Fast and Robust Sparsity Learning over Networks: A Decentralized Surrogate Median Regression Approach

Weidong Liu, Xiaojun Mao, *Member, IEEE,* and Xin Zhang

***Abstract*—Decentralized sparsity learning has attracted a significant amount of attention recently due to its rapidly growing applications. To obtain the robust and sparse estimators, a natural idea is to adopt the non-smooth median loss combined with a ℓ_1 sparsity regularizer. However, most of the existing methods suffer from slow convergence performance caused by the *double* non-smooth objective. To accelerate the computation, in this paper, we proposed a decentralized surrogate median regression (deSMR) method for efficiently solving the decentralized sparsity learning problem. We show that our proposed algorithm enjoys a linear convergence rate with a simple implementation. We also investigate the statistical guarantee, and it shows that our proposed estimator achieves a near-oracle convergence rate without any restriction on the number of network nodes. Moreover, we establish the theoretical results for sparse support recovery. Thorough numerical experiments and real data study are provided to demonstrate the effectiveness of our method.**

I. Introduction

In recent years, decentralized machine learning (ML) has received growing research interest due to its advantages in system stability, data privacy, and computation efficiency [1, 2]. In contrast to the traditional centralized distributed architecture coordinated by a master machine, decentralized ML works with peer-to-peer networked systems, where workers can perform local computation and pass the message through the network links. The goal of decentralized ML is to learn a global ML model by having workers optimize their own models and share local model information with their neighbors. So far, decentralized ML has achieved significant success in many scientific and engineering areas, including distributed sensing in wireless sensor networks[3, 4, 5, 6], multi-agent robotic systems[7, 8, 9], smart grids[10, 11] etc.

However, in spite of the increasing adoption in applications, the performances of most decentralized ML methods are not robust and are vulnerable to the following three aspects: 1) *Data Heterogeneity*. Due to the lack of the global information aggregated by the central master, workers in decentralized network systems learn the model heavily relied on the local data and neighboring information. Thus, data heterogeneity across workers often causes a so-called 'consensus error' that reduces the model estimation efficiency [2, 12, 13]; 2) *Noise Contamination*. In many real-world ML applications, training data are contaminated by heavy-tailed noises or outlier values. Under this circumstance, adopting the mean square loss or Huber loss [14] in decentralized ML framework become less efficient or even infeasible [15]; 3) *Cyber Attacks*. Without the central master, it is difficult for decentralized ML methods to identify the cyber attackers, which inject abnormal data into ML training. Typically, these injected fake data will affect the training process and bias the estimated model.

In this paper, we aim to improve the estimation efficiency and robustness for decentralized ML. Specifically, we focus on the problem of decentralized sparsity learning on linear model. Consider a network with m nodes, each of which holds a dataset $D_j = \{\mathbf{x}_i, y_i\}_{i=1}^{n_j}$ for $j \in \{1, \ldots, m\}$, where $y \in \mathbb{R}$ is the observed response values, $\mathbf{x} = (x_1, \ldots, x_p)^\top \in \mathbb{R}^p$ is a p-dimensional covariate vector. In the following we always use the notation $[t]$ as the set of indices $\{1, \ldots, t\}$ where the number t can be changed to be the length of the set. We suppose that the data follows a linear model:

$$y_i = \mathbf{x}_i^\top \boldsymbol{\beta} + \epsilon_i, \quad \forall i \in [n_j], j \in [m], \tag{1}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^\top$ is the globally common linear coefficient of dimension p, and ϵ_i is the measurement noise which is independent with covariate $\mathbf{x}_i$. Additionally, we consider that the coefficient $\boldsymbol{\beta}$ follows a sparse structure with s non-zero elements. We denote the support of $\boldsymbol{\beta}$ as $S = \{1 \leq j \leq p : \beta_j \neq 0\}$ and $|S| = s$. Practically, we allow the data to be heterogeneous across nodes, i.e., the covariance structure $\boldsymbol{\Sigma}_j = \mathbb{E}_{\mathbf{x}_i \in D_j}(\mathbf{x}_i \mathbf{x}_i^\top)$ and distributions for measurement noise ϵ_i could be different with respect to node index j. Our primary interest is to estimate $\boldsymbol{\beta}$ and recover its sparsity support by utilizing all the data across the network $D = \cup_{j=1}^m D_j$, where the size of D is $N = \sum_{j=1}^m n_j$. For the ease of presentation, we assume that the data are evenly collected and each node has the sample size $|D_j| = n_j = n$ which implies $N = mn$.

There exist several decentralized sparsity learning methods [16, 17, 18, 19] that enjoy a fast linear convergence rate. However, such convergence advantage benefits from the traditional least square loss, which is vulnerable to the abnormal value or outlier noise. In this work, we allow the noise ϵ to be a heavy-tailed noise, of which the variance can be infinite. To mitigate the effect of heavy-tailed noise, a natural idea is to replace the mean square loss with the median loss, so that $\boldsymbol{\beta}$ can be estimated by solving the following minimization

W. Liu is with the School of Mathematical Sciences, School of Life Sciences and Biotechnology, MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University, Shanghai, 200240 China (e-mail: weidongl@sjtu.edu.cn).

X. Mao is with School of Data Science, Fudan University, Shanghai, 200433 China (e-mail: maoxj@fdu.edu.cn).

X. Zhang is with Department of Statistics, Iowa State University, Ames, IA, 50011, USA (e-mail: xinzhang@iastate.edu).

problem,

$$\min_{\boldsymbol{\beta} 2 \mathrm{R}^p} \frac{1}{N} \sum_{j=1}^{m} \sum_{(\mathbf{x}_i, y_i) 2 D_j} \left| y_i \quad \mathbf{x}_i^{\mathsf{T}} \boldsymbol{\beta} \right|, \tag{2}$$

where $j\ j$ is for the absolute value. To further pursue the sparsity structure of $\boldsymbol{\beta}$, we regularize the problem (2) and define the sparse robust estimator for $\boldsymbol{\beta}$ as

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta} 2 \mathrm{R}^p} \frac{1}{N} \sum_{j=1}^{m} \sum_{(\mathbf{x}_i, y_i) 2 D_j} \left| y_i \quad \mathbf{x}_i^{\mathsf{T}} \boldsymbol{\beta} \right| + \lambda_N j \boldsymbol{\beta} j_1, \tag{3}$$

where $j\boldsymbol{\beta}j_1$ is the ℓ_1-regularization term and $\lambda_N > 0$ is the regularization parameter. However, due to the non-smooth property of both the median loss and the ℓ_1-regularization term, it's a non-trivial task to optimize the above *double* non-smooth minimization problem (3), let alone solving it in a decentralized setting. Although several decentralized subgradient methods were proposed to address the non-smooth median loss in network systems (e.g., [20, 21]), the estimation efficiency is not satisfactory because of their sublinear convergence rate. The limitations of these existing works naturally arise the following question: *Could we develop a decentralized sparsity learning method for ℓ_1-regularized median loss that can achieve a fast linear convergence rate?*

We give *affirmative* answer to the above question in this paper. We develop a Decentralized Surrogate Median Regression (deSMR) method and its key features are summarized in the followings:

To avoid the tremendous complexity caused by the non-smooth loss, we suggerate the median loss with a smooth least square loss with transformed responses. The idea of response transformation is inspired by Newton-Raphson updating. We show that our method only needs a constant-order round of reformulation to achieve high accurate estimation;

For collaborative solving the reformulated sparsity-penalized least square loss, we derive a decentralized generalized ADMM algorithm, which enjoys a *linear* convergence. Our algorithm only requires nodes to find a closed-form solution for locally approximated sub-problems and share local model parameters with their neighbors, and thus it is easy to implement.

By incorporating the algorithm convergence rate, we first establish the one-round node-wise statistical consistency rate for our proposed estimators in Theorem 2. We show that our estimators recover the exact sparsity support under some mild conditions. Furthermore, we can iteratively improve the estimators over the networks by replacing the initial estimators. We show that after a constant number of iterations of loss surrogate and enough number of ADMM iterations, our estimators achieve a near-optimal rate of $\sqrt{s \log(N _ p)/N}$ in Theorem 5. This rate matches with the optimal rate $\sqrt{s/N}$ obtained under the true support is known up to a logarithmic factor. Due to the space limitation, we leave the corresponding results of support recovery in the Supplementary Material.

A. Related Work

In the literature, many methods have been developed for solving distributed sparse learning problem. However, most of the literature focus on the centralized distributed setting [22, 23, 24, 25, 26, 27], of which the bottleneck is the central master: for example, the limited bandwidth of the central master will undermine the training performance and failure of the central master will directly ruin the system [28]. The authors of [16] are the first few to investigate the sparsity learning problem in the peer-to-peer networked systems. Specifically, they proposed two novel algorithms, distributed quadratic programming Lasso (DQP-Lasso) and distributed coordinate descent Lasso (DCD-Lasso), based on the alternating-direction method of multipliers [29]. But the two algorithms suffer the high computation complexity caused by updates for primal variable. In [17], the authors proposed Inexact Consensus-ADMM and showed that the proposed algorithm enjoys the linear convergence rate. Furthermore, the authors of [18, 30] proposed the decentralized proximal gradient descent algorithms to bypass introducing the Lagrangian dual. However, the above works mainly focus on the advances of the optimization convergence, while it remains unclear whether they could provide statistical guarantees on the consistency of model estimation and sparsity recovery. Additionally, these works study on the smooth loss for data fitting, e.g. the least square loss, which is not robust in term of heavy-tailed noise and outlier data.

To learn a robust sparse ML model, the quantile loss is widely adopted in statistics community [31, 32, 33, 34]. It is worthwhile noting that the median loss is a special case of the well known quantile loss $\rho_\tau(x) = x(\tau \quad \mathrm{I}[x \quad 0])$ [35] by setting the pre-specified quantile level $\tau = 0.5$. Indeed our method can be easily extended to general quantile loss ρ_τ by shifting a constant to the response. For ease of presentation, we focus on analyzing the median loss. Motivated by the success of the parallel computing technique, a few recent works studied and developed distributed QR approaches. [36] proposed a divide-and-conquer approach to solving the QR problem in the centralized distributed system. However, to maintain the statistical accuracy of the divide and conquer estimator, it requires that each local machine stores at least $o(N^{1/2} \log N)$ data, which would be infeasible when the memory of the local machine is limited. And also, it requires that the local data are homogeneous, i.e., i.i.d. sample, which is usually violated in the network system. To address these issues, [37] proposed a linear estimator of QR (LEQR), by using a kernel smoothing technique. The LEQR approach requires the master machine to store a subset of data and find an initial QR estimator, which will be broadcast to the local machines. But LEQR focuses on the low-dimensional QR problem and cannot identify the sparsity of the QR regression coefficient. [15] studied distributed ℓ_1-regularized problem under the QR loss with sparsity recovery. To deal with the quantile loss with the ℓ_1 penalty, they transformed the responses and converted the distributed QR problem to a distributed ordinary linear regression. Then the approximate Newton method was adopted to solve the distributed linear regression estimation problem.

However, their proposed algorithm also requires that the local data are homogeneous. The above-mentioned approaches focus on the distributed system with a central master machine [1].

The most related to ours is [20], which is also addressing the QR problem in decentralized network systems. In [20], the authors proposed a decentralized subgradient algorithm. Their algorithm finds the global estimation via iteratively performing local subgradient updatings and neighboring aggregations: the machines iteratively update the local estimates with the local subgradients, then share the local estimate with their network neighbors and calculate a weighted average based on the collected neighbors' estimates. Following [20], a few other methods [38, 21] were proposed to reduce the communication cost for solving QR problem in decentralized networks. But these works adopt subgradient-based algorithms, which suffer the low convergence speed and cannot guarantee sparsity recovery.

The rest of this paper is organized as follows. Section II provides the decentralized median regression over the network and introduces the detailed algorithm. The theoretical results of the proposed estimators are established under mild regularity assumptions in Section III. Section IV reports the numerical experiments on the simulated and a real-life example is investigated in Section V. A brief summary is provided in Section VI, and the technical details are given in Supplementary Materials.

II. Methodology

In this section, we first introduce the decentralized median regression in a networked distributed system and consider the equivalent consensus form. To handle the non-smooth loss, we propose a square loss transformation method. By constructing pseudo-response and encourage sparsity, we transform it into an ℓ_1-penalized least square regression problem, which becomes much computationally feasible. Based on this idea, we develop a generalized consensus ADMM algorithm in the networked system.

A. Decentralized Median Regression with Sparsity

In this section, we consider solving the above median regression problem (2) in decentralized framework, of which there are a set of geographically dispersed computing nodes that form a network. We represent a networked distributed computing system as an undirected connected network $G = (N, E)$, where N and E are the sets of nodes and edges, respectively, with $|N| = m$ as the node number. For precisely, denote N_j to be the j-th node. In this connected network, we assume N data samples are collected locally in these m nodes $N_1, \ldots, N_m$. As described in the introduction, each node N_j collects its local data $D_j = \{(\mathbf{x}_i, y_i) : i \in [n]\}$. To be more practical, we allow the data to be heterogeneous across nodes.

The nodes have local computation capabilities and are only able to communicate with their neighbors via the edges in E. Its connectivity is modeled by an adjacency matrix $\mathbf{W}$ [39]. More precisely, $\mathbf{W}_{jk} \in \{0, 1\}$ denotes the connection between nodes N_j and N_k. Nodes N_j and N_k have a pairwise communication link if and only if the corresponding weights $\mathbf{W}_{jk} = \mathbf{W}_{kj}$ are 1; otherwise 0. For the network with no self-loops, the diagonal elements of $\mathbf{W}$ are 0s. In this work, we consider the network is connected and has no self-loops.

Our goal is to have the computational nodes *distributively* and *collaboratively* solving the network-wide median regression problem as follows:

$$\min_{\beta \in \mathbb{R}^p} L(\{(\mathbf{x}_i, y_i) \in D\}, \beta) = \min_{\beta \in \mathbb{R}^p} \frac{1}{m} \sum_{j=1}^{m} L(\{(\mathbf{x}_i, y_i) \in D_j\}, \beta), \tag{4}$$

where each local median loss function $L(\{(\mathbf{x}_i, y_i) \in D_j\}, \beta)$, $\frac{1}{n} \sum_{(\mathbf{x}_i, y_i) \in D_j} |y_i - \mathbf{x}_i^{\mathsf{T}} \beta| + \lambda_N |\beta|_1$ is only observable to node k. Note that in the loss function L, λ_N is the parameter to adjust the sparsity of the estimated median regression coefficient, which is tuned to be universal over the network G. Because in decentralized network system, there is no central controller to maintain a common model parameter during the iterations [2, 40, 41]. Thus, to solve problem (4) in a decentralized fashion, we reformulate it in the following equivalent consensus form:

$$\begin{aligned} &\min_{\mathbf{B} \in \mathbb{R}^{mp}} \frac{1}{m} \sum_{j=1}^{m} L(\{(\mathbf{x}_i, y_i) \in D_j\}, \beta^{(j)}), \\ &\text{s.t. } \beta^{(j)} = \beta^{(k)}, \quad \forall (j,k) \in E, \end{aligned} \tag{5}$$

where $\mathbf{B} = [\beta^{(1)\mathsf{T}}, \ldots, \beta^{(m)\mathsf{T}}]^{\mathsf{T}} \in \mathbb{R}^{mp}$ is the concatenated version of the parameters $\beta^{(j)}$. Here $\beta^{(j)}$ is an introduced local parameter copy at node j. In problem (5), the constraints enforce that the local parameter copy at each node is equal to those of its neighbors, hence the name "consensus". Clearly, the solution of problem (4) solves problem (5) and vice versa [2, 42, 12].

However, it is non-trivial to solve the above decentralized robust and sparse learning problem (5) because of the two factors: 1). The non-smooth median loss function makes decentralized gradient/Newton-based algorithms infeasible; 2). Composited with the sparsity ℓ_1 regularizer, there is no close-form solution for the consensus loss function. To address the above two challenges, in the following, we will propose our method named Decentralized Surrogate Median Regression (deSMR) method to solve the above problem (5). Two key components are included in our method. We first perform a transformation on the response y and surrogate the median loss with a least-square loss; Then, we adopt the generalized alternating direction method of multipliers (ADMM) framework to solve the decentralized penalized least square problem. We will show that our method enjoys a linear convergence rate and thus a low complexity.

B. Square Loss Transformation for Outer Loop

The penalized median loss in (4) follows a non-smooth loss plus non-smooth regularizer structure. To the best of our knowledge, the existing decentralized algorithms typically solve

[1] Although [37] proposed a tree-structured system to solve the QR problem in sensor network, their method still required a central root machine.

such *doubly* non-smooth loss functions by using subgradient-based algorithm [20, 38, 21]. However, one of the critical weaknesses of these works is that the subgradient-based algorithm suffers a slow *sublinear* convergence rate, which leads to the intensive computation and communication complexities in the decentralized network system. To avoid the tremendous complexity for solving the *doubly* non-smooth loss, in the following, we propose a square loss transformation to replace the median loss.

To faster the algorithm, we adopt the transformation methods in [37] to reformulate the median regression problem (2). Inspired by the Newton-Raphson method, given the initial estimator $\boldsymbol{\beta}_0$, [37] proposed the new response variable $\widetilde{y}$ as

$$\widetilde{y} = \mathbf{x}^{\mathsf{T}}\boldsymbol{\beta}_0 \quad f^{-1}(0)(\mathrm{I}[y \quad \mathbf{x}^{\mathsf{T}}\boldsymbol{\beta}_0] \quad 1/2),$$

where $f(0)$ denotes the density of ϵ at 0. With this transformation, it has been shown that the population version of the Newton-Raphson iteration to solve problem (2) induced an updated solution $\boldsymbol{\beta}_1$ as the least square solution of the problem $\mathrm{E}(\widetilde{y} \quad \mathbf{x}^{\mathsf{T}}\boldsymbol{\beta})^2$.

In network node N_j with the dataset D_j from (1), suppose that we have the initial estimator $\widehat{\boldsymbol{\beta}}_0^{(j)}$ obtained by only using D_j locally in the node N_j. We choose the corresponding density estimator of $f(0)$ to be

$$\widehat{f}^{(j)}(0) = \frac{1}{nh^{(j)}} \sum_{(\mathbf{x}_i, y_i) \in D_j} K_j\Big(\frac{y_i \quad \mathbf{x}_i^{\mathsf{T}}\widehat{\boldsymbol{\beta}}_0^{(j)}}{h^{(j)}}\Big), \tag{6}$$

where $K_j(x)$ is a kernel function which satisfies the assumption (A3) (see Section III) and $h^{(j)} \to 0$ are the bandwidths for each $j \in [m]$. The selection of bandwidths $\{h^{(j)} : j \in [m]\}$ will be discussed in our theoretical results (see Section III). Moreover, for each $(\mathbf{x}_i, y_i) \in D_j$, we construct the corresponding pseudo-response by

$$\widetilde{y}_i = \mathbf{x}_i^{\mathsf{T}}\widehat{\boldsymbol{\beta}}_0^{(j)} \quad (\widehat{f}^{(j)}(0))^{-1}(\mathrm{I}\left[y_i \quad \mathbf{x}_i^{\mathsf{T}}\widehat{\boldsymbol{\beta}}_0^{(j)}\right] \quad 1/2). \tag{7}$$

To further encourage the sparsity of the estimator, it is natural to consider the following ℓ_1-regularized problem,

$$\widehat{\boldsymbol{\beta}}_1 = \arg\min_{\boldsymbol{\beta} \in R^p} \frac{1}{2N} \sum_{(\mathbf{x}_i, y_i) \in D} (\widetilde{y}_i \quad \mathbf{x}_i^{\mathsf{T}}\boldsymbol{\beta})^2 + \lambda_{N,0}\|\boldsymbol{\beta}\|_1. \tag{8}$$

Denote $\widetilde{L}(\{(\mathbf{x}_i, y_i) \in D_j\}, \boldsymbol{\beta}) = \frac{1}{2n}\sum_{(\mathbf{x}_i, y_i) \in D_j}(\widetilde{y}_i - \mathbf{x}_i^{\mathsf{T}}\boldsymbol{\beta})^2 + \lambda_{N,0}\|\boldsymbol{\beta}\|_1$, in the decentralized framework, we focused on finding the minimizer of the following problem as surrogate estimator:

$$\min_{\boldsymbol{\beta} \in R^p} \widetilde{L}(\{(\mathbf{x}_i, y_i) \in D\}, \boldsymbol{\beta}) = \min_{\boldsymbol{\beta} \in R^p} \frac{1}{m}\sum_{j=1}^{m} \widetilde{L}(\{(\mathbf{x}_i, y_i) \in D_j\}, \boldsymbol{\beta}). \tag{9}$$

Similarly as problem (5), we reformulate the above transformed general problem into the consensus form as:

$$\widehat{\mathbf{B}}_1 = \arg\min_{\mathbf{B} \in R^{mp}} \frac{1}{2mn}\sum_{j=1}^{m}\left|\widetilde{\boldsymbol{y}}^{(j)} \quad \mathbf{X}^{(j)}\boldsymbol{\beta}^{(j)}\right|_2^2 + \lambda_{N,0}\left|\boldsymbol{\beta}^{(j)}\right|_1,$$
$$\text{s.t.} \quad \boldsymbol{\beta}^{(j)} = \boldsymbol{\beta}^{(k)}, \quad \forall (j,k) \in E, \tag{10}$$

where $\widehat{\mathbf{B}}_1 = [\widehat{\boldsymbol{\beta}}_1^{(1)\mathsf{T}}, \quad , \widehat{\boldsymbol{\beta}}_1^{(m)\mathsf{T}}]^{\mathsf{T}} = \mathbf{1}_m \quad (\frac{1}{m}\sum_{j=1}^{m}\widehat{\boldsymbol{\beta}}^{(j)}) \in R^{mp}$, $\widetilde{\boldsymbol{y}}^{(j)} = (\widetilde{y}_1^{(j)}, \ldots, \widetilde{y}_n^{(j)})$ is the concatenated vector of dimension p, and $\mathbf{X}^{(j)} = (\mathbf{x}_1^{(j)}, \ldots, \mathbf{x}_n^{(j)})$ is the concatenated covariate matrix of dimension $p \quad n$ in node N_j.

It can be shown in Theorem (2) that the convergence rate of $\widehat{\mathbf{B}}_1$ is not optimal under certain initial estimators. To further refine the estimator, we update the initial estimators iteratively by plug-in $\widehat{\mathbf{B}}_{v-1}$ in v-th iteration. Thus at v-th iteration, we consider the ℓ_1-regularized problem to be

$$\widehat{\boldsymbol{\beta}}_v = \arg\min_{\boldsymbol{\beta} \in R^p} \frac{1}{2N} \sum_{(\mathbf{x}_i, y_i) \in D} (\widetilde{y}_{i,v} \quad \mathbf{x}_i^{\mathsf{T}}\boldsymbol{\beta})^2 + \lambda_{N,v}\|\boldsymbol{\beta}\|_1. \tag{11}$$

And the corresponding consensus form of problem (11) is:

$$\widehat{\mathbf{B}}_v = \arg\min_{\mathbf{B} \in R^{mp}} \frac{1}{2mn}\sum_{j=1}^{m}\left|\widetilde{\boldsymbol{y}}_v^{(j)} \quad \mathbf{X}^{(j)}\boldsymbol{\beta}^{(j)}\right|_2^2 + \lambda_{N,v}\left|\boldsymbol{\beta}^{(j)}\right|_1,$$
$$\text{s.t.} \quad \boldsymbol{\beta}^{(j)} = \boldsymbol{\beta}^{(k)}, \quad \forall (j,k) \in E, \tag{12}$$

where $\widehat{\mathbf{B}}_v = [\widehat{\boldsymbol{\beta}}_v^{\mathsf{T}}, \quad , \widehat{\boldsymbol{\beta}}_v^{\mathsf{T}}]^{\mathsf{T}}$.

The detailed algorithm can be found in Algorithm 1.

C. Generalized Consensus ADMM for Inner Loop

With the loss transformation proposed in Section II-B, in the following, we focus on the derivation of detailed algorithm to solving the v-th iteration (12) consensus form problem. By introducing pseudo variables $\boldsymbol{T} = [\mathbf{t}^{(jk)}]$, we can equivalently write the v-th iteration (12) as:

$$\min_{\mathbf{B}, \boldsymbol{T}} \frac{1}{2mn}\sum_{j=1}^{m}\left|\widetilde{\boldsymbol{y}}_v^{(j)} \quad \mathbf{X}^{(j)}\boldsymbol{\beta}^{(j)}\right|_2^2 + \lambda_{N,v}\left|\boldsymbol{\beta}^{(j)}\right|_1,$$
$$\text{s.t.} \quad \boldsymbol{\beta}^{(j)} = \boldsymbol{\beta}^{(k)} = \mathbf{t}^{(jk)}, \quad \forall k \in N(j), j \in [m]. \tag{13}$$

Due to the consensus constraint, a natural idea is to use the popular ADMM, which has been shown to be particularly efficient for solving linear constrained minimization problem. Following classic ADMM [43], we can construct an augmented Lagrangian with penalty parameter $\tau > 0$ as:

$$L_{v,\tau}(\mathbf{B}, \boldsymbol{T}, \mathbf{U}, \mathbf{V}) = \frac{1}{2mn}\sum_{j=1}^{m}\left|\widetilde{\boldsymbol{y}}_v^{(j)} \quad \mathbf{X}^{(j)}\boldsymbol{\beta}^{(j)}\right|_2^2 + \lambda_{N,v}\left|\boldsymbol{\beta}^{(j)}\right|_1$$
$$+ \sum_{j=1}^{m}\sum_{k \in N(j)}\big(\langle \mathbf{u}^{(jk)}, \boldsymbol{\beta}^{(j)} \quad \mathbf{t}^{(jk)}\rangle + \langle \mathbf{v}^{(jk)}, \boldsymbol{\beta}^{(k)} \quad \mathbf{t}^{(jk)}\rangle$$
$$+ \frac{\tau}{2}\left|\boldsymbol{\beta}^{(j)} \quad \mathbf{t}^{(jk)}\right|_2^2 + \frac{\tau}{2}\left|\boldsymbol{\beta}^{(k)} \quad \mathbf{t}^{(jk)}\right|_2^2\big), \tag{14}$$

where $\mathbf{U} = \{\mathbf{u}^{(jk)}\}_{j=1,k}^{m,N(j)}$ and $\mathbf{V} = \{\mathbf{v}^{(jk)}\}_{j=1,k}^{m,N(j)}$ are the dual variables.According to [16], we derive the updating rules for $(\mathbf{B}, \boldsymbol{T}, \mathbf{U}, \mathbf{V})$ in the followings. For the t-th iteration in the inner loop, given primal variables $(\mathbf{B}_{v,t}, \boldsymbol{T}_{v,t})$, the dual variables can be updated with

$$\mathbf{u}_{v,t+1}^{(jk)} = \mathbf{u}_{v,t}^{(jk)} + \tau(\boldsymbol{\beta}_{v,t}^{(j)} \quad \mathbf{t}_{v,t}^{(jk)}), \tag{15a}$$
$$\mathbf{v}_{v,t+1}^{(jk)} = \mathbf{v}_{v,t}^{(jk)} + \tau(\boldsymbol{\beta}_{v,t}^{(k)} \quad \mathbf{t}_{v,t}^{(jk)}). \tag{15b}$$

Then for primal variables, we can update them by solving the corresponding subproblems:

$$\begin{aligned}
&\beta^{(j)}_{v;t+1} = \arg\min_{\beta\in R^p} \frac{1}{2mn}\|\tilde{y}^{(j)}_v - X^{(j)}\beta^{(j)}\|_2^2 + \lambda_{N;v}\|\beta^{(j)}\|_1 + \sum_{k\in N(j)} \langle u^{(jk)}_{v;t+1} + v^{(jk)}_{v;t+1}, \beta^{(j)}\rangle + \frac{\rho}{2}\|\beta^{(j)} - \theta^{(jk)}_{v;t}\|_2^2 + \frac{\rho}{2}\|\beta^{(j)} - \theta^{(kj)}_{v;t}\|_2^2 \\
&= \arg\min_{\beta\in R^p} \frac{1}{2mn}\|\tilde{y}^{(j)}_v - X^{(j)}\beta^{(j)}\|_2^2 + \lambda_{N;v}\|\beta^{(j)}\|_1 + \sum_{k\in N(j)} \langle u^{(jk)}_{v;t+1} + v^{(jk)}_{v;t+1}, \beta^{(j)}\rangle + \rho\left\|\beta^{(j)} - \frac{\theta^{(jk)}_{v;t} + \theta^{(kj)}_{v;t}}{2}\right\|_2^2, \quad (16)
\end{aligned}$$

$$\begin{aligned}
\theta^{(jk)}_{v;t+1} &= \arg\min_{t\in R^p} \langle u^{(jk)}_{v;t+1} + v^{(jk)}_{v;t+1}, t - \theta^{(jk)}\rangle + \frac{\rho}{2}\|\beta^{(j)}_{v;t+1} - t\|_2^2 + \frac{\rho}{2}\|\beta^{(k)}_{v;t+1} - t\|_2^2 \\
&= \frac{1}{2}(u^{(jk)}_{v;t+1} + v^{(jk)}_{v;t+1}) + \frac{1}{2}(\beta^{(j)}_{v;t+1} + \beta^{(k)}_{v;t+1}). \quad (17)
\end{aligned}$$

If it holds that $u^{(jk)}_{v;t} + v^{(jk)}_{v;t} = 0\ \forall j,k$, then we have $\theta^{(jk)}_{v;t} = \frac{1}{2}(\beta^{(j)}_{v;t} + \beta^{(k)}_{v;t})$, which leads to $u^{(jk)}_{v;t+1} + v^{(jk)}_{v;t+1} = (\beta^{(j)}_{v;t} + \beta^{(k)}_{v;t} - 2\theta^{(jk)}_{v;t}) = 0$. By setting $u^{(jk)}_{v;0} + v^{(jk)}_{v;0} = 0\ \forall j,k$, we can simplify the udpates (15)-(17) as

$$\begin{aligned}
u^{(jk)}_{v;t+1} &= u^{(jk)}_{v;t} + \frac{\rho}{2}(\beta^{(j)}_{v;t} - \beta^{(k)}_{v;t}), \\
v^{(jk)}_{v;t+1} &= v^{(jk)}_{v;t} + \frac{\rho}{2}(\beta^{(k)}_{v;t} - \beta^{(j)}_{v;t}), \quad (18a)
\end{aligned}$$

$$\begin{aligned}
\beta^{(j)}_{v;t+1} &= \arg\min_{\beta} \frac{1}{2mn}\|\tilde{y}^{(j)}_v - X^{(j)}\beta^{(j)}\|_2^2 + \lambda_{N;v}\|\beta^{(j)}\|_1 \\
&\quad + \sum_{k\in N(j)} \langle u^{(jk)}_{v;t+1} + v^{(jk)}_{v;t+1}, \beta^{(j)}\rangle + \frac{\rho}{2}\left\|\beta^{(j)} - \frac{\beta^{(j)}_{v;t} + \beta^{(k)}_{v;t}}{2}\right\|_2^2. \quad (18b)
\end{aligned}$$

Furthermore, by defining $p^{(j)}_{v;t} = \sum_{k\in N(j)}(u^{(jk)}_{v;t} + v^{(kj)}_{v;t})$, we can rewrite (18a) as

$$p^{(j)}_{v;t+1} = p^{(j)}_{v;t} + \sum_{k\in N(j)}(\beta^{(j)}_{v;t} - \beta^{(k)}_{v;t}), \quad (19)$$

with $p^{(j)}_0 = 0$, and (18b) as

$$\begin{aligned}
\beta^{(j)}_{v;t+1} &= \arg\min_{\beta} \frac{1}{2mn}\|\tilde{y}^{(j)}_v - X^{(j)}\beta^{(j)}\|_2^2 + \lambda_{N;v}\|\beta^{(j)}\|_1 \\
&\quad + \langle p^{(j)}_{v;t+1}, \beta^{(j)}\rangle + \sum_{k\in N(j)} \rho\left\|\beta^{(j)} - \frac{\beta^{(j)}_{v;t} + \beta^{(k)}_{v;t}}{2}\right\|_2^2. \quad (20)
\end{aligned}$$

Thus, (12) can be solved by recursively performing (19) and (20).

Unfortunately, the minimization problem in (20) has no closed-form solution with a non-orthogonal matrix $X^{(j)}$, and it commonly requires multiple optimizing iterations to find an approximated minimizer [44, 45]. To address this issue, we consider the generalized ADMM framework [46, 47, 48, 17, 44], by adding a quadratic term to (20),

$$\begin{aligned}
\beta^{(j)}_{v;t+1} &= \arg\min \frac{1}{2mn}\|\tilde{y}^{(j)}_v - X^{(j)}\beta^{(j)}\|_2^2 + \lambda_{N;v}\|\beta^{(j)}\|_1 + \langle p^{(j)}_{v;t+1}, \beta^{(j)}\rangle + \sum_{k\in N(j)} \rho\left\|\beta^{(j)} - \frac{\beta^{(j)}_{v;t} + \beta^{(k)}_{v;t}}{2}\right\|_2^2 + (\beta^{(j)} - \beta^{(j)}_{v;t})^T \\
&\quad (\tau_j I - \frac{1}{mn}X^{(j)T}X^{(j)})(\beta^{(j)} - \beta^{(j)}_{v;t}) \\
&= \arg\min \frac{1}{\omega_j}\|\beta^{(j)} - \omega_j\tau_j\beta^{(j)}_{v;t} + \omega_j\frac{1}{mn}X^{(i)T}(X^{(j)}\beta^{(j)}_{v;t} - \tilde{y}^{(j)}_v) - \omega_j p^{(j)}_{v;t+1} + \rho\sum_{k\in N(j)}(\beta^{(j)}_{v;t} + \beta^{(k)}_{v;t})\|_2^2 + \lambda_{N;v}\|\beta^{(j)}\|_1 \\
&= S_{\lambda_{2}\omega_j}(\omega_j\tau_j\beta^{(j)}_{v;t} - \omega_j\frac{1}{mn}X^{(j)T}(X^{(j)}\beta^{(j)}_{v;t} - \tilde{y}^{(j)}_v) - \omega_j p^{(j)}_{v;t+1} + \rho\sum_{k\in N(j)}(\beta^{(j)}_{v;t} + \beta^{(k)}_{v;t})), \quad (21)
\end{aligned}$$

where $\omega_j = 1/(|N(j)|\rho + \tau_j)$, $S_t(x)$ is the coordinate-wise soft-thresholding operator with $[S_t(x)]_i = (1 - t/|x_i|)_+ x_i$ and $(t)_+ = t$ if $t > 0$, 0 otherwise. To summarize, we state our proposed decentralized algorithm for solving (4) in Algorithm 1. Note that the inner loop updates in Algorithm 1 can share the same rules with the Inexact Consensus ADMM [17] for ℓ_1 penalized least square problem (13).

Algorithm 1 Decentralized Suggerated Median Regression with Generalized Consensus ADMM.

Input: Local data $D_j = \{X^{(j)}, y^{(j)}\}$ at node N_j, the number of outer iterations V and inner iterations T, kernel functions $K_j(\cdot)$, bandwidths $\{h^{(j)}_v\}_{v=1;j=1}^{V;m}$, the tunning parameters $\lambda_{0;j}$ for initial estimator and universal tunning parameters $\lambda_{N;v}$ over the network.

1: Locally compute the initial estimators $\beta^{(j)}_0$ at each node N_j:

$$b^{(j)}_0 = \arg\min_{\beta\in R^p} \frac{1}{n}\|y^{(j)} - X^{(j)T}\beta\|_1 + \lambda_{0;j}\|\beta\|_1; \quad (22)$$

2: for $v = 0, \ldots, V$ do
3: Construct pseudo responses $\tilde{y}^{(j)}_v$ as in (7) at each node N_j locally;
4: Set $\beta^{(j)}_{v;0} = b^{(j)}_v$ and $p^{(j)}_{v;0} = 0$;
5: for $t = 1, \ldots, T$ do
6: Communicate local parameter $\beta^{(j)}_{v;t}$ with neighboring nodes;
7: Update $p^{(j)}_{v;t}$ and $\beta^{(j)}_{v;t}$ with (19) and (21), respectively;
8: end for
9: Set $b^{(j)}_{v+1} = \beta^{(j)}_{v;T+1}$;
10: end for

III. Theoretical Results

For a vector $v = (v_1, \ldots, v_p)^T$, define the ℓ_1 norm $|v|_1 = \sum_{i=1}^p |v_i|$ and the ℓ_2 norm $|v|_2 = \sqrt{\sum_{i=1}^p v_i^2}$. For a matrix $A = (a_{ij}) \in R^{m\times n}$, define $|A|_1 = \max_{1\le i\le m;1\le j\le n}|a_{ij}|$ as the infinity norm, $\|A\|_F = \sqrt{\sum_{i=1}^m\sum_{j=1}^n a_{ij}^2}$ as the

relative state," *IEEE Transactions on Automatic Control*, vol. 64, no. 6, pp. 2352–2367, 2018.
[39] E. W. Weisstein, "Adjacency matrix," *https://mathworld. wolfram. com/*, 2007.
[40] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial informatics*, vol. 9, no. 1, pp. 427–438, 2012.
[41] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.
[42] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.
[43] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976.
[44] Y. Zhu, "An augmented admm algorithm with application to the generalized lasso problem," *Journal of Computational and Graphical Statistics*, vol. 26, no. 1, pp. 195–204, 2017.
[45] S. Tao, D. Boley, and S. Zhang, "Local linear convergence of ista and fista on the lasso problem," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 313–336, 2016.
[46] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," *Journal of Scientific Computing*, vol. 66, no. 3, pp. 889–916, 2016.
[47] X. Zhang, J. Liu, and Z. Zhu, "Distributed linear model clustering over networks: A tree-based fused-lasso admm approach," *arXiv preprint arXiv:1905.11549*, 2019.
[48] Y. Gu, J. Fan, L. Kong, S. Ma, and H. Zou, "Admm for high-dimensional sparse penalized quantile regression," *Technometrics*, vol. 60, no. 3, pp. 319–331, 2018.
[49] P. Yaskov, "Lower bounds on the smallest eigenvalue of a sample covariance matrix." *Electronic Communications in Probability*, vol. 19, pp. 1–10, 2014.
[50] P. Zhao and B. Yu, "On model selection consistency of lasso," *Journal of Machine Learning Research*, vol. 7, no. Nov, pp. 2541–2563, 2006.
[51] M. J. Wainwright, "Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (lasso)," *IEEE Transactions on Information Theory*, vol. 55, no. 5, pp. 2183–2202, 2009.
[52] P. Bühlmann and S. Van De Geer, *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.
[53] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.
[54] J. Fan, Y. Fan, and E. Barut, "Adaptive robust variable selection," *The Annals of Statistics*, vol. 42, no. 1, pp. 324–351, 2014.
[55] S. A. Van de Geer, "High-dimensional generalized linear models and the lasso," *Annals of Statistics*, vol. 36, no. 2, pp. 614–645, 2008.
[56] M. Redmond, "Communities and Crime Data Set ," http://archive.ics.uci.edu/ml//datasets/Communities+and+Crime, 2009.
[57] X. Yang, X. Yan, and J. Huang, "High-dimensional integrative analysis with homogeneity and sparsity recovery," *Journal of Multivariate Analysis*, vol. 174, p. 104529, 2019.