Smoothing quantile regression for a distributed system

Rong Jiang[a], Keming Yu[b,*]

[a]*Donghua University, Shanghai, 201620, People's Republic of China*
[b]*Brunel University, London UB83PH, UK*

Abstract

Quantile regression has become a popular alternative to least squares regression for providing a comprehensive description of the response distribution, and robustness against heavy-tailed error distributions. However, the nonsmooth quantile loss poses new challenges to distributed estimation in both computation and theoretical development. To address this challenge, we use a convolution-type smoothing approach and its Taylor expression to transform the nondifferentiable quantile loss function into a convex quadratic loss function, which admits a fast and scalable algorithm to perform optimization under massive and high-dimensional data. The proposed distributed estimators are both computationally and communication efficient. Moreover, only the gradient information is communicated at each iteration. Theoretically, we show that, after a certain number of iterations, the resulting estimator is statistically as efficient as the global estimator without any restriction on the number of machines. Both simulations and data analysis are conducted to illustrate the finite sample performance of the proposed methods.

Keywords: quantile regression, massive data, high-dimensional data,

*Corresponding author
Email addresses: jrtrying@dhu.edu.cn (Rong Jiang), keming.yu@brunel.ac.uk (Keming Yu)

distributed estimator

2010 MSC: 60G08, 62G20

1. Introduction

Technology advancement in applications such as e-commerce has led to massive data sets that cannot be processed with stand alone computers due to processor, memory or disk bottlenecks. Consequently, the dataset must be stored and processed on many connected computer nodes, which thereafter are referred to as a distributed system. More precisely, a distributed system refers to a large cluster of computers, which are typically connected with each other. Specifically, to obtain some estimates of the parameters of interest under a large-scale computation problem, one can compute an estimate of the parameters or some other summary of the data locally and then send this more compact piece of information to a central machine. Accordingly, a solution path can be obtained on the master node. In this work, we consider a "master-and-worker"-type distributed system, in which workers do not need to communicate with each other directly. However, they should be connected to a master node, which is a computer with outstanding capacities. The related references can be found in Yang (1997), Jiang et al. (2018), Chen et al. (2021), and so on.

In a distributed system, because data are often collected from different sources with different times and locations, the homoscedasticity assumption is often not valid (Fan et al., 2014). Moreover, heavy-tailed noise, with an infinite variance (e.g., Cauchy distribution), is prevalent in practice. For such heavy-tailed noise, most existing theories for distributed systems based on least squares will no longer be applicable. Quantile regression (QR) proposed by Koenker and Bassett (1978) is a powerful tool for learning the relationship between a scalar response and multivariate predictors in the presence of heavier tails and/or data heterogeneity, which makes QR a natural candidate as an analysis tool for distributed systems. Other references about the QR method can be found in Koenker (2005), Wang and Wang (2013), Lv et al. (2013), Yu et al. (2016), He et al. (2021), and so on.

Modern data acquisitions have facilitated the collection of massive (Alex et al., 2009) and high-dimensional data (Jiang, 2018) with complex structures. The main purpose of the paper is to provide a new estimation approach for high-dimensional QR in a distributed system and establish the theoretical results. More specifically, we consider the following linear model:

$$\mathbf{Y} = \mathbf{X}^\top \beta_0 + \varepsilon, \tag{1.1}$$

where $\mathbf{X}$ is a random vector of p-dimensional covariates, β_0 is a vector of unknown parameters of interest, and the noise ε satisfies $P(\varepsilon \leq 0|\mathbf{X}) = \tau$. Note that β_0 and ε should truly be $\beta_{0,\tau}$ and ε_τ, and we omit the subscript τ for notational convenience. Theoretically, the true parameter vector β_0 in model (1.1) solves the following minimization problem:

$$\beta_0 = \arg\min_{\beta} E\left\{\rho_\tau\left(\mathbf{Y} - \mathbf{X}^\top \beta\right)\right\}, \tag{1.2}$$

where $\rho_\tau(r) = \tau r - rI(r < 0)$ is the check loss function with $I(v)$ denoting the indicator function that takes value 1 if v is true, and zero otherwise.

In this paper, we consider the setting in which p increases with sample size n and even p can be much larger than n. Therefore, the number of available covariates is typically large, but only a small number of covariates are related to the response. By a variable selection technique, one can discover the important variables with high probability. In recent decades, various variable selection techniques have been well studied, such as the least absolute shrinkage and selection operator (LASSO) proposed by Tibshirani (1996), smooth clipped absolute deviation (SCAD) proposed by Fan and Li (2011), and minimax concave penalty (MCP) proposed by Zhang (2010). Existing approaches to penalized QR for a distributed system mostly focus on LASSO and develop corresponding algorithms (see Zhao et al. (2020), Wang and Lian (2020) and Chen et al. (2020)). Therefore, we consider the following penalized QR estimator based on LASSO:

$$\hat{\beta}_L = \arg\min_{\beta} \sum_{i=1}^{n} \rho_\tau \left(Y_i - \mathbf{X}_i^\top \beta \right) + \lambda_n \|\beta\|_1, \tag{1.3}$$

where $\{Y_i, \mathbf{X}_i\}_{i=1}^n$ are independent and identically distributed samples from $(\mathbf{Y}, \mathbf{X})$, $\|\beta\|_1$ is the ℓ_1-regularization of β, and λ_n is the regularization parameter.

For problem (1.2) under massive data sets with fixed p, Xu et al. (2020) studied a "one-shot" QR method. It only used one round of communication, in which the estimators were obtained in parallel on local machines, and then communicated and combined into a central node to form an estimator by averaging the local estimators. However, the one-shot method suffers from the following drawbacks. It requires a large sample size on each local machine, as their theories heavily rely on asymptotic expansions of certain estimators. Moreover, the number of machines K should be $Ke^{\sqrt{K}} = n$ to ensure the effectiveness of the methods, but this feature may not be desirable in distributed settings. For example, in a sensor network with a vast number of sensors, the number of machines may exceed the constraint set for the optimal rate. Furthermore, for high-dimensional regression using a penalized model, Lee et al. (2017) noticed that simple averaging does not work due to the unignorable bias size. Volgushev et al. (2019) and Chen and Zhou (2020) also studied a one-short QR method by weighted averaging of the local estimators, which requires the conditions $K = o(\sqrt{n})$ and $K = O(\tilde{n}^r)$ with $0 \leq r < 1/3$ to achieve the optimal rate of convergence, where $\tilde{n}$ is the sample size of the local machine.

For problem (1.3) under massive and high-dimensional datasets, Zhao et al. (2020) proposed a bias-correction method (van de Geer et al., 2014) to obtain a debiased estimator before taking the average. However, the debiasing step is computationally time-consuming and suffers the drawbacks of the one-short method. Wang and Lian (2020) extend the communication-efficient surrogate likelihood method proposed by Jordan et al. (2019) for distributed QR. However, their iterative algorithm cannot improve estimation accuracy. Chen et al. (2020) transformed QR loss to a least squares loss and provided a distributed estimator that is both computationally and communicatively efficient. However, their method only works when the error term ε is independent of the covariates $\mathbf{X}$, which is a highly restrictive assumption.

In view of the existing literature, one question that naturally arises from the analysis of the QR method under massive and high-dimensional datasets is as follows: can we possibly develop a communication-efficient estimation that is valid under some regularity conditions, to relax the restriction on the number of machines, sample size of local machines, and error term? Motivated by this question, our paper presents a multiround distributed algorithm for estimating β_0 in model (1.1). Note that the QR loss function $\rho_\tau(\cdot)$ is nondifferentiable and lacks strong convexity. Thus, QR remains computationally expensive for large-scale data when both the sample size and dimension are very large. To circumvent the nondifferentiability of the QR loss function, Horowitz (1998) proposed smoothing the indicator part of the check function via a kernel smoothing survival function. Chen et al. (2019) applied Horowitz's smoothing quantile regression to solve problem (1.2) under a massive dataset. However, their method requires that each local machine computes and communicates a $p \times p$-dimensional matrix to the master machine, where p is the dimension of covariance. Communication is expensive when p is very large. Moreover, their method cannot be extended to solve problem (1.3). The smoothing method proposed by Horowitz (1998) gains smoothness at the cost of convexity, which inevitably raises optimization issues. In general, computing a global minimum of a nonconvex function is intractable. To address the aforementioned issue, Fernandes et al. (2021) proposed a convolution-type smoothing method that yields a convex and twice differentiable loss function, which yields a lower mean squared error than that of the estimator in Horowitz (1998) and a more accurate Bahadur-Kiefer representation than the standard QR estimator. Since the convolution-type smoothing loss function is globally convex and locally strongly convex, the standard Newton-Raphson iteration method can be used to solve problem (1.2). However, the convolution-type smoothing loss function cannot be directly applied to (1.3) under a distributed system, because it contains the function of unknown parameters. Thus, we use a Taylor expansion of the convolution-type smoothing loss function, and then the standard QR loss can be transformed to a convex quadratic loss. Therefore, several efficient optimization methods in the optimization literature can be adopted. For example, the Projected Scaled Subgradient, sign projection (PSSsp) algorithm (Schmidt, 2010) and the active set algorithm (Solntsev et al., 2015) were used.

To summarize, we aim to make the following important contributions to the existing literature.

(1) We develop a communication-efficient estimation that turns the non-differentiable quantile loss function into a convex quadratic loss function.

(2) We show that, after a constant number of iterations, our method achieves a near-oracle rate under some regularity conditions, and then relaxes the restriction on the number of machines, sample size of the local machine and error term. The proposed method only needs one machine (master machine) with certain computing power. Other machines can perform poorly because they only need to perform simple calculations.

This paper is organized as follows. We start with a brief review of the convolution-type smoothing method and its application in distributed systems in Section 2. In Section 3, the distributed smoothing QR estimator is proposed. The variable selection method is developed in Section 4. Both simulation examples and the application on real data are given in Section 5 to illustrate the proposed procedures. We conclude this paper with a brief discussion in Section 6. All technical proofs are deferred to the Appendix.

Notation: We fix some notations that will be used throughout this paper. For a vector $\mathbf{v} = (v_1, \ldots, v_n)^\top$, define $\|\mathbf{v}\|_1 = \sum_{i=1}^n |v_i|$, $\|\mathbf{v}\|_2 =$

$\sqrt{\sum_{i=1}^{n} v_i^2}$, and $\|\mathbf{v}\|_\infty = \max_{1\le i\le n} |v_i|$. If $\mathbf{A}$ is a $p\times q$ matrix and the element of row i and column j is a_{ij}, we use $\|\mathbf{A}\|$ to denote its spectral norm, defined by $\|\mathbf{A}\| = \max_{\mathbf{v}\in S^{n-1}} \|\mathbf{A}\mathbf{v}\|_2$, where $S^{n-1} = \{\mathbf{v} \in R^n : \|\mathbf{v}\|_2 = 1\}$ is the unit sphere in R^n and $\|\mathbf{A}\|_\infty = \max_{1\le i\le p}\sum_{j=1}^{p} |a_{ij}|$. For two sequences of real numbers $\{a_n\}_{n\ge1}$ and $\{b_n\}_{n\ge1}$, $a_n \lesssim b_n$ denotes $a_n \le Cb_n$ for some constant $C > 0$, $a_n \gtrsim b_n$ if $b_n \lesssim a_n$, and $a_n \asymp b_n$ if $a_n \lesssim b_n$ and $b_n \lesssim a_n$.

2. Smoothing quantile regression

2.1. Standard smoothing quantile regression

Theoretically, the true parameter vector β_0 in model (1.1) solves the following minimization problem:

$$\beta_0 = \arg\min_{\beta} E\left[\rho_\tau\left\{e(\beta)\right\}\right] = \arg\min_{\beta} \int \rho_\tau(t)dF(t,\beta), \tag{2.1}$$

where $e(\beta) = \mathbf{Y} - \mathbf{X}^\top\beta$ and $F(t,\beta) = P(e(\beta) \le t)$. With regard to (2.1), the QR estimator of β_0 can be solved by minimizing the following objective function:

$$\frac{1}{n}\sum_{i=1}^{n} \rho_\tau\left\{e_i(\beta)\right\} = \int \rho_\tau(t)d\hat{F}(t,\beta), \tag{2.2}$$

where $e_i(\beta) = Y_i - \mathbf{X}_i^\top\beta$ and $\hat{F}(\cdot,\beta)$ denotes the empirical distribution function of $e_i(\beta)$. For large-scale data when both n and p are large, we apply kernel smoothing (Fernandes et al., 2021) to the empirical objective function in (2.2) to address the nondifferentiability and lack of strong convexity of the loss function, and then have

$$\begin{aligned}\hat{\beta} &= \arg\min_{\beta} S_h(\beta) \equiv \arg\min_{\beta} \int \rho_\tau(t)d\hat{F}_h(t,\beta)\\ &= \arg\min_{\beta}\left[(1-\tau)\int_{-\infty}^{0} \hat{F}_h(t,\beta)dt + \tau\int_{0}^{+\infty}\{1-\hat{F}_h(t,\beta)\}dt\right],\end{aligned} \tag{2.3}$$

where $\hat{F}_h(t,\beta) = \int_{-\infty}^{t} \hat{f}_h(u,\beta)du$, $\hat{f}_h(u,\beta) = n^{-1}\sum_{i=1}^{n} K_h\left(u - e_i(\beta)\right)$, $K_h(\cdot) = K(\cdot/h)/h$, $K(\cdot)$ is a smooth kernel function and h is a bandwidth. Now, $S_h(\beta)$ is twice continuously differentiable with the gradient and Hessian matrix

$$S_h^{(1)}(\beta) = n^{-1}\sum_{i=1}^{n} \mathbf{X}_i \left\{\tilde{K}(-e_i(\beta)/h) - \tau\right\},$$
$$S_h^{(2)}(\beta) = n^{-1}\sum_{i=1}^{n} \mathbf{X}_i\mathbf{X}_i^{\top} K_h(-e_i(\beta)),$$

respectively, where $\tilde{K}(t) = \int_{-\infty}^{t} K(u)du$. Moreover, provided that the kernel function $K(\cdot)$ is nonnegative, $S_h(\beta)$ is a convex function. Below we list the explicit expressions of the check functions in (2.3) under several commonly used kernels.

(1) Uniform kernel $K(u) = (1/2)I(|u| \le 1)$:

$$\begin{aligned} S_h(\beta) =& \frac{1}{n}\sum_{i=1}^{n}\left[\frac{h}{4}\{e_i^2(\beta) + h^2\}I\{|e_i(\beta)| \le h\}\right. \\ &\left. + \frac{1}{2}|e_i(\beta)|I\{|e_i(\beta)| > h\} + (\tau - 1/2)e_i(\beta)\right]; \end{aligned} \tag{2.4}$$

(2) Epanechnikov kernel $K(u) = (3/4)(1 - u^2)I(|u| \le 1)$:

$$\begin{aligned} S_h(\beta) =& \frac{1}{n}\sum_{i=1}^{n}\left[\frac{h}{16}\{6 + 6e_i^2(\beta) - e_i^4(\beta)\}I\{|e_i(\beta)| \le h\}\right. \\ &\left. + \frac{1}{2}|e_i(\beta)|I\{|e_i(\beta)| > h\} + (\tau - 1/2)e_i(\beta)\right]; \end{aligned} \tag{2.5}$$

(3) Gaussian kernel $K(u) = (2\pi)^{-1/2}\exp(-u^2/2)$:

$$S_h(\beta) = \frac{1}{n}\sum_{i=1}^{n}\left[h(2\pi)^{-1/2}\exp(-e_i^2(\beta)/(2h^2)) - e_i(\beta)\Phi(-e_i(\beta)/h) + \tau e_i(\beta)\right]. \tag{2.6}$$

2.2. Smoothing quantile regression for a distributed system

Now let us discuss how to use (2.3) to develop a distributed estimator. Suppose that n observations are distributed across K local machines. Define $M = \{1, \ldots, n\}$ to be all sample observations. Decompose $M = \cup_{k=1}^{K} M_k$, where M_k collects the observations distributed to the kth worker, $|M_k| = n_k$ and $n = \sum_{k=1}^{K} n_k$.

Note that under a distributed system, each machine cannot communicate $S_h(\beta)$ in (2.3) to the master machine, due to the functionals of unknown parameter β, such as $I\{|e_i(\beta)| > h\}$ in (2.4) and (2.5) or $\Phi(-e_i(\beta)/h)$ in (2.6). We then use a Taylor expansion of $S_h(\beta)$ around an initial estimator $\hat{\beta}^0$ of β_0. This yields:

$$\begin{aligned} S_h(\beta) =& S_h(\hat{\beta}^0) + (\beta - \hat{\beta}^0)^\top S_h^{(1)}(\hat{\beta}^0) + \frac{1}{2}(\beta - \hat{\beta}^0)^\top S_h^{(2)}(\hat{\beta}^0)(\beta - \hat{\beta}^0) + o_p(\|\beta - \hat{\beta}^0\|_2^2), \\ =& \tilde{S}_h(\beta, \hat{\beta}^0) + o_p(\|\beta - \hat{\beta}^0\|_2^2), \end{aligned}$$

where

$$\tilde{S}_h(\beta, \hat{\beta}^0) = S_h(\hat{\beta}^0) + (\beta - \hat{\beta}^0)^\top S_h^{(1)}(\hat{\beta}^0) + \frac{1}{2}(\beta - \hat{\beta}^0)^\top S_h^{(2)}(\hat{\beta}^0)(\beta - \hat{\beta}^0). \tag{2.7}$$

Note that (2.7) is a convex quadratic loss function; thus, it is convenient to apply for a distributed system. Under a distributed system, $S_h^{(1)}(\hat{\beta}^0)$ and $S_h^{(2)}(\hat{\beta}^0)$ can be rewritten as

$$S_h^{(1)}(\hat{\beta}^0) = \frac{1}{n}\sum_{k=1}^{K} n_k S_{h,k}^{(1)}(\hat{\beta}^0), \quad S_h^{(2)}(\hat{\beta}^0) = \frac{1}{n}\sum_{k=1}^{K} n_k S_{h,k}^{(2)}(\hat{\beta}^0), \tag{2.8}$$

where $S_{h,k}^{(1)}(\hat{\beta}^0) = n_k^{-1} \sum_{i \in M_k} \mathbf{X}_i \left\{ \tilde{K}(-e_i(\hat{\beta}^0)/h) - \tau \right\}$ and $S_{h,k}^{(2)}(\hat{\beta}^0) = n_k^{-1} \sum_{i \in M_k} \mathbf{X}_i \mathbf{X}_i^\top K_h(-e_i(\hat{\beta}^0))$.

From (2.7) and (2.8), we can see that each local machine computes and communicates a p-dimensional vector $S_{h,k}^{(1)}(\hat{\beta}^0)$, and a $p \times p$-dimensional matrix $S_{h,k}^{(2)}(\hat{\beta}^0)$ to the first machine M_1 (master machine). However, when $p \to \infty$, $p \times p$ is very large. The burden of communication is heavy. Note that (2.7) can be written as

$$\begin{aligned}
\tilde{S}_h(\beta, \hat{\beta}^0) =& S_h(\hat{\beta}^0) + (\beta - \hat{\beta}^0)^\top S_h^{(1)}(\hat{\beta}^0) + \frac{1}{2}(\beta - \hat{\beta}^0)^\top S_{h_1,1}^{(2)}(\hat{\beta}^0)(\beta - \hat{\beta}^0) \\
&+ \frac{1}{2}(\beta - \hat{\beta}^0)^\top \left\{ S_h^{(2)}(\hat{\beta}^0) - S_{h_1,1}^{(2)}(\hat{\beta}^0) \right\} (\beta - \hat{\beta}^0) \\
=& \bar{S}_{h,h_1}(\beta, \hat{\beta}^0) + C + o_p\left(\|\beta - \hat{\beta}^0\|_2^2 \right),
\end{aligned}$$

where $S_{h_1,1}^{(2)}(\hat{\beta}^0) = n_1^{-1} \sum_{i \in M_1} \mathbf{X}_i \mathbf{X}_i^\top K_{h_1}(-e_i(\hat{\beta}^0))$ and h_1 are based on the first sample, $C = S_h(\hat{\beta}^0) + (\hat{\beta}^0)^\top S_h^{(1)}(\hat{\beta}^0) + 1/2(\hat{\beta}^{0\top})^\top S_{h_1,1}^{(2)}(\hat{\beta}^0)\hat{\beta}^0$ is constant which is independent of β, and

$$\bar{S}_{h,h_1}(\beta, \hat{\beta}^0) = \beta^\top \left\{ S_h^{(1)}(\hat{\beta}^0) - S_{h_1,1}^{(2)}(\hat{\beta}^0)\hat{\beta}^0 \right\} + \frac{1}{2}\beta^\top S_{h_1,1}^{(2)}(\hat{\beta}^0)\beta. \tag{2.9}$$

The last identity in the equation is due to $\|S_h^{(2)}(\hat{\beta}^0) - S_{h_1,1}^{(2)}(\hat{\beta}^0)\| = o_p(1)$, which is detailed in the proof of Theorem 3.1 in the Appendix.

By now we have transformed the nondifferentiable quantile loss function (2.2) into a convex quadratic loss function (2.9), which admits a fast and scalable algorithm to perform optimization under massive and high-dimensional data. Moreover, the communication complexity is p only, as there is no need to communicate the $(K-1) \times p \times p$ sample covariance matrixes $\{S_{h,k}^{(2)}(\hat{\beta}^0)\}_{k=2}^K$.

3. Distributed smoothing QR estimator

Section 2 shows that the estimation of β_0 in model (1.1) can be implemented by solving quadratic optimization (2.9) to obtain

$$\hat{\beta}_H^1 = \arg\min_{\beta} \bar{S}_{h,h_1}(\beta, \hat{\beta}^0) = \hat{\beta}^0 - \left\{ S_{h_1,1}^{(2)}(\hat{\beta}^0) \right\}^{-1} S_h^{(1)}(\hat{\beta}^0). \tag{3.1}$$

To establish the asymptotic properties of the proposed estimator, the following technical conditions are imposed.

C1. The kernel function $K(\cdot)$ is a symmetric, bounded and nonnegative function that integrates into one, that is, $K(u) = K(-u)$, $0 \leq K(u) \leq \infty$ for all $u \in R$ and $\int K(u)du = 1$. In addition, $\min_{|u|\leq 1} K(u) > 0$ and $\int u^2 K(u)du < \infty$.

C2. There exists $l_2 \geq l_1 > 0$ such that $l_1 \leq f_{\varepsilon|\mathbf{X}}(0) \leq l_2$ almost surely (for all X), where $f_{\varepsilon|\mathbf{X}}(\cdot)$ is the conditional density function of ε given $\mathbf{X}$. Moreover, there exists a constant $l_3 > 0$ such that $\left| f_{\varepsilon|\mathbf{X}}(t) - f_{\varepsilon|\mathbf{X}}(0) \right| \leq l_3|t|$ for all $t \in R$ almost surely.

C3. The random vector $\mathbf{X}$ is sub-exponential: there exist $\tilde{c}_1 \geq 0$ such that

$$P\left(|\mathbf{X}^\top \Sigma^{-1/2}\delta| \geq \tilde{c}_1 \|\delta\|_2 t \right) \leq 2\exp(-t),$$

for all $\delta \in R^p$ and $t \geq 0$, where $\Sigma = E(\mathbf{X}\mathbf{X}^\top)$ is a positive definite matrix. Moreover, assume that $0 < \Lambda_{\min}(\Sigma) \leq \Lambda_{\max}(\Sigma) < \infty$, where $\Lambda_{\min}(\Sigma)$ and $\Lambda_{\max}(\Sigma)$ are the smallest and largest eigenvalues of Σ, respectively.

C4. The random vector $\mathbf{X}$ is sub-Gaussian: there exist $\tilde{c}_2 \geq 0$ such that

$$P\left(|\mathbf{X}^\top \Sigma^{-1/2}\delta| \geq \tilde{c}_2 \|\delta\|_2 t \right) \leq 2\exp(-t^2/2),$$

for all $\delta \in R^p$ and $t \geq 0$.

Remark 3.1. *Condition* **C1** *is a mild condition on $K(\cdot)$ for smoothing approximation. For example, for Gaussian kernel $K(u) = (2\pi)^{-1/2}\exp(-u^2/2)$, it is easy to see that it satisfies condition* **C1**. *Condition* **C2** *is a regular condition on the smoothness of the conditional density function $f_{\varepsilon|\mathbf{X}}(\cdot)$.* **C3** *assumes a sub-exponential condition on the random covariates, which encompasses the bounded case considered by Fernandes et al. (2021). Moreover, the conditions* **C2** *and* **C3** *ensure that $\mathbf{D} = E\{f_{\varepsilon|\mathbf{X}}(0)\mathbf{X}\mathbf{X}^\top\}$ is positive definite, which means that $\mathbf{D}^{-1}$ exists. Conditions* **C3** *and* **C4** *are the standard irrepresentable condition. Condition* **C4** *is slightly more stringent than Conditions* **C3**. *If* **X** *is Gaussian, then* **C3** *and* **C4** *hold. Heavier-tailed distributions of* **X** *are excluded here so that we can expect standard rates of convergence for the quantile regression estimates. Conditions* **C2**-**C4** *are standard conditions, which are commonly used in high-dimensional quantile regression (see He et al. (2020), Wang and Lian (2020) and Chen et al. (2020)).*

We are ready to present the theoretical results for our distributed smoothing QR estimator.

Theorem 3.1. *Suppose we have an initial estimator $\hat{\beta}^0$ with $\|\hat{\beta}^0 - \beta_0\|_2 = O_p(a_n)$ with $a_n \asymp n^{-c_1}$ for some constant $c_1 > 0$ and $a_n \gtrsim \sqrt{p/n}$. The dimension $p \asymp n^{c_2}$ for some constant $0 < c_2 < 1$. $h_1 \to 0$ and $(p/n)^{1/2} \lesssim h \lesssim (p/n)^{1/4}$. Assume that conditions* **C1**-**C3** *are satisfied, we can obtain*

$$\|\hat{\beta}_H^1 - \beta_0\|_2 = O_p\left(\sqrt{\frac{p}{n}} + a_n\left\{\sqrt{\frac{p\log n}{nh}} + \sqrt{\frac{p\log n_1}{n_1 h_1}} + a_n + h + h_1\right\}\right). \tag{3.2}$$

Note that, (3.2) means that one round of aggregation enables a refinement of the estimator with its bias reducing from $O_p(a_n)$ to $o_p(a_n)$, when

$p \log n_1/(n_1 h_1) = o(1)$. Therefore, an iterative refinement of the initial estimator will successively improve the estimation accuracy. Now we are ready to present the theoretical results for the estimator in (3.1) with multiple rounds of aggregations.

Theorem 3.2. *Suppose that all conditions in Theorem 3.1 hold,* $\|\hat{\beta}^0 - \beta_0\|_2 = O_p(\sqrt{p/n_1})$ *with* $n_1 = n^r, 0 < r \leq 1$, *and* $p \asymp n^{c_3}$ *with* $0 < c_3 < r$, $h_1 \asymp (p \log n_1/n_1)^{1/3}$. *We have*

$$\|\hat{\beta}_H^q - \beta_0\|_2 = O_p\left(\sqrt{\frac{p}{n}} + \sqrt{\frac{p}{n_1}}\left(\frac{p \log n_1}{n_1}\right)^{q/3}\right). \tag{3.3}$$

The initial estimator of β_0 can be obtained by minimizing (2.2) based on the first machine M_1, thus we have $\|\hat{\beta}^0 - \beta_0\|_2 = O_p(\sqrt{p/n_1})$ under some regularity conditions; see He and Shao (2000). It can be shown that when the iteration number q is sufficiently large,

$$q \geq Q_1 \equiv \varphi(1 + 3/2 \log(n_1/n)/\log(p \log n_1/n_1)), \tag{3.4}$$

where $\varphi(\cdot)$ is an integer up function, the second term in (3.3) is dominated by the first term, and the convergence rate in (3.3) becomes

$$\|\hat{\beta}_H^q - \beta_0\|_2 = O_p\left(\sqrt{p/n}\right),$$

which is the convergence rate of the smoothed QR estimator by (2.3) in a single machine setup; see Theorem 3.1 in He et al. (2020). Next, we have the asymptotic distribution of $\hat{\beta}_H^q$ as follows.

Theorem 3.3. *Assume that Conditions* **C1-C4** *are satisfied.* $\|\hat{\beta}^0 - \beta_0\|_2 = O_p(\sqrt{p/n_1})$ *with* $n_1 = n^r, 0 < r \leq 1$, $p \asymp n^{c_4}$ *with* $0 < c_4 < \min(3/8, r)$,

$h \asymp (p/n)^{2/5}$, $h_1 \asymp (p \log n_1/n_1)^{1/3}$, and $n \to \infty$. The number of iterations q satisfies (3.4). Then, for any $\alpha \in R^p$ with $\alpha \neq \mathbf{0}$, we have

$$\frac{n^{1/2}\alpha^\top(\hat{\beta}_H^q - \beta_0)}{\sqrt{\alpha^\top \mathbf{D}^{-1}\Sigma\mathbf{D}^{-1}\alpha}} \xrightarrow{L} \mathcal{N}\left(0, \tau(1-\tau)\right),$$

where $\xrightarrow{L}$ stands for convergence in the distribution.

Theorem 3.3 shows that $\hat{\beta}_H^q$ achieves the same asymptotic efficiency as the estimator computed directly on all the samples; see Theorem 3.3 in He et al. (2020). Note that in a common scenario when $n_1 = n^r, 0 < r \leq 1$, the number of iterations Q_1 in (3.4) is bounded by a constant. It is also important to note that the required number of rounds Q_1 is usually quite small. Some examples are present in Table 1.

Based on the above analysis, we now introduce a distributed smoothing QR method for estimating β_0. The procedure is summarized in Algorithm 1.

Algorithm 1: Distributed smoothing QR method.

```
Input: The number of iterations $Q_1$, the quantile level $\tau$, kernel function $K(\cdot)$, and bandwidths $h$ and $h_1$;
1: Calculate the initial estimator $\hat{\beta}^0$ based on machine $M_1$:
       $\hat{\beta}^0 = \arg\min_{\beta} \sum_{i \in M_1} \rho_\tau\left(Y_i - \mathbf{X}_i^\top \beta\right);$     (3.5)
2: for $q = 0 : Q_1 - 1$ do
3:     Transmit the current iterate $\hat{\beta}_H^q$ to local machines $\{M_k\}_{k=1}^K$;
4:     for $k = 1 : K$ do
           Compute the local gradient $S_{h,k}^{(1)}(\hat{\beta}_H^q)$ at machine $M_k$;
           Transmit the local gradient $S_{h,k}^{(1)}(\hat{\beta}_H^q)$ to machine $M_1$;
```

```
5:      end for
6:      Calculate the global gradient $S_h^{(1)}(\hat{\beta}_H^q) = n^{-1}\sum_{k=1}^K n_k S_{h,k}^{(1)}(\hat{\beta}_H^q)$,
        and $S_{h_1,1}^{(2)}(\hat{\beta}_H^q)$ in machine $M_1$. Compute the estimator:
            $\hat{\beta}_H^{q+1} = \hat{\beta}^0 - \left\{S_{h_1,1}^{(2)}(\hat{\beta}_H^q)\right\}^{-1} S_h^{(1)}(\hat{\beta}_H^q);$    (3.6)
7:  end for
Output: The final estimator is $\hat{\beta}_H^{Q_1}$.
```

It is worthwhile to note that there are many methods to solve (3.5). In our experiments, we adopt the coordinate descent algorithm (Pietrosanu et al., 2020) to obtain $\hat{\beta}^0$. Moreover, (3.6) may be computationally expensive for calculating the inverse matrix of $S_{h_1,1}^{(2)}(\hat{\beta}_H^q)$ under a large p and may need modifications if $S_{h_1,1}^{(2)}(\hat{\beta}_H^q)$ is ill-conditioned, especially when h_1 is too small. In this situation, we could use a Barzilai-Borwein algorithm (Barzilai and Borwein, 1988) to obtain a simple approximation of the inverse matrix of $S_{h_1,1}^{(2)}(\hat{\beta}_H^q)$.

4. Distributed penalized smoothing QR estimator

To further encourage the sparsity of the estimator which is discussed in Section 2, it is natural to consider the following ℓ_1-regularized problem:

$$\begin{aligned}\hat{\beta}_L^1 &= \arg\min_\beta \left\{\bar{S}_{h,h_1}(\beta, \hat{\beta}_L^0) + \lambda_n\|\beta\|_1\right\}\\ &= \arg\min_\beta \left[\beta^\top \left\{S_h^{(1)}(\hat{\beta}_L^0) - S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\hat{\beta}_L^0\right\} + \frac{1}{2}\beta^\top S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\beta + \lambda_n\|\beta\|_1\right],\end{aligned} \tag{4.1}$$

where $\hat{\beta}_L^0$ is defined as an initial estimator of β_0 under the LASSO penalty. By (4.1), the estimation of the high-dimensional sparse β_0 can be implemented by solving a penalized quadratic optimization instead of the penalized QR optimization. Therefore, several efficient optimization methods in the optimization literature, such as the PSSsp algorithm and the active set algorithm can be adopted. Next, we present the main theoretical results.

Theorem 4.1. *Suppose that we have an initial estimator $\hat{\beta}_L^0$ with $\|\hat{\beta}_L^0 - \beta_0\|_2 = O_p(\tilde{a}_n)$ with $\tilde{a}_n \asymp n^{-c_5}$ for some constant $c_5 > 0$. The dimension $p = o(\exp(n_1))$ and $(s/n)^{1/2} \lesssim h \lesssim (\log(n \vee p)/n)^{1/4}$, where s is the sparsity of β_0, $s = \sum_{j=0}^{p} I(\beta_{0,j} \neq 0)$ and $n \vee p = \max(n, p)$. Take*

$$\lambda_n = \bar{C}\left\{\sqrt{\frac{\log(n \vee p)}{n}} + \tilde{a}_n\left(\sqrt{\frac{s\log(n \vee p)}{nh}} + \sqrt{\frac{s\log(n_1 \vee p)}{n_1 h_1}} + \tilde{a}_n + h + h_1\right)\right\}, \tag{4.2}$$

with $\bar{C}$ being a sufficiently large constant. Under conditions **C1**, **C2** *and* **C4**, *we can obtain*

$$\|\hat{\beta}_L^1 - \beta_0\|_2 = O_p\left(\sqrt{s}\lambda_n\right). \tag{4.3}$$

Suppose that $\tilde{a}_n \asymp \sqrt{s\log(n_1 \vee p)/n_1}$ and $h_1 \asymp (s\log(n_1 \vee p)/n_1)^{1/3}$, then (4.2) is equal to

$$\lambda_n = O_p\left(\sqrt{\frac{\log(n \vee p)}{n}} + \tilde{a}_n\left(\frac{s\log(n_1 \vee p)}{n_1}\right)^{1/3}\right),$$

and (4.3) can be written as

$$\|\hat{\beta}_L^1 - \beta_0\|_2 = O_p\left(\sqrt{\frac{s\log(n \vee p)}{n}} + \tilde{a}_n\sqrt{s}\left(\frac{s\log(n_1 \vee p)}{n_1}\right)^{1/3}\right),$$

For the estimator with its bias reducing from $O_p(\widetilde{a}_n)$ to $o_p(\tilde{a}_n)$, assume $\sqrt{s}\,(s\log(n_1 \vee p)/n_1)^{1/3} = o(1)$. Now, we are ready to present the theoretical results for the estimator in (4.1) with multiple rounds of aggregations.

Theorem 4.2. *Suppose we have an initial estimator $\hat{\beta}_L^0$ with $\|\hat{\beta}_L^0 - \beta_0\|_2 = O_p(\sqrt{s\log(n_1 \vee p)/n_1})$ with $n_1 = n^{c_6}, 0 < c_6 \leq 1$. Let $p = o(\exp(n^{c_6}))$, $s = o((n_1/\log(n_1 \vee p))^{2/5})$, $h_1 \asymp (s\log(n_1 \vee p)/n_1)^{1/3}$ and $(s/n)^{1/2} \lesssim h \lesssim (\log(n \vee p)/n)^{1/4}$. Take*

$$\lambda_n^q = \bar{C}\left(\sqrt{\frac{\log(n \vee p)}{n}} + s^{5q/6}\left(\frac{\log(n_1 \vee p)}{n_1}\right)^{(2q+3)/6}\right),$$

with $\bar{C}$ being a sufficiently large constant. Under conditions **C1**, **C2** *and* **C4**, *we can obtain*

$$\|\hat{\beta}_L^q - \beta_0\|_2 = O_p\left(\sqrt{\frac{s\log(n \vee p)}{n}} + s^{(5q+3)/6}\left(\frac{\log(n_1 \vee p)}{n_1}\right)^{(2q+3)/6}\right). \quad (4.4)$$

The initial estimator β_0 can also be obtained by minimizing (4.1) based on M_1, thus we have $\|\hat{\beta}_L^0 - \beta_0\|_2 = O_p(\sqrt{s\log(n_1 \vee p)/n_1})$ under some regularity conditions; see Theorem 2 in Belloni and Chernozhukov (2011). It can be shown that when the iteration number q is sufficiently large,

$$q \geq Q_2 \equiv \varphi(1 + 3\log(\log(n \vee p)^{n_1}/\log(n_1 \vee p)^n)/\log(s^5\log^2(n_1 \vee p)/n_1^2)), \quad (4.5)$$

the second term in (4.4) is dominated by the first term, and the convergence rate in (4.4) becomes

$$\|\hat{\beta}_L^q - \beta_0\|_2 = O_p\left(\sqrt{s\log(n \vee p)/n}\right),$$

which is the convergence rate of the ℓ_1-regularized quantile regression estimator in a single machine setup; see Theorem 2 in Belloni and Chernozhukov (2011).

The following theorems provide results on support recovery of the proposed estimators in (4.1). Let $T = \{j \in \{1, \ldots, p\} : |\beta_{0,j}| > 0\}$ is the support of β_0 and $\hat{T}^1 = \{j \in \{1, \ldots, p\} : |\hat{\beta}_{L,j}^1| > 0\}$.

Theorem 4.3. *Suppose that all conditions in Theorem 4.1 hold.*
(i) We have $\hat{T}^1 \subseteq T$ with probability tending to one;
(ii) In addition, suppose that for a sufficiently large constant $\tilde{C} > 0$,

$$\begin{aligned}\min_{j \in T} |\beta_{0,j}| \geq & \tilde{C} \|\mathbf{D}_{T\times T}^{-1}\|_\infty \Big\{ \sqrt{\frac{\log(n \vee p)}{n}} \\ & + \tilde{a}_n \Big(\sqrt{\frac{s \log(n \vee p)}{nh}} + \sqrt{\frac{s \log(n_1 \vee p)}{n_1 h_1}} + \tilde{a}_n + h + h_1 \Big) \Big\},\end{aligned}$$

we have $\hat{T}^1 = T$ with probability tending to one.

The support recovery result for $\hat{\beta}_L^q$ can be present in the following theorem. Denote $\hat{T}^q = \{j \in \{1, \ldots, p\} : |\hat{\beta}_{L,j}^q| > 0\}$.

Theorem 4.4. *Suppose that all conditions in Theorem 4.2 hold.*
(i) We have $\hat{T}^q \subseteq T$ with probability tending to one;
(ii) In addition, suppose that for a sufficiently large constant $\tilde{C} > 0$,

$$\min_{j \in T} |\beta_{0,j}| \geq \tilde{C} \|\mathbf{D}_{T\times T}^{-1}\|_\infty \left\{ \sqrt{\frac{\log(n \vee p)}{n}} + s^{5q/6} \left(\frac{\log(n_1 \vee p)}{n_1} \right)^{(2q+3)/6} \right\}, \tag{4.6}$$

we have $\hat{T}^q = T$ with probability tending to one.

When q satisfies (4.5), the condition (4.6) reduces to $\min_{j \in T} |\beta_{0,j}| \geq \tilde{C} \|\mathbf{D}_{T\times T}^{-1}\|_\infty \sqrt{\log(n \vee p)/n}$, which matches the rate of the lower bound for the "beta-min" condition in LASSO in a single machine setting (Wainwright, 2009).

Based on the above analysis, we introduce a distributed ℓ_1-regularized smoothing QR method for estimating β_0.

Algorithm 2: Distributed penalized smoothing QR estimator

```
Input: The number of iterations $Q_2$, the quantile level $\tau$, kernel function $K(\cdot)$, the regularization parameter $\lambda_n$, and bandwidths $h$ and $h_1$;
1: Calculate the initial estimator $\hat{\beta}_L^0$ based on $M_1$:
       $$\hat{\beta}_L^0 = \arg\min_{\beta} \sum_{i \in M_1} \rho_\tau \left( Y_i - \mathbf{X}_i^\top \beta \right) + \lambda_n \|\beta\|_1; \qquad (4.7)$$
2: for $q = 0 : Q_2 - 1$ do
3:     Transmit the current iterate $\hat{\beta}_L^q$ to local machines $\{M_k\}_{k=1}^K$;
4:     for $k = 1 : K$ do
           Compute the local gradient $S_{h,k}^{(1)}(\hat{\beta}_L^q)$ at machine $M_k$;
           Transmit the local gradient $S_{h,k}^{(1)}(\hat{\beta}_L^q)$ to machine $M_1$;
5:     end for
6:     Calculate the global gradient $S_h^{(1)}(\hat{\beta}_L^q) = n^{-1} \sum_{k=1}^K n_k S_{h,k}^{(1)}(\hat{\beta}_L^q)$,
       and $S_{h_1,1}^{(2)}(\hat{\beta}_L^q)$ in machine $M_1$. Compute the estimator:
       $$\hat{\beta}_L^{q+1} = \arg\min_{\beta} \left\{ \bar{S}_{h,h_1}(\beta, \hat{\beta}_L^q) + \lambda_n \|\beta\|_1 \right\}; \qquad (4.8)$$
7: end for
Output: The final estimator is $\hat{\beta}_L^{Q_2}$.
```

It is worthwhile to note that optimization problems (4.7) and (4.8) have been extensively studied in the optimization literature, and several efficient optimization methods have been developed. In our experiments, we adopt the coordinate descent algorithm (Pietrosanu et al., 2020) to obtain $\hat{\beta}_L^0$ in (4.7), and the PSSsp algorithm (Schmidt, 2010) to solve (4.8). Moreover, the number of iterations Q_2 in (4.5) is bounded by a constant and is usually quite small. Some examples of Q_1 and Q_2 are present in Table 1, and we take $s = (n_1/\log(n_1 \vee p))^{1/3}$ for Q_2.

Table 1: The numbers of iterations Q_1 and Q_2 under different n, n_1 and p.

n	p	n_1	Q_1	p	n_1	Q_1	p	n_1	Q_2	p	n_1	Q_2
10^{10}	10^2	10^4	10	10^3	10^5	10	10^5	10^5	13	10^{15}	10^5	14
		10^5	5		10^6	5		10^6	9		10^6	10
		10^6	4		10^7	3		10^7	6		10^7	6
		10^7	3		10^8	2		10^8	4		10^8	4
10^{15}	10^3	10^6	9	10^4	10^6	17	10^{10}	10^7	14	10^{20}	10^7	15
		10^7	6		10^7	8		10^8	11		10^8	11
		10^8	4		10^8	5		10^9	8		10^9	9
		10^9	3		10^{10}	3		10^{11}	2		10^{11}	2
10^{20}	10^3	10^6	13	10^5	10^8	12	10^{10}	10^8	17	10^{30}	10^8	19
		10^7	8		10^9	8		10^9	14		10^9	15
		10^8	6		10^{10}	6		10^{10}	12		10^{10}	13
		10^{10}	4		10^{12}	4		10^{12}	4		10^{12}	4

5. Numerical studies

In this section, we first use Monte Carlo simulation studies to assess the finite sample performance of the proposed procedures and then demonstrate the application of the proposed methods with two real data analyses. All programs are written in R code. The Gaussian kernel $K(u) = (\sqrt{2\pi})^{-1}\exp(-u^2/2)$ is used in this section.

5.1. Simulation example 1

In this section, we compare the proposed algorithm (3.1) in Section 3 with the following existing methods under full data analysis: interior points algorithm (IP) in Portnoy and Koenker (1997), the ADMM algorithm in Pietrosanu et al. (2020) and the Conquer algorithm in He et al. (2020).

We generate data from the following linear model:

$$Y_i = \mathbf{X}_i^\top \beta_0 + \varepsilon_i, \quad i = 1, \ldots, n,$$

where $\mathbf{X}_i = (1, X_{i1}, \ldots, X_{ip})^\top$ is a $(p+1)$-dimensional covariate vector and $(X_{i1}, \ldots, X_{ip})$ is drawn from a multivariate normal distribution $N(0, \Sigma)$. The covariance matrix Σ is constructed by $\Sigma_{ij} = 0.5^{i-j}$ for $1 \leq i, j \leq p$. Different dimensions $p = 10$, 100 and sample sizes $n = 10^2$, 10^3, 10^4, 10^5, 10^6 are considered in this section. The true value of the parameter is $\beta_0 = (1, \ldots, 1)$ and quantile level $\tau = 0.5$. Two distributions for the model error ε are considered as follows: a standard normal distribution (N(0,1)), and a t distribution with freedom of 5 (t(5)). In this section, we choose the least squares estimation as the initial estimation and $h = (p/n)^{2/5}$ for simplicity.

To evaluate the performance of the four methods, we calculate the mean squared error (MSE) $\|\hat{\beta} - \beta_0\|_2$ and computation time (in seconds). Simulation results are presented in Tables 2 and 3, based on 100 simulation replications. From Tables 2 and 3, the following conclusions can be drawn:

(1) In terms of MSEs in Table 2, we note that (i) all the estimators are close to the true value because the results of MSEs are very small; (ii) for any given sample size n, dimension p and error, it can be seen that the MSEs of Conquer and our proposed method (Proposed) are smaller than those of

IP and ADMM. Moreover, the performance of our proposed method is close to that of Conquer.

(2) In terms of computation time in Table 3, we note that (i) all methods run fast except the case of $p = 100$, $n = 10^6$; (ii) for any given sample size n, dimension p and error, the computation time of Conquer is the shortest. The computation time of our proposed method is close to IP and shorter than ADMM.

Table 2: The means and standard deviations (in parentheses) of MSEs ($\times 100$) under different sample sizes n, dimensions p and distributions of errors for Simulation example 1.

ε	p	n	IP	ADMM	Conquer	Proposed
N(0,1)	10	10^3	15.627 (3.481)	15.683 (3.460)	14.415 (3.194)	12.423 (3.289)
		10^4	5.101 (1.332)	5.127 (1.306)	4.916 (1.274)	4.035 (1.060)
		10^5	1.592 (0.418)	1.607 (0.415)	1.527 (0.417)	1.298 (0.325)
		10^6	0.505 (0.135)	0.496 (0.141)	0.455 (0.130)	0.393 (0.079)
	100	10^3	52.729 (4.704)	52.821 (4.678)	46.366 (4.083)	46.508 (4.095)
		10^4	16.237 (1.109)	16.296 (1.126)	15.177 (1.186)	15.233 (1.195)
		10^5	5.213 (0.307)	5.243 (0.288)	4.953 (0.359)	5.057 (0.366)
		10^6	1.643 (0.106)	1.654 (0.116)	1.577 (0.126)	1.624 (0.120)
t(5)	10	10^3	16.850 (3.711)	16.966 (3.666)	15.804 (3.498)	16.409 (3.732)
		10^4	5.652 (1.404)	5.699 (1.404)	5.431 (1.310)	5.419 (1.244)
		10^5	1.710 (0.460)	1.730 (0.460)	1.656 (0.442)	1.656 (0.430)
		10^6	0.529 (0.137)	0.552 (0.146)	0.500 (0.127)	0.517 (0.133)
	100	10^3	58.381 (5.040)	58.420 (4.966)	52.320 (3.761)	52.439 (3.774)
		10^4	16.909 (1.153)	16.944 (1.286)	15.796 (0.706)	15.841 (0.712)
		10^5	5.476 (0.382)	5.463 (0.399)	5.280 (0.361)	5.351 (0.367)
		10^6	1.659 (0.064)	1.673 (0.071)	1.603 (0.046)	1.626 (0.068)

Table 3: The means of computation times (in seconds) under different sample sizes n, dimensions p and distributions of errors for Simulation example 1.

ε	p	n	IP	ADMM	Conquer	Proposed
N(0,1)	10	10^3	0.003	0.013	0.003	0.003
		10^4	0.028	0.095	0.013	0.023
		10^5	0.382	0.496	0.109	0.231
		10^6	3.941	3.439	1.306	2.164
	100	10^3	0.055	0.118	0.012	0.092
		10^4	0.600	1.324	0.075	0.621
		10^5	6.626	13.128	0.600	5.610
		10^6	80.340	190.381	19.781	88.382
t(5)	10	10^3	0.003	0.013	0.003	0.003
		10^4	0.028	0.095	0.012	0.015
		10^5	0.319	0.491	0.106	0.161
		10^6	3.640	3.477	1.357	1.846
	100	10^3	0.076	0.122	0.015	0.124
		10^4	0.873	1.321	0.074	0.790
		10^5	6.939	13.314	0.605	5.883
		10^6	82.083	184.321	16.440	93.300

5.2. Simulation example 2

In this section, we study the performance of the distributed smoothing QR estimator (DSQR) method in Section 3. Furthermore, we include the following three competitors in our comparison:

(1) the central estimator (Cen), which computes the quantile regression by (2.2) using all data by the coordinate descent algorithm (Pietrosanu et al., 2020) (to be consistent with Simulation example 3);

(2) the one-shot estimator with averaging (Avg); see Xu et al. (2020);

(3) the linear estimator for QR (LEQR); see Chen et al. (2019).

We generate data from the following linear model:

$$Y_i = \mathbf{X}_i^\top \beta_0 + \{\varepsilon_i - F_{\varepsilon_i}^{-1}(\tau)\}, \quad i = 1, \ldots, n,$$

where $\mathbf{X}_i = (1, X_{i1}, \ldots, X_{ip})^\top$ is a $(p+1)$-dimensional covariate vector and $(X_{i1}, \ldots, X_{ip})$ is drawn from a multivariate normal distribution $N(0, \Sigma)$. The covariance matrix Σ is constructed by $\Sigma_{ij} = 0.5^{i-j}$ for $1 \leq i, j \leq p$ with $p = 100$, which makes the irrepresentability condition (condition **C3**) satisfied. The true value of the parameter is $\beta_0 = (1, \ldots, 1)$. The errors ε_i are generated independently from a chi-square distribution with 2 degrees of freedom. We fix the subset sample size $n_1 = 500$ and vary the number of machines K; then, the total sample size $n = 500K$. According to Theorem 3.3, we choose $h = (p/n)^{2/5}$ and $h_1 = (p \log n_1/n_1)^{1/3}$ for simplicity.

To evaluate the performance of the four methods, we calculate the MSE and computation time (in seconds). Simulation results for $\tau = 0.3$, 0.5 and 0.7 are presented in Figure 1 and Table 4, respectively, based on 100 simulation replications. From Figure 1 and Table 4, the following conclusions can be drawn:

(1) In terms of MSEs in Figure 1, we note that (i) all the estimators are close to the true value because the results of MSEs are very small; (ii) for any given number of machines K and quantiles τ, it can be seen that the MSEs of the averaged estimator (Avg) are the largest one. Moreover, its estimated efficiency cannot be improved when K is larger than the subsample size; (iii) when K is small, the performance of the DSQR estimator is better than that of LEQR.

(2) In terms of computation time in Table 4, we note that for any given

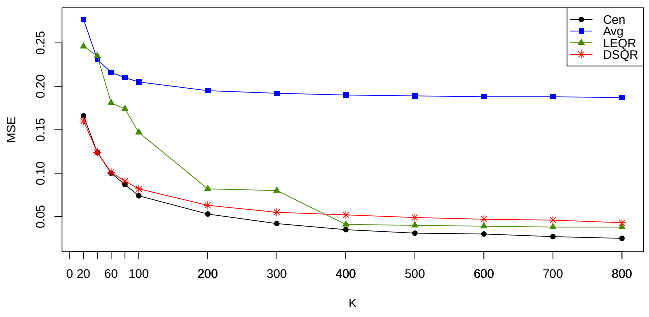
number of machines K and quantile level τ, all estimators are much faster to compute than the Cen, as expected. In particular, our proposed estimator (DSQR) is the fastest one among the estimators. It takes a long time to calculate the one-shot estimator (Avg) on a single machine, but its computing time under a distributed system should be divided by K, which is approximately 0.03, so it runs much fast.

Table 4: The means of computation times (in seconds) under different K and τ for Simulation example 2.

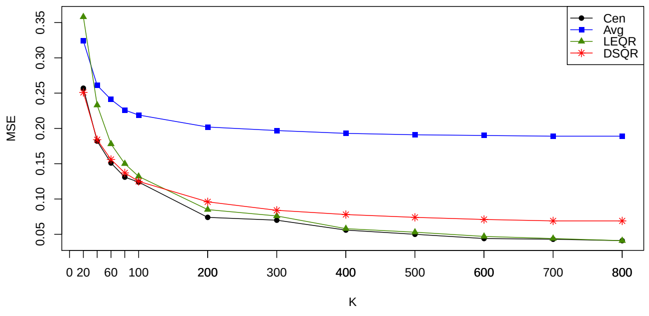
τ	Method\\K	20	40	60	80	100	200	300	400	500	600	700	800
0.3	Cen	3	9	17	26	36	106	197	314	519	698	915	1177
	Avg	0.6	1.2	1.8	2.4	2.9	5.8	8.7	11.6	14.3	17.2	20.0	22.9
	LEQR	0.3	0.6	0.9	1.2	1.4	3.7	5.4	7.2	8.9	10.6	12.3	14.1
	DSQR	0.1	0.2	0.3	0.4	0.5	0.9	1.3	1.8	2.1	2.5	3.0	3.4
0.5	Cen	4	9	18	26	34	98	192	314	463	625	823	1009
	Avg	0.7	1.2	1.8	2.4	2.9	5.8	9.3	12.3	14.8	17.4	20.2	23.3
	LEQR	0.4	0.6	0.9	1.2	1.3	3.6	6.0	7.9	9.2	10.6	12.4	14.3
	DSQR	0.1	0.2	0.2	0.3	0.4	0.8	1.3	1.7	1.9	2.2	2.6	3.0
0.7	Cen	3	9	17	25	35	109	210	295	416	614	825	1035
	Avg	0.6	1.2	1.8	2.3	2.8	5.6	8.4	11.2	14.1	16.9	19.8	22.7
	LEQR	0.3	0.6	0.9	1.1	1.3	3.5	5.3	7.1	8.8	10.6	12.3	14.1
	DSQR	0.1	0.2	0.3	0.3	0.4	0.8	1.1	1.5	1.9	2.3	2.6	3.0

5.3. Simulation example 3

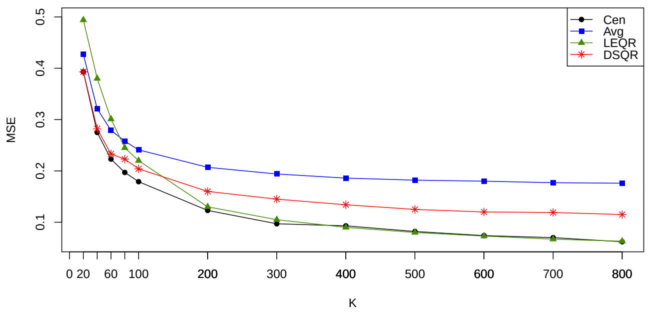
In this section, we study the performances of the distributed penalized smoothing QR estimator (DPSQR) method proposed in Section 4. Furthermore, we include the following five competitors in our comparison:



(a) $\tau = 0.3$



(b) $\tau = 0.5$



(c) $\tau = 0.7$

Figure 1: The means of MSEs versus the number of machines K under different quantiles τ for Simulation example 2.

(1) the central estimator (Cen), which computes the penalized quantile regression based on all data by the coordinate descent algorithm (Pietrosanu et al., 2020);

(2) the subdata estimator (Sub) of our proposed method, which computes (4.1) using only data on the first machine;

(3) the averaging estimator (Avg) of our proposed method, which computes (4.1) on each local machine and combines the local estimators by taking the average;

(4) the communication-efficient estimator (CE); see Wang and Lian (2020);

(5) the robust estimator with LASSO (REL); see Chen et al. (2020).

We conduct a simulation study with $n = 10^6$ and the data are generated from the following linear model:

$$Y_i = \mathbf{X}_i^\top \beta_0 + \{\varepsilon_i - F_{\varepsilon_i}^{-1}(\tau)\}, \quad i = 1, \ldots, n,$$

where $\mathbf{X}_i = (1, X_{i1}, \ldots, X_{ip})^\top$ is a $(p+1)$-dimensional covariate vector and $(X_{i1}, \ldots, X_{ip})$ is drawn from a multivariate uniform distribution on the $[0,1]^p$ with covariance matrix $\Sigma_{ij} = 0.5^{i-j}$ for $1 \leq i, j \leq p$ with $p = 100$. The true value of the parameter is $\beta_0 = (1, 1, 2, 3, 0, \ldots, 0)$. The errors ε_i are generated independently from one of the following two distributions:

(i) Normal errors: $\varepsilon_i \sim N(0, 1)$;

(ii) Heteroscedastic errors: $\varepsilon_i = 0.5(1 + X_{i1}^2)\xi_i$ and $\xi_i \sim t(3)$.

The following five numbers of machines are considered: $K = 100, 200, 500, 800, 1000$, and each subset contains equal observations (n/K). The tuning parameter λ_n is selected by the cross-validation method, which can be obtained by using package cv.hqreg in R. To evaluate the performance of six methods, we calculate the MSE in Simulation example 1, the average proportion of nonzero coefficients correctly estimated to be nonzero (denoted as **C**), and the average proportion of zero coefficients incorrectly estimated to be nonzero (denoted as **IC**). Note that $\mathbf{C} = 1$ and $\mathbf{IC} = 0$ imply perfect recovery. We further study the computational efficiency of our proposed estimator by the computation time (in seconds). Simulation results for $\tau = 0.1$, 0.5 and 0.9 are presented in Tables 5-10, respectively, based on 100 simulation replications.

From Tables 5-10, the following conclusions can be drawn:

(1) In terms of MSEs in Tables 5 and 8, we note that (i) all the estimators are close to the true value because the results of MSEs are very small; (ii) for any given number of machines K, quantile level τ and error terms, the MSEs of subdata estimator (Sub) are the largest one as expected, since it only uses the data on one machine. It can be improved by the averaged estimator (Avg). Furthermore, the proposed estimator (DPSQR) can further reduce the MSEs of Sub; (iii) the performances of the DPSQR estimator are close to Cen and better than other methods under different quantiles and errors.

(2) In terms of **IC**s in Tables 6 and 9, the performance of the all data estimator (Cen) is very good, with $\mathbf{IC} = 0$ under different quantiles and errors. Moreover, for any given number of machines K, quantile level τ and error terms, the performance of our proposed estimator (DPSQR) is better than other methods.

(3) The six methods can select all true predictors in all settings and thus we do not report **C** in the tables.

(4) In terms of computation time in Tables 7 and 10, we note that (i)

Table 5: The means and standard deviations (in parentheses) of MSEs under different K and τ for Simulation example 3. Noises are generated from Normal errors.

τ	K	Cen	Sub	Avg	CE	REL	DPSQR
0.1	100	0.036 (0.006)	0.183 (0.045)	0.143 (0.005)	0.201 (0.049)	0.095 (0.054)	0.042 (0.027)
	200	0.036 (0.006)	0.259 (0.071)	0.146 (0.006)	0.241 (0.068)	0.188 (0.065)	0.044 (0.033)
	500	0.036 (0.006)	0.582 (0.124)	0.160 (0.008)	0.498 (0.121)	0.305 (0.105)	0.102 (0.096)
	800	0.036 (0.006)	0.845 (0.135)	0.178 (0.009)	0.691 (0.151)	0.310 (0.125)	0.094 (0.092)
	1000	0.036 (0.006)	1.024 (0.167)	0.190 (0.010)	0.790 (0.190)	0.313 (0.151)	0.062 (0.065)
0.5	100	0.033 (0.005)	0.188 (0.044)	0.176 (0.004)	0.313 (0.019)	0.113 (0.007)	0.100 (0.043)
	200	0.033 (0.005)	0.202 (0.064)	0.177 (0.004)	0.187 (0.066)	0.163 (0.061)	0.070 (0.055)
	500	0.033 (0.005)	0.349 (0.091)	0.178 (0.005)	0.272 (0.089)	0.213 (0.073)	0.053 (0.054)
	800	0.033 (0.005)	0.520 (0.121)	0.181 (0.005)	0.391 (0.148)	0.246 (0.101)	0.057 (0.067)
	1000	0.033 (0.005)	0.633 (0.139)	0.183 (0.006)	0.455 (0.138)	0.249 (0.108)	0.047 (0.060)
0.9	100	0.039 (0.007)	0.209 (0.062)	0.167 (0.006)	0.212 (0.039)	0.114 (0.079)	0.066 (0.054)
	200	0.039 (0.007)	0.285 (0.077)	0.166 (0.006)	0.263 (0.072)	0.214 (0.071)	0.057 (0.053)
	500	0.039 (0.007)	0.568 (0.118)	0.160 (0.007)	0.486 (0.113)	0.287 (0.105)	0.074 (0.076)
	800	0.039 (0.007)	0.841 (0.140)	0.156 (0.008)	0.697 (0.136)	0.309 (0.127)	0.085 (0.091)
	1000	0.039 (0.007)	1.003 (0.154)	0.154 (0.007)	0.790 (0.146)	0.289 (0.127)	0.071 (0.097)

for any given number of machines K, quantile level τ and error terms, the computation time of Cen is the longest one, as expected. Moreover, other estimators are much faster to compute than Cen; (ii) the computation time of Sub is the shortest, as expected; (iii) our proposed estimator (DPSQR) is slower to compute than Avg for large K because the number of iterations (DPSQR) increases with increasing K; (iv) the CE is faster than our proposed estimator (DPSQR) since it only involves one round of aggregation; (v) our proposed estimator (DPSQR) is faster than REL.

Table 6: The means and standard deviations (in parentheses) of ICs under different K and τ for Simulation example 3. Noises are generated from Normal errors.

τ	K	Cen	Sub	Avg	CE	REL	DPSQR
0.1	100	0.000 (0.000)	0.012 (0.011)	0.034 (0.016)	0.046 (0.063)	0.006 (0.013)	0.007 (0.017)
	200	0.000 (0.000)	0.054 (0.025)	0.180 (0.037)	0.385 (0.098)	0.145 (0.086)	0.021 (0.047)
	500	0.000 (0.000)	0.142 (0.037)	0.325 (0.043)	0.338 (0.109)	0.314 (0.169)	0.136 (0.205)
	800	0.000 (0.000)	0.191 (0.040)	0.362 (0.045)	0.374 (0.089)	0.326 (0.188)	0.148 (0.221)
	1000	0.000 (0.000)	0.209 (0.039)	0.374 (0.047)	0.390 (0.082)	0.327 (0.196)	0.112 (0.188)
0.5	100	0.000 (0.000)	0.010 (0.008)	0.009 (0.004)	0.000 (0.000)	0.000 (0.000)	0.008 (0.005)
	200	0.000 (0.000)	0.013 (0.013)	0.015 (0.007)	0.029 (0.036)	0.036 (0.043)	0.013 (0.019)
	500	0.000 (0.000)	0.068 (0.031)	0.156 (0.032)	0.194 (0.118)	0.170 (0.112)	0.062 (0.093)
	800	0.000 (0.000)	0.107 (0.036)	0.237 (0.038)	0.282 (0.133)	0.252 (0.179)	0.103 (0.155)
	1000	0.000 (0.000)	0.130 (0.039)	0.265 (0.040)	0.335 (0.128)	0.265 (0.205)	0.095 (0.145)
0.9	100	0.000 (0.000)	0.013 (0.011)	0.035 (0.015)	0.001 (0.003)	0.004 (0.012)	0.005 (0.007)
	200	0.000 (0.000)	0.051 (0.025)	0.182 (0.038)	0.054 (0.028)	0.124 (0.076)	0.019 (0.040)
	500	0.000 (0.000)	0.140 (0.035)	0.327 (0.039)	0.252 (0.085)	0.300 (0.148)	0.094 (0.167)
	800	0.000 (0.000)	0.181 (0.038)	0.360 (0.049)	0.363 (0.091)	0.358 (0.184)	0.142 (0.224)
	1000	0.000 (0.000)	0.207 (0.031)	0.371 (0.047)	0.409 (0.092)	0.381 (0.201)	0.143 (0.229)

Table 7: The means of computation times (in seconds) under different K and τ for Simulation example 3. Noises are generated from Normal errors.

τ	K	Cen	Sub	Avg	CE	REL	DPSQR
0.1	100	829	0.18	19.7	2.1	19.6	4.6
	200	829	0.14	6.5	1.3	17.3	5.4
	500	829	0.04	7.1	1.1	25.0	7.9
	800	829	0.01	8.4	1.1	33.4	11.7
	1000	829	0.01	10.0	1.0	41.1	13.3
0.5	100	476	0.13	16.3	2.1	18.8	4.3
	200	476	0.11	6.1	1.1	16.9	5.0
	500	476	0.02	5.9	1.0	24.5	7.7
	800	476	0.01	6.4	1.0	32.9	11.2
	1000	476	0.01	7.0	1.0	40.7	13.2
0.9	100	780	0.17	16.3	2.1	18.9	4.1
	200	780	0.13	6.2	1.3	17.1	4.7
	500	780	0.05	6.7	1.0	24.6	6.9
	800	780	0.01	8.2	1.0	32.9	10.5
	1000	780	0.01	9.8	0.9	40.7	12.3

Table 8: The means and standard deviations (in parentheses) of MSEs under different K and τ for Simulation example 3. Noises are generated from Heteroscedastic errors.

τ	K	Cen	Sub	Avg	CE	REL	DPSQR
0.1	100	0.048 (0.011)	0.197 (0.062)	0.136 (0.006)	0.214 (0.067)	0.112 (0.061)	0.058 (0.044)
	200	0.048 (0.011)	0.297 (0.076)	0.137 (0.006)	0.276 (0.070)	0.206 (0.069)	0.066 (0.050)
	500	0.048 (0.011)	0.596 (0.122)	0.139 (0.007)	0.518 (0.112)	0.274 (0.104)	0.087 (0.079)
	800	0.048 (0.011)	0.886 (0.160)	0.140 (0.007)	0.746 (0.159)	0.311 (0.117)	0.097 (0.092)
	1000	0.048 (0.011)	1.048 (0.189)	0.142 (0.008)	0.874 (0.187)	0.314 (0.151)	0.074 (0.091)
0.5	100	0.016 (0.004)	0.144 (0.032)	0.128 (0.003)	0.247 (0.019)	0.127 (0.004)	0.084 (0.034)
	200	0.016 (0.004)	0.154 (0.038)	0.128 (0.003)	0.132 (0.042)	0.131 (0.036)	0.055 (0.041)
	500	0.016 (0.004)	0.224 (0.064)	0.129 (0.004)	0.161 (0.061)	0.149 (0.052)	0.037 (0.038)
	800	0.016 (0.004)	0.327 (0.074)	0.131 (0.004)	0.209 (0.069)	0.164 (0.060)	0.025 (0.028)
	1000	0.016 (0.004)	0.385 (0.074)	0.132 (0.004)	0.249 (0.075)	0.166 (0.063)	0.018 (0.019)
0.9	100	0.045 (0.010)	0.231 (0.060)	0.199 (0.007)	0.228 (0.040)	0.132 (0.079)	0.077 (0.051)
	200	0.045 (0.010)	0.316 (0.082)	0.200 (0.008)	0.290 (0.074)	0.216 (0.078)	0.067 (0.050)
	500	0.045 (0.010)	0.642 (0.131)	0.209 (0.009)	0.558 (0.124)	0.304 (0.110)	0.097 (0.083)
	800	0.045 (0.010)	0.965 (0.193)	0.225 (0.009)	0.825 (0.205)	0.334 (0.151)	0.115 (0.137)
	1000	0.045 (0.010)	1.171 (0.214)	0.236 (0.010)	0.976 (0.230)	0.356 (0.166)	0.094 (0.104)

Table 9: The means and standard deviations (in parentheses) of ICs under different K and τ for Simulation example 3. Noises are generated from Heteroscedastic errors.

τ	K	Cen	Sub	Avg	CE	REL	DPSQR
0.1	100	0.000 (0.000)	0.016 (0.014)	0.042 (0.017)	0.043 (0.072)	0.013 (0.027)	0.012 (0.032)
	200	0.000 (0.000)	0.060 (0.027)	0.205 (0.038)	0.420 (0.102)	0.218 (0.120)	0.044 (0.087)
	500	0.000 (0.000)	0.145 (0.036)	0.336 (0.040)	0.361 (0.096)	0.335 (0.147)	0.122 (0.183)
	800	0.000 (0.000)	0.190 (0.040)	0.375 (0.038)	0.370 (0.091)	0.349 (0.181)	0.153 (0.217)
	1000	0.000 (0.000)	0.208 (0.040)	0.386 (0.040)	0.371 (0.095)	0.332 (0.199)	0.110 (0.203)
0.5	100	0.000 (0.000)	0.001 (0.001)	0.001 (0.002)	0.000 (0.000)	0.000 (0.000)	0.010 (0.005)
	200	0.000 (0.000)	0.008 (0.008)	0.011 (0.003)	0.025 (0.029)	0.015 (0.012)	0.010 (0.009)
	500	0.000 (0.000)	0.057 (0.028)	0.070 (0.022)	0.171 (0.095)	0.105 (0.074)	0.029 (0.064)
	800	0.000 (0.000)	0.098 (0.029)	0.146 (0.030)	0.308 (0.097)	0.173 (0.106)	0.054 (0.100)
	1000	0.000 (0.000)	0.120 (0.032)	0.181 (0.032)	0.340 (0.098)	0.204 (0.131)	0.041 (0.062)
0.9	100	0.000 (0.000)	0.018 (0.014)	0.058 (0.023)	0.002 (0.006)	0.006 (0.013)	0.006 (0.010)
	200	0.000 (0.000)	0.061 (0.025)	0.228 (0.037)	0.057 (0.025)	0.145 (0.071)	0.018 (0.043)
	500	0.000 (0.000)	0.153 (0.036)	0.346 (0.044)	0.245 (0.076)	0.305 (0.142)	0.101 (0.171)
	800	0.000 (0.000)	0.195 (0.044)	0.369 (0.043)	0.351 (0.088)	0.326 (0.181)	0.129 (0.220)
	1000	0.000 (0.000)	0.219 (0.045)	0.386 (0.038)	0.398 (0.094)	0.333 (0.188)	0.122 (0.219)

Table 10: The means of computation times (in seconds) under different K and τ for Simulation example 3. Noises are generated from Heteroscedastic errors.

τ	K	Cen	Sub	Avg	CE	REL	DPSQR
0.1	100	1305	0.16	20.1	2.9	19.6	4.3
	200	1305	0.13	6.4	1.3	17.3	4.9
	500	1305	0.07	7.5	1.1	24.7	7.3
	800	1305	0.02	8.7	1.0	33.0	10.8
	1000	1305	0.01	10.2	0.9	40.6	12.4
0.5	100	486	0.16	15.0	2.1	18.7	4.3
	200	486	0.09	6.0	1.1	16.9	5.1
	500	486	0.05	5.9	1.0	24.5	7.5
	800	486	0.02	6.1	1.0	32.8	11.0
	1000	486	0.01	6.6	0.9	40.4	12.7
0.9	100	1277	0.15	19.6	3.6	18.7	4.2
	200	1277	0.14	6.2	1.2	16.8	4.6
	500	1277	0.07	7.3	1.0	24.4	6.7
	800	1277	0.04	8.6	1.0	32.4	9.8
	1000	1277	0.02	10.2	0.8	40.0	11.3

Table 11: Covariates and their descriptions for real data example 1.

Name	Description
SO_2	SO_2 concentration (ug/m^3)
NO_2	NO_2 concentration (ug/m^3)
CO	CO concentration (ug/m^3)
TEMP	temperature (degrees Celsius)
PRES	pressure (hPa)
DEWP	dew point temperature (degrees Celsius)
WSPM	wind speed (m/s)

5.4. Real data example 1: Beijing multi-site air-quality data set

We apply the proposed DSQR method in Section 3 to the analysis of Beijing multi-site air-quality dataset. The dataset includes 420768 hourly air pollutant data points from 12 nationally-controlled air-quality monitoring sites. The air-quality data are from the Beijing Municipal Environmental Monitoring Center. The meteorological data at each air-quality site were matched with the nearest weather station from the China Meteorological Administration. The time period is from March 1st, 2013, to February 28th, 2017. The dataset is obtained from online site: https://archive.ics.uci.edu/ml/datasets/Beijing+Site+Air-Quality+Data. More details can be found in the relevant paper by Chen et al. (2020).

In this study, we can use a linear model (1.1) to explore the relationship between the $PM_{2.5}$ concentration (ug/m^3) and seven variables in Table 11.

Because the data are from 12 nationally-controlled air-quality monitoring sites, we consider the number of blocks $K = 12$. The initial estimator in our method is based on the first site (Aotizhongxin site). Figure 2 depicts the changes in estimated coefficients for the Beijing multi-site air-quality data using our proposed DSQR method with quantiles $\tau = 0.1$, 0.3, 0.5, 0.7 and 0.9. From Figure 2, it is easy to see that the estimated coefficients of SO_2, NO_2, DEWP and WSPM increase with quantile τ. Furthermore, we evaluate the performance of the proposed DSQR estimator compared with IP, ADMM, Conquer, AVQR, LEQR and a two-step procedure (TS) proposed by Volgushev et al. (2019), based on the mean absolute fitting error (MAFE):

$$MAFE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|,$$

where n is the total sample size 420768, Y_i is the value of $PM_{2.5}$, and $\hat{Y}_i$ is the fitted value of Y_i at quantile 0.5. The results are present in Table 12. We also calculate the MAFE of the least squares method by full data, which is equal to 30.391. In terms of the MAFE, we can find that quantile regression is better than the least square method for this example. The results of all data analysis by IP, ADMM, and Conquer are the same. Our proposed method is close to all data analysis, and is better than AVQR, TS and LEQR. Moreover, to illustrate the computational advantage of the proposed DSQR method, we also list the running times for quantiles $\tau = 0.1$, 0.3, 0.5, 0.7 and 0.9 together by different methods in Table 12. The results show that the DSQR costs less time than the other six methods. To summarize, our algorithm can handle QR with massive data problems easily, with its running time competitive.

5.5. Real data example 2: Year Prediction MSD data set

As an illustration, we now apply the proposed DPSQR methodology in Section 4 to the Year Prediction MSD dataset. The dataset is collected from the public database of the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/dat

Table 12: The computation times (in seconds) for all quantiles $\tau = 0.1, 0.3, 0.5, 0.7, 0.9$ and MAFE with $\tau = 0.5$ of the IP, ADMM, Conquer, AVQR, TS, LEQR and DSQR estimators for real data example 1.

	IP	ADMM	Conquer	AVQR	TS	LEQR	DSQR
t	10.3	31.0	7.7	44.7	44.8	5.33	4.0
MAFE	29.233	29.233	29.233	29.298	29.299	29.258	29.244

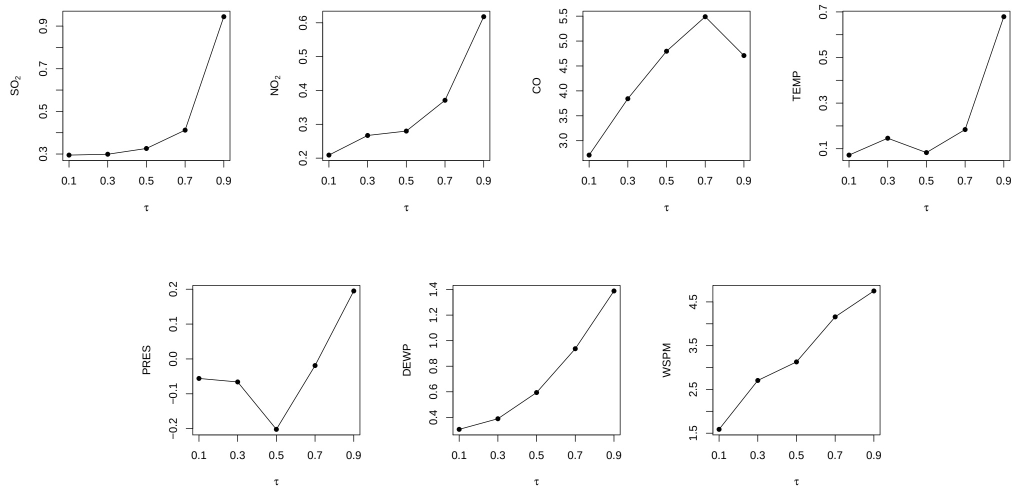


Figure 2: The estimated coefficients of DSQR under different quantiles τ for real data example 1.

The dataset is a freely-available collection of audio features for contemporary popular music tracks ranging from 1922 to 2011. Approximately 515345 observations were recorded with 91 variables: the year of a song, 12 average timbre and 78 timbre covariance variables. The problem of research is to predict the release year of songs from the audio features.

In this study, a linear model (1.1) is used to fit the data, where the year of a song is the dependent variable ($\mathbf{Y}$) and 12 average timbre and 78 timbre covariance variables are covariate variables. To evaluate the performance of our proposed DPSQR method, we first calculate the mean absolute prediction error (MAPE) of the predictions under quantile $\tau = 0.5$. The first 500000 data points are used for the estimation, and the remaining 15345 data points are used for the prediction. Therefore,

$$MAPE = \frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \left| Y_i - \hat{Y}_i \right|,$$

where $\hat{Y}_i$ is the fitted value of Y_i and $\tilde{n} = 15345$. The results are presented in Table 13, and it is easy to show that the performances of the all data estimator (Cen) are the best of all methods, as expected. The MAPEs of other estimators increase with K. Moreover, the MAPEs of the proposed estimator (DPSQR) are smaller than those of Sub, CE and REL.

Furthermore, to illustrate the computational advantage of the proposed DSPQR method, we also list the running time of our method under different K and quantiles τ in Table 14. The results show that DSPQR costs much less time than Cen, Avg and REL, and is close to Sub and CE. In addition, we study the number of variables selected by the DPSQR method. The results are presented in Table 15, which shows that LASSO procedures a small model

Table 13: The MAPE of the Cen, Sub, Avg, CE, REL, DPSQR estimators with $\tau = 0.5$ under different K for real data example 2.

K	Cen	Sub	Avg	CE	REL	DPSQR
50	6.694	6.805	6.738	6.805	6.802	6.801
100	6.694	6.868	6.737	6.869	6.866	6.856
200	6.694	7.046	6.743	7.043	7.028	7.024
300	6.694	7.154	6.749	7.152	7.125	7.118

because the numbers of selected variables under different K and quantiles level τ are all smaller than the case with $p = 90$ variables. Moreover, for any given K, the number of selected variables decreases with quantile level τ. In summary, again, our algorithm can handle QR with massive and high-dimensional data problems easily, with a competitive running time.

6. Discussion

In this article, we considered a communication-efficient QR approach for distributed high-dimensional linear model, where the size n of the data is too large to be processed simultaneously, and the dimension p of covariates is allowed to be much larger than n. We proposed using QR to extract features of the conditional distribution of the response while avoiding tail conditions and taking heteroscedasticity into account. Under appropriate conditions, the established rate of the estimator is the same as the rate of the standard ℓ_1-regularized QR estimator using all data. One key insight from this work is that a nonsmooth QR loss can be transformed into a convex quadratic loss function, which greatly facilitates computation in a distributed setting. We further provide a communication efficient distributed algorithm, which

Table 14: The computation times (in seconds) of the Cen, Sub, Avg, CE, REL, DPSQR estimators under different K and quantiles τ for real data example 2.

τ	K	Cen	Sub	Avg	CE	REL	DPSQR
0.1	50	546	0.37	16	0.77	4.0	1.3
	100	546	0.14	17	0.50	3.5	0.9
	200	546	0.15	21	0.51	3.4	0.8
	300	546	0.13	34	0.51	3.5	1.0
0.3	50	484	0.48	22	0.99	5.6	1.3
	100	484	0.22	19	0.59	3.6	1.4
	200	484	0.14	28	0.63	3.7	0.9
	300	484	0.12	32	0.48	3.6	1.6
0.5	50	402	0.54	20	1.10	4.7	1.6
	100	402	0.26	21	0.91	4.1	1.5
	200	402	0.13	52	0.72	3.8	1.3
	300	402	0.19	47	0.66	4.0	1.1
0.7	50	433	0.41	20	0.83	4.2	1.4
	100	433	0.20	26	0.56	3.7	1.1
	200	433	3.24	102	3.59	6.7	4.1
	300	433	0.11	234	0.59	3.6	1.0
0.9	50	489	0.31	15	0.81	4.0	1.2
	100	489	0.14	17	0.54	3.6	1.0
	200	489	0.07	72	0.42	3.4	1.0
	300	489	1.96	156	2.39	5.4	2.9

Table 15: The number of selected variables by DPSQR method under different K and quantiles τ for real data example 2.

K	$\tau = 0.1$	$\tau = 0.3$	$\tau = 0.5$	$\tau = 0.7$	$\tau = 0.9$
50	50	35	34	24	14
100	42	36	35	29	22
200	58	37	30	23	17
300	53	41	43	38	16

runs iteratively and only communicates p-dimensional gradient information at each iteration (instead of the $p \times p$ matrix information). Compared with the existing methods, our proposed method can deal with regularized quantile regression without additional conditions: limit the number of machines or the homogeneity of error terms.

To facilitate future research, we will discuss several interesting topics. First, the selections of the regularization parameter λ_n and bandwidths h and h_1, and then, the design of corresponding distributed algorithms. Second, Zhao et al. (2020) studied the inference result based on averaging de-biased QR estimators. As we mentioned in the Introduction, this approach might suffer from heavy computational cost. It would be interesting to develop computationally efficient inference approaches. Finally, one could extend our proposed method to quantile instrumental variable models (Kaplan and Sun, 2017).

Acknowledgments

The authors thank editors and referees for their constructive comments and variable suggestions which have greatly improved the article. This research is supported by the National Natural Science Foundation of China (Series number: 11801069).

Appendix A. Proof of main results

Lemma 1. *Suppose the conditions in Theorem 3.1 hold. We have*

$$\left\| S_h^{(2)}(\hat{\beta}^0) - D \right\| = O_p\left(\sqrt{\frac{p \log n}{nh}} + a_n + h \right).$$

Proof. By the proof of Lemma 3 in Cai et al. (2010), we have

$$\left\| S_h^{(2)}(\hat{\beta}^0) - D \right\| \le 5 \sup_{j \le C_1} \left| \nu_j^\top \left\{ S_h^{(2)}(\hat{\beta}^0) - D \right\} \nu_j \right|, \tag{A.1}$$

where $\nu_1, \ldots, \nu_{C_1}$ are some non-random vectors with $\|\nu_j\|_2 = 1$ and $C_1 \le 5^p$. For $\alpha \in R^p$, denote

$$S_j^{(2)}(\alpha) = n^{-1} \sum_{i=1}^{n} \left(\nu_j^\top \mathbf{X}_i \right)^2 K_h(-e_i(\alpha)).$$

Therefore, when $\|\hat{\beta}^0 - \beta_0\|_2 \le a_n$,

$$\sup_{j \le C_1} \left| \nu_j^\top \left\{ S_h^{(2)}(\hat{\beta}^0) - D \right\} \nu_j \right| \le \sup_{j \le C_1} \sup_{\|\alpha - \beta_0\|_2 \le a_n} \left| S_j^{(2)}(\alpha) - \nu_j^\top D \nu_j \right|. \tag{A.2}$$

Denote $\beta_0 = (\beta_{0,1}, \ldots, \beta_{0,p})$. For every j, we divide the interval $[\beta_{0,j} - a_n, \beta_{0,j} + a_n]$ into n^{C_2} small subintervals and each has length $2a_n/n^{C_2}$, where C_2 is a large positive number. Therefore, there exists a set of points in R^p, $\{\alpha_d, 1 \le d \le n^{C_2 p}\}$, such that for any α in the ball $\|\alpha - \beta_0\|_2 \le a_n$, we have $\|\alpha - \alpha_d\|_2 \le 2\sqrt{p} a_n / n^{C_2}$ for some $1 \le j \le n^{C_2 p}$. Thus, for some large enough constant C_3, we have

$$\left| K_h(-e_i(\alpha)) - K_h(-e_i(\alpha_d)) \right| \le C_3 h^{-2} \left| \mathbf{X}_i^\top (\alpha - \alpha_d) \right|.$$

Therefore

$$\sup_j \sup_{\|\alpha-\beta_0\|_2 \le a_n} \left|S_j^{(2)}(\alpha) - \nu_j^\top D\nu_j\right| - \sup_j \sup_d \left|S_j^{(2)}(\alpha_d) - \nu_j^\top D\nu_j\right| \le \frac{C_3 a_n \sqrt{p}}{n^{C_2+1}h^2} \sum_{i=1}^n \|\mathbf{X}_i\|_2^3.$$

Since $\max_{i,j} E|X_{i,j}|^3 < \infty$, by letting C_2 large enough, we have for any $\delta > 0$,

$$\sup_j \sup_{\|\alpha-\beta_0\|_2 \le a_n} \left|S_j^{(2)}(\alpha) - \nu_j^\top D\nu_j\right| - \sup_j \sup_d \left|S_j^{(2)}(\alpha_d) - \nu_j^\top D\nu_j\right| = O_p(n^{-\delta}). \tag{A.3}$$

By the exponential inequality (Cai and Liu, 2011), for some constant C_4, we have

$$\sup_j \sup_d P\left(\left|S_j^{(2)}(\alpha_d) - E\{S_j^{(2)}(\alpha_d)\}\right| \ge C_4\sqrt{\frac{p\log n}{nh}}\right) = O(n^{-\delta p}). \tag{A.4}$$

Moreover, by the Lemma 1 in Fernandes et al. (2021), we can obtain

$$|E\{S_j^{(2)}(\alpha)\} - \nu_j^\top D\nu_j| = O(a_n + h). \tag{A.5}$$

Thus, by (A.1)-(A.5), we have

$$\left\|S_h^{(2)}(\hat\beta^0) - D\right\| = O_p\left(\sqrt{\frac{p\log n}{nh}} + a_n + h\right).$$

This completes the proof. $\square$

Proof of Theorem 3.1. Note that (2.3), we have $S_h^{(1)}(\hat\beta) = 0$, thus

$$S_h^{(1)}(\hat\beta^0) = S_h^{(1)}(\hat\beta^0) - S_h^{(1)}(\hat\beta) = S_h^{(2)}(\tilde\beta)(\hat\beta^0 - \hat\beta),$$

where $\tilde\beta$ is between $\hat\beta^0$ and $\hat\beta$. Then, by the form of (3.1), we can obtain

$$\begin{aligned} S_{h_1,1}^{(2)}(\hat\beta^0)(\hat\beta_H^1 - \beta_0) =& S_{h_1,1}^{(2)}(\hat\beta^0)(\hat\beta^0 - \beta_0) - S_h^{(1)}(\hat\beta^0) \\ =& \left\{S_{h_1,1}^{(2)}(\hat\beta^0) - S_h^{(2)}(\tilde\beta)\right\}(\hat\beta^0 - \beta_0) + S_h^{(2)}(\tilde\beta)(\hat\beta - \beta_0). \end{aligned}$$

Thus, we have

$$\begin{aligned} D(\hat{\beta}_H^1 - \beta_0) &= \left\{ D - S_{h_1,1}^{(2)}(\hat{\beta}^0) \right\} (\hat{\beta}_H^1 - \beta_0) + S_{h_1,1}^{(2)}(\hat{\beta}^0)(\hat{\beta}_H^1 - \beta_0) \\ &= \left\{ D - S_{h_1,1}^{(2)}(\hat{\beta}^0) \right\} (\hat{\beta}_H^1 - \beta_0) + \left\{ S_{h_1,1}^{(2)}(\hat{\beta}^0) - D + D - S_h^{(2)}(\tilde{\beta}) \right\} (\hat{\beta}^0 - \beta_0) \\ &\quad + \left\{ S_h^{(2)}(\tilde{\beta}) - D \right\} (\hat{\beta} - \beta_0) + D(\hat{\beta} - \beta_0). \end{aligned} \tag{A.6}$$

By the Theorem 3.1 in He et al. (2020) and the condition $p = O(n^{c_2})$, we have

$$\sqrt{(\hat{\beta} - \beta_0)^\top \Sigma (\hat{\beta} - \beta_0)} = O_p\left(\sqrt{\frac{p}{n}} + h^2 \right).$$

Thus, by the condition **C3** and $h \lesssim (p/n)^{1/4}$,

$$\|\hat{\beta} - \beta_0\|_2 \le \sqrt{\Lambda_{\min}^{-1}(\Sigma)(\hat{\beta} - \beta_0)^\top \Sigma (\hat{\beta} - \beta_0)} = O_p\left(\sqrt{\frac{p}{n}} \right). \tag{A.7}$$

Note that $\tilde{\beta}$ is between $\hat{\beta}^0$ and $\hat{\beta}$, and $a_n \gtrsim \sqrt{p/n}$, thus

$$\|\tilde{\beta} - \beta_0\|_2 \lesssim \|\hat{\beta}^0 - \beta_0\|_2.$$

Then, by the proof of Lemma 1, we can obtain

$$\left\| S_h^{(2)}(\tilde{\beta}) - D \right\| = O_p\left(\sqrt{\frac{p \log n}{nh}} + a_n + h \right). \tag{A.8}$$

By a similar argument of Lemma 1,

$$\left\| S_{h_1,1}^{(2)}(\hat{\beta}^0) - D \right\| = O_p\left(\sqrt{\frac{p \log n_1}{n_1 h_1}} + a_n + h_1 \right). \tag{A.9}$$

By the conditions **C2** and **C3**, we have

$$\|D(\hat{\beta}_H^1 - \beta_0)\|_2 \ge l_1 \|\Sigma(\hat{\beta}_H^1 - \beta_0)\|_2 \ge l_1 \Lambda_{\min}(\Sigma) \|\hat{\beta}_H^1 - \beta_0\|_2. \tag{A.10}$$

Then, by Lemma 1 and (A.6)-(A.10), we can obtain

$$\begin{aligned}
\|\hat{\beta}_H^1 - \beta_0\|_2 \leq & l_1^{-1}\Lambda_{\min}^{-1}(\Sigma)\|D(\hat{\beta}_H^1 - \beta_0)\|_2 \\
\leq & \left\|D - S_{h_1,1}^{(2)}(\hat{\beta}^0)\right\| \left\|\hat{\beta}_H^1 - \beta_0\right\|_2 \\
& + \left\{\left\|S_{h_1,1}^{(2)}(\hat{\beta}^0) - D\right\| + \left\|D - S_h^{(2)}(\tilde{\beta})\right\|\right\} \left\|\hat{\beta}^0 - \beta_0\right\|_2 \\
& + \left\|S_h^{(2)}(\tilde{\beta}) - D\right\| \left\|\hat{\beta} - \beta_0\right\|_2 + \left\|D(\hat{\beta} - \beta_0)\right\|_2 \\
= & O_p\left(\sqrt{\frac{p}{n}} + a_n\sqrt{\frac{p\log n}{nh}} + a_n\sqrt{\frac{p\log n_1}{n_1h_1}} + a_n^2 + a_nh + a_nh_1\right).
\end{aligned}$$

□

Proof of Theorem 3.2. Note that h_1 is only used to calculate $S_{h_1,1}^{(2)}(\hat{\beta}^0)$, thus we can choose $h_1 = O((p\log n_1/n_1)^{1/3})$ to obtain $\sqrt{p\log n_1/n_1h_1} \asymp h_1$. Then, (3.2) can be write as

$$\|\hat{\beta}_H^1 - \beta_0\|_2 = O_p\left(\sqrt{\frac{p}{n}} + a_n\left\{\sqrt{\frac{p\log n}{nh}} + h + \left(\frac{p\log n_1}{n_1}\right)^{1/3}\right\}\right). \quad \text{(A.11)}$$

Under the condition $(p/n)^{1/2} \lesssim h \lesssim (p/n)^{1/4}$ in Theorem 3.1, we have

$$\sqrt{\frac{p\log n}{nh}} \lesssim \left(\frac{p\log^2 n}{n}\right)^{1/4}. \quad \text{(A.12)}$$

Then, when $n_1/\log n_1 < n^{3/4}(pn_1/\log^6 n)^{1/4}$, $(p\log n_1/n_1)^{1/3}$ is larger than h and $\sqrt{p\log n/(nh)}$. Thus, by (A.11), we can obtain

$$\|\hat{\beta}_H^1 - \beta_0\|_2 = O_p\left(\sqrt{\frac{p}{n}} + a_n\left(\frac{p\log n_1}{n_1}\right)^{1/3}\right).$$

On the other hand, if $n_1/\log n_1 \geq n^{3/4}(p/\log^6 n)^{1/4}$, under condition $a_n \asymp \sqrt{p/n_1}$ and (A.12), we have

$$a_n\left\{\sqrt{\frac{p\log n}{nh}} + h\right\} \lesssim \sqrt{\frac{p}{n_1}}\left(\frac{p\log^2 n}{n}\right)^{1/4} \lesssim \left(\frac{p}{n}\right)^{5/8}\left(\frac{\log^5 n}{\log^2 n_1}\right)^{1/4}.$$

Then, the second term in (A.11) is dominated by the first term. Therefore, under conditions in Theorem 3.1, and $a_n \asymp \sqrt{p/n_1}$, $h_1 = O((p\log n_1/n_1)^{1/3})$, we can obtain

$$\|\hat{\beta}_H^q - \beta_0\|_2 = O_p\left(\sqrt{\frac{p}{n}} + \sqrt{\frac{p}{n_1}}\left(\frac{p\log n_1}{n_1}\right)^{q/3}\right).$$

Then it completes the proof of Theorem 3.2.

□

Proof of Theorem 3.3. Under conditions **C1**, **C2** and **C4**, the number of iterations q satisfies (3.4), and by the result of Theorem 3.3 in He et al. (2020), we have

$$\sup_{x\in R,\alpha\in R^p}\left|P\left\{\frac{n^{1/2}\alpha^\top(\hat{\beta}_H^q - \beta_0)}{\sqrt{\tau(1-\tau)\alpha^\top\mathbf{D}^{-1}\Sigma\mathbf{D}^{-1}\alpha}} \le x\right\} - \Phi(x)\right| \lesssim \frac{p+\log n}{\sqrt{nh}} + n^{1/2}h^2,$$

where $\Phi(\cdot)$ denotes the standard normal distribution function. For obtaining the best condition on p, we choose $h = O((p/n)^{2/5})$, then

$$\frac{p+\log n}{\sqrt{nh}} + n^{1/2}h^2 = (p+\log n)^{4/5}n^{-3/10} \to 0.$$

Thus, $p = o(n^{3/8})$. This completes the proof. □

Lemma 2. *Suppose the conditions in Theorem 4.1 hold. We have*

$$\begin{aligned}&\left\|\bar{S}_{h,h_1}^{(1)}(\beta_0, \hat{\beta}_L^0)\right\|_\infty \\ =&O_p\left(\sqrt{\frac{\log(n\vee p)}{n}} + \tilde{a}_n\sqrt{\frac{s\log(n\vee p)}{nh}} + \tilde{a}_n\sqrt{\frac{s\log(n_1\vee p)}{n_1h_1}} + \tilde{a}_n^2 + \tilde{a}_n^2 h + \tilde{a}_n h_1\right),\end{aligned}$$

where $\bar{S}_{h,h_1}^{(1)}(\beta_0, \hat{\beta}_L^0) = S_h^{(1)}(\hat{\beta}_L^0) - S_{h_1,1}^{(2)}(\hat{\beta}_L^0)(\hat{\beta}_L^0 - \beta_0)$.

Proof. Note that

$$\begin{aligned}\bar{S}^{(1)}_{h,h_1}(\beta_0,\hat{\beta}^0_L) =& S^{(1)}_h(\beta_0) + \left\{S^{(2)}_h(\hat{\beta}^0_L) - S^{(2)}_{h_1,1}(\hat{\beta}^0_L)\right\}(\hat{\beta}^0_L - \beta_0) \\ & - \left\{S^{(1)}_h(\beta_0) - S^{(1)}_h(\hat{\beta}^0_L) - S^{(2)}_h(\hat{\beta}^0_L)(\beta_0 - \hat{\beta}^0_L)\right\}.\end{aligned} \tag{A.13}$$

We first consider $\|S^{(1)}_h(\beta_0)\|_\infty$. Note that

$$\left\|S^{(1)}_h(\beta_0)\right\|_\infty \leq \left\|S^{(1)}_h(\beta_0) - E\left\{S^{(1)}_h(\beta_0)\right\}\right\|_\infty + \left\|E\left\{S^{(1)}_h(\beta_0)\right\}\right\|_\infty. \tag{A.14}$$

By Bernstein's inequality, we can obtain

$$\left\|S^{(1)}_h(\beta_0) - E\left\{S^{(1)}_h(\beta_0)\right\}\right\|_\infty = O_p\left(\sqrt{\frac{\log(n \vee p)}{n}}\right). \tag{A.15}$$

Note that,

$$\begin{aligned}E\left\{S^{(1)}_h(\beta_0)\right\} &= E\left[\frac{1}{n}\sum_{i=1}^n \mathbf{X}_i\left\{\tilde{K}(-e_i(\beta_0)/h) - \tau\right\}\right] \\ &= E\left[\frac{1}{n}\sum_{i=1}^n \mathbf{X}_i E\left\{\tilde{K}(-e_i(\beta_0)/h) - \tau|\mathbf{X}_i\right\}\right].\end{aligned}$$

By conditions **C1** and **C2**,

$$\begin{aligned}& E\left\{\tilde{K}(-e_i(\beta_0)/h) - \tau|\mathbf{X}_i\right\} \\ =& \int_{-\infty}^{+\infty} K(u)\int_0^{-hu}\left\{f(\mathbf{X}_i^\top\beta_0 + t|\mathbf{X}_i^\top) - f(\mathbf{X}_i^\top\beta_0|\mathbf{X}_i^\top)\right\}dtdu \\ \leq& \int_{-\infty}^{+\infty} K(u)\int_0^{-hu}\left|f(\mathbf{X}_i^\top\beta_0 + t|\mathbf{X}_i^\top) - f(\mathbf{X}_i^\top\beta_0|\mathbf{X}_i^\top)\right|dtdu \\ \leq& l_3\int_{-\infty}^{+\infty} K(u)\int_0^{-hu}|t|dtdu = \frac{1}{2}l_3h^2\int_{-\infty}^{+\infty} u^2K(u)du = O(h^2),\end{aligned}$$

and by condition **C4**, we have

$$\left\|E\left\{S^{(1)}_h(\beta_0)\right\}\right\|_\infty \lesssim h^2. \tag{A.16}$$

Thus, by (A.14)-(A.16) and condition $h \lesssim (\log n/n)^{1/4}$, we have

$$\left\| S_h^{(1)}(\beta_0) \right\|_\infty = O_p\left(\sqrt{\frac{\log(n \vee p)}{n}} + h^2 \right) = O_p\left(\sqrt{\frac{\log(n \vee p)}{n}} \right). \tag{A.17}$$

For the initial estimator, we have $\hat{\beta}^0_{L,T^c} = 0$ with high probability, where $T^c = \{j \in \{1, \dots, p\} : \beta_{0,j} = 0\}$. Due to the fact that $\beta_{0,T^c} = 0$, by $\|\hat{\beta}^0_L - \beta_0\|_2 = O_p(\tilde{a}_n)$, and by Lemma1, we have

$$\begin{aligned} & \left\| \left\{ S_h^{(2)}(\hat{\beta}^0_L) - S_{h_1,1}^{(2)}(\hat{\beta}^0_L) \right\} (\hat{\beta}^0_L - \beta_0) \right\|_\infty \\ \leq & \left\| \left\{ S_h^{(2)}(\hat{\beta}^0_L) - S_{h_1,1}^{(2)}(\hat{\beta}^0_L) \right\}_{T \times T} \right\| \cdot \left\| (\hat{\beta}^0_L - \beta_0)_T \right\|_2 \\ = & O_p\left(\tilde{a}_n \sqrt{\frac{s \log(n \vee p)}{nh}} + \tilde{a}_n \sqrt{\frac{s \log(n_1 \vee p)}{n_1 h_1}} + \tilde{a}_n^2 + \tilde{a}_n h + \tilde{a}_n h_1 \right), \end{aligned} \tag{A.18}$$

Finally,

$$\begin{aligned} & \left\| S_h^{(1)}(\beta_0) - S_h^{(1)}(\hat{\beta}^0_L) - S_h^{(2)}(\hat{\beta}^0_L)(\beta_0 - \hat{\beta}^0_L) \right\|_\infty \\ \leq & \left\| \left\{ S_h^{(2)}(\hat{\beta}^0_L) - D \right\} (\hat{\beta}^0_L - \beta_0) \right\|_\infty + \left\| S_h^{(1)}(\beta_0) - S_h^{(1)}(\hat{\beta}^0_L) - D(\beta_0 - \hat{\beta}^0_L) \right\|_\infty . \end{aligned} \tag{A.19}$$

Similar to (A.18), we get

$$\left\| \left\{ S_h^{(2)}(\hat{\beta}^0_L) - D \right\} (\hat{\beta}^0_L - \beta_0) \right\|_\infty = O_p\left(\tilde{a}_n \sqrt{\frac{s \log(n \vee p)}{nh}} + \tilde{a}_n^2 + \tilde{a}_n h \right). \tag{A.20}$$

Denote $\mathbf{A} = S_h^{(1)}(\beta_0) - S_h^{(1)}(\hat{\beta}^0_L) - D(\beta_0 - \hat{\beta}^0_L)$, thus

$$\|\mathbf{A}\|_\infty \leq \|E\mathbf{A}\|_\infty + \|\mathbf{A} - E\mathbf{A}\|_\infty .$$

By conditions **C1** and **C2**,

$$\begin{aligned}
E\mathbf{A} &= E\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{X}_i\mathbf{X}_i^\top\int_{-\infty}^{+\infty}K(u)\left\{f(\mathbf{X}_i^\top\hat{\beta}_L^0 - uh|\mathbf{X}_i^\top) - f(\mathbf{X}_i^\top\beta_0|\mathbf{X}_i^\top)\right\}du\right](\beta_0 - \hat{\beta}_L^0) \\
&\lesssim E\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{X}_i\mathbf{X}_i^\top\int_{-\infty}^{+\infty}K(u)\left\{|\mathbf{X}_i^\top(\beta_0 - \hat{\beta}_L^0)| + |uh|\right\}du\right](\beta_0 - \hat{\beta}_L^0) \\
&\lesssim E\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{X}_i\mathbf{X}_i^\top\left\{|\mathbf{X}_i^\top(\beta_0 - \hat{\beta}_L^0)| + h\right\}\right](\beta_0 - \hat{\beta}_L^0).
\end{aligned}$$

Thus, we have $\|E\mathbf{A}\|_\infty = O_p(\tilde{a}_n^2 + \tilde{a}_n h)$. By the Theorem A.3 in Spokoiny (2013), we can obtain

$$\|\mathbf{A} - E\mathbf{A}\|_\infty = O_p\left(\tilde{a}_n\sqrt{\frac{s\log(n\vee p)}{nh}}\right).$$

Then

$$\left\|S_h^{(1)}(\beta_0) - S_h^{(1)}(\hat{\beta}_l^0) - D(\beta_0 - \hat{\beta}_l^0)\right\|_\infty = O_p\left(\tilde{a}_n\sqrt{\frac{s\log(n\vee p)}{nh}} + \tilde{a}_n^2 + \tilde{a}_n h\right). \tag{A.21}$$

By (A.19)-(A.21), we have

$$\left\|S_h^{(1)}(\beta_0) - S_h^{(1)}(\hat{\beta}_L^0) - S_h^{(2)}(\hat{\beta}_L^0)(\beta_0 - \hat{\beta}_L^0)\right\|_\infty \le O_p\left(\tilde{a}_n\sqrt{\frac{s\log(n\vee p)}{nh}} + \tilde{a}_n^2 + \tilde{a}_n h\right). \tag{A.22}$$

Finally, combing (A.13), (A.17), (A.18) and (A.22), we can obtain

$$\begin{aligned}
\left\|\bar{S}_{h,h_1}^{(1)}(\beta_0, \hat{\beta}_L^0)\right\|_\infty \le& \left\|S_h^{(1)}(\beta_0)\right\|_\infty + \left\|\left\{S_h^{(2)}(\hat{\beta}_L^0) - S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\right\}(\hat{\beta}_L^0 - \beta_0)\right\|_\infty \\
&- \left\|S_h^{(1)}(\beta_0) - S_h^{(1)}(\hat{\beta}_L^0) - S_h^{(2)}(\hat{\beta}_L^0)(\beta_0 - \hat{\beta}_L^0)\right\|_\infty \\
=& O_p\left(\sqrt{\frac{\log n}{n}} + \tilde{a}_n\left\{\sqrt{\frac{s\log(n\vee p)}{nh}} + \sqrt{\frac{s\log(n_1\vee p)}{n_1 h_1}} + \tilde{a}_n + \tilde{a}_n h + h_1\right\}\right).
\end{aligned}$$

This completes the proof. □

Proof of Theorem 4.1. We first proof that

$$\|(\hat\beta_L^1 - \beta_0)_{T^c}\|_1 \le 3\|(\hat\beta_L^1 - \beta_0)_T\|_1,$$

where T is the support of β_0. By the definition of $\hat\beta_L^1$ in (4.1), we have

$$\bar S_{h,h_1}(\hat\beta_L^1, \hat\beta_L^0) - \bar S_{h,h_1}(\beta_0, \hat\beta_L^0) + \lambda_n\|\hat\beta_L^1\|_1 - \lambda_n\|\beta_0\|_1 \le 0.$$

Therefore, we can obtain

$$\begin{aligned}&\bar S_{h,h_1}(\hat\beta_L^1, \hat\beta_L^0) - \bar S_{h,h_1}(\beta_0, \hat\beta_L^0)\\ \le&\lambda_n\Big(\|\beta_0\|_1 - \|\hat\beta_L^1\|_1\Big) = \lambda_n\Big(\|\beta_{0,T}\|_1 - \|\hat\beta_{L,T}^1\|_1 - \|\hat\beta_{L,T^c}^1\|_1\Big)\\ \le&\lambda_n\|(\hat\beta_L^1 - \beta_0)_T\|_1 - \lambda_n\|(\hat\beta_L^1 - \beta_0)_{T^c}\|_1.\end{aligned} \tag{A.23}$$

Note that

$$\begin{aligned}&\bar S_{h,h_1}(\hat\beta_L^1, \hat\beta_L^0) - \bar S_{h,h_1}(\beta_0, \hat\beta_L^0) - (\hat\beta_L^1 - \beta_0)^\top \bar S_{h,h_1}^{(1)}(\beta_0, \hat\beta_L^0)\\ =&\frac{1}{2}(\hat\beta_L^1 - \beta_0)^\top S_{h,h_1}^{(2)}(\hat\beta_L^0)(\hat\beta_L^1 - \beta_0).\end{aligned} \tag{A.24}$$

Take $\bar C$ large enough such that $\lambda_n \ge 2\|\bar S_{h,h_1}^{(1)}(\beta_0, \hat\beta_L^0)\|_\infty$ with probability tending to one, and by the conditions $K(u) \ge 0$ and , we have

$$\begin{aligned}&\bar S_{h,h_1}(\hat\beta_L^1, \hat\beta_L^0) - \bar S_{h,h_1}(\beta_0, \hat\beta_L^0)\\ =&(\hat\beta_L^1 - \beta_0)^\top \bar S_{h,h_1}^{(1)}(\beta_0) + \frac{1}{2}(\hat\beta_L^1 - \beta_0)^\top S_{h,h_1}^{(2)}(\hat\beta_L^0)(\hat\beta_L^1 - \beta_0)\\ \ge&(\hat\beta_L^1 - \beta_0)^\top \bar S_{h,h_1}^{(1)}(\beta_0, \hat\beta_L^0)\\ \ge&-\|\hat\beta_L^1 - \beta_0\|_1\left\|\bar S_{h,h_1}^{(1)}(\beta_0, \hat\beta_L^0)\right\|_\infty\\ \ge&-\frac{1}{2}\lambda_n\|\hat\beta_L^1 - \beta_0\|_1.\end{aligned} \tag{A.25}$$

Thus, by (A.23) and (A.25), we can obtain

$$-\frac{1}{2}\lambda_n\|\hat\beta_L^1 - \beta_0\|_1 \le \lambda_n\|(\hat\beta_L^1 - \beta_0)_T\|_1 - \lambda_n\|(\hat\beta_L^1 - \beta_0)_{T^c}\|_1.$$

After rearranging, we have

$$\|(\hat{\beta}_L^1-\beta_0)_{T^c}\|_1 \le 3\|(\hat{\beta}_L^1-\beta_0)_T\|_1. \tag{A.26}$$

Thus, by (A.26),

$$\|\hat{\beta}_L^1-\beta_0\|_1 \le 4\|(\hat{\beta}_L^1-\beta_0)_T\|_1 \le 4\sqrt{s}\|(\hat{\beta}_L^1-\beta_0)_T\|_2 \le 4\sqrt{s}\|\hat{\beta}_L^1-\beta_0\|_2. \tag{A.27}$$

By (A.23) and (A.24), we have

$$\begin{aligned}
&\frac{1}{2}(\hat{\beta}_L^1-\beta_0)^\top S_{h,h_1}^{(2)}(\hat{\beta}_L^0)(\hat{\beta}_L^1-\beta_0)\\
=&\bar{S}_{h,h_1}(\hat{\beta}_L^1,\hat{\beta}_L^0)-\bar{S}_{h,h_1}(\beta_0,\hat{\beta}_L^0)-(\hat{\beta}_L^1-\beta_0)^\top\bar{S}_{h,h_1}^{(1)}(\beta_0)\\
\le&\lambda_n\left(\|\beta_0\|_1-\|\hat{\beta}_L^1\|_1\right)-(\hat{\beta}_L^1-\beta_0)^\top\bar{S}_{h,h_1}^{(1)}(\beta_0,\hat{\beta}_L^0) \qquad (A.28)\\
\le&\lambda_n\|\hat{\beta}_L^1-\beta_0\|_1+\|\hat{\beta}_L^1-\beta_0\|_1\left\|\bar{S}_{h,h_1}^{(1)}(\beta_0)\right\|_\infty\\
\le&\frac{3}{2}\lambda_n\|\hat{\beta}_L^1-\beta_0\|_1.
\end{aligned}$$

Finally, by the condition **C4**, for any $\delta \in R^p$ and some constants $C_7 > 0$, we get

$$\min_{\delta:|\delta|_1\lesssim\sqrt{s}\|\delta\|_2}\frac{\delta^\top\Sigma\delta}{\|\delta\|_2^2} \ge C_7. \tag{A.29}$$

Then, by Lemma 1, (A.27)-(A.29), we have

$$\begin{aligned}
\|\hat{\beta}_L^1-\beta_0\|_2^2 \le&\Lambda_{\min}^{-1}(\Sigma)(\hat{\beta}_L^1-\beta_0)^\top\Sigma(\hat{\beta}_L^1-\beta_0)\\
\le&l_1^{-1}\Lambda_{\min}^{-1}(\Sigma)(\hat{\beta}_L^1-\beta_0)^\top\mathbf{D}(\hat{\beta}_L^1-\beta_0)\\
=&l_1^{-1}\Lambda_{\min}^{-1}(\Sigma)(\hat{\beta}_L^1-\beta_0)^\top\left\{S_{h,h_1}^{(2)}(\hat{\beta}_L^0)+\mathbf{D}-S_{h,h_1}^{(2)}(\hat{\beta}_L^0)\right\}(\hat{\beta}_L^1-\beta_0)\\
\le&2l_1^{-1}\Lambda_{\min}^{-1}(\Sigma)(\hat{\beta}_L^1-\beta_0)^\top S_{h,h_1}^{(2)}(\hat{\beta}_L^0)(\hat{\beta}_L^1-\beta_0)\\
\le&3l_1^{-1}\Lambda_{\min}^{-1}(\Sigma)\lambda_n\|\hat{\beta}_L^1-\beta_0\|_1\\
\le&12l_1^{-1}\Lambda_{\min}^{-1}(\Sigma)\lambda_n\sqrt{s}\|\hat{\beta}_L^1-\beta_0\|_2.
\end{aligned}$$

Thus, $\|\hat{\beta}_L^1 - \beta_0\|_2 \leq 12 l_1^{-1} \Lambda_{\min}^{-1}(\Sigma)\lambda_n\sqrt{s}$. Finally, by the Lemma 2, we can proof the theorem. $\square$

Proof of Theorem 4.2. We can proved Theorem 4.2 directly from the proof of Theorem 4.1. $\square$

Proof of Theorem 4.3. Theorem 4.3 (i) follows directly from the proof of Theorem 4.1. Now we consider Theorem 4.3 (ii). Define $\tilde{\beta}$ to be the solution of the following optimization problem:

$$\tilde{\beta} = \arg\min_{\beta, \beta_{T^c}=\mathbf{0}} \left[\beta^\top \left\{S_h^{(1)}(\hat{\beta}_L^0) - S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\hat{\beta}_L^0\right\} + \frac{1}{2}\beta^\top S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\beta + \lambda_n\|\beta\|_1\right],$$

where $\beta_{T^c} = \mathbf{0}$ denotes the subset vector with the coordinates of β in $T^c = \{j \in \{1, \ldots, p\} : \beta_{0,j} = 0\}$. Then there exist sub-gradients $\tilde{\mathbf{Z}}$ with $|\tilde{\mathbf{Z}}|_\infty \leq 1$ such that

$$\left\{S_h^{(1)}(\hat{\beta}_L^0) - S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\hat{\beta}_L^0\right\}_T + \tilde{\beta}_T^\top \left\{S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\right\}_{T\times T} + \lambda_n\tilde{\mathbf{Z}}_T = 0.$$

Then, we have

$$\begin{aligned}\tilde{\beta}_T - \beta_{0,T} =& \mathbf{D}_{T\times T}^{-1}\Big[-\lambda_n\tilde{\mathbf{Z}}_T - \left\{S_{h_1,1}^{(2)}(\hat{\beta}_L^0) - \mathbf{D}\right\}_{T\times T}(\tilde{\beta}_T - \beta_{0,T}) \\ & - \left\{S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\right\}_{T\times T}\beta_{0,T} - \left\{S_h^{(1)}(\hat{\beta}_L^0) - S_{h_1,1}^{(2)}(\hat{\beta}_L^0)\hat{\beta}_L^0\right\}_T\Big] \\ =& \mathbf{D}_{T\times T}^{-1}\Big[-\lambda_n\tilde{\mathbf{Z}}_T - \left\{S_{h_1,1}^{(2)}(\hat{\beta}_L^0) - \mathbf{D}\right\}_{T\times T}(\tilde{\beta}_T - \beta_{0,T}) \\ & - \left\{S_h^{(1)}(\hat{\beta}_L^0) - S_{h_1,1}^{(2)}(\hat{\beta}_L^0)(\hat{\beta}_L^0 - \beta_0)\right\}_T\Big].\end{aligned}$$

By Lemmas 1 and 2, for a sufficiently large constant $\tilde{C}$ and the choice of λ_n, we can obtain

$$\begin{aligned}\|\tilde{\beta}_T - \beta_{0,T}\|_\infty \leq& \tilde{C}\|\mathbf{D}_{T\times T}^{-1}\|_\infty\Big\{\sqrt{\frac{\log(n\vee p)}{n}} \\ & + \tilde{a}_n\Big(\sqrt{\frac{s\log(n\vee p)}{nh}} + \sqrt{\frac{s\log(n_1\vee p)}{n_1h_1}} + \tilde{a}_n + h + h_1\Big)\Big\}\end{aligned}$$

Note that $P(\tilde{\beta} = \hat{\beta}_L^1) \to 1$. Then Theorem 4.3 (ii) follows from the above and together with the lower bound condition on $\min_{j \in T} |\beta_{0,j}|$. $\square$

Proof of Theorem 4.4. We can proved Theorem 4.4 directly from the proof of Theorem 4.3. $\square$

References

Alex, N., Hasenfuss, A., Hammer, B., 2009. Patch clustering for massive data sets. Neurocomputing 72, 1455–1469.

Barzilai, J., Borwein, J., 1988. Two-point step size gradient methods. IMA Journal of Numerical Analysis 8, 141–148.

Belloni, A., Chernozhukov, V., 2011. L1-penalized quantile regression in high-dimensional sparse models. The Annals of Statistics 39, 82–130.

Cai, T., Liu, W., 2011. Adaptive thresholding for sparse covariance matrix estimation. Journal of the American Statistical Association 106, 672–684.

Cai, T., Zhang, C.-H., Zhou, H., 2010. Optimal rates of convergence for covariance matrix estimation. The Annals of Statistics 38, 2118–2144.

Chen, L., Zhou, Y., 2020. Quantile regression in big data: A divide and conquer based strategy. Computational Statistics & Data Analysis 144, 106892.

Chen, X., Lee, J., Li, H., Yang, Y., 2021. Distributed estimation for principal component analysis: An enlarged eigenspace analysis. Journal of the American Statistical Association, 1–31.

Chen, X., Liu, W., Mao, X., Yang, Z., 2020. Distributed high-dimensional regression under a quantile loss function. Journal of Machine Learning Research 21, 1–43.

Chen, X., Liu, W., Zhang, Y., 2019. Quantile regression under memory constraint. The Annals of Statistics 47, 3244–3273.

Fan, J., Han, F., Liu, H., 2014. Challenges of big data analysis. National Science Review 1, 293–314.

Fan, J., Li, R., 2011. Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of the American Statistical Association 94, 1348–1360.

Fernandes, M., Guerre, E., Horta, E., 2021. Smoothing quantile regressions. Journal of Business & Economic Statistics 39, 338–357.

He, X., Pan, X., Tan, K. M., Zhou, W., 2020. Smoothed quantile regression with large scale inference. arXiv: Statistics Theory.

He, X., Shao, Q.-M., 2000. On parameters of increasing dimensions. Journal of Multivariate Analysis 73, 120–135.

He, Y., Li, H., Wang, S., Yao, X., 2021. Uncertainty analysis of wind power probability density forecasting based on cubic spline interpolation and support vector quantile regression. Neurocomputing 430, 121–137.

Horowitz, J., 1998. Bootstrap methods for median regression models. Econometrica 66, 1327–1352.

Jiang, H., 2018. Sparse estimation based on square root nonconvex optimization in high-dimensional data. Neurocomputing 282, 122–135.

Jiang, R., Hu, X., Yu, K., Qian, W., 2018. Composite quantile regression for massive datasets. Statistics 52, 980–1004.

Jordan, M., Lee, J., Yang, Y., 2019. Communication-efficient distributed statistical learning. Journal of the American Statistical Association 14, 668–681.

Kaplan, D., Sun, Y., 2017. Smoothed estimating equations for instrumental variables quantile regression. Econometric Theory 33, 105–157.

Koenker, R., 2005. Quantile regression. Cambridge University Press, Cambridge.

Koenker, R., Bassett, G., 1978. Regression quantile. Econometrica 46, 33–50.

Lee, J., Liu, Q., Sun, Y., Taylor, J., 2017. Communication-efficient sparse regression: a one-shot approach. Journal of Machine Learning Research 18, 1–30.

Lv, S.-G., Ma, T.-F., Liu, L., Feng, Y.-L., 2013. Fast learning rates for sparse quantile regression problem. Neurocomputing 108, 13–22.

Pietrosanu, M., Gao, J., Kong, L., Jiang, B., Niu, D., 2020. Advanced algorithms for penalized quantile and composite quantile regression. Computational Statistics, DOI: 10.1007/S00180–020–01010–1.

Portnoy, S., Koenker, R., 1997. The gaussian hare and the laplacian tortoise: computability of squared-error versus absolute-error estimators. Statistical Science 12, 279–300.

Schmidt, M., 2010. Graphical model structure learning with ℓ_1-regularization. PhD thesis, University of British Columbia.

Solntsev, S., Nocedal, J., Byrd, R., 2015. An algorithm for quadratic ℓ_1-regularized optimization with a flexible active-set strategy. Optimization Methods and Software 30, 1213–1237.

Spokoiny, V., 2013. Bernstein-von mises theorem for growing parameter dimension. arXiv:1302.3430.

Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B 58, 267–288.

van de Geer, S., Bhlmann, P., Ritov, Y., Dezeure, R., 2014. On asymptotically optimal confidence regions and tests for high-dimensional models. The Annals of Statistics 42, 1166–1202.

Volgushev, S., Chao, S.-K., Cheng, G., 2019. Distributed inference for quantile regression processes. The Annals of Statistics 47, 1634–1662.

Wainwright, M., 2009. Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (lasso). IEEE Transactions on Information Theory 55, 2183–2202.

Wang, L., Lian, H., 2020. Communication-efficient estimation of high-dimensional quantile regression. Analysis and Applications 18, 1057–1075.

Wang, Y., Wang, S., 2013. Estimating α-frontier technical efficiency with shape-restricted kernel quantile regression. Neurocomputing 101, 243–251.

Xu, Q., Cai, C., Jiang, C., Sun, F., Huang, X., 2020. Block average quantile regression for massive dataset. Statistical Papers 61, 141–165.

Yang, L., 1997. Solving sparse least squares problems on massively distributed memory computers. International Conference on Advances in Parallel and Distributed Computing, 170–177.

Yu, D., Kong, L., Mizera, I., 2016. Partial functional linear quantile regression for neuroimaging data analysis. Neurocomputing 195, 74–87.

Zhang, C.-H., 2010. Nearly unbiased variable selection under minimax concave penalty. The Annals of Statistics 38, 894–942.

Zhao, W., Zhang, F., Lian, H., 2020. Debiasing and distributed estimation for high-dimensional quantile regression. IEEE Transactions on Neural Networks and Learning Systems 31, 2569–2577.