Math. Prog. Comp. (2017) 9:471–497
DOI 10.1007/s12532-017-0117-2

FULL LENGTH PAPER

A branch-and-bound algorithm for instrumental variable quantile regression

Guanglin Xu[1] · **Samuel Burer**[1]

Received: 1 November 2015 / Accepted: 27 December 2016 / Published online: 9 January 2017

Abstract This paper studies a statistical problem called *instrumental variable quantile regression* (IVQR). We model IVQR as a convex quadratic program with complementarity constraints and—although this type of program is generally NP-hard—we develop a branch-and-bound algorithm to solve it globally. We also derive bounds on key variables in the problem, which are valid asymptotically for increasing sample size. We compare our method with two well known global solvers, one of which requires the computed bounds. On random instances, our algorithm performs well in terms of both speed and robustness.

Keywords Convex quadratic programming · Complementarity constraints · Branch-and-bound · Quantile regression

Mathematics Subject Classification 90C20 · 90C26 · 90C33

1 Introduction

Least-squares linear regression [23] estimates the conditional expectation of a random variable $\boldsymbol{b} \in \mathbb{R}$ as a function of random covariates $\boldsymbol{a_1} \in \mathbb{R}^{n_1}$ and a random error term $\boldsymbol{\epsilon} \in \mathbb{R}$ by modeling

$$\boldsymbol{b} = \boldsymbol{a_1}^{\mathrm{T}} x_1^* + \boldsymbol{\epsilon} \quad \text{with} \quad \mathrm{E}[\boldsymbol{\epsilon} \mid \boldsymbol{a_1} = a_1] = 0,$$

✉ Guanglin Xu
guanglin-xu@uiowa.edu

Samuel Burer
samuel-burer@uiowa.edu

1 Department of Management Sciences, University of Iowa, Iowa City, IA 52242-1994, USA

where $x_1^* \in \mathbb{R}^{n_1}$ is a vector of coefficients, E$[\cdot \mid \cdot]$ denotes conditional expectation, and a_1 is any specific realization of $\boldsymbol{a_1}$. Note that we use bold letters to denote random variables and regular letters to denote realizations. Based on m realizations $(b_i, a_{1i}) \in \mathbb{R}^{1+n_1}$ of $(\boldsymbol{b}, \boldsymbol{a_1})$ encoded as a matrix $(b, A_1) \in \mathbb{R}^{m\times(1+n_1)}$, an estimate $\hat{x}_1$ of x_1^* is obtained by minimizing the sum of squared residuals in the sample (b, A_1). Specifically, $\hat{x}_1 := \arg\min_{x_1} \|b - A_1x_1\|_2^2$. The final calculated residuals $b - A_1\hat{x}_1$ can be viewed as m realizations of the error term $\boldsymbol{\epsilon}$.

It is well-known that least-squares regression is sensitive to outliers in data samples. On the other hand, quantile regression [6,17] can be used as an alternative that is less sensitive to outliers. Although we can consider any quantile index $u \in (0, 1)$, we restrict our attention for the sake of simplicity to the median-case where $u = 1/2$:

$$\boldsymbol{b} = \boldsymbol{a_1}^{\mathrm{T}} x_1^* + \boldsymbol{\epsilon} \quad \text{with} \quad P(\boldsymbol{\epsilon} \le 0 \mid \boldsymbol{a_1} = a_1) = \frac{1}{2},$$

where $P(\cdot \mid \cdot)$ denotes conditional probability. The goal here is to find x_1^* so that, for any new realization $(\bar{b}, \bar{a}_1)$ of $(\boldsymbol{b}, \boldsymbol{a_1})$, the probability that $\bar{a}_1^{\mathrm{T}} x_1^*$ exceeds $\bar{b}$ is exactly $1/2$. Given the sample (b, A_1), let us define a loss function for the i-th observation in terms of the quantile index $u = 1/2$:

$$\begin{aligned}\rho_{1/2}(b_i - a_{1i}^{\mathrm{T}}x_1) &= \frac{1}{2} \cdot \max\left\{b_i - a_{1i}^{\mathrm{T}}x_1, 0\right\} + \left(1 - \frac{1}{2}\right) \cdot \max\left\{a_{1i}^{\mathrm{T}}x_1 - b_i, 0\right\} \\ &= \frac{1}{2}\left|b_i - a_{1i}^{\mathrm{T}}x_1\right|.\end{aligned}$$

Then the associated estimation problem of the median-case quantile regression corresponds to calculating $\hat{x}_1 \in \operatorname{Arg\,min}_{x_1} \sum_{i=1}^m \rho_{1/2}(b_i - a_{1i}^{\mathrm{T}}x_1)$, where $\sum_{i=1}^m \rho_{1/2}(b_i - a_{1i}^{\mathrm{T}}x_1)$ is the overall loss or estimation error.

Let us briefly discuss the intuition of the above minimization problem by considering a simpler problem: $\min_{\zeta\in\mathbb{R}} \sum_{i=1}^m \rho_{1/2}(b_i - \zeta)$. One can show that the symmetry of the piecewise linear function $\rho_{1/2}(\cdot)$ ensures that, for an optimal solution ζ^*, it must hold that $|\{i : b_i > \zeta^*\}| = |\{i : b_i < \zeta^*\}|$, i.e., the number of positive errors equals the number of negative errors. Hence, ζ^* is the median of sample b, i.e., $\zeta^* = \operatorname{med}(b_i)$. This analysis applies more generally to the quantile regression estimation of the previous paragraph, so that the estimate $\hat{x}_1$ ensures an equal number of positive and negative errors; see the details in [17,24].

After dropping a constant factor $1/2$ in the objective function, we have

$$\hat{x}_1 \in \operatorname{Arg}\min_{x_1} \|b - A_1x_1\|_1 \longleftrightarrow \begin{array}{ll} \min_{x_1,x_3^+,x_3^-} & e^{\mathrm{T}}x_3^+ + e^{\mathrm{T}}x_3^- \\ \text{s.t.} & x_3^+ - x_3^- = b - A_1x_1 \\ & x_3^+, x_3^- \ge 0. \end{array}$$

This is a linear program (LP) in which the variables $x_3^+, x_3^- \in \mathbb{R}^m$ are auxiliary variables, and $e \in \mathbb{R}^m$ is the vector of all ones. (Note that we reserve the notation x_2 for below.)

When there is sampling bias, i.e., sampling exhibits $P(\boldsymbol{\epsilon} \le 0 \mid \boldsymbol{a_1} = a_1) \neq \frac{1}{2}$, the estimate $\hat{x}_1$ provided by quantile regression may be inaccurate. In such cases, the presence of additional covariates $\boldsymbol{a_2} \in \mathbb{R}^{n_2}$, called *instruments*, can often be exploited to correct the bias [7,15], i.e., sampling with both $\boldsymbol{a_1}$ and $\boldsymbol{a_2}$ properly exhibits $P(\boldsymbol{\epsilon} \le 0 \mid \boldsymbol{a_1} = a_1, \boldsymbol{a_2} = a_2) = \frac{1}{2}$. While this could serve as the basis for a model $\boldsymbol{b} = \boldsymbol{a_1}^{\mathrm{T}} x_1^* + \boldsymbol{a_2}^{\mathrm{T}} x_2^* + \boldsymbol{\epsilon}$, the hope is to minimize the effect of $\boldsymbol{a_2}$ so that the model depends clearly on the endogenous covariates $\boldsymbol{a_1}$, not the instruments $\boldsymbol{a_2}$. For example, the most desirable case would have $x_2^* = 0$.

Hence, in *instrumental variable quantile regression (IVQR)*, we define the estimator $\hat{x}_1$ of x_1^* such that the instruments $\boldsymbol{a_2}$ do not help in the conditional quantile. In other words, we (ideally) choose $\hat{x}_1$ such that

$$0 \in \operatorname{Arg}\min_{x_2} \|b - A_1\hat{x}_1 - A_2 x_2\|_1$$

where $x_2 \in \mathbb{R}^{n_2}$ is a variable and $(b, A_1, A_2) \in \mathbb{R}^{m\times(1+n_1+n_2)}$ is the sample data. Note that such a desirable $\hat{x}_1$ may not exist; see the next paragraph. The corresponding LP is

$$\begin{array}{lll} \min_{x_2, x_3^+, x_3^-} & e^{\mathrm{T}} x_3^+ + e^{\mathrm{T}} x_3^- & \\ \text{s.t.} & x_3^+ - x_3^- + A_2 x_2 = b - A_1\hat{x}_1 & \quad (1) \\ & x_3^+, x_3^- \ge 0 & \end{array}$$

Note that $\hat{x}_1$ is not a variable in (1). Rather, given an estimate $\hat{x}_1$ of x_1^*, the purpose of (1) is to verify that $\hat{x}_2 = 0$ leads to minimal model error. So the overall IVQR problem is to find a value $\hat{x}_1$ having this desired property. This is a type of inverse optimization problem because we desire that part of the optimal solution have a pre-specified value (namely, $\hat{x}_2 = 0$). Following the statistical requirements of IVQR (see, for instance, [4,7]), we force the relation $m > n_2 \ge n_1$ in this paper.

In actuality, there may not exist an $\hat{x}_1$ providing an optimal $\hat{x}_2 = 0$ as just described. So instead we will choose $\hat{x}_1$ such that that $\hat{x}_2$ optimizes (1) with minimum Euclidean norm. We will show in Sect. 2.1 that the resulting problem is a *convex quadratic program with complementarity constraints* (CQPCC), which is generally NP-hard to solve.

The IVQR problem was introduced in [7], where the authors carried out a statistical analysis of the estimation of x_1^* and provided asymptotic normality and standard-error calculations. For $n_1 = 1$, the authors presented a simple, effective enumeration procedure for calculating the estimate. However, they also pointed out that, for larger n_1, their enumeration procedure would suffer from the curse of dimensionality. This provides another perspective on the difficulty of solving the CQPCC mentioned in the previous paragraph.

Our paper is organized as follows. In Sect. 1.1, we briefly review the relevant literature, especially inverse optimization, partial inverse optimization, linear programs with complementarity constraints, and techniques for non-convex quadratic programs, and in Sect. 1.2, we establish the notation we use in the paper. Then in Sect. 2, we discuss the IVQR problem in detail. Section 2.1 formulates IVQR as a CQPCC and proposes a tractable relaxation by dropping the complementarity constraints. In Sect. 2.2, we derive valid bounds on key variables in IVQR, which hold asymptotically for increasing sample size in both light- and heavy-tailed models. The bounds will be used by one of the two global optimization solvers in Sect. 4 but are also of independent interest.

In Sect. 3, we propose a convex-QP-based B&B algorithm to solve IVQR globally. Our B&B algorithm works by enforcing the complementarity constraints via linear constraints in nodes of the tree. We detail the scheme and structure of the B&B algorithm in Sect. 3.1 and describe important implementation issues in Sect. 3.2. Section 4 empirically compares our algorithm with two optimization solvers—Couenne (version 0.4) and CPLEX (version 12.4)—on three types of randomly generated instances. In particular, CPLEX solves a mixed-integer model of the CQPCC using the bounds derived in Sect. 2.2. We conclude that our algorithm is quite efficient and robust. Section 5 gives some final thoughts.

1.1 Related literature

As mentioned above, the IVQR problem is a type of inverse optimization problem [1,31] because ideally we would like the optimal solution $\hat{x}_2$ of (1) to be zero. Since the other variables x_3^+, x_3^- in (1) do not have desired values, IVQR is in fact a type of *partial inverse optimization problem*, which is similar to a regular inverse problem except that only certain parts of the desired optimal solution are specified. Research on partial inverse optimization problems has been active in recent decades; see [5, 12,18,27–30]. In particular, Heuberger [14] was one of the first to investigate partial inverse optimization problems. In many cases, partial inverse optimization is NP-hard. For example, solving partial inverse linear programming typically involves explicitly handling the complementarity conditions of the primal-dual optimality conditions.

In Sect. 2.1, we will show that IVQR can be formulated as a convex QP with complementarity constraints (CQPCC). Even *linear programs with complementarity constraints* (LPCCs) are known to be NP-hard since, for example, they can be used to formulate NP-hard nonconvex quadratic optimization problems [26]; see also [2,16,19,21,22]. It is well known that LPCCs can be formulated as mixed-integer programs when the nonnegative variables involved in the complementarity constraints are explicitly bounded.

In Sect. 4, we will employ a similar technique to reformulate IVQR as a convex quadratic mixed-integer program, which can be solved by CPLEX (version 12.4). Complementarity constraints can also be handled using general techniques for bilinear problems although we do not do so in this paper; see [11,25] for example.

Our IVQR problem can be solved by the recent algorithm of Bai et al. [3], which is a global algorithm for solving general CQPCCs. Their algorithm consists of two stages where the first stage solves a mixed-integer quadratic program with pre-set arbitrary upper bounds on the complementarity variables and the second relaxes the bounds with a logical Bender's decomposition approach. However, we have found by testing code supplied by the authors that this general-purpose algorithm was not competitive with the special-purpose algorithm that we will present in this paper.

Another related work is by Liu and Zhang [20] in which the authors present an algorithm to find global minimizers of general CQPCCs. The main idea of the algorithm is to use an embeded extreme point method to search for a higher-quality locally optimal solution starting with every feasible solution obtained by a branch-and-bound algorithm. The efficiency of the algorithm is based on the hope that the locally optimal solutions can provide better global upper bounds.

1.2 Notation

$\mathbb{R}^n$ refers to n-dimensional Euclidean space represented as column vectors, and $\mathbb{R}^{m\times n}$ is the set of real $m \times n$ matrices. The special vector $e \in \mathbb{R}^n$ consists of all ones. For $v \in \mathbb{R}^n$, both v_i and $[v]_i$ refer to the i-th component of v. For $v, w \in \mathbb{R}^n$, the Hadamard product of v and w is denoted by $v \circ w := (v_1 w_1, \ldots, v_n w_n)^{\mathrm{T}}$. For a matrix A, we denote by the row vector A^i the i-th row of A. For a scalar $p \geq 1$, the p-norm of $v \in \mathbb{R}^n$ is defined as $\|v\|_p := (\sum_{i=1}^n |v_i|^p)^{1/p}$. The ∞-norm is defined as $\|v\|_\infty := \max_{i=1}^n |v_i|$. For a given minimization problem, the notation Arg min denotes the set of optimal solutions. If the optimal solution set is known to be a singleton, i.e., there is a unique optimal solution, we write arg min instead. Our probability and statistics notation is standard. Throughout this paper, bold letters denote random variables. $P(\cdot)$ and $\mathrm{E}[\cdot]$ denote probability and expectation, respectively, and $P(\cdot|\cdot)$ and $\mathrm{E}[\cdot|\cdot]$ are the conditional variants. For an event E, we also write $\mathbb{1}\{E\}$ for the indicator function for E.

2 The IVQR problem and its details

In this section, we formulate the IVQR problem as a convex quadratic program with complementarity constraints (CQPCC) and state a natural CQP (convex quadratic program) relaxation that will serve as the root node in our B&B algorithm discussed in Sect. 3. We also derive asymptotic bounds on critical variables in the CQPCC that hold with high probability as the sample size increases. The bounds will in particular be required by one of the other global solvers in Sect. 4.

2.1 A CQPCC representation of the IVQR problem

Recall problem (1), which expresses our goal that $\hat{x}_2 = 0$ lead to minimal model error given the estimate $\hat{x}_1$ of x_1^*. Its dual is

$$\begin{array}{ll} \max_y & (b - A_1\hat{x}_1)^{\mathrm{T}} y \\ \text{s. t.} & A_2^{\mathrm{T}} y = 0 \\ & -e \leq y \leq e \end{array} \tag{2}$$

Note that $A_2^{\mathrm{T}} y = 0$ reflects that (1) optimizes only with respect to x_2, while $\hat{x}_1$ is considered fixed. The full optimality conditions for (1) and (2), including complementary slackness, are

$$
\begin{aligned}
& x_3^+ - x_3^- + A_2 x_2 = b - A_1 \hat{x}_1 \\
& x_3^+, x_3^- \geq 0 && (3) \\
& A_2^{\mathrm{T}} y = 0 && (4) \\
& -e \leq y \leq e && (5) \\
& x_3^+ \circ (e - y) = x_3^- \circ (y + e) = 0 && (6)
\end{aligned}
$$

Now consider the full IVQR problem in which x_1 is a variable. The optimality conditions just stated allow us to cast the IVQR problem as the task of finding a feasible value of x_1 satisfying the following system, where x_2, x_3^+, x_3^-, and y are also variables:

$$
\begin{aligned}
& x_3^+ - x_3^- + A_1 x_1 + A_2 x_2 = b && (7) \\
& (3)\text{–}(6) \\
& x_2 = 0.
\end{aligned}
$$

In comparison to the preceding optimality conditions, equation (7) highlights that x_1 is a variable, and the equation $x_2 = 0$ expresses our goal that zero is the optimal solution of (1). As mentioned in the Introduction, however, the constraint $x_2 = 0$ may be too stringent, and so we relax it to the weaker goal of finding a solution $(x_1, x_2, x_3^+, x_3^-, y)$ such that x_2 has minimum Euclidean norm:

$$
\begin{array}{ll}
\min_{x_1, x_2, x_3^+, x_3^-, y} & \|x_2\|_2^2 \\
\text{s. t.} & (3)\text{–}(7).
\end{array}
\qquad (8)
$$

This is our CQPCC formulation of the IVQR problem. In particular, the objective taken together with (3)–(5) and (7) form a CQP, and (6) enforces the complementarity constraints. For completeness, we prove that (8) is feasible.

Proposition 1 *The IVQR problem* (8) *is feasible.*

Proof Consider the primal problem (1) with $\hat{x}_1$ fixed. As x_3^+ and x_3^- are nonnegative, their difference $x_3^+ - x_3^-$ is equivalent to a vector of free variables. Hence, (1) is feasible. Furthermore, as x_3^+, and x_3^- are nonnegative, the objective function $e^{\mathrm{T}} x_3^+ + e\mathrm{T} x_3^-$ of (1) is bounded below, and hence (1) has an optimal solution. Then so does the dual (2) by strong duality. Those primal and dual optimal solutions, in addition, satisfy complementary slackness, exhibiting a feasible solution of (8). □

We present an alternative formulation (9) of (8), which will be the basis of the rest of the paper since it will prove convenient for the development in Sect. 3. We first add slack variables to (8) to convert all inequalities (except nonnegativity) into equations:

$$
\begin{array}{ll}
\min_{x_1, x_2, x_3^+, x_3^-, y, s^+, s^-} & \|x_2\|_2^2 \\
\text{s. t.} & x_3^+ - x_3^- + A_1 x_1 + A_2 x_2 = b, \quad x_3^+, x_3^- \geq 0 \\
& A_2^{\mathrm{T}} y = 0, \quad y + s^+ = e, \quad -e + s^- = y, \quad s^+, s^- \geq 0 \\
& x_3^+ \circ s^+ = x_3^- \circ s^- = 0
\end{array}
$$

where $s^+, s^- \in \mathbb{R}^m$. Then we eliminate y:

$$\begin{array}{ll} \min_{x_1,x_2,x_3^+,x_3^-,s^+,s^-} & \|x_2\|_2^2 \\ \text{s.t.} & x_3^+ - x_3^- + A_1x_1 + A_2x_2 = b, \;\; x_3^+, x_3^- \geq 0 \\ & A_2^{\mathrm{T}}(e - s^+) = 0, \;\; e - s^+ = -e + s^-, \;\; s^+, s^- \geq 0 \\ & x_3^+ \circ s^+ = x_3^- \circ s^- = 0. \end{array} \tag{9}$$

Due to the presence of the complementarity constraints, (9) is very likely difficult to solve since even linear programs with complementarity constraints (LPCCs) are NP-hard [21]. We will propose in Sect. 3, however, that (9) can be solved practically by a B&B algorithm, which employs polynomial-time CQP relaxations. For example, if we simply eliminate the complementarities, the resulting relaxation is tractable:

$$\begin{array}{ll} \min_{x_1,x_2,x_3^+,x_3^-,s^+,s^-} & \|x_2\|_2^2 \\ \text{s.t.} & x_3^+ - x_3^- + A_1x_1 + A_2x_2 = b, \;\; x_3^+, x_3^- \geq 0 \\ & A_2^{\mathrm{T}}(e - s^+) = 0, \;\; e - s^+ = -e + s^-, \;\; s^+, s^- \geq 0. \end{array} \tag{10}$$

This relaxation will indeed serve as the root relaxation in Sect. 3, and all node relaxations will be derived from it. It can be easily solved by numerous solvers. Note that (10) is bounded below by 0, and similarly, every node relaxation is bounded, too. Thus the B&B algorithm works even though we do not have upper bounds on the complementarity variables x_3^+ and x_3^-. However, one of the other algorithms, with which we will compare, requires *a priori* upper bounds on the variables x_3^+ and x_3^-. Therefore, we will propose asymptotic statistical bounds in the following section.

2.2 Variable bounds

The B&B algorithm that we will present in Sect. 3 can solve (9) directly, even though the feasible set is unbounded. However, as discussed above, one of the other algorithms, with which we will compare, requires *a priori* bounds on the variables x_3^+ and x_3^- of (9). So in this subsection, we derive bounds for these variables. The derived bounds are also of interest from the statistical point of view.

Since the difference $x_3^+ - x_3^-$ is closely related to the error $\boldsymbol{\epsilon}$ of the model, it suffices to bound $\boldsymbol{\epsilon}$. However, one can expect that $\boldsymbol{\epsilon}$ is unbounded in general, and so some additional assumptions are required to bound $\boldsymbol{\epsilon}$ with high probability as the sample size m grows larger. We will focus on two specific, representative examples—one in which $\boldsymbol{\epsilon}$ has light tails and one in which the tails of $\boldsymbol{\epsilon}$ are heavy—and we prove explicit bounds on $\boldsymbol{\epsilon}$ that hold with high probability for large m. These bounds will subsequently be incorporated into (9) and used in Sect. 4 by one of the solvers.

Suppose that data (b, A) with $\epsilon := b - Ax^*$ is a random sample following the quantile-regression model

$$\boldsymbol{b} = \boldsymbol{a}^{\mathrm{T}}x^* + \boldsymbol{\epsilon} \qquad \text{with} \qquad P(\boldsymbol{\epsilon} \leq 0 \mid \boldsymbol{a} = a) = \tfrac{1}{2}.$$

This is exactly the model considered in this paper except that the subscript 1 appearing on $\boldsymbol{a}$, a, and A has been dropped for notational convenience. We start by stating two lemmas that will facilitate the details of Example 1 (light tails) and Example 2 (heavy tails) below. The proofs are included in the Appendix.

Lemma 1 *For a random sample* $(b, A) \in \mathbb{R}^{m\times(1+n)}$ *with* $\epsilon := b - Ax^*$ *and a given constant* $C > 1$*, the probability* $P\,(\|\epsilon\|_\infty > C\|b\|_\infty)$ *is bounded above by both*

$$P\left(\frac{C}{C+1} < \frac{\|\epsilon\|_\infty}{\|Ax^*\|_\infty} < \frac{C}{C-1}\right)$$

and

$$m\,\max_{i=1}^{m} P\left(|\epsilon_i| > C\max_{k=1}^{m}\left\{|\epsilon_k|\cdot\mathbb{1}\{\epsilon_k a_k^T x^* \geq 0\}\right\}\right).$$

where $\mathbb{1}$ *is the indicator function.*

Lemma 2 *For any normal random variable* $Z \sim \mathcal{N}(0, \sigma^2)$ *and any* $\theta \geq 1$*, it holds that*

$$\frac{1}{2\theta}\cdot\epsilon(\theta) \leq P(Z > \theta\sigma) \leq \frac{1}{\theta}\cdot\epsilon(\theta), \qquad \textit{where } \epsilon(\theta) := \frac{1}{\sqrt{2\pi}}\exp(-\theta^2/2).$$

In addition, consider q *identically distributed copies* $Z_1, \ldots, Z_q$ *of* Z*, where* q *is large enough so that* $\log(q) \geq 1$ *and* $q/(8\pi\log(q)) \geq \sqrt{q}$*. If* $\theta = \sqrt{\log(q)}$*, then*

$$P\left(\max_{1\leq p\leq q} Z_p \leq \theta\sigma\right) \leq \exp(-q^{1/4}).$$

We are now ready to give the light- and heavy-tailed examples that suggest reasonable asymptotic bounds on the error. In particular, both Examples 1 and 2 show that the bound $C\,\|b\|_\infty$ is asymptotically valid for ϵ in theory when $C > 1$. However, in practice, $C = 5$ will be appropriate (with high probability for large m) for many situations of interest. It can work effectively even for small m; see details in Sect. 4.3. So we can enforce $x_3^+ \leq 5\,\|b\|_\infty e$ and $x_3^- \leq 5\,\|b\|_\infty e$ in the formulation (9) of IVQR. For ease of discussion in Examples 1 and 2, we set $C = 5$. Again, the same analysis can be applied to any value $C > 1$.

Example 1 (Light tails) For the case $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2)$, let $(b, A) \in \mathbb{R}^{m\times(1+n)}$ be a random sample with $\epsilon := b - Ax^*$. Then the inequality $\|\epsilon\| \leq 5\|b\|_\infty$ holds almost surely as $m \to \infty$.

To explain Example 1, set $C = 5$. Lemma 1 implies $P(\|\epsilon\|_\infty > 5\|b\|_\infty) \leq m\max_{i=1}^{m} p_i$, where

$$p_i := P\left(|\epsilon_i| > 5\max_{k}\left\{|\epsilon_k|\cdot\mathbb{1}\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\}\right\}\right).$$

We claim that each product $m \cdot p_i \to 0$ as $m \to \infty$. In particular, we will show that, independently of i,

$$p_i \leq \exp\left(-\tfrac{m}{72}\right) + 2\left(\tfrac{3}{m}\right)^2 + \exp\left(-\left(\tfrac{m}{3}\right)^{1/4}\right) \tag{11}$$

so that

$$\begin{aligned} P(\|\epsilon\|_\infty > 5\|b\|_\infty) &\leq m\left(\exp\left(-\tfrac{m}{72}\right) + 2\left(\tfrac{3}{m}\right)^2 + \exp\left(-\left(\tfrac{m}{3}\right)^{1/4}\right)\right) \\ &= \exp\left(\log(m) - \tfrac{m}{72}\right) + \tfrac{18}{m} + \exp\left(\log(m) - \left(\tfrac{m}{3}\right)^{1/4}\right) \\ &\to 0. \end{aligned}$$

This shows that one can asymptotically expect the error to be at most $5\|b\|_\infty$.

To prove the inequality (11), fix the index i; say $i = m$ without loss of generality. If more than $q := \lfloor \frac{m}{3} \rfloor$ of the terms $\epsilon_k a_k^{\mathrm{T}} x^*$ are nonnegative (including the first q terms without loss of generality) and $|\epsilon_m| \leq 5 \max_{1 \leq k \leq q} |\epsilon_k|$, then

$$\begin{aligned} |\epsilon_m| &\leq 5 \max_{k=1}^{q} |\epsilon_k| \\ &= 5 \max_{k=1}^{q} \left\{ |\epsilon_k| \cdot \mathbb{1}\left\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\right\}\right\} \\ &\leq 5 \max_{k=1}^{m} \left\{ |\epsilon_k| \cdot \mathbb{1}\left\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\right\}\right\}. \end{aligned}$$

Logically, this ensures the contrapositive implication

$$\begin{aligned} &|\epsilon_m| > 5 \max_{k=1}^{m} \left\{ |\epsilon_k| \cdot \mathbb{1}\left\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\right\}\right\} \\ &\qquad \Longrightarrow \sum_{k=1}^{m} \mathbb{1}\left\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\right\} \leq q \quad \text{or} \quad |\epsilon_m| > 5 \max_{k=1}^{q} |\epsilon_k|. \end{aligned}$$

So $p_m \leq \alpha + \beta$, where

$$\begin{aligned} \alpha &:= P\left(\sum_{k=1}^{m} \mathbb{1}\left\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\right\} \leq q\right) \\ \beta &:= P\left(|\epsilon_m| > 5 \max_{k=1}^{q} |\epsilon_k|\right). \end{aligned}$$

We next bound α and β separately.

Because each $\epsilon_k \sim \mathcal{N}(0, \sigma^2)$ conditional on a_k with $P(\epsilon_k \geq 0 \mid \boldsymbol{a} = a_k) = P(\epsilon_k \leq 0 \mid \boldsymbol{a} = a_k) = \frac{1}{2}$, we have $P(\epsilon_k a_k^{\mathrm{T}} x^* \geq 0 \mid \boldsymbol{a} = a_k) = \frac{1}{2}$. Then, interpreting both ϵ_k and a_k as random, this means $P(\epsilon_k a_k^{\mathrm{T}} x^* \geq 0) = \frac{1}{2}$. Therefore each $Y_k := \mathbb{1}\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\} - \frac{1}{2}$ is a bounded random variable with mean 0. By Azuma's inequality, we have

$$\alpha = P\left(\sum_{k=1}^{m} Y_k \le q - \frac{m}{2}\right) \le P\left(\sum_{k=1}^{m} Y_k \le -\frac{m}{6}\right) \le \exp\left(-\tfrac{m}{72}\right).$$

Finally, to bound β, set $t = 2\sqrt{\log(q)}$. Logically,

$$|\epsilon_m| > 5\max_{k=1}^{q} |\epsilon_k| \quad \Longrightarrow \quad |\epsilon_m| > t\sigma \quad \text{or} \quad 5\max_{k=1}^{q} |\epsilon_k| \le t\sigma.$$

So $\beta \le \gamma + \delta$, where

$$\gamma := P\left(|\epsilon_1| > t\sigma\right)$$
$$\delta := P\left(5\max_{k=1}^{q} |\epsilon_k| \le t\sigma\right).$$

To bound γ, we use Lemma 2 with $\theta = t$ to show that, for m large enough,

$$\begin{aligned}\gamma &= P(|\epsilon_1| > t\sigma) = 2P(\epsilon_1 > t\sigma)\\ &\le \frac{2}{t}\cdot\frac{1}{\sqrt{2\pi}}\exp(-t^2/2) = \frac{1}{\sqrt{\log(q)}}\cdot\frac{1}{\sqrt{2\pi}}\exp(-2\log(q))\\ &\le q^{-2} \le 2\left(\tfrac{3}{m}\right)^2.\end{aligned}$$

To bound δ, we apply Lemma 2 with $\theta = \frac{1}{5}\sqrt{\log(q)}$ to conclude $\delta \le \exp(-q^{1/4}) \le \exp(-(\frac{m}{3})^{1/4})$.

In total, we have $p_i \le \alpha + \beta \le \alpha + \gamma + \delta \le \exp(-\frac{m}{72}) + 2(\frac{3}{m})^2 + \exp(-(\frac{m}{3})^{1/4})$, which is (11).

Example 2 (heavy tails) Consider the case when $\mathrm{E}[|\boldsymbol{a}^{\mathrm{T}}x^*|^q] \le K$ for some integer $q > 0$ and scalar $K > 0$ and $\boldsymbol{\epsilon}$ satisfies $P(|\boldsymbol{\epsilon}| < t) \le 1 - t^{-k}$, where $k + 1 < q$. Let $(b, A) \in \mathbb{R}^{m\times(1+n)}$ be a random sample with $\epsilon := b - Ax^*$. Then the inequality $\|\epsilon\| \le 5\|b\|_\infty$ holds almost surely as $m \to \infty$.

From Hölder's inequality, we see

$$\begin{aligned}\mathrm{E}\left[\|Ax^*\|_\infty\right] &= \mathrm{E}\left[\{(\|Ax^*\|_\infty)^q\}^{1/q}\right]\\ &\le \left(\mathrm{E}\left[(\|Ax^*\|_\infty)^q\right]\right)^{1/q}\\ &\le \left(\mathrm{E}\left[\sum_{i=1}^{m}|a_i^{\mathrm{T}}x^*|^q\right]\right)^{1/q}\\ &= \left(\sum_{i=1}^{m}\mathrm{E}\left[|a_i^{\mathrm{T}}x^*|^q\right]\right)^{1/q}\\ &\le m^{1/q}K^{1/q}\end{aligned}$$

Hence, by Markov's inequality,

$$P\left(\|Ax^*\|_\infty > m^{1/(k+1)}K^{1/q}\right) \le \frac{E[\|Ax^*\|_\infty]}{m^{1/(k+1)}K^{1/q}} \le \frac{m^{1/q}K^{1/q}}{m^{1/(k+1)}K^{1/q}} = \frac{m^{1/q}}{m^{1/(k+1)}},$$

which goes to 0 as $m \to \infty$ because $k+1 < q$. Moreover,

$$P(\|\epsilon\|_\infty < t) = \Pi_{i=1}^m P(|\epsilon_i| < t) \le (1 - t^{-k})^m,$$

which, substituting $t = Cm^{1/(k+1)}K^{1/q}$, implies

$$\begin{aligned} P\left(\|\epsilon\|_\infty < Cm^{1/(k+1)}K^{1/q}\right) &\le \left(1 - (Cm^{1/(k+1)}K^{1/q})^{-k}\right)^m \\ &= \left(1 - C^{-k}m^{-k/(k+1)}K^{-k/q}\right)^m. \end{aligned}$$

Note that the last quantity goes to 0 as $m \to \infty$ because $k/(k+1) < 0$. By Lemma 1 and taking $C = 5$,

$$\begin{aligned} P(\|\epsilon\|_\infty > 5\|b\|_\infty) &\le P\left(\tfrac{5}{5+1} \le \tfrac{\|\epsilon\|_\infty}{\|Ax\|_\infty} \le \tfrac{5}{5-1}\right) \\ &\le P\left(\tfrac{\|\epsilon\|_\infty}{\|Ax\|_\infty} \le 5\right) \\ &= P\left(\|\epsilon\|_\infty \le 5\|Ax\|_\infty\right) \\ &\le P(\|Ax^*\|_\infty > m^{1/(k+1)}K^{1/q}) + P(\|\epsilon\|_\infty \le 5m^{1/(k+1)}K^{1/q}) \\ &\to 0 + 0 = 0. \end{aligned}$$

An important remark is in order here. For any particular instance we will test in Sect. 4, imposing the bound $5\|b\|_\infty$ might cut off some or all optimal solutions in practice. This is because the bound on the error $\boldsymbol{\epsilon}$ is asymptotic as $m \to \infty$. In practice, we have found, however, that $5\|b\|_\infty$ does not often cut off any optimal solutions. We will discuss this in a bit more detail in Sect. 4.3.

3 A CQP-based branch and bound algorithm

In Sect. 2.1, we mentioned that the CQP relaxation (10)—and ones derived from it—would be used within a branch-and-bound (B&B) algorithm to solve IVQR via the reformulation (9). In this section we present the algorithm in detail and discuss important implementation issues. For the moment, we do not refer to the bounds derived in Sect. 2.2 since our own algorithm does not require them; we only need the bounds for CPLEX in Sect. 4.3.

3.1 The scheme and structure of the algorithm

Our B&B algorithm aims to enforce more and more of the complementarity constraints in (9) further and further down in a dynamically constructed tree. Complementarities are enforced using added linear equalities. For example, a single complementarity $[x_3^+]_i[s^+]_i = 0$ is enforced in one branch by $[x_3^+]_i = 0$ and in a second branch by $[s^+]_i = 0$. This is analogous to branching on a 0–1 binary variable z in integer programming, where one branch forces $z = 0$ and another $z = 1$.

To describe the B&B algorithm formally, for each node of the tree, let F_x^+ and F_s^+ be two disjoint subsets of the index set $\{1, \dots, m\}$, and separately let F_x^- and F_s^- be two disjoint subsets of the same. Also define $G^+ := \{1, 2, \dots, m\}\backslash(F_x^+ \cup F_s^+)$ and $G^- := \{1, 2, \dots, m\}\backslash(F_x^- \cup F_s^-)$. The node will enforce complementarities associated with $F_x^+ \cup F_s^+$ and $F_x^- \cup F_s^-$ by solving the following CQP relaxation:

$$\begin{array}{lll} \min_{x_1,x_2,x_3^+,x_3^-,s^+,s^-} & \|x_2\|_2^2 & \\ \text{s.t.} & x_3^+ - x_3^- + A_1x_1 + A_2x_2 = b, \quad x_3^+, x_3^- \geq 0 & \\ & A_2^{\mathrm{T}}(e - s^+) = 0, \quad e - s^+ = -e + s^-, \quad s^+, s^- \geq 0 & (12) \\ & [x_3^+]_i = 0 \ \ \forall\, i \in F_x^+, \quad [s^+]_i = 0 \ \ \forall\, i \in F_s^+ & \\ & [x_3^-]_j = 0 \ \ \forall\, j \in F_x^-, \quad [s^-]_j = 0 \ \ \forall\, j \in F_s^-. & \end{array}$$

This problem is the basic (or root) relaxation (10) with added linear inequalities that enforce the complementarities $[x_3^+]_i[s^+]_i = 0$ for all $i \in F_x^+ \cup F_s^+$ and $[x_3^-]_j[s^-]_j = 0$ for all $j \in F_x^- \cup F_s^-$. On the other hand, any complementarities corresponding to G^+ or G^- are relaxed compared to (9). Note that, while the feasible set of problem (12) is unbounded in the variables x_3^+ and x_3^-, the objective function is bounded below since it is nonnegative. Hence, (12) is always solvable or infeasible.

Now we discuss how to create new nodes in the tree, i.e., how to branch on a given node that is associated with sets $F_x^+, F_s^+, F_x^-, F_s^-, G^+, G^-$. First, select some $i \in G^+$ or $j \in G^-$ corresponding to a complementarity that has yet to be enforced. Then two child nodes are created as follows: one with $F_x^+ \leftarrow F_x^+ \cup \{i\}$ and one with $F_s^+ \leftarrow F_s^+ \cup \{i\}$ if an i was selected, or one with $F_x^- \leftarrow F_x^- \cup \{j\}$ and one with $F_s^- \leftarrow F_s^- \cup \{j\}$ if a j was selected. G^+ and G^- are also updated for the children as: $G^+ \leftarrow G^+\backslash\{i\}$ if i was selected, or $G^- \leftarrow G^-\backslash\{j\}$ if j was selected. This form of branching leads to two special cases worth pointing out: (i) the root node corresponds to $F_x^+ = F_s^+ = F_x^- = F_s^- = \emptyset$ with no complementarities enforced; (ii) leaf nodes correspond to $G^+ = G^- = \emptyset$ with all complementarities enforced.

The next important concept of the B&B algorithm is fathoming, that is, removing a node from further consideration and, in particular, not branching on it. We may fathom a node if we are certain that the relaxation associated with that node does not contain any solutions for (9) that are better than what we have already encountered during the course of the algorithm. Let GUB ("global upper bound") denote the best feasible value of (9) encountered so far, including possibly gotten at the current node, e.g., when the optimal solution to the relaxation (12) is actually feasible for (9). Then we can fathom if the optimal value of (12) is greater than or equal to GUB. This fathoming rule includes a special case when (12) is infeasible, in which case the relaxation optimal value can be considered $+\infty$.

Another equally important concept for our B&B algorithm is how we evaluate the nodes in the tree. Evaluating a node involves solving the CQP relaxation (12) corresponding to the node and then fathoming the node (if possible) or branching on the node (if fathoming is not possible and if it is not a leaf node).

With these basic concepts introduced, we now have a whole picture of our B&B algorithm. Starting with the root node, the algorithm evaluates, adds, and fathoms nodes from the tree at each iteration. The algorithm finishes when all nodes generated by the algorithm have been evaluated and fathomed. The final GUB value is the optimal value, and the solution associated with GUB is an optimal solution.

3.2 Implementation issues

In this subsection, we will describe some important implementation issues for our B&B algorithm.

One of the most crucial issues is the method chosen to solve the CQP relaxation (12) at each node of the tree. As mentioned in Sect. 2.1, we use CPLEX 12.4 in our codes. In particular, we solve the relaxations using the dual simplex CQP pivoting method based on the following two considerations. First, preliminary results indicated that the dual pivoting method was more numerically stable and accurate compared to other methods, including the primal simplex CQP pivoting method and the interior-point (barrier) method. Secondly, the dual pivoting method is particularly useful for re-optimizing a problem when primal constraints are added, which is the case between parent and child nodes in our algorithm.

Another important issue is how to choose the complementarity on which to branch when child nodes are created. (Please refer to the conceptual discussion in the prior subsection.) We apply a maximum-violation approach, which is similar to most-fractional branching in integer programming. Suppose that we have just solved the CQP relaxation (12) at a node and have an associated optimal solution $(\hat{x}_1, \hat{x}_2, \hat{x}_3^+, \hat{x}_3^-, \hat{s}^+, \hat{s}^-)$. We then compute

$$i \in \operatorname{Arg}\max_k \left\{ [\hat{x}_3^+]_k \, [\hat{s}^+]_k \right\} \quad \text{and} \quad j \in \operatorname{Arg}\max_l \left\{ [\hat{x}_3^-]_l \, [\hat{s}^-]_l \right\}$$

and choose i if $[\hat{x}_3^+]_i [\hat{s}^+]_i$ is larger than $[\hat{x}_3^-]_j [\hat{s}^-]_j$ and otherwise choose j.

We also give a bit more detail about handling the optimal solution $(\hat{x}_1, \hat{x}_2, \hat{x}_3^+, \hat{x}_3^-, \hat{s}^+, \hat{s}^-)$ of the relaxation (12) at each node. (In our experience, CPLEX is quite stable and can always deliver an optimal solution or determine that (12) is infeasible.) First, the solution is checked for feasibility in the original problem (9) up to a relative tolerance of 10^{-6}. If so, we update GUB, and the node is fathomed. On the other hand, if the solution is infeasible at this tolerance, then it means that some complementarity is violated, and we must branch.

In addition, we use a relative optimality tolerance for fathoming a node by its relaxed objective value. Given $\varepsilon > 0$, the optimal value LB ("lower bound") of the relaxation, and the current GUB, the node is fathomed if $(\text{GUB} - \text{LB})/\max\{1, \text{GUB}\} < \varepsilon$. Note that GUB is nonnegative in our setting, and we take $\epsilon = 10^{-6}$ in our implementation.

We adopt a heuristic method to generate an initial GUB for our B&B algorithm. From Proposition 1, we know that (1) is feasible and bounded below by 0 and thus has optimal solutions for each fixed x_1. Hence, we first solve the unconstrained linear least squares problem:

$$\hat{x}_1 = \arg\min_{x_1} \|A_1 x_1 - b\|_2.$$

Then, we solve (1) with $\hat{x}_1$ to get an optimal $\hat{x}_2$. We note that $\hat{x}_1$ and $\hat{x}_2$ are part of a feasible solution of the IVQR problem. Then our initial GUB is set to $\|\hat{x}_2\|_2^2$.

We also provide warm-start initial solutions for each child node on the branch-and-bound tree. We store the relaxed optimal solutions of the parent nodes that serve as the warm-start initial solutions of their child nodes. Although the storing takes additional memory, the algorithm benefits from decreasing the computation time for each nodes with the providing initial solutions. Our preliminary testing results indicate that it is worth considering the compromise in our algorithm.

We can also add implied complementarity constraints to the nodes in the B&B tree. Consider a node in which $[s^+]_i = 0$ is enforced. Since the linear constraints of (12) imply $[s^+]_i + [s^-]_i = 2$, we have $[s^-]_i = 2$, which in turn implies $[x_3^-]_i = 0$. Similarly, when a node enforces $[s^-]_j = 0$, we see $[x_3^+]_j = 0$. Thus, the CQP in (12) can be strengthened as follows:

$$\begin{array}{ll} \min_{x_1,x_2,x_3^+,x_3^-,s^+,s^-} & \|x_2\|_2^2 \\ \text{s.t.} & x_3^+ - x_3^- + A_1x_1 + A_2x_2 = b, \quad x_3^+, x_3^- \geq 0 \\ & A_2^T(e - s^+) = 0, \quad e - s^+ = -e + s^-, \quad s^+, s^- \geq 0 \\ & [x_3^+]_i = 0 \ \forall\, i \in F_x^+, \quad [s^+]_i = [x_3^-]_i = 0 \ \forall\, i \in F_s^+ \\ & [x_3^-]_j = 0 \ \forall\, j \in F_x^-, \quad [s^-]_j = [x_3^+]_j = 0 \ \forall\, j \in F_s^-. \end{array}$$

3.2.1 Node selection strategies

Next, let us introduce two alternative strategies to select the next node to evaluate during the course of our B&B algorithm. The two strategies are called the *best-bound search* and the *bi-priority search*. The best-bound search, which is a well known strategy, will be employed for testing small-size instances in Sect. 4, while the bi-priority search, a strategy that we developed for the IVQR problem, will be employed for testing medium-size and large-size instances.

We discuss best-bound first. Before selecting the next node to evaluate, we sort the remaining nodes in the tree by their LBs (lower bounds), which are just the optimal values of the relaxations of their parent nodes. Then the next node is chosen to be the one with the lowest LB. The best-bound search tends to reduce the number of nodes evaluated during the overall course of the algorithm and can improve the LBs in the tree quickly. The downside, however, is that—at any given time—the size of the tree can be very large if the current GUB is far from optimal, e.g., see [13]. Then memory usage can be a concern, and tasks such as sorting the tree can take a long time. We experienced precisely this behavior in early implementations of our algorithm, which degraded the speed and performance of our implementation significantly.

To address the downsides of the best-bound strategy, we developed what we call bi-priority search, and it is intended for larger problems for which the tree can become quite big. We describe the procedure as follows. At any time during our B&B algorithm, we maintain the nodes of the tree in two separate lists: a high-priority list and a low-priority one. For a pre-specified, fixed value $\hat{v}$, if a node has LB $\leq \hat{v}$, then it is a member of the high-priority list; otherwise, it goes in the low-priority list. Then, as long as the high-priority list is nonempty, we employ depth-first search to choose the most recent node appended to the high-priority list. On the other hand, if the higher-priority list is empty, we use best-bound search to make our next choice from the low-priority list.

The point of bi-priority search is to focus the algorithm's attention using depth-first search in parts of the tree where a good GUB could possibly be discovered. These are the nodes with LB $\leq \hat{v}$ in the high-priority list. If a good GUB is indeed found, then it is likely that many of the nodes in the low-priority list will be fathomed. Clearly this strategy depends on an intelligent choice of $\hat{v}$. For the IVQR problem (9), the optimal values are typically very close to 0. Knowing that this is a feature of the class of IVQR problems in general, we choose $\hat{v} = 0$ specifically to help solve any given instance.

Note that our bi-priority search is different from a common two-phase method in which a B&B algorithm first uses depth-first search to find a feasible solution quickly and then uses best-bound search afterwards to improve the LB [13]; see more sophisticated two-phase methods in [9,10]. It is also different from the hybrid method [13] in which a B&B algorithm employs depth-first search from the current node until a node with LB $> \gamma$, where γ is a pre-specified threshold, and then uses a different method, e.g., best-bound search, to select the next node, and then keeps repeating the same two steps.

4 Computational experiments

In this section, we first discuss the procedure to generate test instances and describe three types of instances. Then, for the computational study, we test our B&B algorithm, referred to as *QPBB*, against two well known solvers on the three types of instances. We implemented our algorithm in MATLAB version 8.5.0.197613 (R2015a) and employed CPLEX 12.4 to solve the subproblems which are convex QPs. The CPLEX solver was called through its MATLAB interface. All computational experiments were performed on an Intel Core i7-3770 CPU running at 3.40 GHz with 8 threads (over 4 cores) under the Linux operating system. We note that—when the compared algorithms report optimal solutions on an instance—we have found experimentally that the optimal values agree up to a relative accuracy of 10^{-6}. In other words, when an algorithm claims to have solved an instance optimally, it is independently verified by the other algorithms up to six significant figures.

4.1 Generation and description of instances

Test instances of the IVQR problem are randomly generated by specifying the following parameters: sample size (m), number of endogenous covariates (n_1), and number of instruments (n_2). As discussed in Sect. 1, we have the relation $m > n_2 \geq n_1$. Thus, we first consider the case that $m > n_2 = n_1$. Let $\mathcal{U}(0, 1)$ and $\mathcal{N}(0, 1)$ denote

Table 1 Details of the three types of test instances

Instance type	# Instances	(m, n_1, n_2)	# Complementarities
Small-IVQR	500	(50, 5, 5)	100
Medium-IVQR	300	(100, 5, 5)	200
Large-IVQR	100	(150, 5, 5)	300

the standard uniform and normal distributions, respectively. Then our steps to generate random data $(b, A_1, A_2) \in \mathbb{R}^m \times \mathbb{R}^{m\times n_1} \times \mathbb{R}^{m\times n_2}$ for a single instance are shown in Algorithm 1. This procedure guarantees $P(\boldsymbol{\epsilon} \le 0 \mid \boldsymbol{a_1} = a_1) \neq \frac{1}{2}$ but $P(\boldsymbol{\epsilon} \le 0 \mid \boldsymbol{a_1} = a_1, \boldsymbol{a_2} = a_2) = \frac{1}{2}$. We refer the reader to [7] for details of the procedure and relevant theory.

Algorithm 1 Random Instance Generator

```
Inputs: $m, n_1, n_2, \alpha := (1, 2, \dots, n_1)^T$, and $\beta := 2\alpha$
Outputs: $(b, A_1, A_2)$
for $i = 1, \dots, m$ do
    Generate $u$ from $\mathcal{U}(0, 1)$
    For $j = 1, \dots, n_2$, sample $[A_2]_{ij}$ independently from $\mathcal{N}(0, 1)^2$ ("squared normal")
    For $j = 1, \dots, n_1$, calculate $[A_1]_{ij} = [A_2]_{ij} + u\beta_j$
    Calculate $b_i = \sum_{j=1}^{n_1} (1 - u + u\alpha_j)[A_1]_{ij}$
end for
```

We generate three types of instances using the above procedure. The first type, called *Small-IVQR*, contains 500 random instances with $(m, n_1, n_2) = (50, 5, 5)$. The second type, called *Medium-IVQR*, contains 300 random instances with $(m, n_1, n_2) = (100, 5, 5)$. The *Large-IVQR* type contains 100 random instances with $(m, n_1, n_2) = (150, 5, 5)$. We expect the Large-IVQR instances to be the most challenging since $2m$ equals the number of complementarities in (9), which directly affects the (potential) number of nodes in the B&B tree. Table 1 describes the three types of instances. We also considered the case of $m > n_2 > n_1$, but the computational results were the same as those presented with $n_1 = n_2$. So here we focus on the case $m > n_1 = n_2$.

4.2 Tests against couenne

We first test our algorithm against the open-source global solver Couenne (version 0.4) [8] on the Small-IVQR, Medium-IVQR, and Large-IVQR instances based on the formulation in (9). For our algorithm, we set the feasibility tolerance to 10^{-6} and the fathoming tolerance to 10^{-6}. Couenne incorporates a generic tolerance, which affects a number of aspects of its performance; we choose 10^{-6}. The per-instance time limit is set to 900 s (15 min) for both algorithms on the Small-IVQR instances, 1800 s (30 min) on the Medium-IVQR instances, and 7200 s (2 h) on the Large-IVQR instances. If the time limit is reached for either method on a particular instance, the

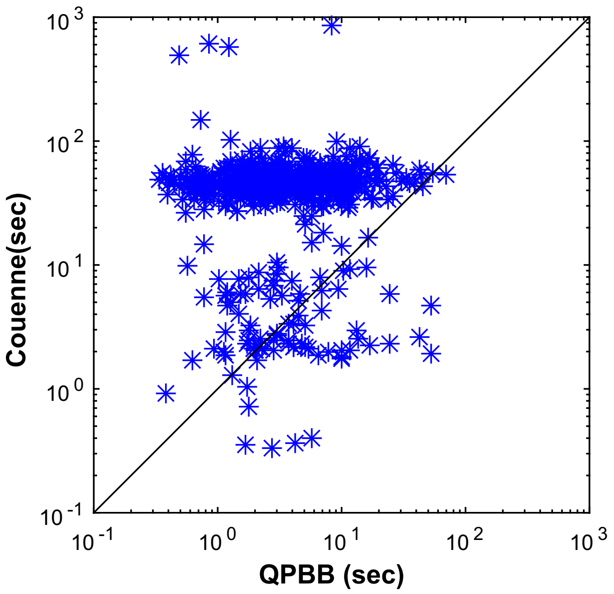

Fig. 1 Comparison of the computation times (in seconds) between our B&B algorithm and Couenne on the 500 Small-IVQR instances. The x- and y-axes are log scales, and the *diagonal line* defines $y = x$. An *asterisk* represents an instance for which both algorithms report optimality

method will return its current best objective value (GUB), but of course it is possible that the GUB returned is not the global optimal value. We employ the best-bound search for our algorithm on the Small-IVQR instances and the bi-priority search on the Medium-IVQR and Large-IVQR instances. We report our main observations as follows.

The comparison results of both algorithms on the 500 Small-IVQR instances are shown in Fig. 1. Specifically, our algorithm outperforms Couenne on 454 instances out of 500 in terms of computation time. Our algorithm performs worse than Couenne only on the remaining 46 instances. Note that we present log–log plots of the CPU times, and the straight line defines $y = x$ in Fig. 1.

Figure 2 shows the computational comparison of the two algorithms on the 300 Medium-IVQR instances. In particular, our algorithm outperforms Couenne on 283 instances out of 300 in terms of computational time. Our algorithm performs worse than Couenne on the remaining 17 instances. Furthermore, our algorithm reports optimality on all the 300 instances while Couenne reports optimality on 293 instances and exceeds the time limit on the remaining 7.

Figure 3 indicates the computational results of the comparison of the two algorithms on the 100 Large-IVQR instances. Our algorithm outperforms Couenne on 99 instances out of 100 in terms of computational time whereas it performs worse than Couenne on the remaining 1 instance. Furthermore, our algorithm reports optimality on all 100 instances while Couenne reports optimality on 94 instances and exceeds the time limit of 7200 seconds on the remaining 6.

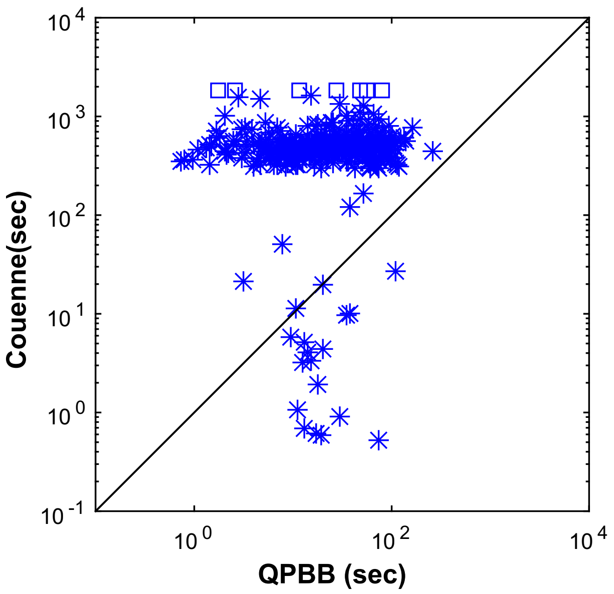

Fig. 2 Comparison of the computation times (in seconds) between our B&B algorithm and Couenne on the 300 Medium-IVQR instances. The x- and y-axes are log scales, and the *diagonal line* defines $y = x$. An *asterisk* represents an instance for which both algorithms report optimality; a *square* represents that Couenne exceeded the time limit of 1800 s

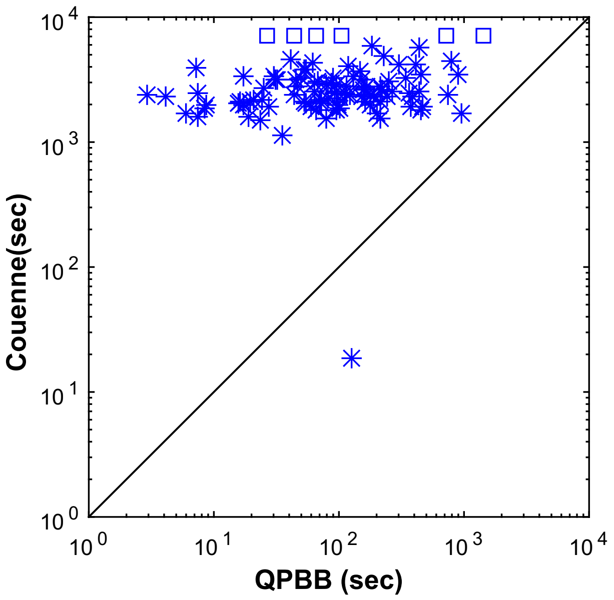

Fig. 3 Comparison of the computation times (in seconds) between our B&B algorithm and Couenne on the 100 Large-IVQR instances. The x- and y-axes are log scales, and the *diagonal line* defines $y = x$. An *asterisk* represents an instance for which both algorithms report optimality; a *square* represents that Couenne exceeded the time limit of 7200 s

4.3 Tests against CPLEX

In this subsection, we compare our algorithm against CPLEX's mixed-integer QP solver (version 12.4) applied to a mixed-integer programming (MIP) formulation of (9). For the MIP formulation, we use a standard technique for handling complementarity constraints by introducing binary variables together with a "big-M" approach. The big-M constants require *a priori* bounds on x_3^+ and x_3^-, and so we employ the bounds derived in Sect. 2.2. The MIP formulation is

$$\begin{array}{lll} \min_{x_1,x_2,x_3^+,x_3^-,s^+,s^-,z^+,z^-} & \|x_2\|_2^2 \\ \text{s. t.} & x_3^+ - x_3^- + A_1x_1 + A_2x_2 = b, \;\; x_3^+, x_3^- \geq 0 \\ & A_2^{\mathrm{T}}(e - s^+) = 0, \;\; e - s^+ = -e + s^-, \;\; s^+, s^- \geq 0 \\ & x_3^+ \leq 5\,\|b\|_\infty\, z^+, \;\; s^+ \leq 2(e - z^+) \\ & x_3^- \leq 5\,\|b\|_\infty\, z^-, \;\; s^- \leq 2(e - z^-) \\ & z^+, z^- \in \{0, 1\}^m, \end{array} \tag{13}$$

where $5\|b\|_\infty$ is the upper bound on both x_3^+ and x_3^- from Sect. 2.2.

We compare our algorithm with CPLEX on the Small-IVQR, the Medium-IVQR, and the Large-IVQR instances. For the comparison, the per-instance time limit is set to 900 s (15 min) for both algorithms on the Small-IVQR instances, 1800 s (30 min) on the Medium-IVQR instances, and 7200 s (2 h) on the Large-IVQR instances. If the time limit is reached for either algorithm on any instance, the algorithm will return its current best GUB. We use the same tolerances as those in Sect. 4.2 for our algorithm, and we use the default tolerances for CPLEX. For a fair comparison, we also introduce the upper bounds $x_3^+ \leq 5\,\|b\|_\infty\, e$ and $x_3^- \leq 5\,\|b\|_\infty\, e$ into formulation (9) and its relaxations within our algorithm. However, note that our algorithm does not technically require the derived upper bounds. Similar to the comparison with Couenne, we employ the best-bound search for our algorithm on the Small-IVQR instances and the bi-priority search on the Medium-IVQR and Large-IVQR instances.

Figure 4 shows the computational comparison of the two algorithms on the 500 Small-IVQR instances. Our algorithm outperforms CPLEX on 343 instances whereas it performs worse than CPLEX on the remaining 157. In addition, both algorithms report optimality for all 500 instances.

Figure 5 shows that the computational results of both the B&B algorithm and CPLEX on 300 Medium-IVQR instances. Specifically, our B&B algorithm reports optimality on all 300 instances. However, CPLEX returns optimal values only for 159 instances out of 300, exceeds the time limit on 138 instances, and has numerical issues on the remaining 3 instances. In addition, our algorithm outperforms CPLEX on 278 instances out of the 297 instances where both algorithms did not have numerical issues. Our algorithm only performs worse on the remaining 19 instances.

Figure 6 shows the comparison results of both B&B algorithm and CPLEX on the 100 Large-IVQR instances. Specifically, our B&B algorithm reports optimality on all 100 instances. However, CPLEX reports optimality only on 17 instances, exceeds

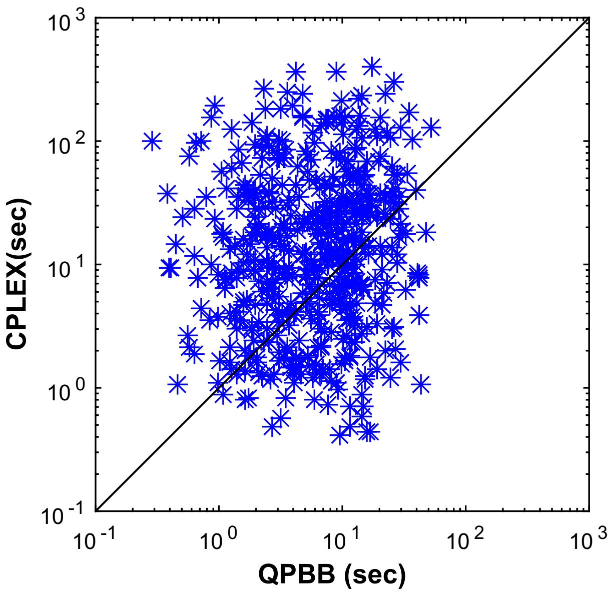

Fig. 4 Comparison of the computation time (in seconds) between our B&B algorithm and CPLEX on 500 Small-IVQR instances. The x- and y-axes are log scales, and the *diagonal line* defines $y = x$. An *asterisk* represents an instance for which both algorithms report optimality

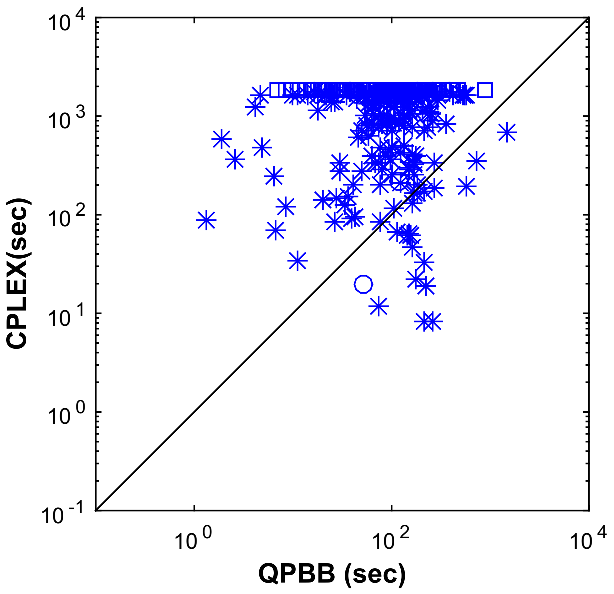

Fig. 5 Comparison of the computation time (in seconds) between our B&B algorithm and CPLEX on the 300 Medium-IVQR instances. The x- and y-axes are log scales, and the *diagonal line* defines $y = x$. An *asterisk* represents an instance where both algorithms reported optimality; a *square* represents an instance where CPLEX exceeded the time limit; a *circle* in the figure represents an instance where CPLEX had numerical issues

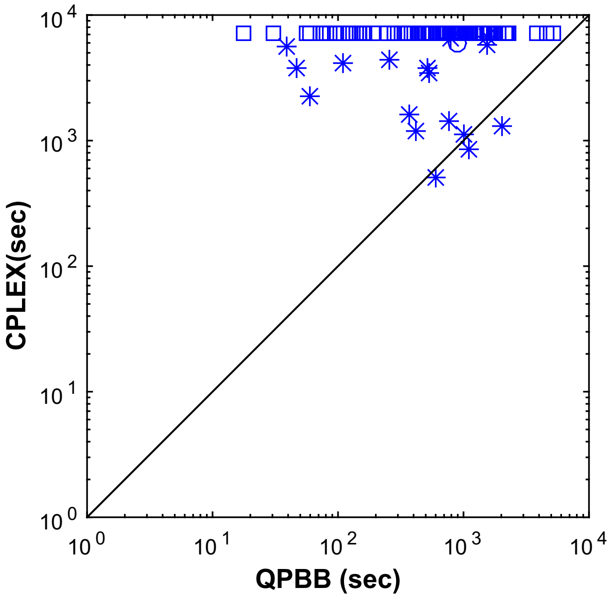

Fig. 6 Comparison of the computation time (in seconds) between our B&B algorithm and CPLEX on the 100 Large-IVQR instances. The x- and y-axes are log scales, and the *diagonal line* defines $y = x$. An *asterisk* represents an instance where both algorithms reported optimality; a *square* represents an instance where CPLEX exceeded the time limit; a *circle* represents an instance where CPLEX had numerical issues

the time limit of 7200 s on 82 instances and has numerical issues on the remaining 1 instance. Furthermore, our B&B algorithm outperforms CPLEX on 96 instances out of the 100.

Note that imposing the asymptotic bound, $5\|b\|_\infty$, does not cut off all optimal solutions for any Small-IVQR, Medium-IVQR, or Large-IVQR instances, which indicates that the bound works effectively in practice. As mentioned, our B&B does not technically require the asymptotic bound.

4.3.1 Sensitivity analysis of the big-M approach

It is well-known that mixed-integer programming is sensitive to larger values of the big-M. So we investigate the impact of this parameter on the computational times of both QPBB and CPLEX.

Our experiment proceeds as follows. We first apply our algorithm on formulation (9) with no upper bounds on x_3^+ and x_3^- and obtain optimal vectors $(x_3^+)^*$ and $(x_3^-)^*$. We then set $M := \max\{|(x_3^+)^*|, |(x_3^-)^*|\}$ and separately provide the pre-specified upper bounds M, $2M$, $5M$, and $10M$ to (9) for the variables x_3^+ and x_3^-. We then test our algorithm against CPLEX on each of the four cases on 100 Small-IVQR instances, and Fig. 7 displays the results. The overall trend is that QPBB performs better than CPLEX as the bounds M, $2M$, $5M$, and $10M$ increase. Furthermore, the computation time for each instance tends to increase as the bounds increase. Finally, note that CPLEX performs slightly worse than QPBB if M is chosen exactly. However,

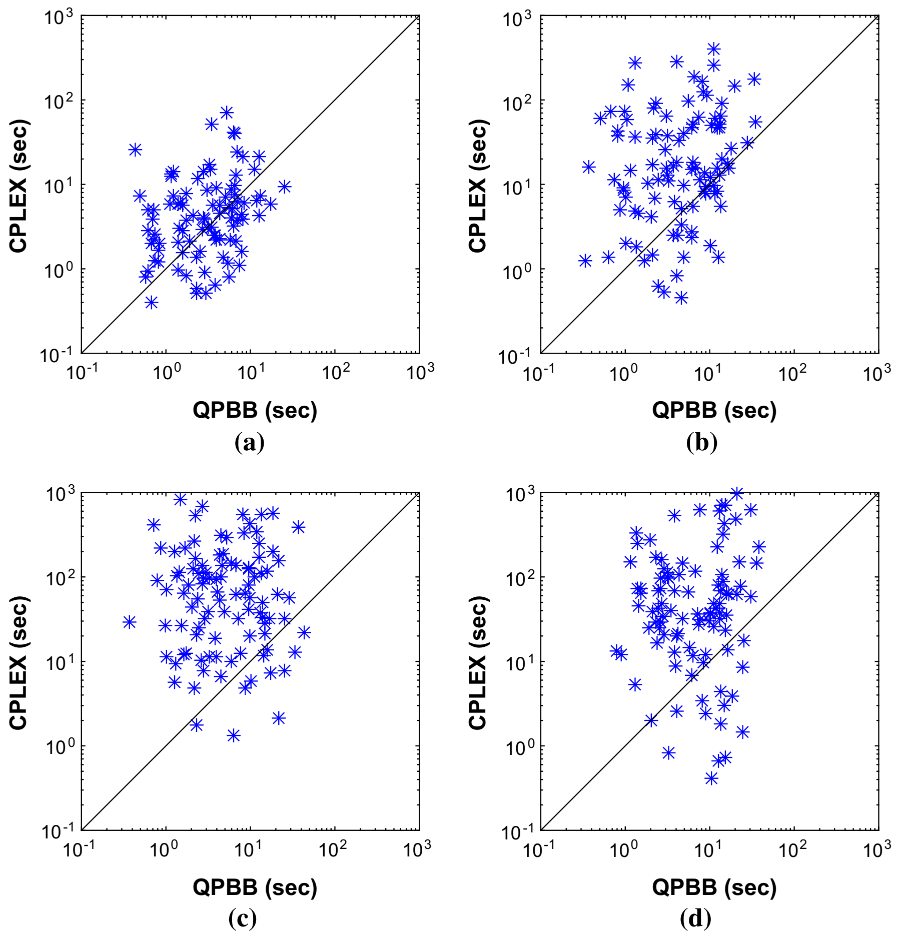

Fig. 7 Comparison of the computation times (in seconds) between our B&B algorithm and CPLEX with four pre-specified upper bounds of x_3^+ and x_3^- on 100 Small-IVQR instances. The x- and y-axes are log scales, and the *diagonal line* defines $y = x$. An *asterisk* represents an instance where both algorithms reported optimality. **a** The pre-specified value of M. **b** The pre-specified value of $2M$. **c** The pre-specified value of $5M$. **d** The pre-specified value of $10M$

CPLEX is more sensitive to the value of big-M and becomes even worse with bigger upper bounds. Of course, it is not a trivial task for CPLEX to find the true M.

We also investigate the impact of the pre-specified upper bounds on the number of nodes in the B&B tree solved by QPBB. Figure 8 shows box plots for the number of nodes on the four pre-specified upper bounds as well as no pre-specified upper bound. As shown in Fig. 8, there is a clear upward trend in the number of nodes solved increases as the bound increases.

4.4 Additional experiments

To improve the performance of our algorithm, we developed a variety of variants.

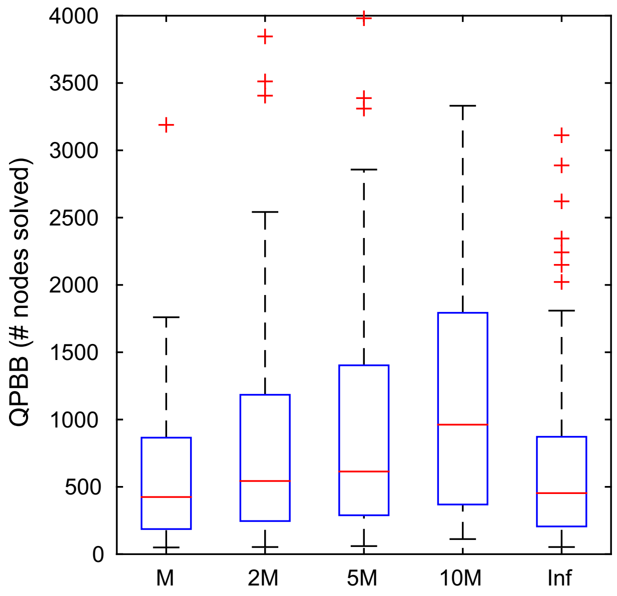

Fig. 8 Illustration of the number of nodes solved by our B&B algorithm on the 100 Small-IVQR instances with the four pre-specified upper bounds as well as no pre-specified upper bound. The y-axis shows the number of nodes solved by QPBB. Inf denotes that no upper bound is forced for x_3^+ and x_3^-

In terms of branching rules, we also tried strong-branching and pseudo-cost branching rules in addition to the maximum-violation rule. It turned out the maximum-violation branching rule performs the best among these rules. This is potentially due to the fact that the optimal values of the IVQR problems in (9) are typically very close to 0, as mentioned above. Generally speaking, there is no improvement on the LBs at the earlier levels of the tree since almost every branching variable returns a relaxation with optimal value $v = 0$. Thus, the extra computation does not return useful information. Similar observations are true for the pseudo-cost branching rule.

For node selection rules, we tried depth-first search and breadth-first search rules besides the two strategies we tested in Sect. 4. Results indicated that the best-bound search and bi-priority search strategies performed best.

5 Conclusions

In this paper, we have studied a problem arising in statistics called instrumental variable quantile regression (IVQR) and proposed a CQPCC formulation. We have introduced a relaxation scheme, which we then incorporated into a CQP-based B&B algorithm to solve the IVQR problem. We have tested our algorithm on three types of randomly generated instances against two well-known global solvers, Couenne and CPLEX. The computational results show that our B&B algorithm solves IVQR efficiently and robustly.

We have chosen to use the 2-norm of x_2 as the objective function for the IVQR problem because we understand from the IVQR literature that the 2-norm may be more amenable to analysis of the statistical properties of the errors in IVQR. However, in principle, one could use a different norm (e.g., the 1-norm or ∞-norm) to formulate the objective. This would lead to LP subproblems in the B&B algorithm, which could be an interesting direction for future research. Furthermore, as the main goal of IVQR problem is to minimize the norm of the estimate of instrumental variable, we have not investigated the tradeoff between this norm and the model's final estimate error. This could also be interesting for future research.

Acknowledgements The authors would like to acknowledge the participation and contribution of Alexandre Belloni, who supplied the IVQR problem as an interesting global, quadratic problem. He also provided the code to generate instances of the IVQR problem consistent with the statistics literature. The authors also express their sincere thanks to two anonymous referees and one anonymous technical editor for their helpful and insightful comments to improve the paper.

Appendix A Proof of Lemma 1

Proof Since $Ax^* + \epsilon = b$, we have from the triangle inequality that $\|\epsilon\|_\infty \leq \|b\|_\infty + \|Ax^*\|_\infty$ and $\|Ax^*\|_\infty \leq \|b\|_\infty + \|\epsilon\|_\infty$. So the event $\|\epsilon\|_\infty > C\|b\|_\infty$ implies

$$\|\epsilon\|_\infty > C\|b\|_\infty \geq C(\|\epsilon\|_\infty - \|Ax^*\|_\infty) = C\|\epsilon\|_\infty - C\|Ax^*\|_\infty$$

and

$$\|\epsilon\|_\infty > C\|b\|_\infty \geq C(\|Ax^*\|_\infty - \|\epsilon\|_\infty) = C\|Ax^*\|_\infty - C\|\epsilon\|_\infty,$$

which together imply

$$\frac{C}{C+1} < \frac{\|\epsilon\|_\infty}{\|Ax^*\|_\infty} < \frac{C}{C-1}.$$

This proves the first bound. To prove the second, we note that, by definition,

$$\|\epsilon\|_\infty > C\|b\|_\infty \Longleftrightarrow \max_{i=1}^{m} |\epsilon_i| > C \max_{k=1}^{m} \left| a_k^{\mathrm{T}} x^* + \epsilon_k \right|,$$

and so $\|\epsilon\|_\infty > C\|b\|_\infty$ implies that $|\epsilon_i| > C \max_{k=1}^{m} |a_k^{\mathrm{T}} x^* + \epsilon_k|$ holds for at least one specific i. Hence,

$$\begin{aligned} P\left(\|\epsilon\|_\infty > C\|b\|_\infty\right) &\leq \sum_{i=1}^{m} P\left(|\epsilon_i| > C \max_{k=1}^{m} \left| a_k^{\mathrm{T}} x^* + \epsilon_k \right| \right) \\ &\leq m \max_{i=1}^{m} P\left(|\epsilon_i| > C \max_{k} \left| a_k^{\mathrm{T}} x^* + \epsilon_k \right| \right). \qquad (14) \end{aligned}$$

Next, we claim $|a_k^{\mathrm{T}} x^* + \epsilon_k| \geq |\epsilon_k| \cdot \mathbb{1}\{\epsilon_k a_k^{\mathrm{T}} x^* \geq 0\}$ for each k. If $\epsilon_k a_k^{\mathrm{T}} x^* < 0$, this is certainly true. Otherwise, if $\epsilon_k a_k^{\mathrm{T}} x^* \geq 0$, then ϵ_k and $a_k^{\mathrm{T}} x^*$ have the same (possibly zero) signs and hence $|a_k^{\mathrm{T}} x^* + \epsilon_k| \geq |\epsilon_k|$. Thus,

$$|\epsilon_i| > C \max_{k=1}^{m} \left| a_k^{\mathrm{T}} x^* + \epsilon_k \right| \implies |\epsilon_i| > C \max_{k=1}^{m} \left\{ |\epsilon_k| \cdot \mathbb{1}\left\{ \epsilon_k a_k^{\mathrm{T}} x^* \geq 0 \right\} \right\}$$

and so

$$P\left(|\epsilon_i| > C \max_{k=1}^{m} \left| a_k^{\mathrm{T}} x^* + \epsilon_k \right| \right) \leq P\left(|\epsilon_i| > C \max_{k=1}^{m} \left\{ |\epsilon_k| \cdot \mathbb{1}\left\{ \epsilon_k a_k^{\mathrm{T}} x^* \geq 0 \right\} \right\} \right). \tag{15}$$

Combining inequalities (14) and (15), we achieve the second bound. □

Appendix B Proof of Lemma 2

Proof The first two-sided inequality is a standard fact about the normal distribution. For the last inequality, we have

$$P\left(\max_{1 \leq p \leq q} Z_p \leq \theta\sigma \right) = \Pi_{p=1}^{q} P(Z_p \leq \theta\sigma) = (1 - P(Z > \theta\sigma))^q.$$

By the first part of the lemma and the standard fact that $0 < x < 1$ and $a > 0$ imply $(1-x)^a \leq \exp(-ax)$,

$$\begin{aligned} (1 - P(Z > \theta\sigma))^q &\leq \left(1 - \frac{1}{2\theta} \cdot \epsilon(\theta)\right)^q \\ &\leq \exp\left(-\frac{q}{2\theta} \cdot \epsilon(\theta)\right). \end{aligned}$$

Now substituting the definition of $\epsilon(\theta)$ and $\theta = \sqrt{\log(q)}$, we see

$$\begin{aligned} \exp\left(-\frac{q}{2\theta} \cdot \epsilon(\theta)\right) &= \exp\left(-\frac{q}{2\theta} \cdot \frac{1}{\sqrt{2\pi}} \exp(-\log(q)/2)\right) \\ &= \exp\left(-\frac{q}{2\theta} \cdot \frac{1}{\sqrt{2\pi}} q^{-1/2}\right) \\ &= \exp\left(-\frac{\sqrt{q}}{2\sqrt{\log(q)}} \cdot \frac{1}{\sqrt{2\pi}}\right) \\ &\leq \exp\left(-q^{1/4}\right), \end{aligned}$$

where the last inequality follows from the assumption that $q/(8\pi \log(q)) \geq \sqrt{q}$. This proves the result. □

References

1. Ahuja, R.K., Orlin, J.B.: Inverse optimization. Oper. Res. **49**(5), 771–783 (2001)
2. Audet, C., Savard, G., Zghal, W.: New branch-and-cut algorithm for bilevel linear programming. J. Optim. Theory Appl. **38**(2), 353–370 (2007)
3. Bai, L., Mitchell, J.E., Pang, J.S.: On convex quadratic programs with linear complementarity constraints. Comput. Optim. Appl. **54**(3), 517–554 (2013)
4. Belloni, A., Chen, D., Chernozhukov, V., Hansen, C.: Sparse models and methods for optimal instruments with an application to eminent domain. Econometrica **80**(6), 2369–2429 (2012)
5. Cai, M.C., Duin, C.W., Yang, X.G., Zhang, J.Z.: The partial inverse minimum spanning tree problem when weight increase is forbidden. Eur. J. Oper. Res. **188**(2), 23–28 (2008)
6. Chen, C.: An introduction to quantile regression and QUANTREG procedure. In: 30th SAS user group international conference. SAS institute INC, Philadelphia (2005)
7. Chernozhukov, V., Hansen, C.: Instrumental variable quantile regression: a robust inference approach. J. Econom. **142**(1), 379–398 (2008)
8. Couenne, a solver for non-convex MINLP problems. Technical report, http://www.coin-or.org/Couenne/
9. Eckstein, J.: Parallel branch-and-bound algorithms for general mixed integer programming on the cm-5. SIAM J. Optim. **4**, 794–814 (1994)
10. Forrest, J.J.H., Hirst, J.P.H., Tomlin, J.A.: Practical solution of large scale mixed integer programming problems with umpire. Manag. Sci. **20**, 736–773 (1974)
11. Gabriel, S.A., Gareia-Bertrand, R., Sahakij, P., Concjo, A.J.: A practical approach to approximate bilinear functions in mathematical programming problems by using Schur's decomposition and SOS type 2 variables. J. Oper. Res. Soc. **57**(8), 995–1004 (2006)
12. Gentry, S.: Partial inverse linear programming. Technical Report LIDS-P-2532, Massachusetts Inst. Tech., Lab for Information and Decision Systems (2001)
13. Glankwamdee, W.: Topics in Branch and Bound on Computational Grids. ProQuest (2008)
14. Heuberger, C.: Inverse combinatorial optimization: a survey on problems, methods, and results. J. Comb. Optim. **8**(3), 329–361 (2004)
15. Horowitz, J., Lee, S.: Nonparametric instrumental variables estimation of a quantile regression model. Econometrica **75**(4), 1191–1208 (2007)
16. Hu, J., Mitchell, J.E., Pang, J.S., Bennett, K.P., Kunapuli, G.: On the global solution of linear programs with linear complementarity constraints. SIAM J. Optim. **19**(1), 445–471 (2008)
17. Koenker, R., Hallock, K.F.: Quantile regression. J. Econ. Perspect. **15**(4), 143–156 (2001)
18. Lai, T.C., Orlin, J.B.: The complexity of preprocessing. Technical report, Sloan School of Management, MIT (2003)
19. Lee, Y.C., Pang, J.S., Mitchell, J.E.: An algorithm for global solution to bi-parametric linear complementarity constrained linear programs. SIAM J. Optim. **62**(2), 263–297 (2015)
20. Liu, G.S., Zhang, J.Z.: A new branch-and-bound algorithm for solving quadratic programs with linear complementarity constraints. J. Comput. Appl. Math. **146**(1), 77–87 (2002)
21. Mangasarian, O.L.: Regularized linear programs with equilibrium constraints. In: Fukushima, M., Qi, L. (eds.) Reformulation-nonsmooth, piecewise smooth, semismooth and smoothing methods, pp. 259–268. Kluwer Academic Publisher, Dordrecht (1998)
22. Mitchell, J.E., Pang, J.S., Yu, B.: Obtaining tighter relaxations of mathematical programs with complementarity constraints. In Terlaky, T., Curtis, F. (eds) Modeling and Optimization: Theory and Applications, volume 21 of Springer Proceedings in Mathematics and Statistics, chapter 1. pp. 1–23. Springer, New York (2012)
23. Montgomery, D., Peck, E., Vining, G.: Introduction to Linear Regression Analysis. Wiley, New York (1982)
24. Sabo, K., Scitovski, R.: The best least absolute deviations line-properties and two efficient methods for its derivation. Anziam J. **50**, 185–198 (2008)
25. Sherali, H.D., Alameddine, A.: A new reformulation—linearization technique for bilinear programming problems. J. Glob. Optim. **2**(4), 379–410 (1992)
26. Vandenbussche, D., Nemhauser, G.: A polyhedral study of nonconvex quadratic programs with box constraints. Math. Program. **102**(3), 531–557 (2005)
27. Wang, Q.: Partial inverse most unbalanced spanning tree problem. Przeglad Elektrotechniczny **88**(1b), 111–114 (2012)

28. Yang, X.G.: Complexity of partial inverse assignment problem and partial inverse cut problem. RAIRO—Oper. Res. **35**(1), 117–126 (2001)
29. Yang, X.G., Zhang, J.Z.: Inverse sorting problem by minimizing the total weighted number of changes and partial inverse sorting problems. Comput. Optim. Appl. **36**(1), 55–66 (2007)
30. Yang, X.G., Zhang, J.Z.: Partial inverse assignment problems under l_1 norm. Oper. Res. Lett. **35**(1), 23–28 (2007)
31. Zhang, J., Liu, Z.: Calculating some inverse linear programming problems. J. Comput. Appl. Math. **72**(2), 215–434 (1996)