Quantile Regression for Large-scale Applications

Jiyan Yang JIYAN@STANFORD.EDU
ICME, Stanford University, Stanford, CA 94305

Xiangrui Meng XIMENG@LINKEDIN.COM
LinkedIn Corporation, 2029 Stierlin Ct, Mountain View, CA 94043

Michael W. Mahoney MMAHONEY@CS.STANFORD.EDU
Dept. of Mathematics, Stanford University, Stanford, CA 94305

Abstract

Quantile regression is a method to estimate the quantiles of the conditional distribution of a response variable, and as such it permits a much more accurate portrayal of the relationship between the response variable and observed covariates than methods such as Least-squares or Least Absolute Deviations regression. It can be expressed as a linear program, and interior-point methods can be used to find a solution for moderately large problems. Dealing with very large problems, *e.g.*, involving data up to and beyond the terabyte regime, remains a challenge. Here, we present a randomized algorithm that runs in time that is nearly linear in the size of the input and that, with constant probability, computes a $(1+\epsilon)$ approximate solution to an arbitrary quantile regression problem. Our algorithm computes a low-distortion subspace-preserving embedding with respect to the loss function of quantile regression. Our empirical evaluation illustrates that our algorithm is competitive with the best previous work on small to medium-sized problems, and that it can be implemented in MapReduce-like environments and applied to terabyte-sized problems.

1. Introduction

Quantile regression is a method to estimate the quantiles of the conditional distribution of a response variable, expressed as functions of observed covariates (Koenker & Bassett, 1978), in a manner analogous to the way in which Least-squares regression estimates the conditional mean. The Least Absolute Deviations regression (*i.e.*, ℓ_1 regression) is a special case of quantile regression that involves computing the median of the conditional distribution. In contrast with ℓ_1 regression and the more popular ℓ_2 or Least-squares regression, quantile regression involves minimizing asymmetrically-weighted absolute residuals. Doing so, however, permits a much more accurate portrayal of the relationship between the response variable and observed covariates, and it is more appropriate in certain non-Gaussian settings. For these reasons, quantile regression has found applications in many areas (Buchinsky, 1994; Koenker & Hallock, 2001; Buhai, 2005). As with ℓ_1 regression, the quantile regression problem can be formulated as a linear programming problem, and thus simplex or interior-point methods can be applied (Koenker & D'Orey, 1993; Portnoy & Koenker, 1997; Portnoy, 1997). Most of these methods are efficient only for problems of small to moderate size, and thus to solve very large-scale quantile regression problems more reliably and efficiently, we need new computational techniques.

In this paper, we provide a fast algorithm to compute a $(1+\epsilon)$ relative-error approximate solution to the over-constrained quantile regression problem. Our algorithm constructs a low-distortion subspace embedding of the form that has been used in recent developments in randomized algorithms for matrices and large-scale data problems; and our algorithm runs in time that is nearly linear in the number of nonzeros in the input data.

In more detail, recall that a quantile regression problem can be specified by a (design) matrix $A \in \mathbb{R}^{n\times d}$, a (response) vector $b \in \mathbb{R}^n$, and a parameter $\tau \in (0,1)$, in which case the quantile regression problem can be

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

solved via the optimization problem

$$\text{minimize}_{x\in\mathbb{R}^d} \quad \rho_\tau(b - Ax), \tag{1}$$

where $\rho_\tau(x) = \sum_{i=1}^d \rho_\tau(x_i)$, for $x \in \mathbb{R}^d$, where

$$\rho_\tau(z) = \begin{cases} \tau z, & z \geq 0; \\ (\tau - 1)z, & z < 0, \end{cases} \tag{2}$$

for $z \in \mathbb{R}$, is the corresponding loss function. In the remainder of this paper, we will use A to denote the augmented matrix $\begin{bmatrix} b & -A \end{bmatrix}$, and we will consider $A \in \mathbb{R}^{n\times d}$. With this notation, the quantile regression problem of (1) can equivalently be expressed as a constrained problem with a single linear constraint,

$$\text{minimize}_{x\in\mathcal{C}} \quad \rho_\tau(Ax), \tag{3}$$

where $\mathcal{C} = \{x \in \mathbb{R}^d \mid c^T x = 1\}$ and c is a unit vector with the first coordinate set to be 1. We will focus on problems with size $n \gg d$.

Our main algorithm depends on a technical result, presented as Lemma 3 below, of independent interest. Let $A \in \mathbb{R}^{n\times d}$ be an input matrix, and let $S \in \mathbb{R}^{s\times n}$ be a random sampling matrix constructed based on the following importance sampling probabilities,

$$p_i = \min\{1, s \cdot \|U_{(i)}\|_1 / \|U\|_1\},$$

where $\|\cdot\|_1$ is the element-wise norm, and where $U_{(i)}$ is the i-th row of an ℓ_1 well-conditioned basis U for the range of A (see Definition 2 below). Then, Lemma 3 states that, for a sampling complexity s that depends on d but is independent of n,

$$(1 - \epsilon)\rho_\tau(Ax) \leq \rho_\tau(SAx) \leq (1 + \epsilon)\rho_\tau(Ax)$$

will be satisfied for every $x \in \mathbb{R}^d$.

Although one could use, *e.g.*, the algorithm of (Dasgupta et al., 2009) to compute such a well-conditioned basis U and then "read off" the 1-norm of the rows of U, doing so would be much slower than the time allotted by our main algorithm. Thus, to apply the ideas from Lemma 3 for fast quantile regression, we provide two algorithms, Algorithm 1 and Algorithm 2 below. As quantified by Lemma 4 and Lemma 5, respectively, these two algorithms provide fast construction of the well-conditioned basis U and fast estimation of the ℓ_1 norms of $U_{(i)}$, respectively; and both run in $\mathcal{O}(\text{nnz}(A) \cdot \log n)$ time, where $\text{nnz}(A)$ is the number of nonzero elements of A.

Given these results, our main algorithm for quantile regression is presented as Algorithm 3. Our main theorem for this algorithm, Theorem 1 below, states that, with constant probability, this algorithm returns a $(1 + \epsilon)$-approximate solution to the quantile regression problem; and that this solution can be obtained in time $\mathcal{O}(\text{nnz}(A)\cdot\log n)$ plus the time for solving the subproblem, whose size is $\mathcal{O}(\mu d^3 \log(\mu/\epsilon)/\epsilon^2) \times d$, where $\mu = \frac{\tau}{1-\tau}$, for $\tau \geq 1/2$, independent of n.

Our empirical evaluation results show that the output of our algorithm is 2-digit accurate in terms of both objective value and solution to quantile regression by sampling, *e.g.*, about 0.001% of the data. It outperforms other conditioning-based methods, and the running time of the proposed algorithm is comparable to existing sampling methods in RAM. In addition, our algorithm can be implemented in MapReduce-like environments and applied to terabyte-sized problems.

The best previous algorithm for moderately large quantile regression problems is due to (Portnoy & Koenker, 1997) and (Portnoy, 1997). Their algorithm uses an interior-point method on a smaller problem that has been preprocessed by randomly sampling a subset of the data. Their preprocessing step involves predicting the sign of each $A_{(i)}x^* - b_i$, where $A_{(i)}$ and b_i are the i-th row of the input matrix and the response vector, respectively, and x^* is an optimal solution to the original problem. When compared with our approach, they compute an optimal solution, while we compute an approximate solution; but we provide worst-case analysis that with high probability our algorithm is guaranteed to work, while they do not. Also, the sampling complexity of their algorithm depends on the higher dimension n, while the number of samples required by our algorithm depends only on the lower dimension d; but our sampling is with respect to a carefully-constructed nonuniform distribution, while they sample uniformly at random.

For a detailed overview of recent work on using randomized algorithms to compute approximate solutions for least-squares regression and related problems, see the recent review (Mahoney, 2011). Most relevant for our work is the algorithm of (Dasgupta et al., 2009) that constructs a well-conditioned basis by ellipsoid rounding and a subspace-preserving sampling matrix in order to approximate the solution of general ℓ_p regression problems, for $p \in [1, \infty)$, in roughly $\mathcal{O}(nd^5 \log n)$; the algorithms of (Sohler & Woodruff, 2011) and (Clarkson et al., 2013) that use the "slow" and "fast' versions of Cauchy Transform to obtained a low-distortion ℓ_1 embedding matrix and solve the over-constrained ℓ_1 regression problem in $\mathcal{O}(nd^{1.376+})$ and $\mathcal{O}(nd \log n)$ time, respectively; and the algorithm of (Meng & Mahoney, 2013) that constructs low-distortion embeddings in "input sparsity"

time and uses those embeddings to approximate the solution of the over-constrained ℓ_1 regression problem in $\mathcal{O}(\mathrm{nnz}(A)\cdot\log n+\mathrm{poly}(d)\log(1/\epsilon)/\epsilon^2)$ time. In particular, we will use the method in (Meng & Mahoney, 2013) for constructing ℓ_1-norm well-conditioned basis matrices in nearly input-sparsity time.

A more detailed empirical evaluation and the proofs of our main (and additional) results may be found in the associated technical report (Yang et al., 2013).

2. Preliminaries

We use $\|\cdot\|_1$ to denote the element-wise ℓ_1 norm for both vectors and matrices; and we use $[n]$ to denote the set $\{1,2,\ldots,n\}$. For any matrix A, $A_{(i)}$ and $A^{(j)}$ denote the i-th row and the j-th column of A, respectively; and $\mathcal{A}$ denotes the column space of A. For simplicity, we assume A has full column rank; and we always assume that $\tau\geq\frac{1}{2}$. All the results hold for $\tau<\frac{1}{2}$ by simply switching the positions of τ and $1-\tau$.

Although $\rho_\tau(\cdot)$ is not a norm, since the loss function does not have the positive linearity, it satisfies some "good" properties, as stated in the following lemma:

Lemma 1. *Suppose that $\tau\geq\frac{1}{2}$. Then, for any $x,y\in\mathbb{R}^d, a\geq 0$, the following hold: $\rho_\tau(x+y)\leq\rho_\tau(x)+\rho_\tau(y)$; $(1-\tau)\|x\|_1\leq\rho_\tau(x)\leq\tau\|x\|_1$; $\rho_\tau(ax)=a\rho_\tau(x)$; and $|\rho_\tau(x)-\rho_\tau(y)|\leq\tau\|x-y\|_1$.*

The following notion of a low-distortion embedding will be crucial for our method. In this paper, the "measure functions" we will consider are $\|\cdot\|_1$ and $\rho_\tau(\cdot)$.

Definition 1 (Low-distortion embedding)**.** *Given $A\in\mathbb{R}^{n\times d}$, a measure function of vectors $f(\cdot)$, a matrix $S\in\mathbb{R}^{s\times n}$ is a $(1\pm\epsilon)$-distortion embedding matrix of $(\mathcal{A},f(\cdot))$ if $s=\mathrm{poly}(d)$ and for all $x\in\mathbb{R}^d$,*

$$(1-\epsilon)f(Ax)\leq f(SAx)\leq(1+\epsilon)f(Ax).$$

We will say that S is a $(1\pm\epsilon)$-distortion sampling matrix if S is a $(1\pm\epsilon)$-distortion embedding matrix and there is only one nonzero element per row in S.

The following notion of a basis, originally introduced by (Dasgupta et al., 2009), that is well-conditioned for the ℓ_1 norm will also be crucial for our method.

Definition 2 (Well-conditioned basis)**.** *Given $A\in\mathbb{R}^{n\times d}$, a basis U of $\mathcal{A}$ is (α,β)-conditioned if $\|U\|_1\leq\alpha$ and for all $x\in\mathbb{R}^q$, $\|x\|_\infty\leq\beta\|Ux\|_1$. We will say that U is a well-conditioned basis of A if α and β are low-degree polynomials in d, independent of n.*

For completeness, note that two important ingredients for proving subspace preservation are γ-nets and tail inequalities. Suppose that Z is a point set and $\|\cdot\|$ is a metric on Z. A subset Z_γ is called as a γ-net for some $\gamma>0$ if for every $x\in Z$ there is a $y\in Z_\gamma$ such that $\|x-y\|\leq\gamma$. It is well-known that the unit ball of a d-dimensional subspace has a γ-net with size at most $(3/\gamma)^d$ (Bourgain et al., 1989). Also, we use the standard Bernstein inequality to prove concentration results for the sum of independent random variables.

Lemma 2 (Bernstein inequality)**.** *Let $X_1,\ldots,X_n$ be independent random variables with zero-mean. Suppose that $|X_i|\leq M$, for $i\in[n]$, then for any positive number t, we have*

$$\Pr\left[\sum_{i\in[n]}X_i>t\right]\leq\exp\left(-\frac{t^2/2}{\sum_{i\in[n]}\mathbf{E}\,X_j^2+Mt/3}\right).$$

3. Main Theoretical Results

Here, we present our main technical results on low-distortion subspace-preserving embeddings and our fast randomized algorithm for quantile regression.

3.1. Main technical ingredients

We start with a result which says that if we sample sufficiently many (but still only $\mathrm{poly}(d)$) rows according to an appropriately-defined non-uniform importance sampling distribution (of the form given in Eqn. (4) below), then we obtain a $(1\pm\epsilon)$-distortion embedding matrix with respect to the loss function of quantile regression. Note that the form of this lemma, the proof of which may be found in (Yang et al., 2013), is based on ideas from (Dasgupta et al., 2009; Clarkson et al., 2013).

Lemma 3 (Subspace-preserving Sampling Lemma)**.** *Given $A\in\mathbb{R}^{n\times d}$, let $U\in\mathbb{R}^{n\times d}$ be an (α,β)-conditioned basis for $\mathcal{A}$. For $s>0$, define*

$$\hat{p}_i\geq\min\{1,s\cdot\|U_{(i)}\|_1/\|U\|_1\},\tag{4}$$

and let $S\in\mathbb{R}^{n\times n}$ be a random diagonal matrix with $S_{ii}=1/\hat{p}_i$ with probability $\hat{p}_i$, and 0 otherwise. Then when $\epsilon<1/2$ and

$$s\geq\frac{\tau}{1-\tau}\frac{27\alpha\beta}{\epsilon^2}\left(d\log\left(\frac{\tau}{1-\tau}\frac{18}{\epsilon}\right)+\log\left(\frac{4}{\delta}\right)\right),$$

with probability at least $1-\delta$, for every $x\in\mathbb{R}^d$,

$$(1-\varepsilon)\rho_\tau(Ax)\leq\rho_\tau(SAx)\leq(1+\varepsilon)\rho_\tau(Ax).\tag{5}$$

Remark. It is not hard to see that for any matrix S satisfying (5), the rank of A is preserved.

Algorithm 1 Fast Algorithm for Computing Well-conditioned Basis (Meng & Mahoney, 2013)

1: **Input:** $A \in \mathbb{R}^{n\times d}$ with full column rank.
2: **Output:** $R^{-1} \in \mathbb{R}^{d\times d}$ such that AR^{-1} is an (α,β)-conditioned basis with $\alpha\beta \le 6d^2$.
3: Construct a low-distortion embedding matrix $\Pi_1 \in \mathbb{R}^{r_1\times n}$ of $(\mathcal{A}, \|\cdot\|_1)$.
4: Construct $\tilde{R} \in \mathbb{R}^{d\times d}$ such that $A\tilde{R}^{-1}$ is a well-conditioned basis for $\mathcal{A}$ (For example, by QR factorization of $\Pi_1 A$).
5: Compute a $(1\pm 1/2)$-distortion embedding matrix $\tilde{S} \in \mathbb{R}^{\mathrm{poly}(d)\times n}$ of $(\mathcal{A}, \|\cdot\|_1)$.
6: Compute $R \in \mathbb{R}^{d\times d}$ by ellipsoid rounding such that $\tilde{S}AR^{-1}$ is a (α,β)-conditioned basis with $\alpha\beta \le 2d^2$.

Remark. Given such a low-distortion subspace-preserving sampling matrix, it is not hard to show that, by solving the sub-sampled problem induced by S, *i.e.*, solving $\min_{x\in\mathcal{C}} \rho_\tau(SAx)$, then one obtains a $(1+\epsilon)/(1-\epsilon)$-approximate solution to the original problem. For more details, see (Yang et al., 2013).

In order to apply Lemma 3, we need to compute the sampling probabilities in Eqn. (4). This requires two steps: first, find a well-conditioned basis U; and second, compute the row norms of U. We now present two algorithms that will perform these two steps in the allotted $\mathcal{O}(\mathrm{nnz}(A)\cdot\log n)$ time.

Consider, first, Algorithm 1, which computes a well-conditioned basis for $\mathcal{A}$. This algorithm originally appeared as first four steps of Algorithm 2 in (Meng & Mahoney, 2013), but it is included here for completeness. Our main result for Algorithm 1 is given in Lemma 4.

Lemma 4. *Given $A \in \mathbb{R}^{n\times d}$ with full rank, Algorithm 1 takes $\mathcal{O}(\mathrm{nnz}(A)\cdot\log n)$ time to compute a matrix $R \in \mathbb{R}^{d\times d}$ such that with a constant probability, AR^{-1} is an (α,β)-conditioned basis for $\mathcal{A}$ with $\alpha\beta \le 6d^2$.*

Remark. The output of Algorithm 1 is *not* the well-conditioned matrix U, but instead it is the matrix R, the inverse of which transforms A into U.

Remark. A well-conditioned basis for $\mathcal{A}$ can also be computed by other (typically more expensive) approaches, *e.g.*, the methods from (Dasgupta et al., 2009; Sohler & Woodruff, 2011; Clarkson et al., 2013) or the other algorithm of (Meng & Mahoney, 2013).

Consider, next, computing $\hat{p}_i$ from U (or from A and R^{-1}), and note that forming U explicitly is expensive both when A is dense and when A is sparse. In practice, however, we will not need to form U explicitly, and we will not need to compute the exact value of the ℓ_1-norm of each row of U. Indeed, it suffices to get estimates of $\|U_{(i)}\|_1$, in which case we can adjust the sampling complexity s to maintain a small approximation factor. Algorithm 2 provides a way to compute the estimates of the ℓ_1 norm of each row of U fast and construct the sampling matrix. The same technique was used in (Clarkson et al., 2013). Our main result for Algorithm 2 is presented in Lemma 5, the proof of which can be found in (Yang et al., 2013)

Algorithm 2 Fast Construction of $(1\pm\epsilon)$-distortion Sampling Matrix of $(\mathcal{A}, \rho_\tau(\cdot))$

1: **Input:** $A \in \mathbb{R}^{n\times d}, R \in \mathbb{R}^{d\times d}$ such that AR^{-1} is (α,β)-conditioned, $\epsilon \in (0,1/2)$, $\tau \in [1/2,1)$.
2: **Output:** Sampling matrix $S \in \mathbb{R}^{n\times n}$.
3: Let $\Pi_2 \in \mathbb{R}^{d\times r_2}$ be a matrix of independent Cauchys with $r_2 = 15\log(40n)$.
4: Compute $R^{-1}\Pi_2$ and construct $\Lambda = AR^{-1}\Pi_2 \in \mathbb{R}^{n\times r_2}$.
5: For $i \in [n]$, compute $\lambda_i = \mathrm{median}_{j\in[r_2]}|\Lambda_{ij}|$.
6: For $s = \frac{\tau}{1-\tau}\frac{81\alpha\beta}{\epsilon^2}\left(d\log\left(\frac{\tau}{1-\tau}\frac{18}{\epsilon}\right)+\log 80\right)$ and $i \in [n]$, compute probabilities

$$\hat{p}_i = \min\left\{1, s\cdot\frac{\lambda_i}{\sum_{j\in[n]}\lambda_j}\right\}.$$

7: Let $S \in \mathbb{R}^{n\times n}$ be diagonal with independent entries

$$S_{ii} = \begin{cases} \frac{1}{\hat{p}_i}, & \text{with probability } \hat{p}_i; \\ 0, & \text{with probability } 1-\hat{p}_i. \end{cases}$$

Lemma 5 (Fast Construction of $(1\pm\epsilon)$-distortion Sampling Matrix)**.** *Given a matrix $A \in \mathbb{R}^{n\times d}$, and a matrix $R \in \mathbb{R}^{d\times d}$ such that AR^{-1} is a (α,β)-conditioned basis for $\mathcal{A}$, Algorithm 2 takes $\mathcal{O}(\mathrm{nnz}(A)\cdot\log n)$ time to compute a sampling matrix $S \in \mathbb{R}^{t\times n}$ (with only one nonzero per row), such that with probability at least 0.9, S is a $(1\pm\epsilon)$-distortion sampling matrix. That is for all $x \in \mathbb{R}^d$,*

$$(1-\epsilon)\rho_\tau(Ax) \le \rho_\tau(SAx) \le (1+\epsilon)\rho_\tau(Ax). \qquad (6)$$

Further, with probability at least $1-o(1)$, $t = \mathcal{O}\left(\mu\alpha\beta d\log(\mu/\epsilon)/\epsilon^2\right)$, where $\mu = \frac{\tau}{1-\tau}$.

3.2. Main algorithm

Here, we state our main algorithm for computing an approximate solution to the quantile regression problem. Recall that, to compute a relative-error approxi-

Algorithm 3 Fast Randomized Algorithm for Quantile Regression

1: **Input:** $A \in \mathbb{R}^{n\times d}$ with full column rank, $\epsilon \in (0, 1/2)$, $\tau \in [1/2, 1)$.
2: **Output:** An approximate solution $\hat{x} \in \mathbb{R}^d$.
3: Compute $R \in \mathbb{R}^{d\times d}$ such that AR^{-1} is a well-conditioned basis for $\mathcal{A}$ via Algorithm 1.
4: Compute a $(1\pm\epsilon)$-distortion embedding $S \in \mathbb{R}^{s\times n}$ of $(\mathcal{A}, \rho_\tau(\cdot))$ via Algorithm 2.
5: Return $\hat{x} \in \mathbb{R}^d$ that minimizes $\rho_\tau(SAx)$ with respect to $x \in \mathcal{C}$.

mate solution, it suffices to compute a $(1\pm\epsilon)$-distortion embedding matrix S. To construct S, we first compute a well-conditioned basis U by Algorithm 1, and we then apply Algorithm 2 to approximate the ℓ_1 norm of each row of U. These procedures are summarized in Algorithm 3. The main quality-of-approximation result for this algorithm is stated in Theorem 1, the proof of which can be found in (Yang et al., 2013)

Theorem 1 (Fast Quantile Regression)**.** *Given $A \in \mathbb{R}^{n\times d}$ and $\varepsilon \in (0, 1/2)$, Algorithm 3 returns a vector $\hat{x}$ that, with probability at least 0.8, satisfies*

$$\rho_\tau(A\hat{x}) \le \left(\frac{1+\varepsilon}{1-\varepsilon}\right)\rho_\tau(Ax^*),$$

where x^* is an optimal solution to the original problem. In addition, the algorithm to construct $\hat{x}$ runs in time

$$\mathcal{O}(\mathrm{nnz}(A)\cdot\log n) + \phi\left(\mathcal{O}(\mu d^3\log(\mu/\epsilon)/\epsilon^2), d\right),$$

where $\mu = \frac{\tau}{1-\tau}$ and $\phi(s, d)$ is the time to solve a quantile regression problem of size $s \times d$.

4. Empirical Evaluation

Here, we present a brief summary of our empirical evaluation of Algorithm 3. We considered both simulated data and real data, and we considered both medium-sized data as well as terabyte-scale data. Many more details may be found in (Yang et al., 2013).

4.1. Medium-scale Quantile Regression

We start with a test on simulated data with size $1e6 \times 12$ which is generated in the following way.

1. Each row of the design matrix A is a canonical vector. The number of measurements on the j-th column is twice as that on the $(j-1)$-th column, for $j = 2, \ldots, 12$. A is roughly a $1e6 \times 12$ matrix.
2. The true coefficient vector x^* is a vector of size 12 with independent Gaussian entries. Let $b^* = Ax^*$.
3. The noise vector ϵ is generated with independent Laplacian entries. The response vector is given by

$$b_i = \begin{cases} 500\epsilon_i & \text{with probability } 0.001; \\ b_i^* + \epsilon_i & \text{otherwise.} \end{cases}$$

Recall, as we point out in a remark after Lemma 4, that we can use other methods for the conditioning step, *i.e.*, for finding the well-conditioned basis $U = AR^{-1}$ in the first step of Algorithm 3. Here, we will consider the empirical performance of five methods for doing so, namely, ISPC, SPC, SC, NOCO, and UNIF: ISPC is the implementation of Algorithm 3 where ISPC stands for the Improved Sparse Cauchy Transform (called SPC2 in (Yang et al., 2013)); SPC, SC and NOCO are the variations of Algorithm 3 by using, respectively, the Sparse Cauchy Transform (Meng & Mahoney, 2013), the Slow Cauchy Transform (Sohler & Woodruff, 2011), and no conditioning (for all these, we compute the row norms of the well-conditioned basis exactly instead of estimating them, as this may reduce the error due to the estimating step); and, finally, UNIF is the uniform sampling method, which we add here for completeness.

Rather than determining the sample size from a given tolerance ϵ, we let the sample size s vary from 100 to $1e5$ as an input of the algorithm. Consider Figure 1, where we show the results when $\tau = 0.95$, where we draw the first and the third quartiles of the relative errors of the objective value and solution measured in infinity norm from 50 independent trials. We limit the Y axis to 100 to show more details. The relative performance of each method doesn't change substantially when τ takes other values.

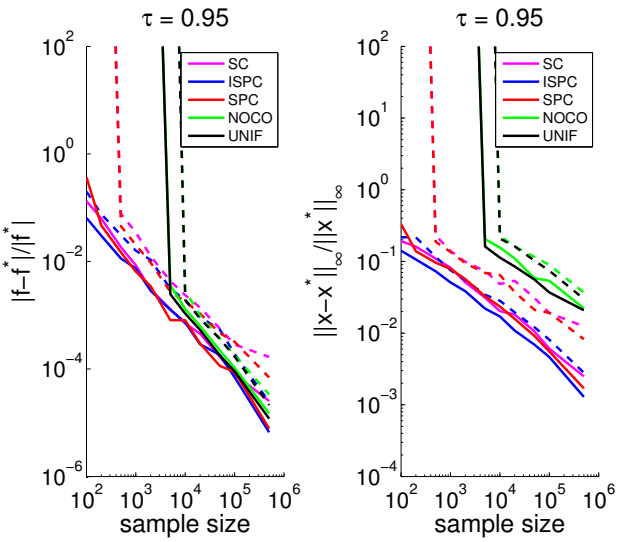

Figure 1. Setting $\tau = 0.95$, the plot on left shows the first (solid) and the third (dashed) quartiles of the relative error of the objective value using five different methods, while the one on right shows the relative ℓ_∞ error of the solution.

From the plots we may see, for the objective value, all the three conditioning-based methods perform similarly. They give 2-digit accuracy when the sampling complexity is about 1000. Among these, ISPC performs slightly better. As expected, the two naive methods UNIF and NOCO yield large relative error when the sampling complexity s is below $1e4$. They are not reliable if we want to solve the problem quickly while maintaining reasonable accuracy.

Although our theory is about estimating the objective value, our conditioning-based methods also yield high accuracy on the solution, *i.e.*, the vector achieving the optimum. See Table 1. As we can see, ISPC performs the best. NOCO is likely to sample the outliers and UNIF performs badly due the the imbalance measurements in the design matrix. In order to see better the difference in the behavior of these methods, we fix the sample size to be 5000 and record the relative errors on solutions measured in three different norms.

Table 1. The first and the third quartiles of relative errors in ℓ_1, ℓ_2, and ℓ_∞ norms. ISPC clearly performs the best. SC and SPC have slightly worse accuracy. NOCO and UNIF generate large errors.

	$\|x - x^*\|_2/\|x^*\|_2$	$\|x - x^*\|_1/\|x^*\|_1$	$\|x - x^*\|_\infty/\|x^*\|_\infty$
ISPC	[0.024, 0.036]	[0.026, 0.307]	[0.019, 0.033]
SPC	[0.031, 0.058]	[0.032, 0.053]	[0.026, 0.051]
SC	[0.034, 0.071]	[0.035, 0.060]	[0.027, 0.081]
NOCO	$[0.160, 5.3 \times 10^7]$	$[0.152, 3.5 \times 10^7]$	$[0.133, 5.4 \times 10^7]$
UNIF	$[0.155, 6.0 \times 10^6]$	$[0.146, 3.8 \times 10^7]$	$[0.131, 6.3 \times 10^7]$

Next, we consider a real data set consisting of a 5% sample of the U.S. 2000 Census data[1], consisting of annual salary and related features on people who reported that they worked 40 or more weeks in the previous year and worked 35 or more hours per week. The size of the design matrix is 5×10^6 by 11. The performance of the methods on objective value is similar to that on the simulated data. Here, we will explore more on the running times. In particular, we compare the running time of our method with some competing methods when the data size increases. They are the primal-dual method, referred to as `ipm`, and that with preprocessing, referred to as `prqfn`; see (Portnoy & Koenker, 1997) for more details. Fix $d = 11$, we let n, the large dimension, changes from $5e5$ to $5e6$. For each n, we extract the leading $n \times d$ submatrix of the census data, and record the running time of each method. The result is shown in Figure 2. From the plot we can see, ISPC runs faster than `prqfn` in most cases and appears to have a linear rate.

[1] http://www.census.gov/census2000/PUMS5.html

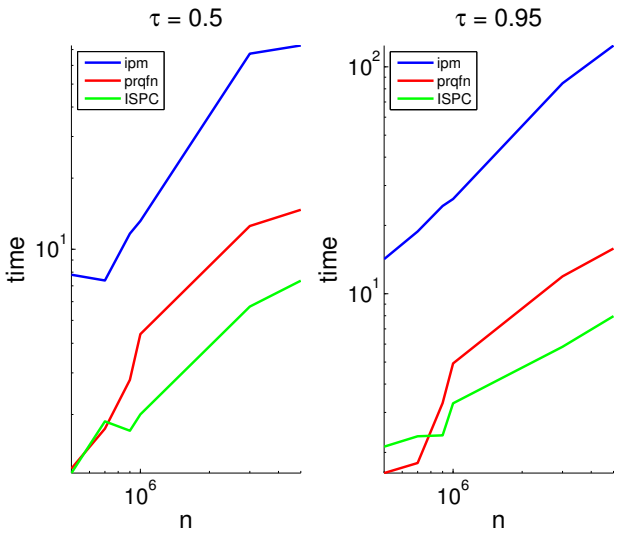

Figure 2. The running time in seconds for each method as the data size changes from $5e5$ to $5e6$. `ipm` is the standard interior-point method and `prqfn` is the `ipm` with preprocessing. ISPC is the implementation of Algorithm 3.

4.2. Large-scale Quantile Regression

We continue our empirical evaluation with a terabyte-scale problem that we generate by stacking 2000 copies of the census data we used in the previous section. This leads to a problem of size roughly $10^{10} \times 12$ whose optimal solutions at different quantiles are known. At this terabyte scale, `ipm` has two major issues: memory requirement and running time. Though shared memory machines with more than a terabyte of RAM exist, they are rare in practice. When $n = 10^{10}$, even we have enough RAM to use, it might take a very long time to solve the problem using `ipm`.

The MapReduce framework is now the *de facto* standard parallel environment for large data analysis. Apache Hadoop[2], an open source implementation of MapReduce, is extensively used by many companies and institutions, *e.g.*, Facebook, LinkedIn, and Yahoo!. Since our sampling algorithm only needs a few passes through the data and is embarrassingly parallel, it is straightforward to implement it on Hadoop.

In Table 2, we list part of the solution computed by our randomized algorithm with a sample size $1e5$ at different quantiles, as well as the corresponding optimal solution. As we may see, for most coefficients, our algorithm provides 2-digit accuracy. The quantile regression result reveals some interesting facts. For example, marriage may entail a higher salary in lower

[2] Apache Hadoop, http://hadoop.apache.org/

Table 2. Quantile regression results for the U.S. Census 2000 data. The response is the total annual income. Except for the intercept and the terms involved with education, all the covariates are $\{0, 1\}$ binary indicators.

Covariate	$\tau = 0.1$	$\tau = 0.25$	$\tau = 0.5$	$\tau = 0.75$	$\tau = 0.9$
intercept	8.9812	9.3022	9.6395	10.0515	10.5510
	[8.9673, 8.9953]	[9.2876, 9.3106]	[9.6337, 9.6484]	[10.0400, 10.0644]	[10.5296, 10.5825]
female	-0.2609	-0.2879	-0.3227	-0.3472	-0.3774
	[-0.2657, -0.2549]	[-0.2924, -0.2846]	[-0.3262, -0.3185]	[-0.3481, -0.3403]	[-0.3792, -0.3708]
Age $\in$ [30, 40)	0.2693	0.2649	0.2748	0.2936	0.3077
	[0.2610, 0.2743]	[0.2613, 0.2723]	[0.2689, 0.2789]	[0.2903, 0.2981]	[0.3027, 0.3141]
Age $\in$ [40, 50)	0.3173	0.3431	0.3769	0.4118	0.4416
	[0.3083, 0.3218]	[0.3407, 0.3561]	[0.3720, 0.3821]	[0.4066, 0.4162]	[0.4386, 0.4496]
Age $\in$ [50, 60)	0.3316	0.3743	0.4188	0.4612	0.5145
	[0.3190, 0.3400]	[0.3686, 0.3839]	[0.4118, 0.4266]	[0.4540, 0.4636]	[0.5071, 0.5230]
Age $\in$ [60, 70)	0.3237	0.3798	0.4418	0.5072	0.6027
	[0.3038, 0.3387]	[0.3755, 0.3946]	[0.4329, 0.4497]	[0.4956, 0.5162]	[0.5840, 0.6176]
Age $\geq$ 70	0.3206	0.4132	0.5152	0.6577	0.8699
	[0.2962, 0.3455]	[0.4012, 0.4359]	[0.5036, 0.5308]	[0.6371, 0.6799]	[0.8385, 0.8996]
non_white	-0.0953	-0.1018	-0.0922	-0.0871	-0.0975
	[-0.1023, -0.0944]	[-0.1061, -0.0975]	[-0.0985, -0.0902]	[-0.0932, -0.0860]	[-0.1041, -0.0932]
married	0.1175	0.1117	0.0951	0.0870	0.0953
	[0.1121, 0.1238]	[0.1059, 0.1162]	[0.0918, 0.0989]	[0.0835, 0.0914]	[0.0909, 0.0987]
education	-0.0152	-0.0175	-0.0198	-0.0470	-0.1062
	[-0.0179, -0.0117]	[-0.0200, -0.0149]	[-0.0225, -0.0189]	[-0.0500, -0.0448]	[-0.1112, -0.1032]
education^2	0.0057	0.0062	0.0065	0.0081	0.0119
	[0.0055, 0.0058]	[0.0061, 0.0064]	[0.0064, 0.0066]	[0.0080, 0.0083]	[0.0117, 0.0122]

quantiles. Education^2, whose value ranged from 0 to 256, has a strong impact on the total income, especially in the higher quantiles. Also, the difference in age doesn't affect the total income much in lower quantiles, but becomes a significant factor in higher quantiles.

To summarize, our algorithm can handle terabyte-sized quantile regression problems easily, *e.g.*, obtaining 2 digits of accuracy by sampling about $1e5$ rows on a problem of size $1e10 \times 11$; and the running time is competitive with the best existing random sampling algorithms, and it can be applied in parallel and distributed environments.

References

Bourgain, J., Lindenstrauss, J., and Milman, V. Approximation of zonoids by zonotopes. *Acta Math*, 162:73–141, 1989.

Buchinsky, M. Changes is US wage structure 1963-87: an application of quantile regression. *Econometrica*, 62:405–408, 1994.

Buhai, I. S. Quantile regression: Overview and selected applications. *Ad Astra*, 4, 2005.

Clarkson, K. L., Drineas, P., Magdon-Ismail, M., Mahoney, M. W., Meng, X., and Woodruff, D. P. The Fast Cauchy Transform and faster robust linear regression. In *Proc. of the 24th Annual SODA*, 2013.

Dasgupta, A., Drineas, P., Harb, B., Kumar, R., and Mahoney, M. W. Sampling algorithms and corsets for ℓ_p regression. *SIAM J. Comput.*, 38(5):2060–2078, 2009.

Koenker, R. and Bassett, G. Regression quantiles. *Econometrica*, 46(1):33–50, 1978.

Koenker, R. and D'Orey, V. Computing regression quantiles. *J, Roy. Statist. Soc. Sr. C(Appl, Statis.)*, 43:410 – 414, 1993.

Koenker, R. and Hallock, K. Quantile regression. *J. of Economic Perspectives*, 15(4):143–156, 2001.

Mahoney, M. W. *Randomized algorithms for matrices and data.* Foundations and Trends in Machine Learning. NOW Publishers, Boston, 2011. Also available at: arXiv:1104.5557.

Meng, X. and Mahoney, M. W. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Proc. of the 45th Annual STOC*, 2013.

Portnoy, S. On computation of regression quantiles: Making the Laplacian tortoise faster. *Lecture Notes-Monograph Series, Vol. 31, L_1-Statistical Procedures and Related Topics*, pp. 187–200, 1997.

Portnoy, S. and Koenker, R. The Gaussian hare and the Laplacian tortoise: Computability of squared-error versus absolute-error estimators, with discussion. *Statistical Science*, 12(4):279–300, 1997.

Sohler, C. and Woodruff, D. P. Subspace embedding for the ℓ_1-norm with applications. In *Proc. of the 43rd Annual ACM STOC*, pp. 755–764, 2011.

Yang, J., Meng, X., and Mahoney, M. W. Quantile regression for large-scale applications. Technical report, 2013. Preprint: arXiv:1305.0087.