x) \leq C_2 g(x)$.

the literature focuses on clustering the obeservation y, rather than the relationship between y and covariate $\mathbf{x}$. The authors of [16, 17] are the first few to investigate the network clustering problem under the subgroup analysis framework. Specifically, they considered the pairwise fusion penalty term for clustering the intercepts and the regression coefficients, respectively. In [22], the authors proposed a fused-lasso method termed FLARCC (fused lasso approach in regression coefficients clustering) to identify heterogeneity patterns of coefficients and to merge the homogeneous parameter clusters across multiple datasets in regression analysis with $\Theta(K)$ computational complexity. However, FLARCC does not exploit any spatial network structure to further improve the performance. The authors of [15] proposed a spatially clustered coefficient (SCC) regression method, which is based on a MST of the network graph to capture the spatial relationships among the nodes. By contrast, in our work, we adopt the penalty function based framework to recover clusters identities by adding a penalty term $P_\lambda(\mathbf{w}_1, \cdots, \mathbf{w}_K)$ to the ordinary least square problem for (1). Unlike all the above methods that are implemented on single centralized machine, a key distinguishing feature of our work is that we need to conduct clustering in a *distributed* fashion. As will be shown later, we improve the tree-based fusion penalty approach proposed in [15] to enhance the estimation efficiency as well as significantly reduce the computation and communication load for distributed algorithm design.

Our work also contributes to the theory of distributed optimization over networks, which has attracted a flurry of recent research (see, e.g., [18, 25, 21, 6]). In the general framework of distributed optimization, all K nodes in a connected network distributively and collaboratively solve an optimization problem in the form of: $\min_{\mathbf{w}} f(\mathbf{w}) \triangleq \sum_{i=1}^{K} f_i(\mathbf{w})$, where each f_i is the objective function observable only to the i-th node and $\mathbf{w}$ is a global decision variable across all nodes. By introducing a local copy $\mathbf{w}_i$, the above distributed optimization problem can be reformulated in the following penalized version of the so-called *consensus* form: $\min_{\mathbf{w}_i, i=1,\cdots,K} \sum_{i=1}^{K} f_i(\mathbf{w}_i) + \frac{\lambda}{2} \sum_{i,j=1, i\neq j}^{K} \pi_{i,j} \|\mathbf{w}_i - \mathbf{w}_j\|_2^2$, where $\pi_{i,j}$ is a weight parameter for penalizing the disagreement between the i-th and j-th nodes. Interestingly, in this work, opposite to traditional distributed algorithms that focus on the consensus problems stated above, we consider whether there exists *disagreements* among the true $\{\mathbf{w}_i\}_{i=1}^{K}$: The nodes are to be classified to several clusters and the nodes in each cluster share the same $\mathbf{w}$. We note that the authors of [11] and [24] also focused on discovering the clustering patterns among the nodes with decentralized algorithms. However, they adopted a *pairwise* penalty function to obtain consensus of the inner-cluster weights, which can be reformulated as the well-known Laplacian penalty [1]. A main limitation of the Laplacian penalty is that it cannot shrink the pairwise differences of the parameter estimates to zero (which is also verified in our simulations).

The most related work to ours is [9], where the network lasso method was introduced. In the network lasso method in [9], the authors adopted an ℓ_2 penalty for each edge in the network graph. They also proposed a distributed alternating direction method of multipliers (ADMM) to solve the network lasso problem. Our work differs from [9] in the following key aspects: 1) The number of the penalty terms in [9] depends on the number of edges in the network graph, which yields an $O(K^2)$ computation complexity and is *unscalable* for the large-sized networks. In this paper, we consider a *tree-based* penalty function, which contains exactly $K-1$ penalty terms; 2) The penalty function in the network lasso method [9] adopted the ℓ_2 norm for the $\mathbf{w}_i - \mathbf{w}_j$ difference, while we consider an *adaptive* ℓ_1 norm for the vector difference, which enjoys elegant *oracle* properties (i.e., the selection consistency and the asymptotic normality); 3) The algorithm in [9] is based on the classical ADMM algorithm with two constraints on each edge, while we propose a new generalized ADMM method with only *one* constraint on each edge, which significantly reduces the algorithm's implementation complexity; 4) We rigorously prove the statistical consistency and algorithmic convergence of our proposed approach, both of which were not studied in [9].

3 Model and problem statement

Given a network $G = (V, E)$, where V and E represent the node and edge sets, respectively, our goal is to estimate the coefficients $\{\mathbf{w}_i\}_{i=1}^{K}$ and determine the cluster membership for each node. This problem can be formulated as minimizing the following loss function:

$$L_{\text{Graph}}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{K} \|\mathbf{y}_i - \mathbf{X}_i \mathbf{w}_i\|^2 + \sum_{(v_i, v_j) \in E} P_{\lambda_N}(\mathbf{w}_i - \mathbf{w}_j), \tag{2}$$

where $v_i \in V$ denotes the i-th node in the network; $\mathbf{y}_i = [y_{i,1}, \cdots, y_{i,n}]^\top \in \mathbb{R}^n$ and $\mathbf{X}_i = [\mathbf{x}_{i,1}, \cdots, \mathbf{x}_{i,n}]^\top \in \mathbb{R}^{n \times d}$ represent the reponses and design matrix at the i-th node, respectively; and P_{λ_N} is a penalty function with tuning parameter λ_N. Note that the objective function in (2) consists of two parts: the first part is an ordinary least square (OLS) problem for all the coefficients $\mathbf{w} \triangleq [\mathbf{w}_1^\top, \cdots, \mathbf{w}_K^\top]^\top \in \mathbb{R}^{Kn}$; the second term is a penalty term designed to shrink the difference of any two coefficient vectors if the corresponding nodes are connected. Note that the second term in (2) depends on the network topology. Thus, we make the following assumption that is necessary to guarantee that the problem is well-defined in terms of estimation accuracy:

Assumption 1 *Given a connected network $G = (V, E)$, for any node v_i from a cluster with more than two members, there exists another node v_j from the same cluster such that the edge $(v_i, v_j) \in E$.*

Under Assumption 1, each node is connected with its members if the cluster size is larger than one. Hence, by removing inter-cluster edges, i.e., identifying edges with *non-zero* coefficient difference, the original network graph can be reduced into S subgraphs, which are the subgroup clusters. For the objective function in (2), several important remarks are in order:

Remark 1 First, the penalty terms in the objective function (2) consist of all *pairwise* coefficient differences among all edges in the network graph. If the penalty function is chosen as $P_{\lambda_N}(\mathbf{w}_i, \mathbf{w}_j) = \lambda_N \|\mathbf{w}_i - \mathbf{w}_j\|_2$, then Eq. (2) has the same form as in the network lasso method [9]. Second, the objective function (2) can also be viewed as a variant of the method proposed in [16], where the penalty terms are all pairwise differences of the nodes, and hence the total number of the penalty terms is exactly $(K-1)K/2$. Thanks to Assumption 1, we only need to consider the difference of end nodes of edges. Thus, the number of penalty terms can be reduced to exactly $|E|$. Third, the value of $|E|$ still implies that the number of penalty terms in (2) could scale as $O(K^2)$ if the network is dense, which will in turn result in heavy computation and communication loads as the network size increases. To address the problem, we will propose a simplified tree-based penalty function in Section 4.

4 Problem reformulation: a tree-based approach

As mentioned earlier and has been long noted in statistics (see, e.g., [22, 15]) and optimization (see, e.g., [4]) communities, directly including all edges in penalty terms will incur high computational and communication complexity. To reduce the redundant penalty terms, several strategies have been proposed, including the order method in [12, 22] and the MST approach in [15]. Specifically, in [12, 22], the authors first determined the OLS estimation of the coefficients and then ordered the coefficients. They then presumed that similar coefficients will be neighbors with high probability and only considered regularization terms associated with the adjacent coefficients. By contrast, in [15], the authors used the *spatial* distance to constructed an MST, and preserved the penalty terms in the tree. In essence, these two strategies are tree-based approaches, with the only difference being the definitions of distance measure for the tree: the first one uses model similarity, while the second one uses spatial distances. In this paper, we propose a new tree-based approach, where the distance measure for the tree can be viewed as integrating the above two measures in some sense. Yet, we will show that this new distance measure achieves surprising performance gains.

Specifically, we construct an MST as follows: First, *local* OLS estimators are determined in each node individually: $\hat{\mathbf{w}}_{i,OLS} = [\mathbf{X}_i^\top \mathbf{X}_i]^{-1}[\mathbf{X}_i^\top \mathbf{y}_i]$. Then, the weight for two nodes is defined based on their local model similarity and their connection relationship in the graph as follows:

$$\tilde{s}_{i,j} = \begin{cases} \|\hat{\mathbf{w}}_{i,OLS} - \hat{\mathbf{w}}_{j,OLS}\|, & \text{if } (v_i, v_j) \in E, \\ \infty, & \text{otherwise.} \end{cases} \quad (3)$$

The weight $\tilde{s}$ in (3) contains two important pieces of information: one is the network topology, which is characterized by spatial distances (e.g., in a sensor network, the nodes can only be connected within a certain communication range); the other is the local model similarity, which implies the likelihood of two nodes being in the same cluster. Based on (3), an MST can be constructed so that only penalty terms associated with the MST are considered in the objective function:

$$L_{\text{MST}_s}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{K} \|\mathbf{y}_i - \mathbf{X}_i \mathbf{w}_i\|^2 + \sum_{(v_i, v_j) \in \text{MST}_s} P_\lambda(\mathbf{w}_i - \mathbf{w}_j), \quad (4)$$

where the notation MST$_s$ signifies that the MST is based on the model **s**imilarity. Note that the estimation efficiency and clustering accuracy significantly depend on the penalty function. The following lemma guarantees that the nodes in the same cluster are connected in the MST$_s$ based on the weight defined in (3) (see Section 2.1 in supplementary material for proof details).

Lemma 1 *Under Assumption 1, given an MST$_s$ based on the weights defined in (3), as the local sample size $n \to \infty$, then with probability 1, for any node v_i in a cluster s with more than two members, there exists a node v_j from the same cluster such that the edge (v_i, v_j) is in the MST$_s$.*

With Lemma 1, the number of inter-cluster edges is $S - 1$. Thus, the MST$_s$ is a connected graph with the *smallest* possible number of inter-cluster edges. Also, the MST$_s$ can be separated into S clusters by identifying these inter-cluster edges. We note that there exist distributed methods to find the MST$_s$ (e.g., the Gallager-Humblet-Spira algorithm [8]) and their implementation details are beyond the scope of this paper.

5 Statistical model: an adaptive fused-lasso based approach

For convenience, we use $[\mathbf{v}]_p$ to denote the p-th element of vector $\mathbf{v}$. Based on the MST constructed in Section 4, we specialize the loss function in (4) by adopting the following adaptive lasso penalty:

$$L_{\text{MST}_s}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{y}_i - \mathbf{X}_i\mathbf{w}_i\|^2 + \frac{\lambda_N}{2}\sum_{i=1}^{K}\sum_{j\in\mathcal{N}_i}\sum_{p=1}^{d}[\hat{\boldsymbol{\pi}}_{i,j}]_p\big|[\mathbf{w}_i]_p - [\mathbf{w}_j]_p\big|, \tag{5}$$

where $\mathcal{N}_i$ represents the set of the neighboring nodes of node i in the MST$_s$, $\hat{\boldsymbol{\pi}}_{i,j} \in \mathbb{R}^d$ is an adaptive weight vector defined as $[\hat{\boldsymbol{\pi}}_{i,j}]_p = 1/\big|[\hat{\mathbf{w}}_{i,OLS}]_p - [\hat{\mathbf{w}}_{j,OLS}]_p\big|^\gamma$ for some constant $\gamma > 0$. Therefore, our proposed estimator is $\hat{\mathbf{w}}_{\text{MST}_s} = \arg\min_{\mathbf{w}} L_{\text{MST}_s}(\mathbf{w})$.

Remark 2 Here, our use of an adaptive lasso penalty is motivated by: 1) Adaptive lasso is known to be an oracle procedure for related variable selection problems in statistics [7]; 2) With an adaptive lasso penalty, the objective function in (5) is strongly convex as long as the design matrix $\mathbf{X}$ is of full row rank. This implies that the minimum of (5) is unique. In [16, 17], similar clustering methods were proposed based on the minimax concave penalty (MCP) and the smoothly clipped absolute deviations (SCAD) penalty, both of which are concave penalties and have been shown to be statistically efficient. However, from an optimization perspective, concave penalties will render the objective function non-convex, which in turn leads to intractable algorithm design. In [15], lasso penalty was also adopted, but there is no proof for the oracle properties of their estimator.

For more compact notation in the subsequent analysis, we rewrite the objective function (5) in the following matrix form:

$$L_{\text{MST}_s}(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda_N \sum_{p=1}^{d(K-1)}[\hat{\boldsymbol{\pi}}]_p \cdot |[(\mathbf{H}\otimes\mathbf{I}_K)\mathbf{w}]_p| \tag{6}$$

where $\mathbf{y} = [\mathbf{y}_1^\top, \cdots, \mathbf{y}_K^\top]^\top$, $\mathbf{X} = \text{diag}(\mathbf{X}_1, \cdots, \mathbf{X}_K)^\top$ and $\mathbf{w} = [\mathbf{w}_1^\top, \cdots, \mathbf{w}_K^\top]^\top$ are the response vector, the design matrix, and coefficient vector, respectively; and $\otimes$ denotes the Kronecker product. In (6), $\mathbf{H}$ is the incident matrix of the MST$_s$, which is full row rank and each entry in $\mathbf{H}$ defined as:

$$[\mathbf{H}]_{l,i} = \begin{cases} 1, & \text{if } i = s(l), \\ -1, & \text{if } i = e(l), \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

where $s(l)$ and $e(l)$ denote the starting and ending node indices of edge l in the MST$_s$, respectively, with $s(l) < e(l)$. In (6), $[\hat{\boldsymbol{\pi}}]_p \triangleq 1/[\underline{\mathbf{H}} \cdot \mathbf{w}_{OLS}]_p^\gamma$, where $\underline{\mathbf{H}} \triangleq \mathbf{H}\otimes\mathbf{I}_K$ and $\mathbf{w}_{OLS} = [\mathbf{w}_{1,OLS}^\top, \cdots, \mathbf{w}_{K,OLS}^\top]^\top$ is the vector form of the OLS estimations. Note that adding one more row to $\mathbf{H}$, we can form a square and full rank matrix: $\tilde{\mathbf{H}} = \begin{bmatrix} \mathbf{H} \\ \frac{1}{\sqrt{K}}\mathbf{1}^\top \end{bmatrix}$ [15], and the objective function (6) can be equivalently rewritten as:

$$L_{\text{MST}_s}(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda_N \sum_{p=1}^{dK}[\hat{\boldsymbol{\pi}}]_p \cdot |[\tilde{\underline{\mathbf{H}}}\mathbf{w}]_p|, \tag{8}$$

where $\underline{\tilde{\mathbf{H}}} \triangleq \tilde{\mathbf{H}} \otimes \mathbf{I}_K$ is a full rank square matrix. Define $\boldsymbol{\Delta} = \underline{\tilde{\mathbf{H}}}\mathbf{w}$ as the difference of the connected nodes' weights. It then follows that the objective function in (8) can be rewritten in terms of Δ as:

$$L_{\mathrm{MST}_s}(\boldsymbol{\Delta}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\underline{\tilde{\mathbf{H}}}^{-1}\boldsymbol{\Delta}\|^2 + \lambda_N \sum_{p=1}^{dK}[\hat{\boldsymbol{\pi}}]_p \cdot |[\boldsymbol{\Delta}]_p|. \tag{9}$$

Our estimator then becomes: $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s} = \arg\min_{\boldsymbol{\Delta}} L_{\mathrm{MST}_s}(\boldsymbol{\Delta})$. Since there is a one-to-one transformation between $\hat{\mathbf{w}}_{\mathrm{MST}_s}$ and $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s}$ (i.e., $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s} = \underline{\tilde{\mathbf{H}}}\hat{\mathbf{w}}_{\mathrm{MST}_s}$), we can instead focus on the theoretical properties of $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s}$. We denote the true coefficients as $\mathbf{w}_* = [\mathbf{w}_{1,*}^\top, \cdots, \mathbf{w}_{K,*}^\top]^\top$, and $\boldsymbol{\Delta}_* = \underline{\tilde{\mathbf{H}}}\mathbf{w}_*$. Note that if the two connected nodes are from the same cluster, the corresponding elements in $\boldsymbol{\Delta}_*$ are zero. We denote the set of non-zero elements in $\boldsymbol{\Delta}_*$ as $\mathcal{A}_*$. Similarly, the set of non-zero elements in $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s}$ is denoted as $\hat{\mathcal{A}}_N$. To prove the oracle properties of $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s}$, we need the following assumptions for the linear model in (1):

Assumption 2 *For the linear model in (1): i) the errors are i.i.d. random variables with zero mean and variance σ^2; ii) $\frac{1}{N}(\mathbf{X}\underline{\tilde{\mathbf{H}}}^{-1})^\top \mathbf{X}\underline{\tilde{\mathbf{H}}}^{-1} \xrightarrow{p} \mathbf{C}$ for some positive definite matrix $\mathbf{C}$ as $N \to \infty$.*

We note that in the second condition in Assumption 2, since $\underline{\tilde{\mathbf{H}}}$ is a full rank square matrix, $\mathbf{C}$ is positive definite if $\mathbf{X}$ is full column rank. Now, we state the oracle properties of $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s}$ as follows:

Theorem 1 *Suppose that $\lambda_N/\sqrt{N} \to 0$ and $\lambda_N N^{(\gamma-1)/2} \to \infty$. Under Assumptions 1 and 2, the estimator $\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s}$ satisfies the following two oracle properties: i) (Selection Consistency) $\lim_{n\to\infty} \mathbb{P}(\hat{\mathcal{A}}_N = \mathcal{A}_*) = 1$; and ii) (Asymptotic Normality) $\sqrt{N}([\widehat{\boldsymbol{\Delta}}_{\mathrm{MST}_s}]_{\mathcal{A}_*} - [\boldsymbol{\Delta}_*]_{\mathcal{A}_*}) \xrightarrow{d} \mathcal{N}(0, \sigma^2\mathbf{C}_{\mathcal{A}_*}^{-1})$ as $N \to \infty$, where $\mathbf{C}_{\mathcal{A}_*}^{-1}$ is the submatrix of corresponding to set $\mathcal{A}_*$.*

The proof of Theorem 1 is relegated to supplementary material due to space limitation. Based on Theorem 1, the asymptotic normality for $\hat{\mathbf{w}}_{\mathrm{MST}_s}$ can be derived by a simple linear transformation.

6 Optimization algorithm: an ADMM based distributed approach

In this section, we will design a distributed algorithm for minimizing (5). Due to the penalty structure in (5), one natural idea is to use the popular ADMM method [2], which has been shown to be particularly suited for solving lasso related problems (e.g., [16, 17, 23, 26]). However, in what follows, we will first illustrate why it is challenging to use a regular ADMM approach to solve the MST_s-based fused-lasso clustering problem over networks in a distributed fashion. As a result, it is highly non-trivial to design a new ADMM-based algorithm by exploiting special problem structure in the MST_s regularizer. To this end, we first note that the penalty term in (5) can be written as:

$$\frac{1}{2}\sum_{i=1}^{K}\sum_{j\in\mathcal{N}_i}\sum_{p=1}^{d}[\hat{\boldsymbol{\pi}}_{i,j}]_p\big|[\mathbf{w}_i]_p - [\mathbf{w}_j]_p\big| = \sum_{e_l\in\mathrm{MST}_s}\sum_{p=1}^{d}[\hat{\boldsymbol{\pi}}_l]_p\big|[\mathbf{w}_{s(l)}]_p - [\mathbf{w}_{e(l)}]_p\big|, \tag{10}$$

where e_l represents the l-th edge in MST_s. In (10), $s(l)$ and $e(l)$ denote the starting and ending node indices of edge l, respectively, with $s(l) < e(l)$; and $\hat{\boldsymbol{\pi}}_l = \hat{\boldsymbol{\pi}}_{s(l),e(l)}$ is the corresponding adaptive weight vector for the l-th edge. With the same notation as in Section 5, the weight difference at edge l is $\boldsymbol{\Delta}_l = \mathbf{w}_{s(l)} - \mathbf{w}_{e(l)}$ and $\boldsymbol{\Delta} = [\Delta_1^\top, \cdots, \Delta_{K-1}^\top]^\top = \underline{\mathbf{H}}\mathbf{w}$. Note that there are $K-1$ edges in the MST_s. Thus, the problem of minimizing the loss function in (5) can be reformulated as:

$$\begin{aligned}&\text{Minimize } \ \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{y}_i - \mathbf{X}_i\mathbf{w}_i\|^2 + \lambda_N\sum_{l=1}^{K-1}\sum_{p=1}^{d}[\hat{\boldsymbol{\pi}}_l]_p\big|[\boldsymbol{\Delta}_l]_p\big| \\ &\text{subject to } \ \boldsymbol{\Delta} = \underline{\mathbf{H}}\mathbf{w}.\end{aligned} \tag{11}$$

Then, we can construct an augmented Lagrangian with penalty parameter $\tau > 0$ for (11) as follows:

$$L_\tau(\mathbf{w}, \boldsymbol{\Delta}, \mathbf{z}) = \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{y}_i - \mathbf{X}_i\mathbf{w}_i\|^2 + \lambda_N\sum_{l=1}^{K-1}\sum_{p=1}^{d}[\hat{\boldsymbol{\pi}}_l]_p\big|[\boldsymbol{\Delta}_l]_p\big| - \langle\mathbf{z}, \underline{\mathbf{H}}\mathbf{w} - \boldsymbol{\Delta}\rangle + \frac{\tau}{2}\|\underline{\mathbf{H}}\mathbf{w} - \boldsymbol{\Delta}\|^2, \tag{12}$$

where $\mathbf{z} \in \mathbb{R}^{d(K-1)}$ is the vector of dual variables corresponding to the $K-1$ edges. In what follows, we derive the updating rules for $(\mathbf{w}^{t+1}, \boldsymbol{\Delta}^{t+1}, \mathbf{z}^{t+1})$. First, given the primal and dual pair $\mathbf{w}^t, \mathbf{z}^t$, for the l-th edge with end nodes $s(l)$ and $e(l)$, to determine the weight difference $\boldsymbol{\Delta}^{t+1}$, we need to solve the subproblem $\boldsymbol{\Delta}^{t+1} = \arg\min_{\boldsymbol{\Delta}} L_\tau(\mathbf{w}^t, \boldsymbol{\Delta}, \mathbf{z}^t)$, and hence for the l-th edge with end nodes $s(l)$ and $e(l)$, it follows that (see Section 1 in supplementary material for derivation details):

$$\boldsymbol{\Delta}_l^{t+1} = S_{\lambda_N \hat{\boldsymbol{\pi}}_l/\tau}\Big(\mathbf{w}_{s(l)}^t - \mathbf{w}_{e(l)}^t - \frac{1}{\tau}\mathbf{z}_l^t\Big), \tag{13}$$

where $S_{\lambda_N \hat{\boldsymbol{\pi}}_l/\tau}$ is the coordinate-wise soft-thresholding operator with $[\lambda_N \hat{\boldsymbol{\pi}}_l/\tau]_p = \lambda_N [\hat{\boldsymbol{\pi}}_l]_p/\tau$. Next, we derive the updating rule for $\mathbf{w}^{t+1}$. Let $\mathbf{L} = \mathbf{H}^\top \mathbf{H}$ be the Laplacian matrix for the MST_s. With the classical ADMM, it can be shown that (see Section 1 in supplementary material for derivation details):

$$\begin{aligned}\mathbf{w}^{t+1} &= \arg\min_{\mathbf{w}} L_\tau(\mathbf{w}, \boldsymbol{\Delta}^{t+1}, \mathbf{z}^t) \\ &= [\mathbf{X}^\top \mathbf{X} + \tau \mathbf{L} \otimes \mathbf{I}_d]^{-1}[\mathbf{X}^\top \mathbf{y} + \underline{\mathbf{H}}^\top(\tau \boldsymbol{\Delta}^{t+1} + \mathbf{z}^t)].\end{aligned} \tag{14}$$

Unfortunately, the matrix inverse in (14) *cannot* be computed in a distributed fashion due to the coupled structure of the Laplacian matrix $\mathbf{L}$. Here, we show that the generalized ADMM studied in [5] can be leveraged to derive an updating rule for $\mathbf{w}^{t+1}$, which can be implemented in a parallel fashion. To this end, instead of directly solving the subproblem $\mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} L_\tau(\mathbf{w}, \boldsymbol{\Delta}^{t+1}, \mathbf{z}^t)$, we add a quadratic regularization term $\frac{1}{2}(\mathbf{w} - \mathbf{w}^t)^\top \mathbf{P}(\mathbf{w} - \mathbf{w}^t)$ in the subproblem as in [5] ($\mathbf{P}$ is positive semidefinite):

$$\begin{aligned}\mathbf{w}^{t+1} &= \arg\min_{\mathbf{w}} L_\tau(\mathbf{w}, \boldsymbol{\Delta}^{t+1}, \mathbf{z}^t) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^t)^\top \mathbf{P}(\mathbf{w} - \mathbf{w}^t) \\ &= [\mathbf{X}^\top \mathbf{X} + \tau \underline{\mathbf{H}}^\top \underline{\mathbf{H}} + \mathbf{P}]^{-1}[\mathbf{X}^\top \mathbf{y} + \underline{\mathbf{H}}^\top(\tau \boldsymbol{\Delta}^{t+1} + \mathbf{z}^t) + \mathbf{P}\mathbf{w}^t].\end{aligned}$$

Now, the *key step* is to recognize that we can choose the matrix $\mathbf{P} = -\tau \underline{\mathbf{H}}^\top \underline{\mathbf{H}} + \mathbf{D} = -\tau \mathbf{L} \otimes \mathbf{I}_d + \mathbf{D}$, where $\mathbf{D} = \text{diag}(D_1, \cdots, D_K) \otimes \mathbf{I}_d$ with positive scalars D_i for node i. It follows that $\mathbf{w}^{t+1} = [\mathbf{X}^\top \mathbf{X} + \mathbf{D}]^{-1}[\mathbf{X}^\top \mathbf{y} + \underline{\mathbf{H}}^\top(\tau \boldsymbol{\Delta}^{t+1} + \mathbf{z}^t) + \mathbf{P}\mathbf{w}^t]$. Plugging in $\mathbf{P} = -\tau \mathbf{L} \otimes \mathbf{I}_d + \mathbf{D}$, we have the following local weight update:

$$\mathbf{w}_i^{t+1} = [\mathbf{X}_i^\top \mathbf{X}_i + D_i \mathbf{I}_d]^{-1}\Big[\mathbf{X}_i^\top \mathbf{y}_i + \sum_{v_i \in e_l} [\mathbf{H}]_{li}(\tau \boldsymbol{\Delta}_l^{t+1} + \mathbf{z}_l^t) + (D_i - \tau \text{deg}(i) \mathbf{w}_i^t + \tau \sum_{j \in \mathcal{N}_i} \mathbf{w}_j^t\Big], \tag{15}$$

where $v_i \in e_l$ means node v_i is an end node of edge e_l, deg(i) is the degree of the node v_i (i.e., deg$(i) = |\mathcal{N}_i|$). Thus, the updating of $\mathbf{w}_i^{t+1}$ only requires the local and connected neighbor's information, which facilitates *distributed* implementation. Also, matrix $\mathbf{D}$ plays an important role on the algorithm convergence. Recall that $\mathbf{P} = \mathbf{D} - \tau L \otimes \mathbf{I}_d = [\text{diag}(D_1, \cdots, D_K) - \tau L] \otimes \mathbf{I}_d$. To guarantee $\mathbf{P} \succ 0$, based on the Gershgorin circle theorem, we can choose $\mathbf{D}$ as $D_i > 2\text{deg}(i)$. Lastly, the dual variables $\mathbf{z}^{t+1}$ can be updated as $\mathbf{z}^{t+1} = \mathbf{z}^t - \tau(\underline{\mathbf{H}}\mathbf{w}^{t+1} - \Delta^{t+1})$, and hence for the l-th edge, the corresponding dual update is (see Section 1 in supplementary material for details):

$$\mathbf{z}_l^{t+1} = \mathbf{z}_l^t - \tau\Big(\mathbf{w}_{s(l)}^{t+1} - \mathbf{w}_{e(l)}^{t+1} - \boldsymbol{\Delta}_l^{t+1}\Big). \tag{16}$$

Note, however, that the updating rules (13) and (16) are edge-based while (15) is node-based. To make the updating rules consistent, we define several additional notations: At node $s(l)$, we let $\boldsymbol{\Delta}_{s(l)}^t = \boldsymbol{\Delta}_l^t$ and $\mathbf{z}_{s(l)}^t = \mathbf{z}_l^t$; At node $e(l)$, we let $\boldsymbol{\Delta}_{e(l)}^t = -\boldsymbol{\Delta}_l^t$ and $\mathbf{z}_{e(l)}^t = -\mathbf{z}_l^t$. With simple derivations, it can be verified that if $\boldsymbol{\Delta}_{s(l)}^t = -\boldsymbol{\Delta}_{e(l)}^t = \boldsymbol{\Delta}_l^t$ and $\mathbf{z}_{s(l)}^t = -\mathbf{z}_{e(l)}^t = \mathbf{z}_l^t$ are satisfied in iteration t, then in iteration $t+1$, $\boldsymbol{\Delta}_{s(l)}^{t+1} = -\boldsymbol{\Delta}_{e(l)}^{t+1} = \boldsymbol{\Delta}_l^{t+1}$ and $\mathbf{z}_{s(l)}^{t+1} = -\mathbf{z}_{e(l)}^{t+1} = \mathbf{z}_l^{t+1}$ still hold based on the following node-based updating rules: $\forall i \in \{s(l), e(l)\}$ and $j = \{s(l), e(l)\}/\{i\}$,

$$\begin{cases}\boldsymbol{\Delta}_i^{t+1} = S_{\lambda_N \hat{\boldsymbol{\pi}}_l/\tau}\Big(\mathbf{w}_i^t - \mathbf{w}_j^t - \frac{1}{\tau}\mathbf{z}_i^t\Big), \\ \mathbf{w}_i^{t+1} = [\mathbf{X}_i^\top \mathbf{X}_i + D_i \mathbf{I}_d]^{-1}\Big[\mathbf{X}_i^\top \mathbf{y}_i + \sum_{v_i \in e_l}(\tau \boldsymbol{\Delta}_i^{t+1} + \mathbf{z}_i^t) + (D_i - \tau \text{deg}_i)\mathbf{w}_i^t + \tau \sum_{j \in \mathcal{N}_i} \mathbf{w}_j^t\Big], \\ \mathbf{z}_i^{t+1} = \mathbf{z}_i^t - \tau\Big(\mathbf{w}_i^{t+1} - \mathbf{w}_j^{t+1} - \Delta_i^{t+1}\Big).\end{cases} \tag{17}$$

Algorithm 1 Decentralized Generalized ADMM for Minimizing L_{MST_s} in (5).

```
Require: Data $\{\mathbf{X}_i, \mathbf{y}_i\}_{i=1}^K$, tuning parameter $\lambda_N$;
Ensure: $\mathbf{w}^T$ and $\mathbf{\Delta}^T$;
Each node finds the local OLS estimation and sets $\mathbf{w}_i^0 = \mathbf{w}_{i,OLS}$;
Each node sends $\mathbf{w}_i^0$ to its neighboring nodes in the network and calculates the weight (3);
The network constructs an MST$_s$ based on $\mathbf{w}_i^0$;
The nodes $s(l)$ and $e(l)$ of edge $l$ set $\mathbf{z}_{s(l)}^0 = \mathbf{z}_{e(l)}^0 = \mathbf{0}$ and $\mathbf{\Delta}_{s(l)}^0 = -\mathbf{\Delta}_{e(l)}^0 = \mathbf{w}_{s(l)}^0 - \mathbf{w}_{e(l)}^0$.
while not converged do
    Each node sends its current $\mathbf{w}_i^t$ to its neighboring nodes in the MST$_s$;
    Each node updates the primal and dual variables using the rules in (17).
end while
```

Thus, we can set $\mathbf{\Delta}_{s(l)}^0 = -\mathbf{\Delta}_{e(l)}^0 = \mathbf{w}_s(l)^0 - \mathbf{w}_e(l)^0$ and $\mathbf{z}_{s(l)} = \mathbf{z}_{e(l)} = \mathbf{0}$, $\forall l$, which satisfy the above conditions. Note that the updating rule for $\mathbf{w}_{e(l)}^{t+1}$ has the same structure as $\mathbf{w}_{s(l)}^{t+1}$ because $[\mathbf{H}]_{l,e(l)} = -1$ and $\mathbf{z}_{e(l)}^t = -\mathbf{z}_l^t$, $\mathbf{\Delta}_{e(l)}^t = -\mathbf{\Delta}_l^t$. Our method is summarized in Algorithm 1. The outputs of the algorithm are the estimated coefficient $\hat{\mathbf{w}}$ and the coefficient difference $\widehat{\Delta}$. Whether two nodes are in the same cluster can be determined by checking $\widehat{\mathbf{\Delta}} : \widehat{\mathbf{\Delta}}_{s(l)} = \widehat{\mathbf{\Delta}}_{e(l)} = \mathbf{0}$ if $s(l)$ and $e(l)$ are in the same cluster. The following theorem guarantees the convergence speed of Algorithm 1.

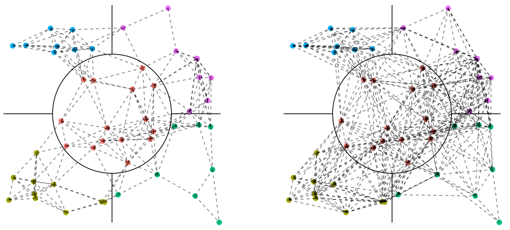
Theorem 2 *Denote the KKT (Karush-Kuhn-Tucker) point for the objective function (11) as* $\mathbf{u}_* = (\mathbf{w}_*^\top, \mathbf{\Delta}_*^\top, \mathbf{z}_*^\top)^\top$. *With a proper selection of* $\mathbf{D}$ *such that* $\mathbf{P} \succ 0(positive definite)$, *the iterates* $\{\mathbf{u}^t\}_{t=1}^\infty$ *converge to* $\mathbf{u}_*$ *in the sense of* $\mathbf{G}$*-norm:* $\|\mathbf{u}^t - \mathbf{u}_*\|_\mathbf{G} \to 0$, *where* $\|\cdot\|_\mathbf{G}$ *represents the semi-norm* $\|\mathbf{x}\|_\mathbf{G}^2 \triangleq \mathbf{x}^\top \mathbf{G} \mathbf{x}$, *where* $\mathbf{G}$ *is defined as:*

$$\mathbf{G} \triangleq \begin{bmatrix} \mathbf{D} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \frac{1}{\tau}\mathbf{I}_{d(K-1)} \end{bmatrix},$$

Further, the convergence rate is linear, i.e., $\exists\, \delta > 0$, *such that* $\|\mathbf{u}^{t+1} - \mathbf{u}_*\|_\mathbf{G}^2 \le (1+\delta)^{-1} \|\mathbf{u}^t - \mathbf{u}_*\|_\mathbf{G}^2$.

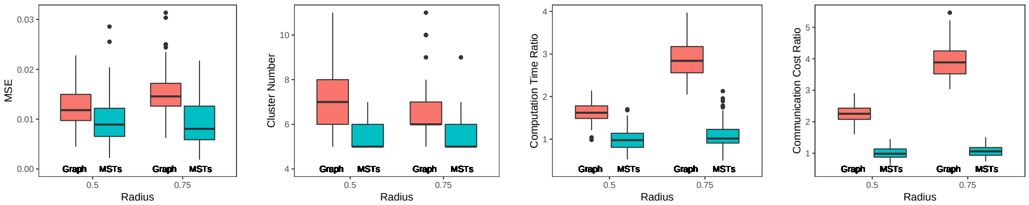
7 Numerical Results

Due to space limitation, we only provide the numerical results of the impacts of the choices of regularization on accuracy and cost. More detailed numerical studies can be found in the supplementary materials. We compare our MST$_s$-based ℓ_1 regularization ($\Theta(K)$ penalty terms) to the pairwise ℓ_1 regularization ($O(K^2)$ penalty tems), which will be referred to as Graph-ℓ_1 regularization in this section. Both models are solved by our proposed generalized ADMM algorithm distributively. In the distributed algorithm, the nodes need to update the local $\mathbf{w}_i$, $\mathbf{\Delta}_{i,l}$ and $\mathbf{z}_{i,l}$ in each iteration. Clearly, the amount of data being transmitted grows as the graph becomes denser. We simulate a 50-node network and each node contains 50 samples. We adjust the network denseness by changing the connection radius r. Two settings are compared: $r = 0.50$ and $r = 0.75$ (see Figure 1 (a) – (b)). We compare the accuracy and costs of the two models with 100 simulations. The MSEs and the estimated cluster number $\hat{S}$ are used for measuring accuracy. We set the baseline to be the average computation time and the average communication cost for the MST$_s$ ℓ_1 model under $r = 0.50$. The boxplots for the accuracy, the computation time ratios, and the communication cost ratios are shown in Figure 2. We can see that our method outperforms in all aspects: Our method improves the MSE at least 21%, while reducing at least 38% computation time and 55% communication cost.



(a) $r = 0.5$. (b) $r = 0.75$.

Figure 1: A 50-node network with two connection radiuses. Solid lines show the clusters and dash lines represent the edges in the network.



(a) Mean square error. (b) Cluster number. (c) Computational time. (d) Communication cost.

Figure 2: The boxplots of MSEs of $\hat{\mathbf{w}}$, the estimated group numbers $\hat{S}$, the computation time ratio and the communication cost ratio of the graph ℓ_1 regularization and MST$_s$ ℓ_1 regularization.

8 Conclusion

In this work, we considered the problem of distributively learning the regression coefficient heterogeneity over networks. We developed a new minimum spanning tree based adaptive fused-lasso model and a low-complexity distributed generalized ADMM algorithm to solve the problem. We investigated the theoretical properties of both the model consistency and algorithm convergence. We showed that our model enjoys the oracle properties (i.e., selection consistency and asymptotic normality) and our distributed optimization algorithm has a linear convergence rate. An interesting future topic is to extend our framework to a more general class of regression problems including generalized linear model and semi-parametric linear model.

References

[1] Rie K Ando and Tong Zhang. Learning on graph with laplacian regularization. In *Advances in neural information processing systems*, pages 25–32, 2007.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1 – 122, 2011.

[3] Arun Tejasvi Chaganty and Percy Liang. Spectral experts for estimating mixtures of linear regressions. In *International Conference on Machine Learning*, pages 1040–1048, 2013.

[4] Yat-Tin Chow, Wei Shi, Tianyu Wu, and Wotao Yin. Expander graph and communication-efficient decentralized optimization. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 1715–1720. IEEE, 2016.

[5] Wei Deng and Wotao Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. *Journal of Scientific Computing*, 66(3):889–916, 2016.

[6] Mark Eisen, Aryan Mokhtari, and Alejandro Ribeiro. Decentralized quasi-newton methods. *IEEE Transactions on Signal Processing*, 65(10):2613–2628, 2017.

[7] Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

[8] Robert G. Gallager, Pierre A. Humblet, and Philip M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Transactions on Programming Languages and systems (TOPLAS)*, 5(1):66–77, 1983.

[9] David Hallac, Jure Leskovec, and Stephen Boyd. Network lasso: Clustering and optimization in large graphs. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 387–396. ACM, 2015.

[10] Trevor Hastie and Robert Tibshirani. Discriminant analysis by gaussian mixtures. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):155–176, 1996.

[11] Zhanhong Jiang, Kushal Mukherjee, and Soumik Sarkar. On consensus-disagreement tradeoff in distributed optimization. In *2018 Annual American Control Conference (ACC)*, pages 571–576. IEEE, 2018.

[12] Zheng Tracy Ke, Jianqing Fan, and Yichao Wu. Homogeneity pursuit. *Journal of the American Statistical Association*, 110(509):175–194, 2015.

[13] Jakub Konecny, Brendan McMahan, and Daniel Ramage. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*, 2015.

[14] D-J Lee, Z Zhu, and P Toscas. Spatio-temporal functional data analysis for wireless sensor networks data. *Environmetrics*, 26(5):354–362, 2015.

[15] Furong Li and Huiyan Sang. Spatial homogeneity pursuit of regression coefficients for large datasets. *Journal of the American Statistical Association*, (just-accepted):1–37, 2018.

[16] Shujie Ma and Jian Huang. A concave pairwise fusion approach to subgroup analysis. *Journal of the American Statistical Association*, 112(517):410–423, 2017.

[17] Shujie Ma, Jian Huang, and Zhiwei Zhang. Exploration of heterogeneous treatment effects via concave fusion. *arXiv preprint arXiv:1607.03717*, 2018.

[18] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48, 2009.

[19] Karl Rohe, Sourav Chatterjee, Bin Yu, et al. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

[20] Juan Shen and Xuming He. Inference for subgroup analysis with a structured logistic-normal mixture model. *Journal of the American Statistical Association*, 110(509):303–312, 2015.

[21] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, 2014.

[22] Lu Tang and Peter XK Song. Fused lasso approach in regression coefficients clustering: learning parameter heterogeneity in data integration. *The Journal of Machine Learning Research*, 17(1):3915–3937, 2016.

[23] Bo Wahlberg, Stephen Boyd, Mariette Annergren, and Yang Wang. An admm algorithm for a class of total variation regularized estimation problems. *IFAC Proceedings Volumes*, 45(16):83–88, 2012.

[24] Weiran Wang, Jialei Wang, Mladen Kolar, and Nathan Srebro. Distributed stochastic multi-task learning with graph regularization. *arXiv preprint arXiv:1802.03830*, 2018.

[25] Kun Yuan, Qing Ling, and Wotao Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

[26] Yunzhang Zhu. An augmented admm algorithm with application to the generalized lasso problem. *Journal of Computational and Graphical Statistics*, 26(1):195–204, 2017.

Supplementary Material for “Learning Coefficient Heterogeneity over Networks: A Distributed Tree-Based Fused-Lasso Approach”

Anonymous Author(s)
Affiliation
Address
email

1 Detailed derivations for the proposed generalized ADMM updating rules

First, given the primal and dual pair $\mathbf{w}^t, \mathbf{z}^t$, to determine the weight difference $\boldsymbol{\Delta}^{t+1}$, we have:

$$\begin{aligned}
\boldsymbol{\Delta}^{t+1} &= \arg\min_{\Delta} L_\tau(\mathbf{w}^t, \boldsymbol{\Delta}, \mathbf{z}^t) \\
&\overset{(a)}{=} \arg\min_{\boldsymbol{\Delta}} \lambda_N \sum_{l=1}^{K-1}\sum_{p=1}^{d} [\hat{\boldsymbol{\pi}}_l]_p \big|[\boldsymbol{\Delta}_l]_p\big| - \langle \mathbf{z}^t, -\boldsymbol{\Delta}\rangle + \frac{\tau}{2}\|\underline{\mathbf{H}}\mathbf{w}^t - \boldsymbol{\Delta}\|^2 \\
&\overset{(b)}{=} \arg\min_{\boldsymbol{\Delta}} \lambda_N \sum_{l=1}^{K-1}\sum_{p=1}^{d} [\hat{\boldsymbol{\pi}}_l]_p \big|[\boldsymbol{\Delta}_l]_p\big| + \boldsymbol{\Delta}^\top \mathbf{z}^t + \frac{\tau}{2}\boldsymbol{\Delta}^\top\boldsymbol{\Delta} - \tau\boldsymbol{\Delta}^\top\underline{\mathbf{H}}\mathbf{w}^t \\
&= \arg\min_{\boldsymbol{\Delta}} \lambda_N \sum_{l=1}^{K-1}\sum_{p=1}^{d} [\hat{\boldsymbol{\pi}}_l]_p \big|[\boldsymbol{\Delta}_l]_p\big| + \frac{\tau}{2}\boldsymbol{\Delta}^\top\boldsymbol{\Delta} - \tau\boldsymbol{\Delta}^\top\Big[\underline{\mathbf{H}}\mathbf{w}^t - \frac{1}{\tau}\mathbf{z}^t\Big] \\
&\overset{(c)}{=} \arg\min_{\boldsymbol{\Delta}} \lambda_N \sum_{l=1}^{K-1}\sum_{p=1}^{d} [\hat{\boldsymbol{\pi}}_l]_p \big|[\boldsymbol{\Delta}_l]_p\big| + \frac{\tau}{2}\Big\|\boldsymbol{\Delta} - (\underline{\mathbf{H}}\mathbf{w}^t - \frac{1}{\tau}\mathbf{z}^t)\Big\|^2,
\end{aligned}$$

where (a) follows from (15) by ignoring constant terms; (b) follows from expanding $\|\underline{\mathbf{H}}\mathbf{w}^t - \boldsymbol{\Delta}\|^2$ and ignoring constant terms; and (c) follows from adding $\frac{\tau}{2}\|\underline{\mathbf{H}}\mathbf{w}^t - \frac{1}{\tau}\mathbf{z}^t\|^2$ and forming the square term. To compute the element $[\boldsymbol{\Delta}_l^{t+1}]_p$, the subgradient can be evaluated as:

$$g([\boldsymbol{\Delta}_l]_p) = \lambda_N[\hat{\boldsymbol{\pi}}_l]_p \nabla\big|[\boldsymbol{\Delta}_l]_p\big| + \tau\Big[[\boldsymbol{\Delta}_l]_p - \big([\mathbf{w}_{s(l)}^t]_p - [\mathbf{w}_{e(l)}^t]_p - \frac{1}{\tau}[\mathbf{z}_l^t]_p\big)\Big], \tag{A1}$$

Setting the subgradient to zero, we have that

$$[\boldsymbol{\Delta}_l^{t+1}]_p = S_{\lambda_N[\hat{\boldsymbol{\pi}}_l]_p/\tau}\Big([\mathbf{w}_{s(l)}^t]_p - [\mathbf{w}_{e(l)}^t]_p - \frac{1}{\tau}[\mathbf{z}_l^t]_p\Big), \tag{A2}$$

where $S_{\lambda_N[\hat{\boldsymbol{\pi}}_l]_p/\tau}$ is the soft-thresholding operator (i.e., $S_a(x) = \operatorname{sign}(x)(|x| - a)_+$). To simplify the notation, we define $S_{\lambda_N\hat{\boldsymbol{\pi}}_l/\tau}$ as the coordinate-wise soft-thresholding operator with $[\lambda_N\hat{\boldsymbol{\pi}}_l/\tau]_p = \lambda_N[\hat{\boldsymbol{\pi}}_l]_p/\tau$. Hence, for the l-th edge with end nodes $s(l)$ and $e(l)$, it follows that:

$$\boldsymbol{\Delta}_l^{t+1} = S_{\lambda_N\hat{\boldsymbol{\pi}}_l/\tau}\Big(\mathbf{w}_{s(l)}^t - \mathbf{w}_{e(l)}^t - \frac{1}{\tau}\mathbf{z}_l^t\Big). \tag{A3}$$

Next, we derive the updating rule for $\mathbf{w}^{t+1}$. In the classical ADMM, it can be shown that:

$$\begin{aligned}
\mathbf{w}^{t+1} &= \arg\min_{\mathbf{w}} L_\tau(\mathbf{w}, \boldsymbol{\Delta}^{t+1}, \mathbf{z}^t) \\
&= [\mathbf{X}^\top\mathbf{X} + \tau\mathbf{L}\otimes\mathbf{I}_d]^{-1}[\mathbf{X}^\top\mathbf{y} + \underline{\mathbf{H}}^\top(\tau\boldsymbol{\Delta}^{t+1} + \mathbf{z}^t)].
\end{aligned} \tag{A4}$$

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

Unfortunately, the matrix inverse in (A4) *cannot* be computed in distributed fashion due to the coupled structure of the Laplacian matrix $\mathbf{L}$. Here, we adopt the generalized ADMM studied in [1], with which the updating can be implemented in parallel. Instead of directly solving the subproblem $\mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} L_\tau(\mathbf{w}, \boldsymbol{\Delta}^{t+1}, \mathbf{z}^t)$, we add a quadratic term $\frac{1}{2}(\mathbf{w} - \mathbf{w}^t)^\top \mathbf{P}(\mathbf{w} - \mathbf{w}^t)$ in the subproblem:

$$\begin{aligned}\mathbf{w}^{t+1} &= \arg\min_{\mathbf{w}} L_\tau(\mathbf{w}, \boldsymbol{\Delta}^{t+1}, \mathbf{z}^t) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^t)^\top \mathbf{P}(\mathbf{w} - \mathbf{w}^t)\\ &= \arg\min_{\mathbf{w}} \frac{1}{2}\sum_{i=1}^{K} \|\mathbf{y}_i - \mathbf{X}_i\mathbf{w}_i\|^2 - \langle \mathbf{z}^t, \underline{\mathbf{H}}\mathbf{w}\rangle + \frac{\tau}{2}\|\underline{\mathbf{H}}\mathbf{w} - \boldsymbol{\Delta}^{t+1}\|^2 + \frac{1}{2}(\mathbf{w} - \mathbf{w}^t)^\top \mathbf{P}(\mathbf{w} - \mathbf{w}^t)\\ &= \arg\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^\top[\mathbf{X}^\top\mathbf{X} + \tau\underline{\mathbf{H}}^\top\underline{\mathbf{H}} + \mathbf{P}]\mathbf{w} - \mathbf{w}^\top[\mathbf{X}^\top\mathbf{y} + \underline{\mathbf{H}}^\top(\tau\boldsymbol{\Delta}^{t+1} + \mathbf{z}^t) + \mathbf{P}\mathbf{w}^t]\\ &= [\mathbf{X}^\top\mathbf{X} + \tau\underline{\mathbf{H}}^\top\underline{\mathbf{H}} + \mathbf{P}]^{-1}[\mathbf{X}^\top\mathbf{y} + \underline{\mathbf{H}}^\top(\tau\boldsymbol{\Delta}^{t+1} + \mathbf{z}^t) + \mathbf{P}\mathbf{w}^t].\end{aligned}$$

Now, we choose the matrix $\mathbf{P} = -\tau\underline{\mathbf{H}}^\top\underline{\mathbf{H}} + \mathbf{D} = -\tau\mathbf{L}\otimes\mathbf{I}_d + \mathbf{D}$, where the diagnoal matrix $\mathbf{D} = \mathrm{diag}(D_1, \cdots, D_K)\otimes\mathbf{I}_d$ with positive scalars D_i for node i and $\mathbf{L} = \mathbf{H}^\top\mathbf{H}$ is the Laplacian matrix for the MST$_s$. It then follows that

$$\mathbf{w}^{t+1} = [\mathbf{X}^\top\mathbf{X} + \mathbf{D}]^{-1}[\mathbf{X}^\top\mathbf{y} + \underline{\mathbf{H}}^\top(\tau\boldsymbol{\Delta}^{t+1} + \mathbf{z}^t) + \mathbf{P}\mathbf{w}^t], \tag{A5}$$

and for each node, plugging in $\mathbf{P} = -\tau\mathbf{L}\otimes\mathbf{I}_d + \mathbf{D}$, the local coefficient can be updated as

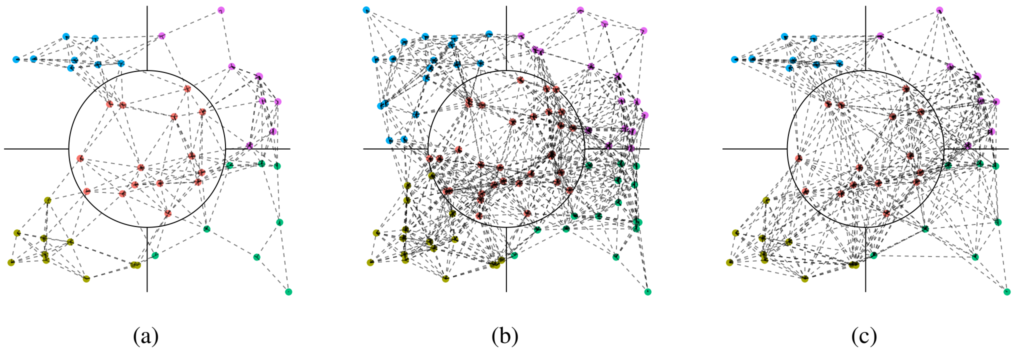
$$\mathbf{w}_i^{t+1} = [\mathbf{X}_i^\top\mathbf{X}_i + D_i\mathbf{I}_d]^{-1}\Big[\mathbf{X}_i^\top\mathbf{y}_i + \sum_{v_i\in e_l}[\mathbf{H}]_{li}(\tau\boldsymbol{\Delta}_l^{t+1} + \mathbf{z}_l^t) + (D_i - \tau\mathrm{deg}(i)\mathbf{w}_i^t + \tau\sum_{j\in\mathcal{N}_i}\mathbf{w}_j^t\Big], \tag{A6}$$

where $v_i \in e_l$ means node v_i is an end node of edge e_l, deg(i) is the degree of the node v_i (i.e., deg$(i) = |\mathcal{N}_i|$). Thus, the updating of $\mathbf{w}_i^{t+1}$ only requires the local and connected neighbor's information, which facilitates *distributed* implementation. Lastly, the dual variables $\mathbf{z}^{t+1}$ can be updated as $\mathbf{z}^{t+1} = \mathbf{z}^t - \tau(\underline{\mathbf{H}}\mathbf{w}^{t+1} - \Delta^{t+1})$, and hence for the l-th edge, the corresponding dual update is:

$$\mathbf{z}_l^{t+1} = \mathbf{z}_l^t - \tau\Big(\mathbf{w}_{s(l)}^{t+1} - \mathbf{w}_{e(l)}^{t+1} - \boldsymbol{\Delta}_l^{t+1}\Big). \tag{A7}$$

2 Further numerical results

In this section, we empirically examine the statistical performance of our proposed estimator $\hat{\mathbf{w}}_{\mathrm{MSTs}}$. In Section 2.1, we compare our MST$_s$ method with serveral exising methods on the random generated networks. A comparison is provided in Section 2.2 to study the impact of the choice of regularization on clustering accuracy and costs. Each simulation result is based on 100 independent repetitions and the random seeds are from 1 to 100.



(a) (b) (c)

Figure 1: Simulation network settings: (a) random design with 50 nodes and the radius 0.5; (c) random design with 100 nodes and the radius 0.5;(c) random design with 50 nodes and the radius 0.75. The solid lines show the underlying partition and the dashed lines represent the edges in network graphs.

Table 1: The results of Simulation 2 with BIC criterion. The results are based on 100 repetitions.

Case	Method	MSE($\hat{\mathbf{w}}$)	$\hat{S}$	Sensitivity	Specificity
	Laplacian	0.0329	NA	NA	NA
n=50	Graph	0.0123	7.28	0.9325	1
K=50	MST$_d$	0.0134	11.87	0.6777	1
	MST$_s$	0.0097	5.55	0.9681	1
	Laplacian	0.0154	NA	NA	NA
n=100	Graph	0.0061	6.83	0.9449	1
K=50	MST$_d$	0.0067	11.58	0.7051	1
	MST$_s$	0.0039	5.35	0.9759	1
	Laplacian	0.0331	NA	NA	NA
n=50	Graph	0.0107	6.94	0.9717	1
K=100	MST$_d$	0.0132	18.62	0.3855	1
	MST$_s$	0.0055	5.89	0.9140	1

2.1 Simulation 1: random design

In this part of simulation, we consider the following network setting (see Figure 1 (a-b)): The nodes are uniformly located in the space $[-1, 1]^2$ and the numbers of the nodes are 50 and 100, respectively. There are five underlying clusters, as shown in Figure 1 (a-b). The covariate $\mathbf{x}$ are generated from multivariate normal distribution with zero mean and covariance Cov$(\mathbf{x}_i, \mathbf{x}_j) = 0.5^{|i-j|}$. The random error ε follows the standard normal distribution. The true coefficients are randomly generated as $\mathbf{w}_{G_1,*} = [4.59, 2.60, -5.12]^\top$, $\mathbf{w}_{G_2,*} = [-2.88, 1.51, 0.59]^\top$, $\mathbf{w}_{G_3,*} = [3.04, 0.53, -4.74]^\top$, $\mathbf{w}_{G_4,*} = [-8.09, -3.20, -2.45]^\top$ and $\mathbf{w}_{G_5,*} = [-0.28, -4.25, -1.28]^\top$. In the connected graph, if the distance of two nodes is smaller than 0.5, then there is an edge between them. Note that with 0.5 as the radius, Assumption 1 is satisfied (see in Figure 1 (a-b)).

We focus on four different types of regularizer: 1) the Laplacian regularizer [6], which can be regarded as a variant of ℓ_2 penalty; 2) the Graph ℓ_1 regularizer as the penalty in (2), which considers all the edges in the graph; 3) the MST$_d$ ℓ_1 regularizer proposed in [3], in which the MST is generated according to spatial distances; 4) our MST$_s$ ℓ_1 regularizer, which generates an MST based on model similarity. We use the Bayesian information criterion (BIC) to select the tuning parameter λ_N. Note that the BIC is widely used in the related works, including homogeneity pursuit methods [2, 3] and subgroup analysis [4, 5]. We simulate three cases: 1) the total number of nodes is $K = 50$ and each node contains $n = 50$ samples; 2) the total number of nodes is $K = 50$ and each node contains $n = 100$ samples; 3) the total number of nodes is $K = 100$ and each node contains $n = 50$ samples. Note that Cases 1) and 2) have different local sample sizes, Cases 1) and 3) have different numbers of nodes and network structures, and Cases 2) and 3) have the same total sample size.

we compare in terms of the following performance metrics: 1) the accuracy of model estimation, MSE($\hat{\mathbf{w}}$)= $\frac{1}{K}\sum_{i=1}^{K} \|\hat{\mathbf{w}}_i - \mathbf{w}_{i,*}\|_2^2$; 2) the estimated group number $\hat{S}$; 3) sensitivity, which measures the proportion of node pairs from the same cluster that are correctly identified; 4) specificity, which measures the proportion of node pairs from the different clusters that are correctly identified. Note that the values of sensitivity and specificity are in the range $[0, 1]$. The closer to 1, the better the prediction is. The simulation results are reported in Table 1 and Figure 2.

From Table 1 and Figure 2, we can see that our MST$_s$ ℓ_1 method outperforms the other methods under all the three circumstances: First of all, we can see that the the MSE from the Laplacian regularizer is higher than those of the other ℓ_1 based regularizer and also the Laplacian regularizer cannot find the nodes' membership. This is because the Laplacian penalty, which is a variant of ℓ_2 penalty, cannot shrink the coefficient difference to zero when two nodes are from the same cluster. Compared to the Graph ℓ_1 regularization, our method improves the efficiency by reducing about $21\%, 36\%, 49\%$ in MSE for the three cases, respectively, while the estimated cluster numbers are closer to five, which is the true group number. Keeping the sample nodes and doubling the local samples in Case 2, the MSE of our proposed regularization reduces to the half of Case 1, which validates our Theorem 1. Comparing Cases 1 and 2, the estimation efficiencies for all three regularizations are improved. This is because by adding more nodes, the total sample size is larger. However, for Cases 2 and 3, although

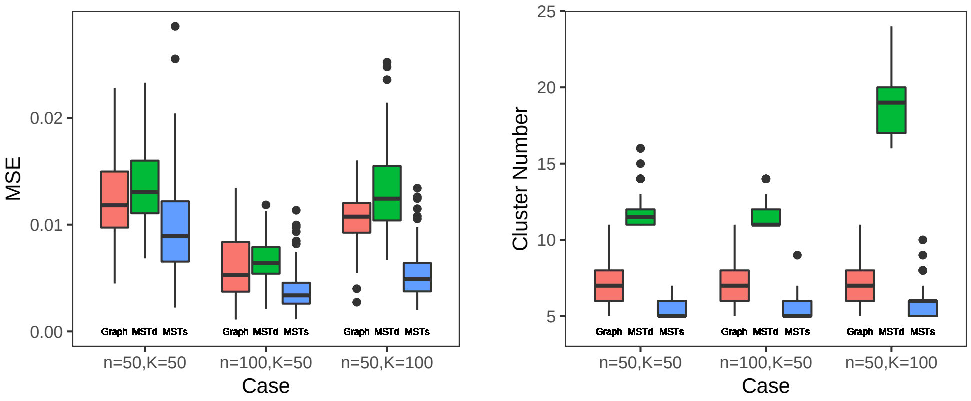


Figure 2: The boxplots of MSEs of $\hat{\mathbf{w}}$ and the estimated group numbers $\hat{S}$ using the three ℓ_1 penalty methods under three cases.

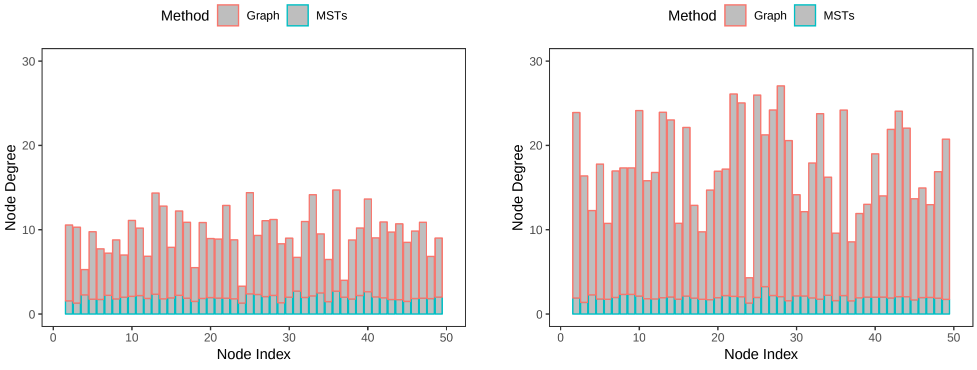
they have the same total sample size, the estimation gets better with fewer node and simpler network topology. Additionally, note that the estimated cluster numbers of the MST$_d$ ℓ_1 regularization is much worse than the Graph ℓ_1 and our regularizations. This is because the MST constructed by the spatial distant cannot guarantee that the nodes from the same group are connected in the tree. This encourages us to use model similarity as the weights when constructing the MST in our MST$_s$ method.

2.2 Simulation 2: network complexity

In this section, we use simulations to illustrate the impact of the choice of regularization on the accuracy, computation time and communication cost. The computations are performed on a Windows computer with a 2.93 GHz Intel(R) Core(TM) i7 CPU processor and 16.0 GB memory. We focus on the two regularizations: the Graph ℓ_1 regularization method and our MST$_s$ ℓ_1 regularization method. In the distributed algorithm, the nodes need to update and store the local $\mathbf{w}_i$, $\mathbf{\Delta}_{i,l}$ and $\mathbf{z}_{i,l}$ in each iteration. Note that the numbers of $\{\mathbf{\Delta}_{i,l}\}_l$ and $\{\mathbf{z}_{i,l}\}_l$ are the same as those the penalty terms associated with node i. Meanwhile, the nodes are required to send the local $\mathbf{w}_i$ to their neighbor nodes in the graph or MST. Clearly, the amount of data being transmitted grows as the graph becomes denser. Here, we consider 50 nodes with the same setting as in Simulation 2. Each node contains 50 samples. We adjust the network denseness by changing the connection radius threshold value r. Two setting are compared, $r = 0.50$ and $r = 0.75$ (See Figure 1 (a) and Figure 1 (c)).

As discussed above, the costs for computation and communication depend on the node degrees of the nodes in the graph or MST. Based on the simulation setting, the connected degrees are shown in Figure 3. The node degrees are deterministic for the graph ℓ_1 regularization. For MST$_s$ ℓ_1 regularization, the node degrees are stochastic because the trees are varying with the local samples. Thus, we repeat 100 trials and compute the average degrees for the nodes. In the case with $r = 0.50$, the maximum degrees for the graph ℓ_1 and MST$_s$ ℓ_1 regularizations are 12 and 2.72, respectively; while in the case with $r = 0.75$, the corresponding maximum degrees are 25 and 3.25, respectively.

Next, with the same local samples from the above 100 simulations, we compare the accuracy and costs for the two regularizations. The MSEs and the estimated group number $\hat{S}$ are used to measure the accuracy. Here we only consider the synchronous algorithm for the computation time approximation. Note that the node with more edges take longer time to calculate more varibles. Thus, the computation time for each iteration is the time for the nodes with the maximum node degree, and the total computation time is the summation of the running times of all interations. The communication cost is defined as the total amount of transmitted messages, which is proportional to the product of the iterations and the edges. We set the baseline as the average computation time and the average communication cost for the MST$_s$ ℓ_1 method under $r = 0.50$. The boxplots for the accuracy, the computation time ratios, and the communication cost ratios are shown in Figure 4. We can see that our MST$_s$ ℓ_1 method outperforms in all aspects. By pruning redundant edges, our MST$_s$ ℓ_1 method enjoys both lower computation and communication costs, as well as the higher



(a) $r = 0.50$

(b) $r = 0.75$

Figure 3: The barcharts of the node degrees for the two setting2 in Simulation 3.

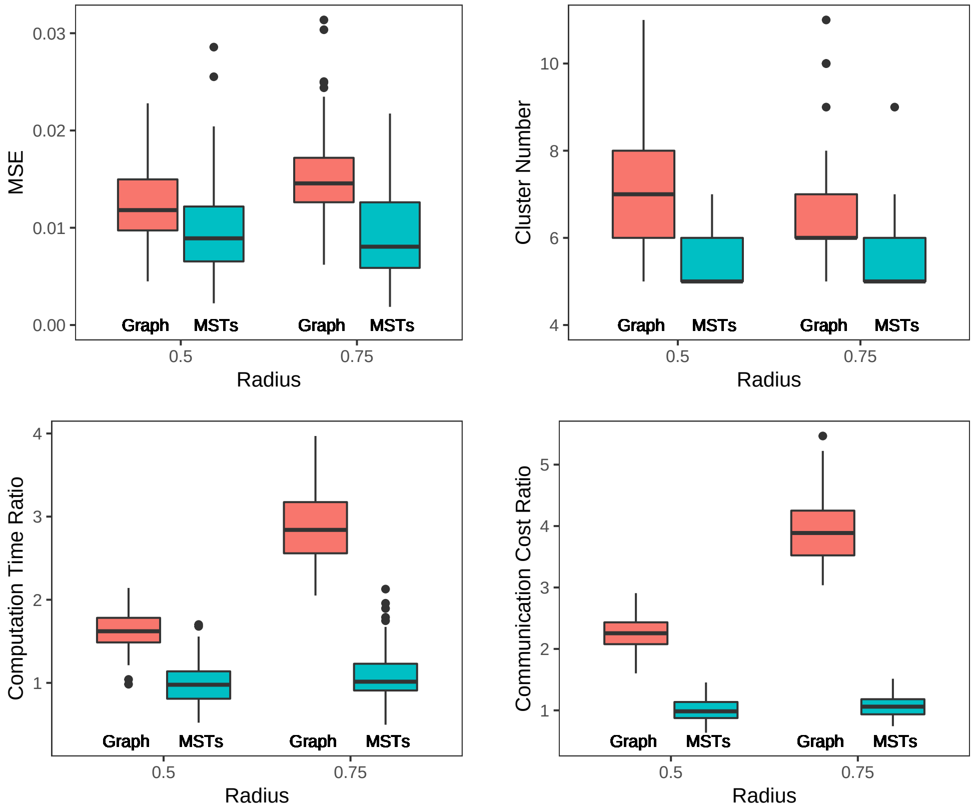


Figure 4: The comparison boxplots of MSEs of $\hat{\mathbf{w}}$, the estimated group numbers $\hat{S}$, the computation time ratio and the communication cost ratio of the graph ℓ_1 regularization and MST$_s$ ℓ_1 regularization.

estimation accuracy. In contrast, for the Graph ℓ_1 regularization, more edges in the graph result in longer computation time and higher communication cost, as well as less accurate estimation.

3 Proofs of the theoretical results

3.1 Proof of Lemma 1

First, we show that under Assumption 1, as the local sample size $n \to \infty$, for any node v_i, the corresponding neighbor node with the minimum weight is from the same cluster with probability 1, i.e., $\lim_{n\to\infty} \mathbb{P}(v_i \sim v_j) = 1$, $j = \arg\min_j \tilde{s}_{i,j}, \forall v_i$, where the notation "$\sim$" means being in the same cluster. Note $\hat{\mathbf{w}}_{i,OLS}$ a root-n consistent estimator of $\mathbf{w}_i$. Thus, the weights for v_i are

$$\tilde{s}_{i,j} = \begin{cases} O_p\Big(\frac{1}{\sqrt{n}}\Big), & \text{if}(v_i, v_j) \in \text{E and } v_i \sim v_j, \\ \|\hat{\mathbf{w}}_{G_k} - \hat{\mathbf{w}}_{G_l}\| + O_p\Big(\frac{1}{\sqrt{n}}\Big), & \text{if}(v_i, v_j) \in \text{E and } v_i \in G_k, v_j \in G_l, \\ \infty, & \text{otherwise.} \end{cases} \tag{A8}$$

where G_k and G_l represent different underlying clusters, $\hat{\mathbf{w}}_{G_k}$ and $\hat{\mathbf{w}}_{G_l}$ are their corresponding coefficients, repectively. Thus, for any node v_i, the event that its corresponding neighboring node with the smallest edge weight is from the same cluster happens with probability 1 as $n \to \infty$.

With the above result, we will show that there is no isolated node in the MST$_s$ with probability 1. We prove it by contradiction. Suppose there is one isolated node v_i. From Assumption 1, we know that there exists a node v_j from the neighbors of v_i in G, such that $\tilde{s}_{i,j} = \min_k \tilde{s}_{i,k}$ and $v_i \sim v_j$. Since v_i is an isolated node, the edge (v_i, v_j) is not in the MST$_s$. Now, we add the edge (v_i, v_j) to the MST$_s$. By the property of the spanning tree, adding one more edge to the MST$_s$ would create a cycle. Thus, after adding the edge (v_i, v_j), we have one cycle C in the new graph: MST$_s$+(v_i, v_j). Since v_i is an isolated node, in the MST$_s$, v_i is connected with a node from different cluster, denoted as v_l, and the edge $(v_i, v_l) \in C$. From the weight (A8), it holds that $\tilde{s}_{i,l} > \tilde{s}_{i,j}$ with probability 1. This suggests that $\tilde{s}_{i,j}$ is not the largest weight in C and there exist another edge in C with the largest weight among all the edges in C. By the cycle property of the minimum spanning tree, the edge with largest weight in C cannot be included in the MST$_s$, contradicting to v_i being an isolated node in the MST$_s$. Therefore, there is no isolated node in the MST$_s$ with probability 1 as $n \to \infty$.

3.2 Proof of Theorem 1

The proof of Theorem 1 is inspired by [7]. Recall the objective function on $\boldsymbol{\Delta}$:

$$L_{\text{MST}_s}(\boldsymbol{\Delta}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1}\boldsymbol{\Delta}\|^2 + \lambda_N \sum_{p=1}^{dK} [\hat{\boldsymbol{\pi}}]_p |[\Delta]_p|, \tag{A9}$$

and weights

$$[\hat{\boldsymbol{\pi}}]_p = \begin{cases} 1/[\tilde{\underline{\underline{\mathbf{H}}}}\mathbf{w}_{OLS}]_p^\gamma, & \text{if } p \bmod K \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{A10}$$

We first prove the asymptotic normality result. Let $\boldsymbol{\Delta} = \boldsymbol{\Delta}_* + \mathbf{u}/\sqrt{N}$ and

$$\Psi_{\text{MST}_s}(\mathbf{u}) = \frac{1}{2}\Big\|\mathbf{y} - \mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1}\Big(\boldsymbol{\Delta}_* + \frac{\mathbf{u}}{\sqrt{N}}\Big)\Big\|^2 + \lambda_N \sum_{p=1}^{dK} [\hat{\boldsymbol{\pi}}]_p \Big|[\boldsymbol{\Delta}_* + \frac{\mathbf{u}}{\sqrt{N}}]_p\Big|. \tag{A11}$$

Denote $\hat{\mathbf{u}}_{\text{MST}_s} = \Psi_{\text{MST}_s}(\mathbf{u})$. Then, we have $\widehat{\boldsymbol{\Delta}}_{\text{MST}_s} = \boldsymbol{\Delta}_* + \hat{\mathbf{u}}_{\text{MST}_s}/\sqrt{N}$. Define $V_N(\mathbf{u}) = \Psi_{\text{MST}_s}(\mathbf{u}) - \Psi_{\text{MST}_s}(\mathbf{0})$, where

$$V_N(\mathbf{u}) = \frac{1}{2}\mathbf{u}^\top\Big[\frac{1}{N}(\mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1})^\top\mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1}\Big]\mathbf{u} - \frac{\boldsymbol{\varepsilon}^\top\mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1}}{\sqrt{N}}\mathbf{u} + \frac{\lambda_N}{\sqrt{N}}\sum_{p=1}^{dK}[\hat{\boldsymbol{\pi}}]_p\sqrt{N}\Big(\Big|\Big[\Delta_* + \frac{\mathbf{u}}{\sqrt{N}}\Big]_p\Big| - |[\Delta_*]_p|\Big). \tag{A12}$$

With Assumption 2, we have that

$$\frac{1}{N}(\mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1})^\top\mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1} \xrightarrow{p} \mathbf{C} \text{ and } \frac{\boldsymbol{\varepsilon}^\top\mathbf{X}\tilde{\underline{\underline{\mathbf{H}}}}^{-1}}{\sqrt{N}} \xrightarrow{d} \mathbf{W} = \mathcal{N}(0, \sigma^2\mathbf{C}).$$

In what follows, we derive the limiting behavior of the third term in (A12). If $[\Delta_*]_p \neq 0$ and $p \bmod K \neq 0$, then $[\hat{\boldsymbol{\pi}}]_p \xrightarrow{p} |[\Delta_*]_p|^{-\gamma}$ and $\sqrt{N}(|[\Delta_* + \frac{\mathbf{u}}{\sqrt{N}}]_p| - |[\Delta_*]_p|) \to [\mathbf{u}]_p \text{sign}([\Delta_*]_p)$. By the Slutsky's theorem, we have $\frac{\lambda_N}{\sqrt{N}}[\hat{\boldsymbol{\pi}}]_p\sqrt{N}(|[\Delta_* + \frac{\mathbf{u}}{\sqrt{N}}]_p| - |[\Delta_*]_p|) \xrightarrow{p} 0$ with $\lambda_N/\sqrt{N} \to 0$. If $[\Delta_*]_p = 0$ and $p \bmod K \neq 0$, then it holds that

$$\sqrt{N}(|[\Delta_* + \frac{\mathbf{u}}{\sqrt{N}}]_p| - |[\Delta_*]_p|) \to |[\mathbf{u}]_p| \tag{A13}$$

$$\frac{\lambda_N}{\sqrt{N}}\hat{\boldsymbol{\pi}}_p = \frac{\lambda_N}{\sqrt{N}} N^{\gamma/2} |\sqrt{N}[\widehat{\Delta}_{\text{OLS}}]_p|^{-\gamma} = O_p(1)\lambda_N N^{(\gamma-1)/2} \to \infty. \tag{A14}$$

Thus, with $[\Delta_*]_p = 0$ and $p \bmod K \neq 0$, we have

$$\frac{\lambda_N}{\sqrt{N}}[\hat{\boldsymbol{\pi}}]_p \sqrt{N}(|[\Delta_* + \frac{\mathbf{u}}{\sqrt{N}}]_p| - |[\Delta_*]_p|) \tag{A15}$$

$$= O_p(1)|[\mathbf{u}]_p|\lambda_N N^{(\gamma-1)/2}] = \begin{cases} 0, & \text{if } [\mathbf{u}]_p = 0, \\ \infty, & \text{otherwise.} \end{cases} \tag{A16}$$

If $p \bmod K = 0$, with the above weight $\hat{\boldsymbol{\pi}}_p = 0$, we have $\frac{\lambda_N}{\sqrt{N}}[\hat{\boldsymbol{\pi}}]_p\sqrt{N}(|[\Delta_* + \frac{\mathbf{u}}{\sqrt{N}}]_p| - |[\Delta_*]_p|) = 0$. Thus, by the Slutsky's theorem, it holds that $V_N(\mathbf{u}) \xrightarrow{d} V(\mathbf{u})$, where

$$V(\mathbf{u}) = \begin{cases} \frac{1}{2}\mathbf{u}_{\mathcal{A}_*}^\top \mathbf{C}_{\mathcal{A}_*}\mathbf{u}_{\mathcal{A}_*} - \mathbf{u}_{\mathcal{A}_*}^\top \mathbf{W}_{\mathcal{A}_*}, & \text{if } [\mathbf{u}]_p = 0, \ \forall p \notin \mathcal{A}_*, \\ \infty, & \text{otherwise.} \end{cases} \tag{A17}$$

Note V_N is convex and the unique minimum of V is $\tilde{\mathbf{u}}$, where $[\tilde{\mathbf{u}}]_{\mathcal{A}_*} = \mathbf{C}_{\mathcal{A}_*}^{-1}\mathbf{W}_{\mathcal{A}_*}$ and $[\tilde{\mathbf{u}}]_{\mathcal{A}_*^c} = \mathbf{0}$. Following the same line as in [7], we have

$$[\hat{\mathbf{u}}_{\text{MST}_s}]_{\mathcal{A}_*} \xrightarrow{d} \mathbf{C}_{\mathcal{A}_*}^{-1}\mathbf{W}_{\mathcal{A}_*} \text{ and } [\hat{\mathbf{u}}_{\text{MST}_s}]_{\mathcal{A}_*^c} \xrightarrow{d} \mathbf{0}. \tag{A18}$$

With $\mathbf{W}_{\mathcal{A}_*} = \mathcal{N}(0, \sigma^2\mathbf{C}_{\mathcal{A}_*})$, the asymptotic normality part is proved.

Next, we show the consistency part. For $p \in \mathcal{A}_*$, the asymptotic normality result shows that $[\widehat{\Delta}_{\text{MST}_s}]_p \xrightarrow{p} [\Delta_*]_p$, therefore, $\mathbb{P}(p \in \hat{\mathcal{A}}_N) \to 1$. Then, we need to show $\forall p' \notin \mathcal{A}_*$, $\mathbb{P}(p \in \hat{\mathcal{A}}_N) \to 0$. Consider the event $p' \in \hat{\mathcal{A}}_N$. With the KKT optimality conditions, it holds that

$$[\mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}]_{p'}^\top[\mathbf{y} - \mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}\widehat{\Delta}_{\text{MST}_s}] = \pm\lambda_N[\hat{\boldsymbol{\pi}}]_{p'}, \tag{A19}$$

where $[\cdot]_{\cdot p}$ represents the pth column of the matrix. Note that on the RHS, we have

$$\lambda_N[\hat{\boldsymbol{\pi}}]_{p'}/\sqrt{N} = \lambda_N N^{(\gamma-1)/2}|\sqrt{N}\widehat{\Delta}_{\text{OLS}}|^{-\gamma} \xrightarrow{p} \infty, \tag{A20}$$

while on the LHS, we have

$$2\frac{[\mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}]_{\cdot p'}^\top[\mathbf{y} - \mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}\widehat{\Delta}_{\text{MST}_s}]}{\sqrt{N}} = 2\frac{[\mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}]_{\cdot p'}^\top\mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}[\Delta_* - \widehat{\Delta}_{\text{MST}_s}]}{\sqrt{N}} + 2\frac{[\mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}]_{\cdot p'}^\top\boldsymbol{\varepsilon}}{\sqrt{N}}. \tag{A21}$$

With the asymptotic normality result and the Slutsky's theorem, $\lambda_N[\hat{\boldsymbol{\pi}}]_{p'}/\sqrt{N}$ asymptotically follows a normal distribution. Thus, we finally have

$$\mathbb{P}(p' \in \hat{\mathcal{A}}_N) \leq \mathbb{P}([\mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}]_{p'}^\top[\mathbf{y} - \mathbf{X}\tilde{\underline{\mathbf{H}}}^{-1}\widehat{\Delta}_{\text{MST}_s}] = \pm\lambda_N[\hat{\boldsymbol{\pi}}]_{p'}) \to 0. \tag{A22}$$

3.3 Proof of Theorem 2

We prove the convergence following the framework of [1]. Recalling the constrained objective function (16), for notational simplicity, we denote $f(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{y}_i - \mathbf{X}_i\mathbf{w}_i\|^2$ and $g(\boldsymbol{\Delta}) = \lambda_N\sum_{l=1}^{K-1}\sum_{p=1}^{d}[\hat{\boldsymbol{\pi}}_l]_p|[\boldsymbol{\Delta}_l]_p|$. Next, we will provide a convergence analysis for our proposed generalized ADMM method for the problem in the following form:

$$\begin{aligned} \text{Minimize} \quad & f(\mathbf{w}) + g(\boldsymbol{\Delta}) \\ \text{subject to} \quad & \underline{\mathbf{H}}\mathbf{w} - \boldsymbol{\Delta} = \mathbf{0}. \end{aligned} \tag{A23}$$

Note that $f(\mathbf{w})$ is the ordinary least square problem, and with Assumption 2, it is strongly convex with a convex modulus $\nu_f > 0$ and has Lipschitz continuous gradients; for $g(\Delta)$, it is a convex function and the corresponding convex modulus $\nu_g = 0$; the constraint $\underline{\mathbf{H}}\mathbf{w} - \boldsymbol{\Delta} = \mathbf{0}$ is linearly independent and $\underline{\mathbf{H}}$ is full row rank.

Now, we state Lemma 2.1 and Theorem 2.2 from [1] as follows (with our notation).

Lemma. *The sequence $\{\mathbf{u}^t\}$ obeys the followings optimality conditions at each iteration:*

$$\underline{\mathbf{H}}^\top \mathbf{z}^{t+1} + \mathbf{P}(\mathbf{w}^t - \mathbf{w}^{t+1}) = \nabla f(\mathbf{w}^{t+1}) \tag{A24}$$

$$-\mathbf{z}^{t+1} + \tau \underline{\mathbf{H}}(\mathbf{w}^t - \mathbf{w}^{t+1}) \in \nabla g(\Delta^{t+1}) \tag{A25}$$

Theorem. *If the matrix $\mathbf{P}$ satisfies that $\mathbf{P} \succ \mathbf{0}$, then there exists $\eta > 0$ such that*

$$\|\mathbf{u}^t - \mathbf{u}_*\|_{\mathbf{G}}^2 - \|\mathbf{u}^{t+1} - \mathbf{u}_*\|_{\mathbf{G}}^2 \geq \eta \|\mathbf{u}^t - \mathbf{u}^{t+1}\|_{\mathbf{G}}^2 + 2\nu_f \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2. \tag{A26}$$

Note that the RHS of (A26) is positive. Hence, $\|\mathbf{u}^{t+1} - \mathbf{u}_*\|_{\mathbf{G}}^2$ and $\mathbf{u}^{t+1}$ are bounded. With the boundedness of sequence $\{\mathbf{u}^t\}$, it follows that there exists a converging subsequence $\{\mathbf{u}^{t_j}\}$ of $\{\mathbf{u}^t\}$. Let $\bar{\mathbf{u}} = \lim_{j\to\infty} \mathbf{u}^{t_j}$. In what follows, we will show that $\bar{\mathbf{u}} = (\bar{\mathbf{w}}^\top, \bar{\Delta}^\top, \bar{\mathbf{z}}^\top)^\top$ is a KKT point. Let $\mathbf{u}_*$ denote an arbirary KKT point for the problem (A23).

From (A26), it can be seen that $\|\mathbf{u}^t - \mathbf{u}_*\|_{\mathbf{G}}^2$ is monotonically nonincreasing and converging. Also, due to $\eta > 0$, $\|\mathbf{u}^t - \mathbf{u}^{t+1}\|_{\mathbf{G}}^2 \to 0$. With the structure of $\mathbf{G}$, it holds that $\mathbf{z}^t - \mathbf{z}^{t+1} \to \mathbf{0}$, and equivalently,

$$\underline{\mathbf{H}}\mathbf{w}^{t+1} - \boldsymbol{\Delta}^{t+1} = \mathbf{0}, \tag{A27}$$

from the updating rule of $\mathbf{z}$. Taking limit on (A27) over the subsequence, we have

$$\underline{\mathbf{H}}\bar{\mathbf{w}} - \bar{\Delta} \to \mathbf{0}, \tag{A28}$$

Since $\|\mathbf{u}^t - \mathbf{u}^{t+1}\|_{\mathbf{G}}^2 \to \mathbf{0}$, it follows that $\mathbf{w}^t - \mathbf{w}^{t+1} \to \mathbf{0}$. From (A24) and (A25), we have that: 1) $\underline{\mathbf{H}}^\top \bar{\mathbf{z}} = \nabla f(\bar{\mathbf{w}})$; and 2) $-\bar{\mathbf{z}} \in \nabla g(\bar{\boldsymbol{\Delta}})$. With (A28), we have that $\bar{\mathbf{u}}$ is a KKT point and thus we have $\mathbf{u}_* = \bar{\mathbf{u}}$. Since $\mathbf{u}^{t_j} \to \mathbf{u}_*$ and the convergence of $\|\mathbf{u}^t - \mathbf{u}_*\|_{\mathbf{G}}^2$, we have $\|\mathbf{u}^t - \mathbf{u}_*\|_{\mathbf{G}}^2 \to \mathbf{0}$.

Next, we prove the linear convergence rate of our algorithm. From (A26), we have that

$$\|\mathbf{u}^t - \mathbf{u}_*\|_{\mathbf{G}}^2 - \|\mathbf{u}^{t+1} - \mathbf{u}_*\|_{\mathbf{G}}^2 \geq \eta \|\mathbf{u}^t - \mathbf{u}^{t+1}\|_{\mathbf{G}}^2 + 2\nu_f \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2. \tag{A29}$$

Hence, we need to show that there exists some $\delta > 0$, such that

$$\eta \|\mathbf{u}^t - \mathbf{u}^{t+1}\|_{\mathbf{G}}^2 + 2\nu_f \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2 \geq \delta \|\mathbf{u}^{t+1} - \mathbf{u}_*\|_{\mathbf{G}}^2, \tag{A30}$$

which is equivalent to

$$\eta \|\mathbf{w}^t - \mathbf{w}^{t+1}\|_{\mathbf{D}}^2 + \frac{\eta}{\tau}\|\mathbf{z}^t - \mathbf{z}^{t+1}\|^2 + 2\nu_f \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2 \geq \delta \|\mathbf{w}^{t+1} - \mathbf{w}_*\|_{\mathbf{D}}^2 + \frac{\delta}{\tau}\|\mathbf{z}^{t+1} - \mathbf{z}_*\|^2 \tag{A31}$$

To this end, we state Lemma 3.2 in [1] as follows:

Lemma. *Suppose that ∇f is Lipschitz continuous with constant L_f. For all $\mu > 1$, we have*

$$\|\mathbf{z}^{t+1} - \mathbf{z}_*\|^2 \leq c_1 \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2 + c_2 \|\mathbf{w}^t - \mathbf{w}^{t+1}\|^2, \tag{A32}$$

where $c_1 = L_f^2(1 - \frac{1}{\mu})^{-1}\lambda_{\min}^{-1}(\underline{\mathbf{H}\mathbf{H}}^\top) > 0$ and $c_2 = \mu\|\mathbf{P}\|^2\lambda_{\min}^{-1}(\underline{\mathbf{H}\mathbf{H}}^\top) > 0$.

Applying the above lemma to the RHS of (A31), we have

$$\begin{aligned} &\eta \|\mathbf{w}^t - \mathbf{w}^{t+1}\|_{\mathbf{D}}^2 + \frac{\eta}{\tau}\|\mathbf{z}^t - \mathbf{z}^{t+1}\|^2 + 2\nu_f \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2 \\ &\qquad \geq c_3 \|\mathbf{w}^t - \mathbf{w}^{t+1}\|^2 + \frac{\eta}{\tau}\|\mathbf{z}^t - \mathbf{z}^{t+1}\|^2 + 2\nu_f \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2, \end{aligned} \tag{A33}$$

where $c_3 = \eta \min D_i$. With c_4 and c_5 satisfying $c_3 - c_1 c_5 \geq c_4$ and $2\nu_f \geq c_2 c_5$, it follows that

$$\begin{aligned} &\eta \|\mathbf{w}^t - \mathbf{w}^{t+1}\|_{\mathbf{D}}^2 + \frac{\eta}{\tau}\|\mathbf{z}^t - \mathbf{z}^{t+1}\|^2 + 2\nu_f \|\mathbf{w}^{t+1} - \mathbf{w}_*\|^2 \\ &\qquad \geq c_4 \|\mathbf{w}^t - \mathbf{w}^{t+1}\|^2 + c_5 \|\mathbf{z}^{t+1} - \mathbf{z}_*\|^2 + \frac{\eta}{\tau}\|\mathbf{z}^t - \mathbf{z}^{t+1}\|^2 \\ &\qquad \geq \frac{c_4}{\max D_i}\|\mathbf{w}^t - \mathbf{w}^{t+1}\|_{\mathbf{D}}^2 + \frac{c_5}{\tau}\tau\|\mathbf{z}^{t+1} - \mathbf{z}_*\|^2 \\ &\qquad \geq \delta \|\mathbf{u}^{t+1} - \mathbf{u}_*\|_{\mathbf{G}}^2, \end{aligned} \tag{A34}$$

where $\delta = \min\{c_4/\max D_i, c_5/\tau\}$. This completes the proof.

References

[1] Wei Deng and Wotao Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. *Journal of Scientific Computing*, 66(3):889–916, 2016.

[2] Zheng Tracy Ke, Jianqing Fan, and Yichao Wu. Homogeneity pursuit. *Journal of the American Statistical Association*, 110(509):175–194, 2015.

[3] Furong Li and Huiyan Sang. Spatial homogeneity pursuit of regression coefficients for large datasets. *Journal of the American Statistical Association*, (just-accepted):1–37, 2018.

[4] Shujie Ma and Jian Huang. A concave pairwise fusion approach to subgroup analysis. *Journal of the American Statistical Association*, 112(517):410–423, 2017.

[5] Shujie Ma, Jian Huang, and Zhiwei Zhang. Exploration of heterogeneous treatment effects via concave fusion. *arXiv preprint arXiv:1607.03717*, 2018.

[6] Weiran Wang, Jialei Wang, Mladen Kolar, and Nathan Srebro. Distributed stochastic multi-task learning with graph regularization. *arXiv preprint arXiv:1802.03830*, 2018.

[7] Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429, 2006.