Int. J. Mach. Learn. & Cyber. (2011) 2:191–207
DOI 10.1007/s13042-011-0031-2

ORIGINAL ARTICLE

Gradient descent algorithms for quantile regression with smooth approximation

Songfeng Zheng

Received: 22 April 2011 / Accepted: 23 June 2011 / Published online: 22 July 2011

Abstract Gradient based optimization methods often converge quickly to a local optimum. However, the check loss function used by quantile regression model is not everywhere differentiable, which prevents the gradient based optimization methods from being applicable. As such, this paper introduces a smooth function to approximate the check loss function so that the gradient based optimization methods could be employed for fitting quantile regression model. The properties of the smooth approximation are discussed. Two algorithms are proposed for minimizing the smoothed objective function. The first method directly applies gradient descent, resulting the gradient descent smooth quantile regression model; the second approach minimizes the smoothed objective function in the framework of functional gradient descent by changing the fitted model along the negative gradient direction in each iteration, which yields boosted smooth quantile regression algorithm. Extensive experiments on simulated data and real-world data show that, compared to alternative quantile regression models, the proposed smooth quantile regression algorithms can achieve higher prediction accuracy and are more efficient in removing noninformative predictors.

Keywords Quantile regression · Gradient descent · Boosting · Variable selection

1 Introduction

The ordinary least square regression aims to estimate the conditional expectation of the response Y given the predictor (vector) $\mathbf{x}$, i.e., $\mathrm{E}(Y|\mathbf{x})$. However, the mean value (or the conditional expectation) is sensitive to the outliers of the data [14]. Therefore, if the data is not homogeneously distributed, we expect the least square regression giving us a poor prediction.

The τth quantile of a distribution is defined as the value such that there is $100\tau\%$ of mass on its left side. Compared to the mean value, quantiles are more robust to outliers [14]. Another advantage of quantile is that we can get a series of quantile values which can describe the whole data distribution better than a single value (e.g., mean) does. Let $Q_\tau(Y)$ be the τth quantile of a random variable Y, it can be proved [12] that

$$Q_\tau(Y) = \arg\min_c \mathrm{E}_Y[\rho_\tau(Y-c)],$$

where $\rho_\tau(r)$ is the "check function" [14] defined by

$$\rho_\tau(r) = rI(r \geq 0) - (1-\tau)r. \quad (1)$$

The function $I(\cdot)$ in Eq. 1 is the indicator function with $I(\cdot) = 1$ if the condition is true, otherwise $I(\cdot) = 0$.

Given data $\{(\mathbf{x}_i, Y_i), \quad i = 1, \ldots, n\}$, with predictor vector $\mathbf{x}_i \in \mathbf{R}^p$ and response $Y_i \in \mathbf{R}$, let $q(\mathbf{x})$ be the τth conditional quantile of Y given $\mathbf{x}$. Similar to the least square regression, quantile regression (QReg) [14] aims at estimating the conditional τth quantile of the response given predictor vector $\mathbf{x}$ and can be formulated as

$$q^*(\cdot) = \arg\min_q \frac{1}{n}\sum_{i=1}^{n} \rho_\tau(Y_i - q(\mathbf{x}_i)). \quad (2)$$

Compared to least square regression, quantile regression is robust to outliers in observations, and can give a more complete view of the relationship between predictor and response. Furthermore, least square regression implicitly assumes normally distributed errors, while such an assumption is not necessary in quantile regression. Since

S. Zheng (✉)
Department of Mathematics, Missouri State University, Springfield, MO 65897, USA
e-mail: SongfengZheng@MissouriState.edu

being introduced in [15], quantile regression has become a popular and effective approach to statistical analysis with wide applications, for example, in economics [11, 17], survival analysis [16], and ecology [4].

The quantile regression model in Eq. 2 can be solved by linear programming algorithms [14, 18] or majorize–minimize algorithms [12]. Li et al. [22] proposed a model for piecewise linear quantile regression function. The idea of support vector regression was introduced for quantile regression model, yielding support vector quantile regression (SV-QReg) [13, 24, 30], which was applied in [28, 29] for microarray analysis. SV-QReg can estimate quantile regression model in non-linear and high dimensional spaces, but it is computationally expensive because it needs to solve a quadratic programming problem. Regression forest was used to estimate the conditional quantiles in [27], but the forest model is not easy to interpret. Langford et al. [20] proposed to use classification technique in estimating the conditional quantile. For a given quantile value, their method trains a set of classifiers $\{c_t\}$ for a series of $t \in [0, 1]$, and the testing stage calculates the average of the outputs of the classifiers. Therefore, this method is time consuming. Recently, L_1 quantile regression (L_1-QReg) [1, 25, 34] was investigated which imposes L_1 constraint on the coefficient vector in the linear quantile regression model, and the theoretical properties of L_1-QReg was investigated in detail in [1].

The gradient based optimization methods usually converge quickly to a local optimum. However, we notice that the objective function used by quantile regression is not differentiable at the origin which makes the gradient based methods not directly applicable. As such, this paper proposes to approximate the check loss function by a smooth function, based on which a gradient descent algorithm for linear quantile regression is derived. We call the resulting algorithm as gradient descent smooth quantile regression (GDS-QReg) model.

Motivated by the idea of gradient boosting algorithms [8, 26], we further propose to estimate the quantile regression function by minimizing the smoothed objective function in the framework of functional gradient descent. In each iteration, we approximate the negative gradient of the objective function by a base procedure, and grow the model in that direction. This results boosted smooth quantile regression (BS-QReg) algorithm. The obtained BS-QReg algorithm converges to a local optimum, more importantly, it enables us to solve problems in high dimensional spaces.

The proposed GDS-QReg and BS-QReg algorithms were tested extensively on various simulated and real-world datasets. The results show that the proposed algorithms can achieve higher prediction accuracy compared to alternatives. Furthermore, the BS-QReg model can automatically select informative predictors, inheriting the variable selection ability of boosting algorithm.

In our previous work [37], the gradient boosting algorithm was directly applied to the check loss function without smooth approximation, which was also independently studied in a technical report [19]. Since the objective function in [37] is not everywhere differentiable, the gradient descent algorithm is not directly applicable. In [37], we also applied gradient ascent for binary classification by predicting quantiles. Since the objective function for classification in [37] is smooth, the methods discussed in this paper will not be applied to the classification problem.

The rest of this paper is organized as follows: Sect. 2 introduces a smooth function to approximate the check loss function used by the QReg model, and the properties of the smooth model are also discussed; based on the smoothed objective function, Sect. 3 proposes the gradient descent algorithm for linear quantile regression model; Sect. 4 reviews the interpretation of boosting as functional gradient descent, and applies the functional gradient descent to minimize the smoothed objective function, yielding the boosted smooth QReg algorithm; Sect. 5 compares the proposed GDS-QReg and BS-QReg to various alternatives via simulated datasets, and the relationship between L_1 constrained QReg and BS-QReg is also investigated; Sect. 6 compares the performance of the proposed algorithms to various alternatives on two real-world datasets; finally, Sect. 7 summarizes this paper and discusses some future research directions.

2 Smooth quantile regression model

This section introduces a smooth approximation to the check loss function and studies the properties of the resulting smooth model.

2.1 Smooth function and its properties

The function $\rho_\tau(x)$ employed by the original quantile regression model is not differentiable at the origin, $x = 0$. The non-differentiability of $\rho_\tau(x)$ makes it difficult to apply gradient based optimization methods in fitting the quantile regression model, although gradient based methods are usually time efficient, easy to implement, and yield a local optimum.

Chen and Mangasarian [5] introduced a class of smooth functions for nonlinear optimization problems and applied this idea to support vector machines [21]. Emulating [21], in this paper, we propose the following smooth function to approximate the check function

$$S_{\tau,\alpha}(x) = \tau x + \alpha \log\left(1 + e^{-\frac{x}{\alpha}}\right), \tag{3}$$

where $\alpha > 0$ is called as the smooth parameter.

The properties of function $S_{\tau,\alpha}(x)$ can be summarized as following:

Lemma 1 *For any given* $\alpha > 0, S_{\tau,\alpha}(x)$ *is a convex function.*

Proof From the definition of $S_{\tau,\alpha}(x)$ in Eq. 3, we calculate that

$$\frac{d^2 S_{\tau,\alpha}(x)}{dx^2} = \frac{1}{\alpha}\frac{e^{-\frac{x}{\alpha}}}{(1+e^{-\frac{x}{\alpha}})^2} > 0 \tag{4}$$

for any $x \in \mathbf{R}$ since $\alpha > 0$, which shows that function $S_{\tau,\alpha}(x)$ is convex everywhere. □

Lemma 2 *Let* $S_{\tau,\alpha}(x)$ *be the function defined in Eq.* 3 *with* $\alpha > 0$, *we have*

$$0 < S_{\tau,\alpha}(x) - \rho_\tau(x) \le \alpha \log 2 \tag{5}$$

for any $x \in \mathbf{R}$. *Thus,*

$$\lim_{\alpha \to 0^+} S_{\tau,\alpha}(x) = \rho_\tau(x). \tag{6}$$

Proof The definition of $\rho_\tau(x)$ in Eq. 1 could be equivalently written as

$$\rho_\tau(x) = \begin{cases} \tau x, & \text{if} \quad x \ge 0 \\ (\tau - 1)x, & \text{if} \quad x < 0. \end{cases}$$

Thus, when $x \ge 0$,

$$S_{\tau,\alpha}(x) - \rho_\tau(x) = \alpha \log\left(1 + e^{-\frac{x}{\alpha}}\right), \tag{7}$$

hence,

$$0 < S_{\tau,\alpha}(x) - \rho_\tau(x) \le \alpha \log 2$$

for $x \ge 0$. When $x < 0$,

$$\begin{aligned} S_{\tau,\alpha}(x) - \rho_\tau(x) &= \tau x + \alpha \log\left(1 + e^{-\frac{x}{\alpha}}\right) - (\tau - 1)x \\ &= \alpha \log\left(1 + e^{\frac{x}{\alpha}}\right). \end{aligned} \tag{8}$$

While

$$0 < \alpha \log\left(1 + e^{\frac{x}{\alpha}}\right) < \alpha \log\left(1 + e^{\frac{0}{\alpha}}\right) = \alpha \log 2,$$

since $x < 0$. This shows that $S_{\tau,\alpha}(x) - \rho_\tau(x)$ also falls in the range $(0, \alpha \log 2)$ for $x < 0$. Thus, Eq. 5 is proved. Eq. 6 follows directly by letting $\alpha \to 0^+$ in Eq. 5. □

It follows from Lemma 2 that the function $S_{\tau,\alpha}(x)$ is always positive because $S_{\tau,\alpha}(x) > \rho_\tau(x) \ge 0$ for all $x \in \mathbf{R}$. From the proof of Lemma 2, it follows that the difference $S_{\tau,\alpha}(x) - \rho_\tau(x)$ is an even function of x and does not depend on τ (see Eqs. 7, 8).

Figure 1 shows the check function with $\tau = 0.5$ and the corresponding smoothed version with the smooth parameter $\alpha = 0.4$. Figure 1a clearly shows that the smoothed check function is always positive, smooth, convex, and dominates the original check function. Figure 1b shows the difference between the check function and its smoothed version, and it is readily observed that the two functions approach each other quickly.

2.2 Smoothed objective function for quantile regression

In this paper, for a given predictor vector $\mathbf{x}$, we assume the τth quantile of the response, Y, could be estimated by function $q(\mathbf{x}, \mathbf{w})$, where $\mathbf{w}$ is the parameter vector and the function $q(\mathbf{x}, \mathbf{w})$ is linear in $\mathbf{w}$, i.e., for constants c_1 and c_2, and two parameter vectors $\mathbf{w}_1$ and $\mathbf{w}_2$, we have

$$q(\mathbf{x}, c_1\mathbf{w}_1 + c_2\mathbf{w}_2) = c_1 q(\mathbf{x}, \mathbf{w}_1) + c_2 q(\mathbf{x}, \mathbf{w}_2). \tag{9}$$

Under this setting, the parameter vector in the quantile regression function is estimated by

$$\hat{\mathbf{w}}_\tau = \arg\min_{\mathbf{w}} \Phi(\mathbf{w}), \tag{10}$$

with

$$\Phi(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \rho_\tau(Y_i - q(\mathbf{x}_i, \mathbf{w})). \tag{11}$$

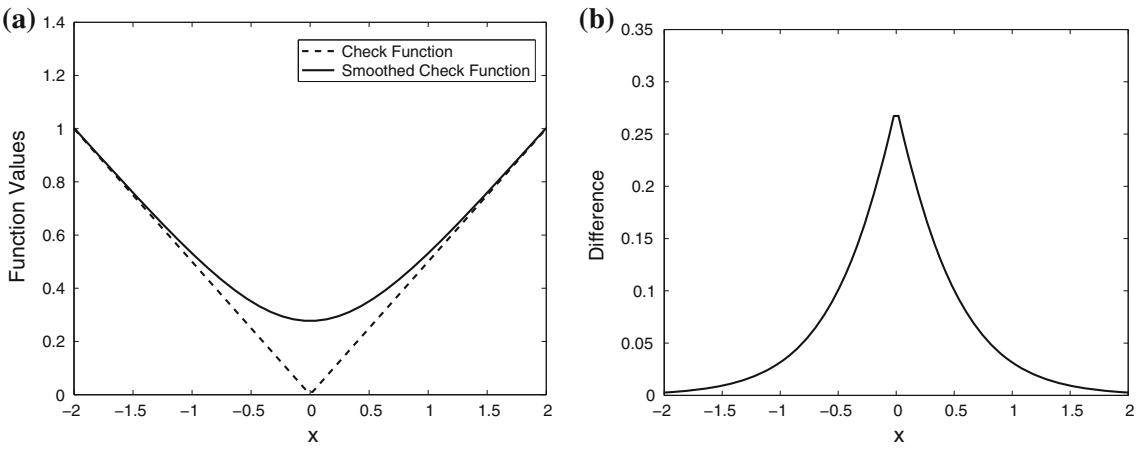

Fig. 1 **a** The check function ($\tau = 0.5$) and the smoothed check function with parameter $\alpha = 0.4$, **b** the difference between the check function and the smoothed check function in **a**

Define

$$\hat{\mathbf{w}}_{\tau,\alpha} = \arg\min_{\mathbf{w}} \Phi_\alpha(\mathbf{w}), \tag{12}$$

where

$$\Phi_\alpha(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} S_{\tau,\alpha}(Y_i - q(\mathbf{x}_i, \mathbf{w})). \tag{13}$$

We call the function $\Phi_\alpha(\mathbf{w})$ the smoothed objective function, and the model obtained by minimizing $\Phi_\alpha(\mathbf{w})$ is called as the smooth quantile regression model with $\hat{\mathbf{w}}_{\tau,\alpha}$ being the estimated parameter vector.

In order to study the property of the smooth quantile regression model, we need the following result:

Lemma 3 *If $f_1(x), f_2(x), \ldots, f_n(x)$ are convex functions on domain Ω, and $c_1, c_2, \ldots, c_n$ are nonnegative constants, then $g(x) = \sum_{i=1}^{n} c_i f_i(x)$* is also a convex function on Ω.

Proof Direct calculation shows that $g''(x) \geq 0$, which implies the convexity of $g(x)$ over Ω. □

Finally, we have the following theorem regarding the relationship between the ordinary quantile regression model and its smoothed counterpart:

Theorem 1 *The functions $\Phi(\mathbf{w})$ and $\Phi_\alpha(\mathbf{w})$ are convex functions of $\mathbf{w}$, thus $\hat{\mathbf{w}}_\tau$ and $\hat{\mathbf{w}}_{\tau,\alpha}$ defined in Eqs. 10 and 12 exist. Furthermore, as $\alpha \to 0^+, \hat{\mathbf{w}}_{\tau,\alpha} \to \hat{\mathbf{w}}_\tau$.*

Proof Let $\Phi_{i,\alpha}(\mathbf{w}) = S_{\tau,\alpha}(Y_i - q(\mathbf{x}_i, \mathbf{w}))$, then for $0 \leq t \leq 1$, and any $\mathbf{w}_1$ and $\mathbf{w}_2$, we have

$$\begin{aligned}\Phi_{i,\alpha}(t\mathbf{w}_1 + (1-t)\mathbf{w}_2) &= S_{\tau,\alpha}(Y_i - q(\mathbf{x}_i, t\mathbf{w}_1 + (1-t)\mathbf{w}_2)) \\ &= S_{\tau,\alpha}(Y_i - tq(\mathbf{x}_i, \mathbf{w}_1) - (1-t)q(\mathbf{x}_i, \mathbf{w}_2)) \qquad (14)\\ &= S_{\tau,\alpha}(t(Y_i - q(\mathbf{x}_i, \mathbf{w}_1)) + (1-t)(Y_i - q(\mathbf{x}_i, \mathbf{w}_2))) \\ &\leq tS_{\tau,\alpha}(Y_i - q(\mathbf{x}_i, \mathbf{w}_1)) + (1-t)S_{\tau,\alpha}(Y_i - q(\mathbf{x}_i, \mathbf{w}_2)) \qquad (15)\\ &= t\Phi_{i,\alpha}(\mathbf{w}_1) + (1-t)\Phi_{i,\alpha}(\mathbf{w}_2), \qquad (16)\end{aligned}$$

where Eq. 14 follows from the linearity of function $q(\mathbf{x}, \mathbf{w})$ defined in Eq. 9, and the " $\leq$ " in Eq. 15 follows from the convexity of $S_{\tau,\alpha}$ proved in Lemma 1. The inequality in Eq. 16 shows that $\Phi_{i,\alpha}(\mathbf{w})$ is a convex function of $\mathbf{w}$. Consequently, by Lemma 3, $\Phi_\alpha(\mathbf{w}) = \sum_{i=1}^{n} \Phi_{i,\alpha}(\mathbf{w})/n$ is convex in $\mathbf{w}$. The convexity of $\Phi(\mathbf{w})$ follows similarly. Thus, the unique solutions to the minimization problems in Eqs. 10 and 12 exist. By Lemma 2, as $\alpha \to 0^+, S_{\tau,\alpha}(Y_i - q(\mathbf{x}_i, \mathbf{w})) \to \rho_\tau(Y_i - q(\mathbf{x}_i, \mathbf{w}))$, thus $\Phi_\alpha(\mathbf{w}) \to \Phi(\mathbf{w})$. Consequently, $\hat{\mathbf{w}}_{\tau,\alpha} \to \hat{\mathbf{w}}_\tau$, as $\alpha \to 0^+$. □

Theorem 1 shows that we can use the solution of smooth quantile regression model with small α to approximate the original quantile regression function. In the following, we provide two algorithms for solving the smooth quantile regression model.

3 Gradient descent smooth quantile regression

In this section, for a given predictor vector $\mathbf{x}$, we assume the estimated τth quantile function of the response, Y, is linear in the predictors, i.e., $q(\mathbf{x}, \mathbf{w}) = \mathbf{x}^T\mathbf{w}$. Note that the vector $\mathbf{x}$ could include the transformed predictors, for example, x_1^2 or $\tan x_2$. The first component of the coefficient vector $\mathbf{w}$ is w_0, which is the constant bias term of the linear quantile regression function. Correspondingly, the first element of $\mathbf{x}$ is 1.

The gradient vector of $\Phi_\alpha(\mathbf{w})$ could be calculated as

$$\nabla\Phi_\alpha(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\left[\tau - \frac{1}{1+\exp\left(\frac{Y_i - \mathbf{x}_i^T\mathbf{w}}{\alpha}\right)}\right]\mathbf{x}_i, \tag{17}$$

and this enables us to minimize $\Phi_\alpha(\mathbf{w})$ by gradient descent, resulting the gradient descent smooth quantile regression (GDS-QReg) algorithm, which is shown as Algorithm 1.

In the third step of Algorithm 1, the step size η_m could be chosen as

$$\begin{aligned}\eta_m &= \arg\min_{\eta} \Phi_\alpha\left(\mathbf{w}^{[m-1]} - \eta\nabla\Phi_\alpha(\mathbf{w}^{[m-1]})\right) \\ &= \arg\min_{\eta} f(\eta), \end{aligned} \tag{18}$$

where $f(\eta) = \Phi_\alpha\left(\mathbf{w}^{[m-1]} - \eta\nabla\Phi_\alpha(\mathbf{w}^{[m-1]})\right)$. The minimization problem in Eq. 18 can be solved by 1-D search algorithms, e.g., Fibonacci search, Golden section search [32]. We choose to use Golden section search for its simplicity, which is given in Algorithm 2 for completeness. Alternatively, we can fix the step size η_m as a small value. This paper will experimentally investigate both the strategies for setting the step size.

Algorithm 1 Gradient Descent Smooth Quantile Regression (GDS-QReg)

0: Initialize $\mathbf{w}^{[0]}$, and set $m = 0$.
1: Increase m by 1. Compute the gradient vector of $\Phi_\alpha(\mathbf{w})$ at $\mathbf{w}^{[m-1]}$ by Eqn. (17).
2: If the vector $\nabla\Phi_\alpha(\mathbf{w}^{[m-1]})$ is close to $\mathbf{0}$ vector, stop.
3: Choose a step size $\eta_m \in \mathbf{R}$ and update the estimated vector $\mathbf{w}$ as

$$\mathbf{w}^{[m]} = \mathbf{w}^{[m-1]} - \eta_m\nabla\Phi_\alpha(\mathbf{w}^{[m-1]}),$$

and go to step 1.

Algorithm 2 Golden section search to minimize function $f(\eta)$ on interval $[a, b]$

0: Given the initial search interval $[a_1, b_1] = [a, b]$ and the tolerance L. let $g = (\sqrt{5} - 1)/2$, $k = 1$, $\lambda_1 = a_1 + (1 - g)(b_1 - a_1)$, $\mu_1 = a_1 + g(b_1 - a_1)$, calculate $f(\lambda_1)$ and $f(\mu_1)$.
1: If $b_k - a_k < L$, stop and return $(a_k + b_k)/2$. Otherwise, if $f(\lambda_k) > f(\mu_k)$, goto step 2; if $f(\lambda_k) \leq f(\mu_k)$, goto step 3.
2: Set $a_{k+1} = \lambda_k$, $b_{k+1} = b_k$, $\lambda_{k+1} = \mu_k$, $\mu_{k+1} = a_{k+1} + g(b_{k+1} - a_{k+1})$, calculate $f(\mu_{k+1})$. Let $k = k + 1$, goto step 1.
3: Set $a_{k+1} = a_k$, $b_{k+1} = \mu_k$, $\mu_{k+1} = \lambda_k$, and $\lambda_{k+1} = a_{k+1} + (1 - g)(b_{k+1} - a_{k+1})$, calculate $f(\lambda_{k+1})$. Let $k = k + 1$, goto step 1.

Algorithm 1 performs the steepest descent which only uses the first order derivative. Second order derivative based gradient methods (e.g., Newton method, conjugate gradient method) [32] use the inverse of the Hessian matrix, thus, they can find a more precise solution and converge in fewer iterations. However, in our application, the Hessian matrix $\nabla^2\Phi_\alpha(\mathbf{w})$ is very close to the zero matrix, which will cause instability in the second order gradient method. As such, we will stick to steepest descent in this work.

4 Boosted smooth quantile regression

The gradient descent method presented in Sect. 3 might converge to a poor solution when the predictor is in high dimensional space. Motivated by the gradient descent explanation of boosting algorithm [8, 26], this section proposes a boosting procedure for smooth quantile regression, which performs coordinate descent in functional space, and is able to work in high dimensional spaces.

4.1 Boosting as functional gradient descent

Boosting [7] is well known for its simplicity and good performance. The powerful feature selection mechanism of boosting makes it suitable to work in high dimensional spaces. Friedman et al. [8, 9] developed a general statistical framework which yields a direct interpretation of boosting as a method for function estimation, which is a "stage-wise, additive model".

Consider the problem of function estimation

$$f^*(\mathbf{x}) = \arg\min_f \mathrm{E}[l(Y, f(\mathbf{x}))|\mathbf{x}],$$

where $l(\cdot, \cdot)$ is a loss function which is typically differentiable and convex with respect to the second argument. Estimating $f^*(\cdot)$ from the given data $\{(\mathbf{x}_i, Y_i), i = 1, \ldots, n\}$ can be performed by minimizing the empirical loss $n^{-1}\sum_{i=1}^n l(Y_i, f(\mathbf{x}_i))$ and pursuing iterative steepest descent in functional space. This leads us to the generic functional gradient descent algorithm [8, 26], and Algorithm 3 shows the version summarized in [2].

Many boosting algorithms can be understood as functional gradient descent with appropriate loss function. For example, if we choose $l(Y, f) = \exp(-(2Y - 1)f)$, we would recover AdaBoost [9], and L_2 Boost [3] corresponds to $l(Y, f) = (Y - f)^2/2$.

4.2 Boosting procedure for smooth quantile regression

Assume $\mathcal{H}$ is the collection of all the weak learners (base procedures), i.e., $\mathcal{H} = \{h_1(\mathbf{x}), \ldots, h_d(\mathbf{x})\}$, where d is the total number of weak learners which could possibly be ∞. Denote the space spanned by the weak learners as

$$\mathcal{L}(\mathcal{H}) = \left\{ \sum_{i=1}^d c_i h_i(\mathbf{x}) \middle| c_i \in \mathbf{R}, \quad i = 1, \ldots, d \right\}.$$

Further assume the τth conditional quantile function of the response, $q(\mathbf{x})$, lies in the space spanned by the functions in $\mathcal{H}$, i.e., $q(\mathbf{x}) \in \mathcal{L}(\mathcal{H})$. Given training data $\{(\mathbf{x}_i, Y_i), i = 1, \ldots, n\}$, the quantile regression function of the response could be estimated by

$$q^*(\cdot) = \arg\min_{q(\cdot) \in \mathcal{L}(\mathcal{H})} \frac{1}{n}\sum_{i=1}^n \rho_\tau(Y_i - q(\mathbf{x}_i)), \tag{19}$$

which is equivalent to finding the combination coefficient vector $(c_1, \ldots, c_d)$. Since the check loss function is not everywhere differentiable, we replace it by its smoothed counterpart with a small α, solving

$$q^*(\cdot) = \arg\min_{q(\cdot) \in \mathcal{L}(\mathcal{H})} \frac{1}{n}\sum_{i=1}^n S_{\tau,\alpha}(Y_i - q(\mathbf{x}_i)). \tag{20}$$

By Theorem 1, the solution to Eq. 20 can be used as the approximated quantile regression function.

We consider solving the minimization problem in Eq. 20 in the general framework of functional gradient descent with the loss function

Algorithm 3 Generic Functional Gradient Descent

0: Set $m = 0$, initialize $f^{[0]}(\cdot) = 0$ or set

$$f^{[0]}(\cdot) = \arg\min_c \frac{1}{n}\sum_{i=1}^{n} l(Y_i, c).$$

1: Increase m by 1. Compute the negative gradient $-\frac{\partial}{\partial f} l(Y, f)$ and evaluate at $f^{[m-1]}(\mathbf{x}_i)$:

$$U_i = -\frac{\partial l(Y_i, f)}{\partial f}\bigg|_{f=f^{[m-1]}(\mathbf{x}_i)}, \quad i = 1, \cdots, n.$$

2: Fit the negative gradient vector $U_1, \cdots, U_n$ to $\mathbf{x}_1, \cdots, \mathbf{x}_n$ by the base procedure (e.g., the weak learner in AdaBoost):

$$\{(\mathbf{x}_i, U_i),\ i = 1, \cdots, n\} \longrightarrow g^{[m]}(\cdot).$$

3: Update the estimation by $f^{[m]}(\cdot) = f^{[m-1]}(\cdot) + \eta_m g^{[m]}(\cdot)$, where η_m is a step size factor for the m-th iteration.

4: Check the stopping criterion, if not satisfied, go to step 1.

$$l(Y, q) = S_{\tau,\alpha}(Y - q(\mathbf{x})),$$

which is convex in the second argument. A direct application of Algorithm 3 yields the boosted smooth quantile regression (BS-QReg) algorithm, which is given in Algorithm 4.

Similar to [8], let the base procedure be $h(\mathbf{x}, \mathbf{a})$, where $\mathbf{a}$ is a parameter vector. The fourth step of Algorithm 4 can be performed by an ordinary least square regression:

$$\mathbf{a}_m = \arg\min_{\mathbf{a}} \sum_{i=1}^{n} [U_i - h(\mathbf{x}_i, \mathbf{a})]^2,$$

hence the function $g^{[m]}(\mathbf{x}) = h(\mathbf{x}, \mathbf{a}_m)$ can be regarded as an approximation of the negative gradient by the base procedure. Same as for GDS-QReg (Algorithm 1), in step 5, the step size factor η_m can be determined by a 1-D search

$$\eta_m = \arg\min_{\eta} \sum_{i=1}^{n} S_{\tau,\alpha}\left[Y_i - q^{[m-1]}(\mathbf{x}_i) - \eta g^{[m]}(\mathbf{x}_i)\right],$$

Algorithm 4 Boosted Smooth Quantile Regression (BS-QReg)

0: Given training data $\{(\mathbf{x}_i, Y_i),\ i = 1, \cdots, n\}$, the desired quantile value τ, the maximum iteration number M, and let th be the sensitive level for the change of the objective function.

1: Set $q^{[0]}(\cdot) = 0$ or initialize $q^{[0]}(\cdot)$ with

$$q^{[0]}(\cdot) = \tau\text{-th quantile of } (Y_1, \cdots, Y_n);$$

calculate the objective function $\Phi_\alpha^{[0]}$ by Eqn. (13).

2: **for** $m = 1$ to M **do**

3: Compute the negative gradient $-\frac{\partial}{\partial q} S_{\alpha,\tau}(Y - q)$ and evaluate at $q^{[m-1]}(\mathbf{x}_i)$:

$$U_i = \tau - \frac{1}{1 + \exp\left(\frac{Y_i - q^{[m-1]}(\mathbf{x}_i)}{\alpha}\right)}, \quad i = 1, 2, \cdots, n.$$

4: Fit the negative gradient vector $U_1, \cdots, U_n$ to $\mathbf{x}_1, \cdots, \mathbf{x}_n$ by the base procedure

$$\{(\mathbf{x}_i, U_i),\ i = 1, \cdots, n\} \longrightarrow g^{[m]}(\cdot).$$

5: Update the estimation by $q^{[m]}(\cdot) = q^{[m-1]}(\cdot) + \eta_m g^{[m]}(\cdot)$, where η_m is a step size factor at step m.

6: Calculate the objective function $\Phi_\alpha^{[m]}$ by Eqn. (13); if $\Phi_\alpha^{[m-1]} - \Phi_\alpha^{[m]} < th$, stop.

7: **end for**

8: Output the estimated τ-th quantile function.

and this paper employs Golden section search presented in Algorithm 2. Alternatively, in each iteration, we could update the fitted function, $q^{[m-1]}(\cdot)$, by a fixed but small step in the negative gradient direction. To guarantee the performance of the resulting model, we fix η_m at a small value as suggested by [2, 8]. Similar to AdaBoost, BS-QReg enables us to select most informative predictors if appropriate base learner is employed, and this will be demonstrated experimentally in Sects. 5 and 6.

In Algorithm 4, if we adopt simple linear regression with only one predictor as base procedure, the time complexity for each base procedure is $O(n)$. Assuming there are p predictors and the algorithm stops with M iterations, then the total time complexity of Algorithm 4 is of the order $O(Mnp)$. The empirical comparison of the time complexity for different algorithms will be given in Sects. 5 and 6.

There is a large volume of literature applying boosting to regression problems, for example, in [6, 7, 35]. However, all these methods estimate the mean value of the response, not quantiles. Compared to the quantile regression models [12, 18, 20, 27, 30] mentioned in Sect. 1, the proposed GDS-QReg and BS-QReg converge to a local optimum and they are easy to implement without requiring additional software packages. BS-QReg is efficient in high dimensional spaces. Furthermore, BS-QReg enjoys the flexibility of choosing various types of weak learners, which enables us to perform variable selection.

5 Simulation studies

This section compares the proposed algorithms to alternatives on various simulated datasets. The tested algorithms include the original QReg algorithm implemented based on an interior point method [14, 18] (IP-QReg), majorize–minimize based quantile regression [12] (MM-QReg), the gradient descent smooth QReg (GDS-QReg, Algorithm 1) with Golden section search for step size and fixed step size, the boosted smooth QReg (BS-QReg, Algorithm 4) with Golden section search for step size and fixed step size, the support vector quantile regression (SV-QReg) [13, 24, 30], and the quantile regression forest [27]. If the step size parameter is determined by 1-D search, we search it in the interval [0, 20] to make sure the optimal step size can be found. The GDS-QReg and BS-QReg models are initialized with 0.

Via simulation, we first justify our choice for the smooth parameter α in the smoothed quantile regression model; then we compare the predictive accuracy and time complexity of the proposed algorithms to those of several alternative models by Simulation 2; the third simulation studies the variable selection ability of the proposed methods in the case of high dimension but small sample size; in Simulation 4, we study the performance of various algorithms in the case of non i.i.d. error terms; finally, Simulation 5 compares the performance of the BS-QReg model with small step size to L_1 quantile regression model.

All the experiments presented in Sects. 5 and 6 were performed on a personal computer with Pentium IV CPU 3.00 GHz and 1.00 GB memory.

Simulation 1 (The choice of α and the step size). In the proposed smooth quantile regression algorithms, the smooth parameter α and the step size parameter will influence the performance of the resulting models.

Intuitively, the second order derivative of a function at a point can be used to measure the smoothness at this point: a larger value of the second order derivative implies that the function is less smooth at this point.[1] In order to investigate the influence of α to the smoothness of the smoothed check loss function in Eq. 3 at $x = 0$, we plot out the curves of the second order derivative (presented in Eq. 4) in Fig. 2 for $\alpha = 0.5$ and $\alpha = 0.1$. It is clear that at point $x = 0$, the magnitude of second order derivative of $S_{\tau,\alpha}(x)$ with small α is large, which indicates that it is less smooth. Thus, in this work, to ensure the smoothness of the approximation, we choose α not too small. On the other hand, Lemma 2 indicates that a small value of α ensures that the smoothed check function and the original check function are close.

Using the above principle as a guide, we further demonstrate the choice of α by data generated according to the model

$$Y = \mathbf{x}^T\mathbf{b} + \epsilon, \tag{21}$$

where $\mathbf{b} = (1, 1, 1, 1, 1)^T$ and $\mathbf{x} \sim N(\mathbf{0}, \mathbf{\Sigma}_{5\times 5})$. The pairwise correlation between x_i and x_j is given by $r^{|i-j|}$ with $r = 0.5$. The error term ϵ follows standard double exponential distribution.[2]

We generated 200 training examples and 1,000 testing examples from the model in Eq. 21. Let $q_\tau(\mathbf{x})$ be the true τth quantile function of Y and $\hat{q}_\tau(\mathbf{x})$ be the estimated τth quantile function, the performance of the fitted model on the testing set is evaluated by the mean absolute deviation which is defined as

$$\text{Mean absolute deviation} = \frac{1}{N_{\text{test}}}\sum_{i=1}^{N_{\text{test}}} |q_\tau(\mathbf{x}_i) - \hat{q}_\tau(\mathbf{x}_i)|, \tag{22}$$

[1] For example, $|x|$ is not smooth at $x = 0$, and its second order derivative at $x = 0$ could be understood as ∞ in some sense.

[2] A random variable X follows a double exponential distribution if its probability density function is $f(x) = \frac{1}{2}e^{-|x|}$.

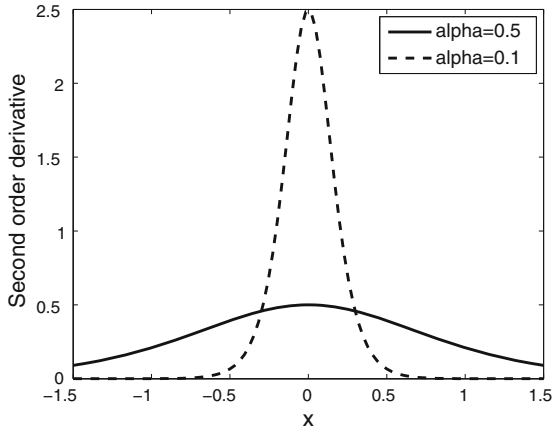

Fig. 2 The second order derivative of the smoothed check loss function Eq. 3 for $\alpha = 0.5$ and $\alpha = 0.1$ with $\tau = 0.5$

where N_{test} is the size of testing set. Under the assumption of Eq. 21, the true τth quantile function of Y can be written out explicitly as

$$q_\tau(\mathbf{x}) = \mathbf{x}^T\mathbf{b} + Q_\tau^{\text{DE}},$$

where Q_τ^{DE} is the τth quantile of the standard double exponential distribution.

The generating-training-testing procedure was repeated 100 times. For three τ values (0.25, 0.5, and 0.75) and different values of the smooth parameter α, Table 1 gives the average of the 100 Mean absolute deviation of the BS-QReg with simple linear regression as base procedure, and the step size parameter was determined by Golden section search. Table 1 shows that the performance of the algorithm is best when α is around 0.5, and this verifies our qualitative analysis, that is, α should be small but not too small. The similar result was observed when using GDS-QReg. In the following simulations, we also tried different α values, and the best performances were observed around $\alpha = 0.5$. The detailed results are not listed due to the limit of space. Thus, we choose $\alpha = 0.5$ in our experiments.

In the generic functional gradient descent algorithm (shown in Algorithm 3), it is claimed in [2] that the choice of the step size factor in step 3 is of minor importance as long as it is "small". A smaller value of fixed step size η_m typically requires a larger number of boosting iterations and thus more computing time, while the predictive accuracy has been empirically found to be good when choosing η_m "sufficiently small" (e.g., $\eta_m = 0.1$) [8]. To balance the predictive accuracy and the computational burden, Bühlmann and Hothorn [2] suggested choose step size as 0.1. By simulations, Friedman [8] showed that the performances are similar if the step size parameter is less than 0.125. Thus, we choose the step size parameter as 0.1 if it needs to be fixed.

Simulation 2 (Comparing the predictive accuracy and time complexity of various methods). We generate data according to

$$Y = \mathbf{x}^T\mathbf{b} + \sigma\epsilon, \tag{23}$$

where $\mathbf{b} = (3, 1.5, 0, 0, 2, 0, 0, 0)^T$ and $\mathbf{x} \sim N(\mathbf{0}, \mathbf{\Sigma}_{8\times 8})$. The pairwise correlation between x_i and x_j is given by $r^{|i-j|}$ with $r = 0.5$. The error term ϵ follows standard normal distribution. We choose $\sigma = 2$, resulting the signal-to-noise (S/N) ratio about 5. The S/N ratio is defined as $\text{Var}(\mathbf{b}^T\mathbf{x})/\text{Var}(\sigma\epsilon)$. This model was considered in many works, for example, in [25, 34].

We generated 100 training examples and 10,000 testing examples from the model in Eq. 23. The performance of the fitted model on the testing set was evaluated by the mean absolute deviation defined in Eq. 22. Under the assumption of Eq. 23, the true τth quantile function of Ycan be written out explicitly as

$$q_\tau(\mathbf{x}) = \mathbf{x}^T\mathbf{b} + \sigma Q_\tau^{\text{Norm}},$$

where Q_τ^{Norm} is the τth quantile of the standard normal distribution.

For fair comparison, we used linear kernel in SV-QReg model, and the simple linear regression model with only one predictor was employed as weak learner in BS-QReg algorithms. The generating-training-testing procedure was repeated 100 times. For three τ values (0.25, 0.50, and 0.75) and each algorithm, we report the averages and standard deviations of the obtained 100 mean absolute deviations in Tables 2 and 3, where the best performance is marked in bold.

Tables 2 and 3 show that, in general, SV-QReg and QReg forest perform worse than the original quantile regression model (IP-QReg); IP-QReg and MM-QReg perform similarly; the proposed gradient based smooth quantile regression models (shown in Table 3) perform better than the alternative models. We further notice that, for the proposed GDS-QReg and BS-QReg algorithms, when the step size parameter is fixed at a small value, the performances are better than the counterparts with the step size found by the 1-D search. The reason is that the 1-D search method is often too greedy while a fixed small step size changes the target function only slightly in each iteration and this could be considered as a less greedy strategy.

Tables 2 and 3 also present the average training time of each model along with the standard deviations.[3] The results show that SV-QReg is most time consuming since it needs to solve a quadratic programming problem; IP-QReg and MM-QReg are most time efficient; among the proposed

[3] Quantile regression forest was implemented based on the R package "quantregForest", while all other algorithms in this paper were implemented using MATLAB, thus the computing time of QReg forest is not comparable to other algorithms. As such, we choose not to provide the training time of QReg forest.

Table 1 The average testing errors of the boosted smooth quantile regression with different values of the smooth parameter α. See text for details

α	0.2	0.3	0.4	0.5	1.0	1.5	2.0
$\tau = 0.25$	0.3155	0.2314	0.2101	0.2052	0.2089	0.2127	0.2152
$\tau = 0.50$	0.1589	0.1538	0.1511	0.1496	0.1490	0.1494	0.1500
$\tau = 0.75$	0.3233	0.2414	0.2205	0.2155	0.2193	0.2235	0.2263

Table 2 The performances of IP-QReg, SV-QReg, quantile regression forest (QReg forest), MM-QReg, and BS-QReg with regression stump as weak learner. Listed are the mean values of the "Mean absolute deviation (MAD)" of 100 runs, and the standard deviations are listed in parentheses. The average training time of each model along with the standard deviations are also shown, with the unit in seconds

Method	IP-QReg	SV-QReg	QReg forest	MM-QReg	BS-QReg stump
MAD					
$\tau = 0.25$	0.706 (0.172)	0.700 (0.032)	0.800 (0.183)	0.704 (0.171)	1.284 (0.148)
$\tau = 0.50$	0.636 (0.147)	0.862 (0.029)	0.619 (0.154)	0.637 (0.145)	1.217 (0.122)
$\tau = 0.75$	0.649 (0.160)	0.690 (0.026)	0.793 (0.175)	0.644 (0.162)	1.295 (0.141)
Time					
$\tau = 0.25$	0.032 (0.009)	56.65 (1.28)	NA	0.034 (0.010)	1.873 (0.300)
$\tau = 0.50$	0.028 (0.008)	56.75 (1.22)	NA	0.029 (0.009)	2.123 (0.282)
$\tau = 0.75$	0.031 (0.007)	56.73 (1.25)	NA	0.035 (0.008)	1.879 (0.271)

Table 3 The performances of Gradient Descent Smooth QReg with Golden section search for step size (GDS-QReg 1) and fixed step size (GDS-QReg 2), boosted smooth QReg with Golden section search for step size (BS-QReg 1) and fixed step size (BS-QReg 2)

Method	GDS-QReg 1	GDS-QReg 2	BS-QReg 1	BS-QReg 2
MAD				
$\tau = 0.25$	0.694 (0.163)	0.692 (0.162)	0.644 (0.154)	**0.580 (0.158)**
$\tau = 0.50$	0.625 (0.138)	0.624 (0.138)	0.587 (0.137)	**0.532 (0.154)**
$\tau = 0.75$	0.633 (0.157)	0.631 (0.155)	0.582 (0.153)	**0.538 (0.159)**
Time				
$\tau = 0.25$	0.230 (0.077)	0.203 (0.057)	0.178 (0.055)	0.140 (0.047)
$\tau = 0.50$	0.209 (0.059)	0.165 (0.043)	0.172 (0.048)	0.115 (0.035)
$\tau = 0.75$	0.233 (0.086)	0.206 (0.055)	0.173 (0.057)	0.141 (0.053)

algorithms (shown in Table 3), the boosting algorithm is more time efficient than the direct gradient descent. Although the 1-D search strategy needs fewer number of iterations (in average, 50 iterations for BS-QReg 1 and 300 iterations for BS-QReg 2) because the searched steps are optimized, the 1-D search itself takes extra time, thus totally the 1-D search strategy is slightly more time consuming than the fixed step size strategy for BS-QReg and GDS-QReg.

For one of the 100 runs, Fig. 3 presents the evolution curves of the objective function in Eq. 13 and the mean absolute deviation in Eq. 22 for GDS-QReg and BS-QReg with fixed step size. It is clear that the objective function decreases monotonically and BS-QReg needs more iterations than GDS-QReg to converge. Although BS-QReg overfits the data slightly, it still has slight advantage in the final predictive precision.

Boosting based methods enjoy the flexibility of choosing other forms of base learner, which is not shared by other quantile regression models in literature. For example, we employed regression stump [31] as the base learner in BS-QReg with fixed step size, and we report the corresponding performance in Table 2. It is observed that the regression stump based BS-QReg does not perform well, and the reason is that it employs an inappropriate form of base learner. The regression stump based BS-QReg also requires considerably more time than the linear regression based BS-QReg because it needs more iterations (in average 1000 iterations) in the case of inappropriate base learner. Nevertheless, this example shows the ability of using flexible base learner by BS-QReg.

Simulation 3 (Dimensionality larger than the sample size). In this experiment, we generated data from model

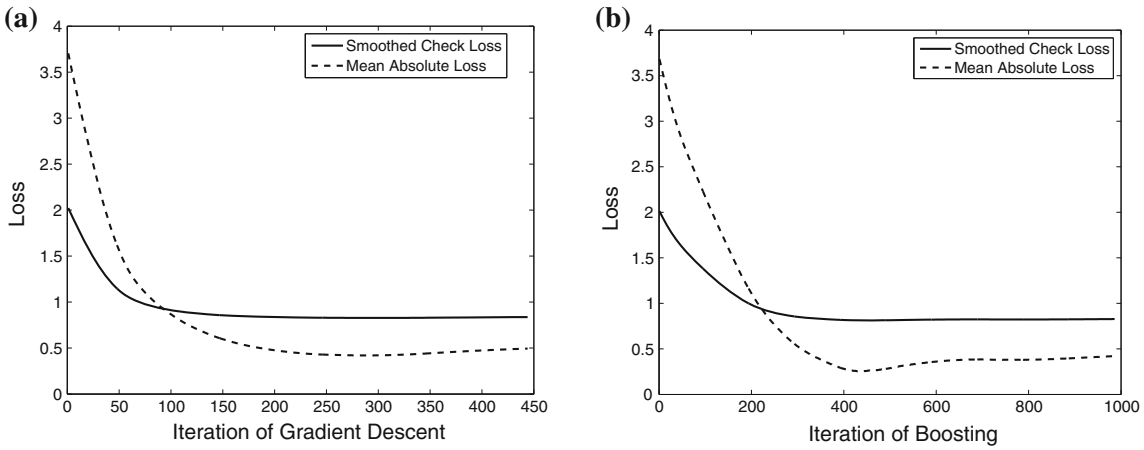

Fig. 3 The evolution curves of objective function and the mean absolute deviation on one of the testing sets: **a** is for Gradient Descent QReg and **b** is for boosted smooth QReg. Both algorithms use the fixed step size 0.1

(23), and augmented the data with 92 noisy variables $x_9, x_{10}, \ldots, x_{100}$, each of which was generated independently from standard normal distribution. The training set consists of 50 examples, and the testing set consists of 10,000 examples. In this way, the dimensionality of the data is much larger than the training sample size, which makes estimation more difficult. Moreover, in this experiment, most of the predictors have no contribution to the response, therefore, it is desirable to identify the informative predictors (i.e., x_1, x_2, and x_5) and suppress the noisy ones.

The models obtained by SV-QReg, GDS-QReg, and BS-QReg are linear functions of the predictors. Since the predictors are at the same scale, it makes sense if we delete the variables with too small coefficients, thus performing variable selection. In applications, variable selection is often needed since it helps us get a simpler model, making the model easier to interpret/understand and identifying informative variables. In this experiment, for any predictor, if the absolute value of its estimated coefficient is less than 0.1 (which is very small compared to the real coefficients of the relevant variables), it is deleted. We calculated the average numbers of the correctly and mistakenly trimmed predictors to measure the variable selection performance.

For three τ values (0.25, 0.5, and 0.75), the experiment was repeated 100 times, and Table 4 presents the average testing errors (i.e., the average mean absolute deviation), the average training time, and the average numbers of correctly and mistakenly trimmed variables of several quantile regression models.[4] Again, the results show that in general, the fixed step size strategy has higher predictive accuracy. We further notice that the testing error of GDS-QReg is almost twice as that of BS-QReg, the reason is that because of the high dimensionality of the data, the gradient descent algorithm converges to a poor solution. The better performance of BS-QReg indicates its adaptability to high dimensional data.

It is observed that the BS-QReg with fixed step size has the best variable selection performance by deleting about 88% of the 97 noninformative variables. Although SV-QReg can achieve a competitive predictive accuracy, it can only delete about 58% of noninformative variables, and fails to exclude some noisy predictors. It is not clear how to identify noninformative predictors by the QReg forest, thus the variable selection result of QReg forest is not reported. From Table 4, we observe that the BS-QReg with fixed step size is again most time efficient among the proposed algorithms.

Simulation 4 (Non *i.i.d.* random errors). This experiment considers the case of non *i.i.d.* random errors to check the robustness of our methods. The datasets were generated from the model

$$y = 1 + x_1 + x_2 + x_3 + (1 + x_3)\epsilon, \tag{24}$$

where x_1 and x_3 were respectively generated from the standard normal distribution and the uniform distribution on [0, 1], $x_2 = x_1 + x_3 + z$ with z being standard normal, and $\epsilon \sim N(0, 1)$. The variables x_1, x_3, z, and ϵ were mutually independent. To study the performance of variable selection, we included five more standard normal noisy variables, $x_4, \ldots, x_8$, independent of each other. In the obtained linear model, we calculate the sum of the absolute values of all the coefficients. Since there are 8 predictors, if the absolute value of the estimated coefficient of a predictor is less than 1% of the total sum, that means the contribution

[4] IP-QReg and MM-QReg were implemented based on the MATLAB code downloaded from http://www.stat.psu.edu/~dhunter/code/qrmatlab/. When the dimensionality is greater than the sample size, the software package gives error message. Thus, the performances of IP-QReg and MM-QReg are not provided.

Table 4 The performance of several algorithms on datasets with dimensionality larger than the sample size. For three τ values, the mean values of the "Mean absolute deviation" of the 100 runs are shown with the standard deviations listed in parentheses. The means and standard deviations of the correctly and mistakenly trimmed variables are listed as measures of variable selection performances. The average training time (in seconds) for each method is also listed. The best performance is marked in bold. Refer to Table 3 for the abbreviations of the methods

τ	Method	Test error	# of trimmed variables		Time
			Correct	Wrong	
0.25	QReg forest	2.8954 (0.2948)	NA	NA	NA
	SV-QReg	1.5461 (0.1667)	55.87 (2.86)	1.54 (0.83)	29.255 (0.195)
	GDS-QReg 1	2.8418 (0.2063)	61.50 (3.12)	0.03 (0.17)	4.268 (1.222)
	GDS-QReg 2	2.8415 (0.2063)	61.50 (3.14)	0.03 (0.17)	4.174 (1.446)
	BS-QReg 1	1.5242 (0.3060)	76.57 (3.03)	0.03 (0.17)	2.065 (0.025)
	BS-QReg 2	**1.3257 (0.4331)**	**86.85 (3.41)**	0.04 (0.20)	0.599 (0.129)
0.50	QReg forest	2.4948 (0.2724)	NA	NA	NA
	SV-QReg	1.5597 (0.0882)	55.87 (2.86)	1.54 (0.83)	29.399 (0.198)
	GDS-QReg 1	2.6416 (0.1991)	61.42 (3.04)	0.03 (0.17)	2.964 (0.865)
	GDS-QReg 2	2.6422 (0.1995)	61.46 (3.18)	0.03 (0.17)	2.349 (0.531)
	BS-QReg 1	1.5085 (0.2545)	74.20 (2.77)	0.00 (0.00)	2.063 (0.024)
	BS-QReg 2	**1.1802 (0.3196)**	**85.08 (3.35)**	0.02 (0.14)	0.463 (0.086)
0.75	QReg forest	2.7906 (0.3119)	NA	NA	NA
	SV-QReg	1.5734 (0.1552)	55.87 (2.86)	1.54 (0.84)	29.431 (0.222)
	GDS-QReg 1	2.8594 (0.1988)	61.52 (3.03)	0.03 (0.17)	4.167 (1.221)
	GDS-QReg 2	2.8590 (0.1988)	61.56 (3.08)	0.03 (0.17)	3.968 (1.122)
	BS-QReg 1	1.5313 (0.2723)	76.80 (2.94)	0.01 (0.10)	2.062 (0.033)
	BS-QReg 2	**1.3312 (0.3749)**	**86.76 (3.05)**	0.06 (0.24)	0.571 (0.121)

of this variable to the model is far less than average, then it is trimmed.

The generating-training-testing was repeated 100 times with training set size 100 and testing set size 10,000. Table 5 shows the average testing errors, the variable selection performance, and time cost of several considered methods. The results in Table 5 once again indicate the advantage of the fixed step size strategy over the 1-D search step size strategy. We further observe that the BS-QReg models yield higher predictive accuracy and are more efficient in removing noninformative variables, and the BS-QReg with fixed small step size has the best overall performance. As in Simulation 2, although BS-QReg with fixed small step size is not as time efficient as IP-QReg and MM-QReg, it is the fastest among the proposed methods.

Simulation 5 (Comparing BS-QReg to L_1 Quantile Regression). The proposed BS-QReg with very small fixed step size belongs to forward stage-wise additive model, which is an approximation to the L_1 constrained model [10]. This simulation experimentally compares the predictive accuracy and variable selection ability of BS-QReg with fixed small step size and L_1 quantile regression (L_1-QReg) [1, 25, 34] which imposes L_1 constraint on the coefficient vector in the linear quantile regression model.

Given data $\{(\mathbf{x}_i, Y_i), i = 1, \ldots, n\}$, assuming the use of linear function $\beta_0 + \boldsymbol{\beta}^T\mathbf{x}$ to estimate the τth quantile of the response, the L_1-QReg is formulated as

$$\begin{aligned} \min_{\beta_0,\boldsymbol{\beta}} \quad & \sum_{i=1}^{n} \rho_\tau(Y_i - \beta_0 - \boldsymbol{\beta}^T\mathbf{x}_i), \\ \text{subject to} \quad & |\beta_1| + \cdots + |\beta_p| \le s. \end{aligned}$$

The solution of L_1-QReg depends on the penalty parameter s, and changing s gives us the evolution of the solution, which is called the solution path [10, 25]. Similarly, for BS-QReg with simple linear regression as base procedure, we can observe the evolution of the regression coefficients with the iteration number.

We compare L_1-QReg and BS-QReg by experiment on simulated data generated according to the model

$$Y = \boldsymbol{\beta}^T\mathbf{x} + \epsilon,$$

where $\boldsymbol{\beta} = (1.5, -1, 1, 0, 0, 0, 0, 0)^T$, the error term ϵ follows standard double exponential distribution, and $\mathbf{x} \in \mathbf{R}^8$ with $\mathbf{x} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_{8\times 8})$. The pairwise correlation between x_i and x_j is given by $r^{|i-j|}$ with $r = 0.5$. 300 training examples and 3,000

Table 5 The performance of several algorithms on the data generated from model Eq. 24. Please refer to the caption of Table 4 for the detailed information

τ	Method	Test error	# of trimmed variables		Time
			Correct	Wrong	
0.25	QReg forest	0.6913 (0.0992)	NA	NA	NA
	IP-QReg	0.5000 (0.1348)	0.68 (0.74)	0.01 (0.10)	0.033 (0.017)
	MM-QReg	0.5004 (0.1339)	0.70 (0.73)	0.01 (0.10)	0.037 (0.012)
	SV-QReg	0.5195 (0.0211)	0.24 (0.50)	0.25 (0.46)	59.858 (0.674)
	GDS-QReg 1	0.4815 (0.1315)	0.61 (0.68)	0.04 (0.19)	0.418 (0.218)
	GDS-QReg 2	0.4761 (0.1259)	0.63 (0.69)	0.04 (0.20)	0.302 (0.161)
	BS-QReg 1	0.4747 (0.1247)	1.18 (0.94)	0.12 (0.33)	0.159 (0.040)
	BS-QReg 2	**0.3638 (0.1298)**	**3.12 (1.12)**	0.40 (0.49)	0.069 (0.018)
0.50	QReg forest	0.6654 (0.0965)	NA	NA	NA
	IP-QReg	0.4379 (0.1185)	0.70 (0.92)	0.02 (0.14)	0.028 (0.008)
	MM-QReg	0.4370 (0.1170)	0.69 (0.88)	0.01 (0.10)	0.029 (0.007)
	SV-QReg	0.6398 (0.0218)	0.27 (0.51)	0.19 (0.44)	59.939 (0.706)
	GDS-QReg 1	0.4256 (0.1153)	0.79 (0.93)	0.01 (0.10)	0.472 (0.200)
	GDS-QReg 2	0.4233 (0.1149)	0.76 (0.94)	0.02 (0.14)	0.282 (0.109)
	BS-QReg 1	0.4210 (0.1099)	1.14 (1.06)	0.03 (0.17)	0.156 (0.039)
	BS-QReg 2	**0.3379 (0.1133)**	**2.87 (1.05)**	0.20 (0.40)	0.058 (0.017)
0.75	QReg forest	0.7456 (0.1010)	NA	NA	NA
	IP-QReg	0.4794 (0.1075)	0.83 (0.91)	0.00 (0.00)	0.030 (0.006)
	MM-QReg	0.4802(0.1068)	0.79 (0.87)	0.00 (0.00)	0.033 (0.005)
	SV-QReg	0.5164 (0.0181)	0.22 (0.42)	0.14 (0.38)	59.890 (0.580)
	GDS-QReg 1	0.4635 (0.1057)	0.78 (0.85)	0.00 (0.00)	0.478 (0.203)
	GDS-QReg 2	0.4591 (0.1069)	0.78 (0.79)	0.00 (0.00)	0.347 (0.151)
	BS-QReg 1	0.4467 (0.1105)	1.17 (0.98)	0.00 (0.00)	0.132 (0.044)
	BS-QReg 2	**0.3891 (0.1217)**	**3.03 (1.13)**	0.07 (0.25)	0.069 (0.018)

testing observations were generated from the model. We fit the L_1-QReg and BS-QReg models for $\tau = 0.5$, and use simple linear regression with only one predictor as base procedure for BS-QReg, the step size in BS-QReg is fixed at 0.1. Define the closeness measure between the estimated model and the true model as $\sum_{i=1}^{8} |\beta_i - \hat{\beta}_i|$, where $\hat{\beta}_i$ is the estimated regression coefficient.

Figure 4 shows the solution paths, the check losses, and the testing errors of the L_1-QReg and BS-QReg. Firstly, we observe that both L_1-QReg and BS-QReg correctly select the informative predictors, i.e., x_1, x_2, and x_3, and all other predictors have coefficients very close to zero. The final estimated coefficients by L_1-QReg and BS-QReg are listed in Table 6. Although the shapes of the solution paths are different as indicated in Fig. 4, it is clear from Table 6 that the two estimates are both very close to the underlying true model with closeness measures 0.1101 for L_1-QReg and 0.1010 for BS-QReg, respectively. Secondly, from the average check loss curves, we see the trends in the check loss and the testing error are similar. The average check losses are 0.4818 for L_1-QReg and 0.4826 for BS-QReg; the mean absolute deviations are 0.0687 for L_1-QReg and 0.0790 for BS-QReg.

The comparison demonstrates that the L_1-QReg and the proposed BS-QReg have similar performances in variable selection and predictive accuracy. However, as we

Table 6 The final estimated regression coefficients by L_1-QReg and BS-QReg

Coefficients	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$
L_1-QReg	1.4648	−1.0266	0.9710	0.0000	0.0000	0.0000	0.0000	0.0193
BS-QReg	1.4575	−1.0134	0.9678	0.0000	0.0000	0.0000	0.0000	0.0129

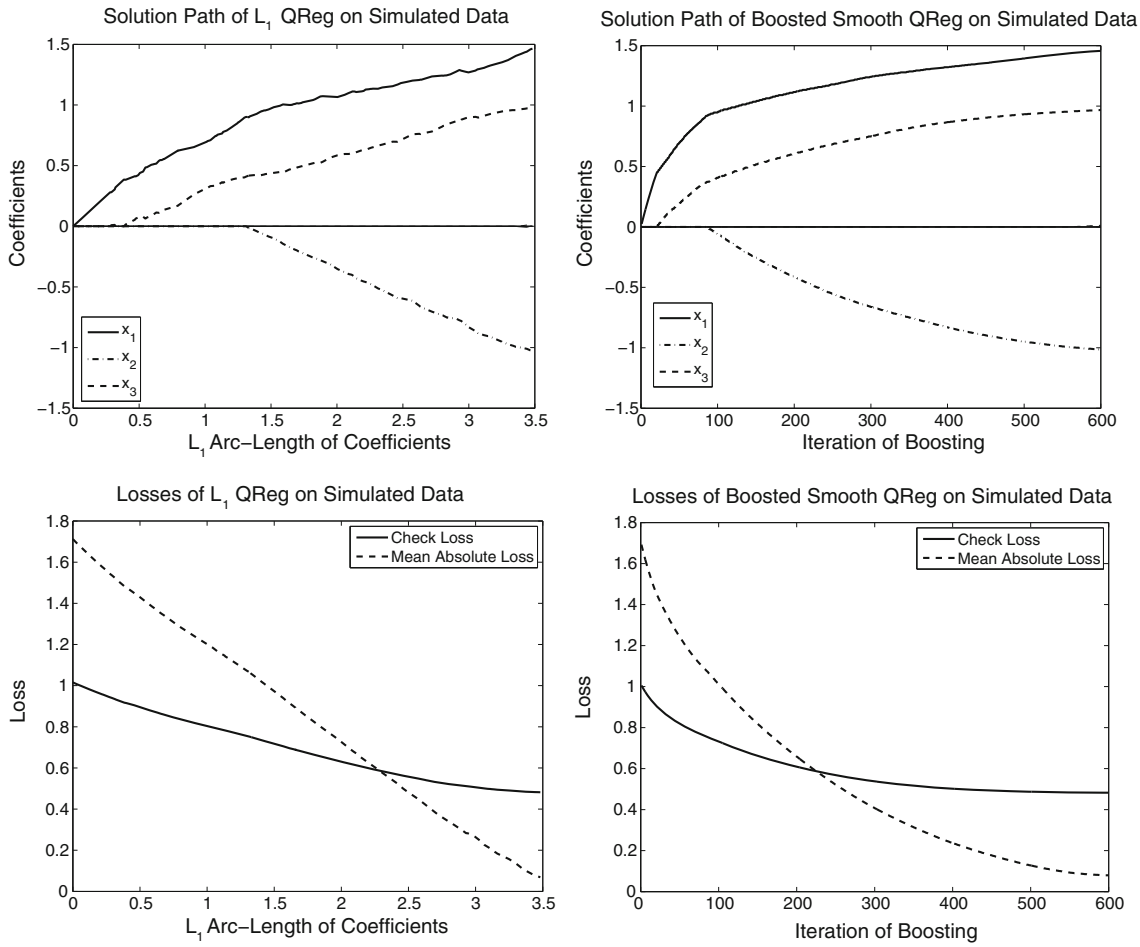

Fig. 4 Solution paths and the average check losses on testing data of L_1-QReg and boosted smooth QReg. In the plots for L_1-QReg, the x-axis is the sum of the absolute values of the solution, i.e. $\sum_{i=1}^{8} |\hat{\beta}_i|$.

mentioned, BS-QReg enjoys the flexibility of using other forms of base learner, while the L_1-QReg in [25, 34] is only for linear model.

6 Results on real data

This section compares the performances of the proposed algorithms to alternative quantile regression models on two real-world datasets. The parameter setting of the algorithms are the same as in Sect. 5. On real-world datasets, although the relevant variables are usually unknown, a simpler model is easier to understand and interpret, thus we none the less prefer a model which can delete more variables. In this section, we adopt the same strategy as in Simulation 4 of Sect. 5 for variable selection.

The considered algorithms include IP-QReg, MM-QReg, QReg forest, SV-QReg with linear kernel, gradient descent smooth QReg with fixed step size and with the step size determined by Golden section search, and boosted smooth QReg using simple linear regression as base procedure with fixed step size and with the step size determined by Golden section search.

6.1 Boston House dataset

The Boston House dataset is available online at http://lib.stat.cmu.edu/datasets/boston_corrected.tx. The dataset contains 506 observations, 15 non-constant predictor variables, and one response variable, CMEDV. We excluded the categorical variable RAD from the predictors. The quantile regression models were trained with the standardized CMEDV as the response and 27 predictor variables including the variable CHAS, the other 13 standardized predictor variables, and their squares.

We randomly select 150 observations (about 30% of the dataset) as training set, and the remaining 356 as testing set. To numerically evaluate the performances, in lack of

the true quantile functions for the dataset, we adopt the average of the "check loss" on the testing set as the error measure, which is defined as

$$L(\tau) = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \rho_\tau(Y_i - \hat{q}_\tau(\mathbf{x}_i)),$$

where N_{test} is the size of the testing set, and $\hat{q}_\tau(\mathbf{x}_i)$ is the estimated τth quantile of the response at $\mathbf{x}_i$. By the definition of quantile, a smaller value of $L(\tau)$ gives a closer estimated quantile to the true quantile.

For three τ values (0.25, 0.50, and 0.75), the partition-training-testing process was repeated 100 times for each algorithm. The means and the standard deviations of the 100 check losses were calculated. We also calculated the average numbers of the trimmed predictors to measure the variable selection performance. Table 7 shows the testing errors, the average training time, and the variable selection performance of the considered algorithms.

From Table 7, we can observe that compared to IP-QReg, MM-QReg, and SV-QReg models, the proposed GDS-QReg and BS-QReg models achieve better performance in terms of predictive accuracy and variable selection ability. We notice that on this real world dataset, the fixed step size strategy still has advantage over the 1-D search strategy in terms of predictive accuracy for both GDS-QReg and BS-QReg. BS-QReg with fixed step size achieves the best testing error and deletes the most non-informative predictors.

The proposed BS-QReg algorithm enables us to select other forms of base learner, for example, regression trees [8, 9]. As an illustration, Table 7 also shows the prediction error of BS-QReg using regression stump [31] as weak learner with fixed step size. Although QReg forest and BS-QReg with regression stump can achieve competitive testing errors, the models obtained by these methods are not linear in the predictors, therefore it is not clear how to perform variable selection.

Table 7 The performance of several algorithms on the Boston House data. For three τ values, we list the mean values of the check loss of 100 runs with the standard deviations listed in parentheses. The means and standard deviations of the deleted variables (DV) are listed as measures of variable selection performance, and the time cost (in seconds) of different algorithms are also given. The best performance is marked in bold

τ	Method	Test error	# of DV	Time
0.25	QReg forest	0.1184 (0.0076)	NA	NA
	IP-QReg	0.1277 (0.0178)	5.76 (1.79)	0.043 (0.009)
	MM-QReg	0.1267 (0.0177)	5.65 (1.52)	0.110 (0.007)
	SV-QReg	0.1263 (0.0188)	5.86 (1.72)	4.995 (0.016)
	GDS-QReg (1-D search step)	0.1202 (0.0100)	5.87 (1.81)	9.305 (0.300)
	GDS-QReg (fixed step)	0.1163 (0.0096)	5.89 (1.85)	5.313 (0.034)
	BS-QReg (1-D search step)	0.1143 (0.0065)	8.37 (2.02)	4.058 (0.028)
	BS-QReg (fixed step)	**0.1112 (0.0056)**	**10.87 (1.82)**	0.829 (0.248)
	BS-QReg stump (fixed step)	0.1246 (0.0116)	NA	2.254 (0.0974)
0.50	QReg forest	0.1480 (0.0105)	NA	NA
	IP-QReg	0.1601 (0.0227)	5.71 (1.82)	0.040 (0.009)
	MM-QReg	0.1576 (0.0161)	5.74 (1.83)	0.088 (0.015)
	SV-QReg	0.1583 (0.0191)	5.77 (1.75)	4.977 (0.017)
	GDS-QReg (1-D search step)	0.1600 (0.0115)	5.91 (1.62)	10.034 (0.417)
	GDS-QReg (fixed step)	0.1486 (0.0091)	5.50 (1.68)	5.312 (0.035)
	BS-QReg (1-D search step)	0.1478 (0.0097)	7.92 (1.76)	7.175 (0.218)
	BS-QReg (fixed step)	**0.1460 (0.0067)**	**9.77 (1.73)**	1.008 (0.251)
	BS-QReg stump (fixed step)	0.1515 (0.0123)	NA	3.336 (0.230)
0.75	QReg forest	0.1338 (0.0108)	NA	NA
	IP-QReg	0.1511 (0.0524)	5.70 (1.60)	0.043 (0.010)
	MM-QReg	0.1465 (0.0231)	5.78 (1.67)	0.113 (0.009)
	SV-QReg	0.1475 (0.0204)	5.69 (1.88)	4.990 (0.014)
	GDS-QReg (1-D search step)	0.1383 (0.0149)	5.76 (1.93)	10.957 (0.951)
	GDS-QReg (fixed step)	0.1348 (0.0104)	5.98 (1.83)	5.307 (0.023)
	BS-QReg (1-D search step)	0.1340 (0.0076)	7.66 (1.94)	5.816 (0.663)
	BS-QReg (fixed step)	**0.1328 (0.0103)**	**10.34 (1.808)**	1.016 (0.299)
	BS-QReg stump (fixed step)	0.1376 (0.0115)	NA	2.306 (0.166)

6.2 BigMac dataset

The R package *alr3* [33] provides the dataset BigMac, which contains the minutes of labor necessary to purchase a Big Mac in 69 cities worldwide, along with 9 predictor variables like primary teacher's gross income, tax rate paid by a primary teacher, primary teacher's hours of work per week, and so on. All the predictor variables and the response variable are standardized to have 0 mean and unit standard deviation.

We randomly selected 20 observations (about 30% of the dataset) for training, and the remaining 49 were used for the testing purpose. We repeated the experiment in Sect. 6.1, and the performance measures of the considered quantile regression models are summarized in Table 8.

Again, on this dataset, the result shows that the fixed step size strategy is a favorable choice for the proposed GDS-QReg and BS-QReg. We observe that on this dataset, QReg forest and BS-QReg with regression stump have the best testing error measures. However, the models obtained by regression forest or regression stump are not linear in the predictors, while the other considered methods get linear models which are easier to interpret and understand. Among all the obtained linear models, we see that the BS-QReg with fixed step size achieves the best testing error (displayed in italic) and the variable selection performance. Same as for the Boston House dataset, the proposed GDS-QReg and BS-QReg models yield better performance than IP-QReg, MM-QReg, and SV-QReg models in terms of predictive accuracy and variable selection power.

From Tables 7 and 8, we see that same as in Sect. 5, on real-world datasets, although BS-QReg with fixed small step size is not as fast as IP-QReg and MM-QReg in training, it is the most time efficient among the proposed methods.

7 Conclusion

The original quantile regression (QReg) minimizes the check loss function which is not differentiable, and this prevents the gradient based optimization methods from

Table 8 The performance of several algorithms on the BigMac data. See the caption of Table 7 for details

τ	Method	Test error	# of DV	Time
0.25	QRegForest	**0.1455 (0.0160)**	NA	NA
	IP-QReg	0.2819 (0.1408)	0.93 (1.02)	0.023 (0.010)
	MM-QReg	0.2631 (0.1022)	0.95 (1.01)	0.025 (0.009)
	SV-QReg	0.2759 (0.1205)	0.95 (1.14)	0.041 (0.008)
	GDS-QReg (1-D search step)	0.2145 (0.0770)	0.66 (0.87)	3.134 (0.823)
	GDS-QReg (fixed step)	0.2172 (0.0717)	0.52 (0.72)	0.911 (0.011)
	BS-QReg (1-D search step)	0.1697 (0.0296)	2.61 (1.42)	0.115 (0.054))
	BS-QReg (fixed step)	*0.1478 (0.0168)*	**4.18 (1.26)**	0.034 (0.020)
	BS-QReg stump (fixed step)	0.1582 (0.0275)	NA	0.131 (0.065)
0.50	QReg forest	**0.2129 (0.0319)**	NA	NA
	IP-QReg	0.3482 (0.1333)	0.79 (0.82)	0.023 (0.008)
	MM-QReg	0.3360 (0.0860)	0.94 (0.91)	0.024 (0.007)
	SV-QReg	0.3318 (0.0955)	1.02 (0.97)	0.041 (0.008)
	GDS-QReg (1-D search step)	0.2899 (0.0687)	0.66 (0.71)	4.198 (0.658)
	GDS-QReg (fixed step)	0.2822 (0.0511)	0.56 (0.69)	0.905 (0.062)
	BS-QReg (1-D search step)	0.2581 (0.0423)	2.74 (1.36)	0.143 (0.081)
	BS-QReg (fixed step)	*0.2329 (0.0271)*	**4.00 (1.34)**	0.033 (0.022)
	BS-QReg stump (fixed step)	0.2186 (0.0367)	NA	0.131 (0.109)
0.75	QReg forest	0.2138 (0.0503)	NA	NA
	IP-QReg	0.3389 (0.1201)	1.15 (0.94)	0.023 (0.008)
	MM-QReg	0.3024 (0.0687)	0.88 (0.91)	0.026 (0.006)
	SV-QReg	0.3278 (0.1011)	0.79 (0.91)	0.041 (0.008)
	GDS-QReg (1-D search step)	0.2834 (0.0608)	0.57 (0.76)	3.306 (0.465)
	GDS-QReg (fixed step)	0.2794 (0.0511)	0.60 (0.69)	0.910 (0.012)
	BS-QReg (1-D search step)	0.2658 (0.0413)	3.27 (1.29)	0.143 (0.059)
	BS-QReg (fixed step)	*0.2465 (0.0421)*	**4.27 (1.28)**	0.038 (0.022)
	BS-QReg stump (fixed step)	**0.2096 (0.0503)**	NA	0.128 (0.110)

being applicable. As such, this paper introduces the smoothed check function, and studies the relationship between the smoothed QReg model and the original QReg model. Two gradient based algorithms were proposed for minimizing the smoothed objective function: the first approach uses gradient descent to minimize the smoothed loss function, resulting the gradient descent smooth quantile regression algorithm; the second method minimizes the smoothed objective function by coordinate descent in functional space spanned by the base learners, which yields boosted smooth quantile regression algorithm. The proposed algorithms are easy to implement, without requiring additional optimization software package. Extensive experiments on simulated data and real-world data show that, compared to alternative quantile regression models, the proposed smooth quantile regression algorithms can achieve better testing accuracy and are more efficient in excluding the noninformative variables from the model.

In the proposed algorithms, we designed two strategies for selecting the step size parameter, i.e., by 1-D search and by fixing it at a small value. The experiments show that the 1-D search based algorithm often performs worse than the fixed step size strategy because 1-D search is usually too greedy. Although the algorithms with 1-D search step size converges in fewer iterations compared to the fixed step size strategy, it needs extra time to perform the search. Our experiments on simulated data and real-world data show that the boosted smooth QReg with small fixed step size is always the most time efficient among the proposed methods. Thus, in applications, we recommend the boosted smooth QReg with small fixed step size.

The proposed boosted smooth QReg algorithms only have "forward" steps, that is, they select variables which cause direct decreasing of the loss function. Moreover, the boosted smooth QReg algorithms are in the framework of coordinate descent, which is greedy. Thus, it is possible that the boosted smooth QReg algorithms are too greedy and pick up some irrelevant variables. In literature, there are "backward" steps in model fitting [10, 23, 36], for example, Forward-Selection and Backward-Elimination, Growing-then-Pruning a tree. As such, it is well motivated to introduce "backward" step in boosted smooth QReg algorithms, that is, deleting some variables at certain stage, and this is our next step of research.

Acknowledgments The short version of this paper [37] was published on the *9th Mexican International Conference on Artificial Intelligence (MICAI)*. The author gratefully acknowledges the anonymous reviewers of MICAI and *International Journal of Machine Learning and Cybernetics* for their constructive comments and would like to extend his gratitude to Prof. Grigori Sidorov for his excellent work in coordinating the preparation and the reviewing of the manuscript. This work was partially supported by a 2011 Summer Faculty Fellowship of Missouri State University.

References

1. Belloni A, Chernozhukov V (2011) ℓ_1-Penalized quantile regression in high-dimensional sparse models. Ann Stat 39(1):82–130
2. Bühlmann P, Hothorn T (2007) Boosting algorithms: regularization, prediction and model fitting. Stat Sci 22:477–505
3. Bühlmann P, Yu B (2003) Boosting with the L_2 Loss: regression and classification. J Am Stat Assoc 98:324–340
4. Cade BS, Noon BR (2003) A gentle introduction to quantile regression for ecologists. Front Ecol Environ 1(8):412–420
5. Chen C, Mangasarian OL (1996) A class of smoothing functions for nonlinear and mixed complementarity problems. Comput Optim Appl 5:97–138
6. Duffy N, Helmbold D (2002) Boosting methods for regression. Mach Learn 47(2–3):153–200
7. Freund Y, Schapire R (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139
8. Friedman J (2001) Greedy function approximation: a gradient boosting machine. Ann Stat 29:1189–1232
9. Friedman J, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting. Ann Stat 28(2):337–407
10. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: data mining, inference, and prediction. 2nd edn. Springer, New York
11. Hendricks W, Koenker R (1992) Hierarchical spline models for conditional quantiles and the demand for electricity. J Am Stat Assoc 93:58–68
12. Hunter DR, Lange K (2000) Quantile regression via an MM algorithm. J Comput Graph Stat 19(1):60–77
13. Hwang C, Shim J (2005) A simple quantile regression via support vector machine. In: Lecture notes in computer science, vol 3610/2005, pp 512–520
14. Koenker R (2005) Quantile regression. Cambridge University Press, Cambridge
15. Koenker R, Bassett G (1978) Regression quantiles. Econometrica 46:33–50
16. Koenker R, Geling R (2001) Reappraising medfly longevity: a quantile regression survival analysis. J Am Stat Assoc 96:458–468
17. Koenker R, Hallock K (2001) Quantile regression. J Econ Perspect 15:143–156
18. Koenker R, Park BJ (1996) An interior point algorithm for nonlinear quantile regression. J Econ 71:265–283
19. Kriegler B, Berk R (2007) Boosting the quantile distribution: a cost-sensitive statistical learning procedure. Technical report, Department of Statitics. University of California, Los Angeles
20. Langford J, Oliveira R, Zadrozny B (2006) Predicting conditional quantiles via reduction to classification. In: Proceedings of uncertainty in artificical intelligence, pp 257–264
21. Lee Y-J, Mangasarian OL (2001) SSVM: a smooth support vector machine for classification. Comput Optim Appl 20(1):5–22
22. Li C, Wei Y, Chappell R, He X (2011) Bent line quantile regression with application to an allometric study of land mammals' speed and mass. Biometrics 67(1):242–249
23. Li SZ, Zhang Z (2004) FloatBoost learning and statistical face detection. IEEE Trans Pattern Anal Mach Intell 26(9):1112–1123
24. Li Y, Liu Y, Zhu J (2007) Quantile regression in reproducing Kernel Hilbert spaces. J Am Stat Assoc 102:255–268
25. Li Y, Zhu J (2008) L_1-norm quantile regression. J Comput Graph Stat 17(1):163–185
26. Mason L, Baxter J, Bartlett PL, Frean M (2000) Boosting algorithms as gradient descent. Adv Neural Inform Process Syst 12:512–518

27. Meinshausen N (2006) Quantile regression forests. J Mach Learn Res 7:983–999
28. Sohn I, Kim S, Hwang C, Lee JW (2008) New normalization methods using support vector machine quantile regression approach in microarray analysis. Comput Stat Data Anal 52(8):4104–4115
29. Sohn I, Kim S, Hwang C, Lee JW, Shim J (2008) Support vector machine quantile regression for detecting differentially expressed genes in microarray analysis. Methods Inf Med 47(5):459–467
30. Takeuchi I, Le QV, Sears TD, Smola AJ (2006) Nonparametric quantile estimation. J Mach Learn Res 7:1231–1264
31. Torralba A, Murphy KP, Freeman WT (2004) Sharing features: efficient boosting procedures for multiclass object detection. In: Proceeding of IEEE conference on computer vision and pattern recognition (CVPR), pp 762–769
32. Walsh GR (1975) Methods of optimization. Wiley, New York
33. Weisberg S (2005) Applied linear regression, 3rd edn. Wiley, New York
34. Wu Y, Liu Y (2009) Variable selection in quantile regression. Stat Sin 19:801–817
35. Zemel R, Pitassi T (2001) A gradient-based boosting algorithm for regression problems. In: Proceedings of advances in neural information processing systems
36. Zhao P, Yu B (2007) Stagewise Lasso. J Mach Learn Res 8:2701–2726
37. Zheng S (2010) Boosting based conditional quantile estimation for regression and binary classification. In: The 9th Mexican international conference on artificial intelligence, Pachuca, LNAI 6438, pp 67–79