Computation of Least Trimmed Squares: A Branch-and-Bound framework with Hyperplane Arrangement Enhancements

Xiang Meng
MIT Operations Research Center, mengx@mit.edu

Andrés Gómez
USC Daniel J. Epstein Department of Industrial and Systems Engineering, gomezand@usc.edu

Rahul Mazumder
MIT Sloan School of Management, Operations Research Center, rahulmaz@mit.edu

Abstract. We study computational aspects of a key problem in robust statistics — the penalized least trimmed squares (LTS) regression problem, a robust estimator that mitigates the influence of outliers in data by capping residuals with large magnitudes. Although statistically attractive, penalized LTS is NP-hard, and existing mixed-integer optimization (MIO) formulations scale poorly due to weak relaxations and exponential worst-case complexity in the number of observations. We propose a new MIO formulation that embeds hyperplane arrangement logic into a perspective reformulation, explicitly enforcing structural properties of optimal solutions. We show that, if the number of features is fixed, the resulting branch-and-bound tree is of polynomial size in the sample size. Moreover, we develop a tailored branch-and-bound algorithm that uses first-order methods with dual bounds to solve node relaxations efficiently. Computational experiments on synthetic and real datasets demonstrate substantial improvements over existing MIO approaches: on synthetic instances with 5000 samples and 20 features, our tailored solver reaches a 1% gap in 1 minute while competing approaches fail to do so within one hour. These gains enable exact robust regression at significantly larger sample sizes in low-dimensional settings.

Key words: Robust statistics, mixed-integer nonlinear optimization, branch-and-bound, hyperplane arrangements

1. Introduction

Outliers—informally speaking, observations that deviate substantially from the bulk of the data—are pervasive in modern datasets. They arise from sensor failures, human entry mistakes, transient process instabilities, or simply corruptions in data. For example, in regression problems, contaminated observations or outliers can severely affect the quality of statistical estimates compared to estimates obtained from clean data. Classical least squares estimation is notoriously sensitive to outliers: a single high-leverage observation can arbitrarily distort the regression coefficients, potentially hurting model prediction and/or statistical inference (Huber 1981, Maronna et al. 2006). Ridge regression (i.e., least squares penalized with a squared ℓ_2-norm penalty) — despite its regularization benefits for ill-conditioned problems, inherits this sensitivity since the squared loss grows without bounds with the residual magnitude. To address this limitation, several robust estimators have been studied in the statistics literature (Rousseeuw and Leroy 1987).

Given a model matrix $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ and response vector $\boldsymbol{y} \in \mathbb{R}^n$, consider the optimization problem

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{z}\in\{0,1\}^n} \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \boldsymbol{x}_i^\top\boldsymbol{\beta}\right)^2(1-z_i) + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 + \mu\sum_{i=1}^{n} z_i, \tag{1}$$

where $\boldsymbol{x}_i^\top$ denotes the i-th row of $\boldsymbol{X}$, $\boldsymbol{\beta} \in \mathbb{R}^p$ denotes the regression coefficients and $r_i := y_i - \boldsymbol{x}_i^\top\boldsymbol{\beta}$, $i \in [n]$ are the residuals. We are interested in the traditional setting of robust statistics where the number of observations is larger than the dimension p, that is $n > p$. Binary variable $z_i \in \{0, 1\}$ indicates whether observation i is an outlier—the weight $(1 - z_i)$ serves to exclude large absolute values of the residual r_i (for $i \in [n]$) and include all other residuals in the computation of the regression coefficients. The excluded points correspond to outliers, and the remaining points are inliers. The parameter $\lambda \geq 0$ controls ridge-regularization, while $\mu \geq 0$ is a parameter that penalizes the number of outliers. Problem (1) is a penalized variant of the Least Trimmed Squares (LTS) estimator introduced by Rousseeuw (1984), which fits a model by minimizing the sum of the smallest squared residuals. The cardinality-constrained LTS places a constraint on the number of trimmed observations, while (1) represents its penalized form that allows the trimming level to be implicitly determined by the penalty μ. Compared to other robust estimators such as Least Median of Squares (LMS)(Rousseeuw 1984), LTS has desirable statistical properties — it enjoys $\sqrt{n}$-consistency, and higher asymptotic efficiency (Rousseeuw and Leroy 1987, Rousseeuw and Van Driessen 2006a).

The estimator in problem (1) can also be interpreted through the lens of a robust loss functions. Indeed, upon minimizing Problem (1) over the binary indicators $\boldsymbol{z}$, the problem reduces to:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \sum_{i=1}^{n} \phi_{\text{cap}}\left(y_i - \boldsymbol{x}_i^\top\boldsymbol{\beta}\right) + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2, \quad \text{where } \phi_{\text{cap}}(r) \stackrel{\text{def}}{=} \min\left\{\frac{1}{2}r^2,\ \mu\right\}. \tag{2}$$

The objective has a capped quadratic loss $\phi_{\text{cap}}(r)$ with an additional ridge regularization with penalty parameter $\lambda/2 \geq 0$. The loss $r \mapsto \phi_{\text{cap}}(r)$ is quadratic for small (absolute) residuals but saturates at the constant level μ for large (absolute) residuals, discouraging grossly contaminated observations from dominating the fit. The nonconvex capped quadratic belongs to a broader class of robust losses that temper growth in the tails, as illustrated in Figure 1, and has close ties to the redescending M-estimators (Maronna et al. 2006). A canonical example of a convex loss function is the Huber loss, which transitions from quadratic to linear

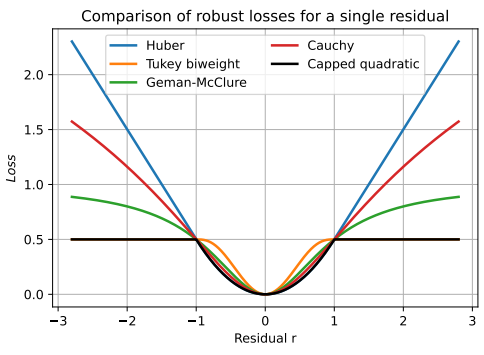

Figure 1: Various robust loss functions.

growth and is a standard baseline in robust regression (Huber 1964, 1981). Well known bounded, non-convex choices of the loss function include Tukey's biweight (Holland and Welsch 1977), Geman–McClure (Chang et al. 2012), and Cauchy/Lorentzian losses (Motulsky and Brown 2006). Relative to these alternatives, the objective in (2) offers a transparent interpretation: observations with sufficiently large residuals contribute a constant effect μ and are effectively trimmed from the fit.

Despite its attractive properties, solving problem (1) poses a formidable challenge. The problem is NP-hard (Bernholt 2006), and early approaches relied on heuristics such as such as the feasible solution algorithm (Hawkins 1994) and its subsequent refinement (Hawkins and Olive 1999), or the widely used FAST-LTS algorithm (Rousseeuw and Van Driessen 2006a). While efficient, these methods provide no optimality guarantees and can produce solutions that are far away from the optimum (Gómez and Neto 2025). In addition, the desirable statistical properties of the LTS estimator are predicated on obtaining a global minimizer of (1), and a suboptimal solution need not inherit any of these guarantees.

Problem (1) can be formulated as a mixed-integer quadratic program, enabling globally optimal solutions via modern mixed-integer optimization (MIO) solvers (Gómez and Neto 2025, Sun et al. 2021, Zioutas and Avramidis 2005a). However, these methods have a worst case complexity of $\mathcal{O}(2^n)$. While MIO methods often exhibit practical runtimes that are substantially better than the theoretical worst-case complexities; unfortunately, for the case of (1), convex relaxations are typically weak and branch-and-bound algorithms require a prohibitive number of nodes to demonstrate optimality. It appears that existing MIO methods for LTS can deliver optimal solutions to problem instances with $n \lesssim 100$ and $p \lesssim 10$ (practical performance is also highly dependent on hyperparameters λ and μ and the dataset itself).

Outside of the MIO literature, some exact combinatorial algorithms have been proposed. Agulló (2001a) proposed both a probabilistic exchange algorithm and an branch-and-bound algorithm for solving LTS. Hofmann et al. (2010) designed a row adding algorithm that computes exact LTS solutions for a range of coverage values, and Klouda (2015) studied an exact algorithm for solving LTS whose computational cost scales as $O(n^{p+1})$. These exact combinatorial methods are generally limited to relatively small or low-dimensional problem instances.

In a different line of work, exact methods based on topological sweeps for solving LTS have been proposed in the statistics literature. These methods inspired by computational geometry are

based on an arrangement of hyperplanes, and require solving $\mathcal{O}(n^p)$ least squares problems. While topological sweep methods can in principle scale to a large number of points provided that p is small, we are unaware of practical implementations apart from the special case of $p = 2$ (e.g., Edelsbrunner and Souvaine 1990, Hössjer 1995). However, as far as we can tell, there are no available practical implementations for the topological sweep method for $p \geq 3$.

Contributions

In this paper we propose new MIO formulations and algorithms for Problem (1), designed with the common regime of $n \gg p$ in mind. The proposed formulations build on state-of-the-art perspective reformulations for Problem (1) with strong convex relaxations (Gómez and Neto 2025), but additionally include additional constraints enforcing hyperplane arrangements logic. The proposed formulation avoids big-M constraints altogether, and branch-and-bound methods based on the proposed formulation terminate after exploring at most $\mathcal{O}(n^{p+1})$ nodes. Unlike topological sweep methods, the practical runtime of branch-and-bound algorithms appear to be substantially better than the worst-case performance. In fact, we propose the first practical implementation of methods based on hyperplane arrangement for Problem (1) that can handle instances with $p \geq 3$. The proposed formulation leads to at least an order-of-magnitude improvement over alternative state-of-the-art MIO approaches (Gómez and Neto 2025, Insolia et al. 2022), and the approach of Bertsimas and Mazumder (2014) for the LMS problem.

In addition, we also develop a tailored branch-and-bound solver to tackle the proposed formulation. Our algorithm is developed in Python, uses first-order methods to solve the continuous subproblems and requires no commercial software. Our code builds upon and significantly extends the earlier work of Hazimeh et al. (2021) proposing tailored branch-and-bound approaches for a different problem sparse linear models (See Section 4 for differences with this work). The computational performance of our algorithm is competitive with state-of-the-art commercial solver Gurobi to solve the proposed formulation, and appears substantially faster than using other commercial MIO solvers. On synthetic datasets, our method scales to significantly larger sample sizes: on instances with $n = 5000$ samples and $p = 20$ features, it reaches a 1% optimality gap in 1 minute, whereas no competing approach does so within a one-hour time limit. On 13 real benchmark regression datasets, our method and Gurobi applied to the proposed formulation achieve the strongest performance, each attaining the best runtime on roughly half of the instances; the remaining baselines are substantially slower or fail to close the gap within the time limit.

Outline

The rest of the paper is organized as follows. In Section 2 we review existing methods for (1) (MIO and topological sweep methods as well as heuristics). In Section 3 we describe the proposed MIO formulation, study its strength and prove its worst-case complexity of $\mathcal{O}(n^{p+1})$. In Section 4 we describe the proposed branch-and-bound algorithm based on first-order methods. Finally, in Section 5 we provide computational experiments with synthetic and real data.

2. Review of existing solution approaches

We review exact solution approaches for Problem (1) from the literature. First, we discuss exact MIO approaches that have recently been proposed in the Operations Research community. We then discuss classical exact solution approaches based on hyperplane arrangements from the statistics literature. Finally, we examine approximate methods (also known as heuristics) for finding good solutions, which are currently the most widely used approach in practice.

2.1. MIO formulations

At a high level, MIO approaches are based on the branch-and-bound method. The natural approach is to branch on the binary variables z, that is, branching on whether a data point should be deemed an outlier or not. Earlier approaches from the mathematical optimization community (Agulló 2001b, Giloni and Padberg 2002) studied formulation (1) directly. However, nonconvexity of the continuous relaxation of Problem (1) due to the presence of cubic terms in the objective makes the design of efficient solution methods challenging. The first exact MIO approaches were proposed by Zioutas and Avramidis (2005b) and Zioutas et al. (2009), which reformulated (1) into mixed-integer quadratically constrained quadratic programs by introducing additional variables representing the loss incurred from fitting each point. More recently, Insolia et al. (2022) proposed an alternative mixed-integer quadratic optimization reformulation by introducing variables representing "corrections" associated with each datapoint, which we review in §2.1.1. Finally, stronger formulations were proposed in the literature, see Gómez (2021) for methods specific to time-series data and Gómez and Neto (2025) for approaches for the general least trimmed squares problem (1), which we review in Section 2.1.2.

In general, MIO methods have a theoretical worst case complexity of $\mathcal{O}(2^n)$, corresponding to all possible combinations of inliers and outliers. Branch-and-bound methods typically explore substantially fewer branch-and-bound nodes (especially if strong relaxations are used). Nonetheless,

both the theoretical complexity and computational experiments reported in the aforementioned papers suggest that the performance of existing MIO methods deteriorates substantially as the sample size n increases. Current MIO approaches have been shown to perform well with real data for $n \approx 100$, but struggle to solve to optimality instances with larger values of n.

2.1.1. Big-M Formulation We first consider a MIO formulation of Problem (1) similar to the one proposed by Insolia et al. (2022), incorporating Big-M constraints. We introduce auxiliary variables $\boldsymbol{w} \in \mathbb{R}^n$ representing corrections to the response vector, where w_i captures the residual when observation i is treated as an outlier—that is, $w_i = y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}$. We define a constant $M > 0$ such that any optimal correction satisfies $|w_i^*| \leq M$. Binary variables $z_i, i \in [n]$, indicate whether observation i is an outlier, enforced through the constraints $-Mz_i \leq w_i \leq Mz_i$, $i \in [n]$. This results in the following Big-M formulation:

$$\begin{aligned} \min_{\boldsymbol{\beta},\boldsymbol{w},\boldsymbol{z}} \ & \frac{1}{2}\sum_{i=1}^{n}\left(y_i - w_i - \boldsymbol{x}_i^\top\boldsymbol{\beta}\right)^2 + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 + \mu\sum_{i=1}^{n} z_i \\ \text{s.t.} \ & -Mz_i \leq w_i \leq Mz_i, \quad i \in [n], \\ & \boldsymbol{\beta} \in \mathbb{R}^p,\ \boldsymbol{w} \in \mathbb{R}^n,\ z_i \in \{0,1\}, \quad i \in [n]. \end{aligned} \tag{3}$$

In the above formulation, if $z_i = 0$, then the big-M constraints force $w_i = 0$; in that case, the loss associated with datapoint i reduces to the standard quadratic loss. On the other hand, if $z_i = 1$, then w_i is free to take any value; in particular, for a fixed β, setting $w_i = y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}$ is optimal, and the associated loss with datapoint i vanishes.

There are two concerns in using formulation (3). Firstly, we need to specify the value of the maximum magnitude M, which can be large for arbitrary outliers. This is different from formulation (1), which doesn't need such a specification. If we select M to be sufficiently large so that a solution to Problem (3) is also a solution to Problem (1) (e.g., using arguments similar to Bertsimas et al. 2016), the value of M can be very conservative, possibly causing numerical issues for an algorithm. Additionally, convex relaxations of (3) are weak since solution $\boldsymbol{\beta} = \mathbf{0}, \boldsymbol{w} = \boldsymbol{y}, \boldsymbol{z} = \epsilon\mathbf{1}$ are feasible provided $\epsilon \geq \|\boldsymbol{y}\|_\infty / M$, with objective values close to 0 for any reasonable value of M. As a consequence, branch-and-bound algorithms struggle to prune nodes and make informed branching decisions, leading to prohibitive solution times.

2.1.2. Perspective Formulation Gómez and Neto (2025) propose stronger, big-M free formulations of Problem (1), obtained by applying the perspective reformulation (Frangioni and Gentile 2006, Günlük and Linderoth 2010).

The key observation is that the objective of Problem (3) can be written as

$$\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{w}-\boldsymbol{X}\boldsymbol{\beta}\|_2^2+\frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2=\frac{1}{2}\|\boldsymbol{y}\|_2^2-\boldsymbol{y}^\top(\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{w})+\frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\boldsymbol{\Sigma}\begin{pmatrix}\boldsymbol{\beta}\\ \boldsymbol{w}\end{pmatrix},$$

where $\boldsymbol{\Sigma}=\begin{pmatrix}\boldsymbol{X}^\top\boldsymbol{X}+\lambda\boldsymbol{I} & -\boldsymbol{X}^\top\\ -\boldsymbol{X} & \boldsymbol{I}\end{pmatrix}$. Given $\boldsymbol{d}\in\mathbb{R}^n_+$, define

$$\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}=\begin{pmatrix}\boldsymbol{X}^\top\boldsymbol{X}+\lambda\boldsymbol{I} & -\boldsymbol{X}^\top\\ -\boldsymbol{X} & \boldsymbol{I}-2\text{Diag}(\boldsymbol{d})\end{pmatrix}$$

and note that the following quadratic term can be decomposed as

$$\frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\boldsymbol{\Sigma}\begin{pmatrix}\boldsymbol{\beta}\\ \boldsymbol{w}\end{pmatrix}=\frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}\begin{pmatrix}\boldsymbol{\beta}\\ \boldsymbol{w}\end{pmatrix}+\sum_{i=1}^n d_iw_i^2, \tag{4}$$

where the first term remains convex provided that matrix $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}$ is positive semi-definite. The perspective reformulation then replaces the separable term $\sum_i d_iw_i^2$ with $\sum_i d_iw_i^2/z_i$. The latter term equals $d_iw_i^2$ when $z_i=1$ and enforces $w_i=0$ when $z_i=0$, using the convention that $0/0=0$ and $x/0=\infty$ for $x>0$. We thus obtain the perspective formulation

$$\begin{aligned}\min_{\boldsymbol{\beta},\boldsymbol{w},\boldsymbol{z}}\ & \frac{1}{2}\|\boldsymbol{y}\|_2^2-\boldsymbol{y}^\top(\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{w})+\frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}\begin{pmatrix}\boldsymbol{\beta}\\ \boldsymbol{w}\end{pmatrix}+\sum_{i=1}^n\left(\mu z_i+\frac{d_iw_i^2}{z_i}\right)\\ \text{s.t.}\ & -Mz_i\le w_i\le Mz_i,\quad i\in[n],\\ & \boldsymbol{\beta}\in\mathbb{R}^p,\ \boldsymbol{w}\in\mathbb{R}^n,\ z_i\in\{0,1\},\quad i\in[n].\end{aligned} \tag{5}$$

Observe that if $\boldsymbol{d}>\boldsymbol{0}$, then the big-M constraints $-M\boldsymbol{z}\le\boldsymbol{w}\le M\boldsymbol{z}$ can be removed from the formulation. The perspective formulation (5) is equivalent to (3) but yields tighter continuous relaxations, leading to smaller branch-and-bound trees and faster solve times.

To find a vector $\boldsymbol{d}$ that results in the best convex relaxation of Problem (5), Gómez and Neto (2025) propose an algorithm that solves a sequence of positive semi-definite programs (SDPs) with cones of order $p+1$. We now describe an alternative and simpler approach we use in our numerical experiments. First, let $\tilde{\lambda}\in(0,\lambda)$ be a small number used to guarantee strong convexity and numerical stability. We note that, by the Schur complement, we can guarantee that $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}\succ\boldsymbol{0}$ holds by ensuring that $\boldsymbol{I}-\boldsymbol{X}(\boldsymbol{X}^\top\boldsymbol{X}+(\lambda-\tilde{\lambda})\boldsymbol{I})^{-1}\boldsymbol{X}^\top-2\text{Diag}(\boldsymbol{d})\succeq\boldsymbol{0}$. Thus, setting

$$d_j=\frac{1}{2}\gamma_{\min}\left(\boldsymbol{I}-\boldsymbol{X}(\boldsymbol{X}^\top\boldsymbol{X}+(\lambda-\tilde{\lambda})\boldsymbol{I})^{-1}\boldsymbol{X}^\top\right)\quad\forall j\in[n], \tag{6}$$

where $\gamma_{\min}(\cdot)$ denotes the minimum eigenvalue of the argument matrix, guarantees that $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}} \succ \mathbf{0}$. While choice (6) results in a weaker relaxation than using the method proposed by Gómez and Neto (2025), this can be computed faster and does not require access to (commercial) conic optimization solvers or specialized algorithms.

2.2. Solution via hyperplane arrangements

Different from MIO approaches, exact algorithms for (1) were proposed in the statistics literature (Rousseeuw and Leroy 2003) based on hyperplane arrangements. The algorithms involved solving $\mathcal{O}(n^p)$ least squares regression problems. Thus, unlike MIO methods, the complexity is polynomial in the number of datapoints n but exponential in the number of features p. As far as we can tell, these algorithms are difficult to implement in practice. We now review methods based on hyperplane arrangements. We note that the method we describe differs from classical algorithms for the least trimmed squares problems, since we describe a method for the penalized version whereas classical approaches tackle cardinality constrained problems with an upper bound on the number of outliers. Nonetheless, the ideas behind the algorithms are similar.

The hyperplane arrangement algorithm relies on two key results: the first (Proposition 1) concerns structure of optimal solutions of (1), and the second (Proposition 2) is a classical result from combinatorial geometry (Cover 1965, Zaslavsky 1975).

PROPOSITION 1. *Any optimal solution* $(\boldsymbol{\beta}^*, \boldsymbol{z}^*)$ *of Problem* (1) *satisfies:*

1. *if* $z_i^* = 0$*, then* $|y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^*| \leq \sqrt{2\mu}$*;*
2. *if* $z_i^* = 1$*, then* $|y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^*| \geq \sqrt{2\mu}$*.*

PROPOSITION 2. *An arrangement of* n *hyperplanes in* $\mathbb{R}^p$ *passing through the origin divide the space into at most* $2\sum_{i=0}^{p-1} \binom{n}{i} = \mathcal{O}(n^p)$ *regions, where the upper bound is attained if the hyperplanes are in general position.*

We can use Proposition 2 to solve Problem (1) as follows. Each datapoint $(\boldsymbol{x_i}, y_i)$ induces three regions in $\mathbb{R}^p$:

$$
\begin{aligned}
B_i^{\leq} &= \left\{ \boldsymbol{\beta} \in \mathbb{R}^p : \boldsymbol{x_i}^\top \boldsymbol{\beta} \leq y_i - \sqrt{2\mu} \right\} \\
B_i^{=} &= \left\{ \boldsymbol{\beta} \in \mathbb{R}^p : y_i - \sqrt{2\mu} \leq \boldsymbol{x_i}^\top \boldsymbol{\beta} \leq y_i + \sqrt{2\mu} \right\} \\
B_i^{\geq} &= \left\{ \boldsymbol{\beta} \in \mathbb{R}^p : \boldsymbol{x_i}^\top \boldsymbol{\beta} \geq y_i + \sqrt{2\mu} \right\}.
\end{aligned}
$$

Note that for any solution $\boldsymbol{\beta}$ in the band defined by $B_i^{=}$, setting $z_i = 0$ is a better choice than $z_i = 1$ in Problem (1)—that is, point i is an inlier. Similarly, for any $\boldsymbol{\beta} \in B_i^{\leq} \cup B_i^{\geq}$ we have that $z_i = 1$ is a better choice in Problem (1), thus point i would be categorized as an outlier. The three regions are defined by two hyperplanes, $\boldsymbol{x_i}^\top \boldsymbol{\beta} = y_i - \sqrt{2\mu}$ and $\boldsymbol{x_i}^\top \boldsymbol{\beta} = y_i + \sqrt{2\mu}$ corresponding to data point $i \in [n]$. Thus, combining all $2n$ hyperplanes creates a division of $\mathbb{R}^p$ into $\mathcal{O}((2n)^p)$ regions, each corresponding to a selection of outliers. Hence an optimal solution to Problem (1) can be obtained by solving a ridge regularized least squares problem for each one of the regions (with a fixed selection of outliers) and choosing the solution that yields the best objective value.

EXAMPLE 1. Consider three points in $\mathbb{R}^2$: $(x_1, y_1) = (-1, 8)$, $(x_2, y_2) = (0, 0.7)$ and $(x_3, y_3) = (1, 1)$. Figure 2 (left) shows the three points in the plane as well as the optimal ridge regularized LTS regression line $y = \beta_0 + \beta_1 x$, obtained by solving (1) with $\lambda = \mu = 1$ – note that the first point is flagged as an outlier in the optimal solution $(\beta_0^*, \beta_1^*) = (0.8, 0.1)$. Figure 2 (right) shows the associated hyperplane arrangement in the $\boldsymbol{\beta}$ space: each point i is associated with a band $B_i^{=}$, and for any solution $(\beta_0, \beta_1) \in B_i^{=}$ the best choice is to keep point i as an inlier; point i is discarded as an outlier for choices $(\beta_0, \beta_1) \notin B_i^{=}$. Since the three bands do not intersect, we can conclude that $\boldsymbol{z} = \mathbf{0}$ cannot be optimal, thus the solution with all three points as inliers should not be considered. ■

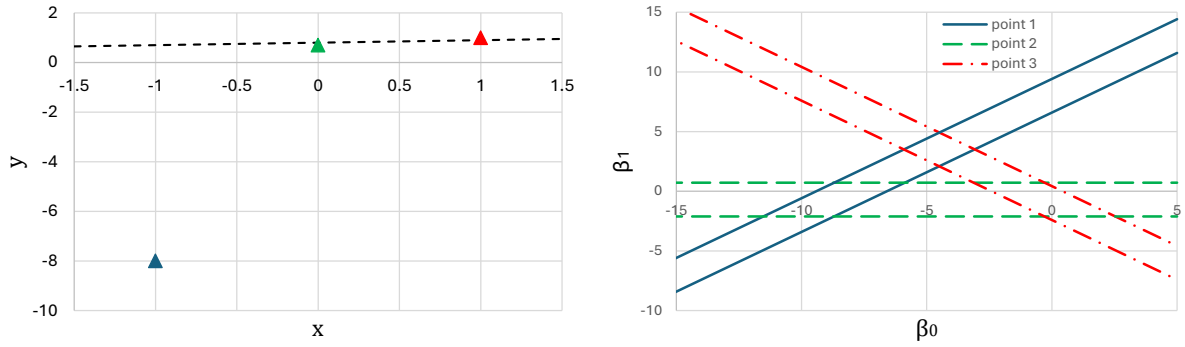

Figure 2 Illustration of Example 1: an ridge regularized LTS regression in the plane (left), and associated hyperplane arrangement (right).

Note that for the most part, the algorithmic idea of traversing all regions while solving least squares problems is is computationally expensive and hence may be impractical. Efficient implementations do exist if $p = 2$ (e.g., Edelsbrunner and Souvaine 1990, Hössjer 1995): using the observation that the intersection of two lines occurs at a single point, one can efficiently enumerate

all regions with proper data structures. Unfortunately, we are not aware of practical implementations for $p > 2$. In theory, it is possible to do so via linear optimization for example (Černỳ et al. 2019), but the additional computational cost can be excessive. Moreover, the computational cost of computing $\mathcal{O}(n^p)$ least squares estimators can be prohibitive even if $p = 3$ for sufficiently large values of n.

2.3. Heuristics

Given the perceived inefficiencies of exact methods, the preferred solution approach for Problem (1) is via heuristics. The most popular heuristic, called FAST-LTS (Rousseeuw and Van Driessen 2006b), is based on alternating minimization and works as follows. Starting from an initial point $\boldsymbol{\beta}$, the heuristic alternates between: *(i)* flag the points with largest residuals as outliers (optimization over $\boldsymbol{z}$); *(ii)* solve a least squares problem for the new inlier/outlier combination (optimization over $\boldsymbol{\beta}$). Variants of this method are common in the literature for problems similar to Problem (1) (Shen and Sanghavi 2019a,b). While alternating minimization heuristics have been observed to work well in high signal-to-noise ratios, and performance guarantees can be established under appropriate conditions (Bhatia et al. 2015), they have also been shown to lead to suboptimal solutions in challenging instances (Gómez and Neto 2025).

3. Enhanced MIO formulation with hyperplane arrangements

The perspective formulation (5) treats each observation independently: it strengthens the relaxation of each w_i^2 term, but does not exploit the relationship between the outlier indicators $\boldsymbol{z}$ and the regression coefficients $\boldsymbol{\beta}$. Hyperplane arrangements, on the other hand, capture precisely this relationship through the threshold structure of Proposition 1. We propose a formulation that incorporates this structure into the perspective formulation, by enforcing the logical relationships

$$z_i = 0 \implies \boldsymbol{\beta} \in B_i^{=} \tag{7a}$$

$$z_i = 1 \implies \boldsymbol{\beta} \in B_i^{\leq} \cup B_i^{\geq} \tag{7b}$$

for all datapoints $i \in [n]$. A direct approach to enforce (7) is by including additional binary variables $\boldsymbol{z}^-, \boldsymbol{z}^+ \in \{0,1\}^n$ such that $\boldsymbol{z}^- + \boldsymbol{z}^+ = \boldsymbol{z}$ and in the big-M constraints

$$y_i + \sqrt{2\mu} - M(1 - z_i^+) \leq \boldsymbol{x_i}^\top \boldsymbol{\beta} \leq y_i - \sqrt{2\mu} + M(1 - z_i^-) \tag{8a}$$

$$y_i - \sqrt{2\mu} - M(z_i^- + z_i^+) \leq \boldsymbol{x_i}^\top \boldsymbol{\beta} \leq y_i + \sqrt{2\mu} + M(z_i^- + z_i^+) \tag{8b}$$

for all $i \in [n]$. The additional variables $\boldsymbol{z}^-, \boldsymbol{z}^+$ have the interpretation that "$z_i^+ = 1$ if and only if $\boldsymbol{\beta} \in B_i^+$" and "$z_i^- = 1$ if and only if $\boldsymbol{\beta} \in B_i^-$". Note that constraints (8) are not valid for (3) or (5) in the usual sense, as they remove feasible integer points. Nonetheless, they do not remove any optimal solutions of the MIO problems and can be used as optimality cuts or constraints.

A limitation of constraints (8) is the presence of the big-M terms. Indeed, the constant M is required to be at least as large as the maximum residual, which in principle can be arbitrarily large. We now propose an alternative formulation that imposes the logical considerations (7) on the perspective formulation of the LTS problem (5) without using big-M constraints:

$$\min_{\substack{\boldsymbol{\beta},\boldsymbol{w},\boldsymbol{z} \\ \boldsymbol{w}^\pm,\boldsymbol{z}^\pm}} \frac{1}{2}\|\boldsymbol{y}\|_2^2 - \boldsymbol{y}^\top(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{w}) + \frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}\begin{pmatrix}\boldsymbol{\beta} \\ \boldsymbol{w}\end{pmatrix} + \sum_{i=1}^n \left(\mu z_i + \frac{d_i(w_i^+)^2}{z_i^+} + \frac{d_i(w_i^-)^2}{z_i^-}\right)$$

$$\text{s.t.} \quad w_i^- \geq \sqrt{2\mu}\, z_i^-, \quad w_i^+ \geq \sqrt{2\mu}\, z_i^+, \quad i \in [n], \tag{9a}$$

$$|y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta} - w_i| \leq \sqrt{2\mu}(1 - z_i^- - z_i^+), \quad i \in [n], \tag{9b}$$

$$z_i = z_i^+ + z_i^-, \quad w_i = w_i^+ - w_i^-, \quad i \in [n], \tag{9c}$$

$$\boldsymbol{\beta} \in \mathbb{R}^p,\ \boldsymbol{w} \in \mathbb{R}^n, \boldsymbol{z} \in \{0,1\}^n, \boldsymbol{w}^-, \boldsymbol{w}^+ \in \mathbb{R}^n_+, \boldsymbol{z}^-, \boldsymbol{z}^+ \in \{0,1\}^n. \tag{9d}$$

Note that in Problem (9), the optimization variables $\boldsymbol{z}$ and $\boldsymbol{w}$ can be projected out and constraints (9c) removed, but we keep them for now to facilitate comparisons with Problem (5). All constraints in problem (9) are either linear or can be linearized in the case of (9b). Intuitively, variables $\boldsymbol{w}^+$ and $\boldsymbol{w}^-$ represent the positive and negative parts of $\boldsymbol{w}$, respectively; variable $z_i^+ = 1$ ($z_i^- = 1$) if point i is an outlier by overestimating (underestimating) the response variable. The rest of this section is devoted to studying formulation (9).

3.1. Formulation Strength

The proposed formulation (9) is at least as strong as the perspective reformulation (5), and avoids using big-M constraints such as (8). The perspective reformulation (5) is obtained from the convex hull of the epigraph of the quadratic regularization term and indicator constraints, that is,

$$Z_{\text{persp}} = \left\{(z,w,t) \in \{0,1\} \times \mathbb{R}^2 : t \geq w^2,\ w(1-z) = 0\right\}.$$

Our formulation is based on a richer structure, involving a residual r_i corresponding to a term of the form $y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}$ for some $i \in [n]$. Namely, given $b \in \mathbb{R}_+$, we consider the set

$$Z_{\text{HA}}(b) = \left\{(z,w,r,t) \in \{0,1\} \times \mathbb{R}^3 : t \geq w^2,\ w(1-z) = 0,\ |r|(1-z) \leq b(1-z),\ |r|z \geq bz,\ w = rz\right\}.$$

Using decomposition (4), we can reformulate Problem (1) as the MIO

$$\min_{\boldsymbol{\beta},\boldsymbol{w},\boldsymbol{z},\boldsymbol{t}} \quad \frac{1}{2}\|\boldsymbol{y}\|_2^2 - \boldsymbol{y}^\top(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{w}) + \frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}\begin{pmatrix}\boldsymbol{\beta}\\ \boldsymbol{w}\end{pmatrix} + \sum_{i=1}^n (\mu z_i + d_i t_i) \tag{10a}$$

$$\text{s.t.} \quad (z_i, w_i, y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}, t_i) \in Z_{\text{HA}}(\sqrt{2\mu}) \quad \forall i \in [n] \tag{10b}$$

$$\boldsymbol{\beta} \in \mathbb{R}^p,\ \boldsymbol{w} \in \mathbb{R}^n,\ \boldsymbol{z} \in \{0,1\}^n,\ \boldsymbol{t} \in \mathbb{R}^n. \tag{10c}$$

Note that constraints (10b) include the epigraph constraint $t_i \geq w_i^2$. In addition:

1. if $z_i = 0$, then the constraint in (10b) reduces to $w_i = 0$ and $|y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}| \leq \sqrt{2\mu}$, encapsulating precisely relationship (7a).
2. if $z_i = 1$, then the constraint (10b) reduces to $|y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}| \geq \sqrt{2\mu}$ –corresponding precisely to (7b)– and $w = y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}$, encapsulating the additional optimality condition that the additional variables w_i are exactly the residuals to ensure that the associated term vanishes.

Naturally, set $Z_{\text{HA}}(b)$ is highly nonconvex due to the presence of binary variables and quadratic constraints. Thus, instead of relying directly on it, our formulation relaxes constraints (10b) to $(z_i, w_i, y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}, t_i) \in \text{cl conv}\left(Z_{\text{HA}}(\sqrt{2\mu})\right),\ \forall i \in [n]$. The description of this convex hull is given in Proposition 3, and the proof is deferred to Appendix A.1.

PROPOSITION 3. *The closure of the convex hull of $Z_{HA}(b)$ is given by cl conv $(Z_{HA}(b)) =$ proj$_{(z,w,r,t)}\bar{Z}(b)$ where*

$$\bar{Z}(b) = \Big\{(z,w,r,t,w^-,w^+,z^-,z^+) : t \geq \frac{(w^+)^2}{z^+} + \frac{(w^-)^2}{z^-},\ w^- \geq bz^-,\ w^+ \geq bz^+,\\ |r-w| \leq b(1-z^- - z^+),\ z^- + z^+ = z \leq 1,\ w = w^+ - w^-,\ z^-, z^+ \geq 0\Big\}.$$

The constraints defining set $\bar{Z}(b)$ in Proposition 3 are precisely (9a)-(9c). We formalize this property in the following corollary.

COROLLARY 1. *Formulation* (9) *is a correct formulation for Problem* (1)*, obtained from Problem* (10) *by replacing constraints* (10b) *with* $(z_i, w_i, y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}, t_i) \in$ *cl conv* $\left(Z_{HA}(\sqrt{2\mu})\right),\ \forall i \in [n]$.

So far we have established that formulation (9) is strong, in the sense of directly using the convex hull of a region involving a mix of feasibility and optimality conditions. The formulation is indeed the strongest in its class, that is, no stronger formulation can exist unless additional structure is included in the set (e.g., considering sets with multiple datapoints with non-trivial interactions). However, we state in Proposition 4 that the proposed relaxation does not improve the continuous relaxation of (5). This result is a consequence of the more general Proposition 7, and we defer its proof until then.

PROPOSITION 4. *Constraints* (9a)-(9b) *are redundant for the continuous relaxation of Problem* (9)*.*

COROLLARY 2. *The optimal solutions and optimal objective values of the convex relaxations of formulations* (5) *and* (9) *coincide.*

In light of Proposition 4, preferring formulation (9) over formulation (5) seems counterintuitive: the continuous relaxation is harder to solve due to the presence of additional constraints, and there appear to be no associated gains in terms of relaxation quality. However, we note that the result of Proposition 4 holds only for the relaxation of Problem (9), and need not hold if any additional constraint is added or objective terms are modified. In particular, the result of Proposition 4 does not hold when additional branching constraints of the form $z_i^{\pm} \leq 0$ or $z_i^{\pm} \geq 1$ are introduced. Thus, even if the root relaxations obtained from both formulations coincide, using formulation (9) results in better relaxations at every node other than the root node of the branch-and-bound tree. In fact,

as we show in the next section, the inclusion of constraints (9a)-(9b) guarantees that branch-and-bound algorithms explore a number of nodes polynomial in the number of datapoints n when the dimension p is fixed.

3.2. Computational Cost

Every time a branching constraint $z_i^{\pm} \leq 1$ or $z_i^{\pm} \geq 1$ is introduced, the feasible values for $\boldsymbol{\beta} \in \mathbb{R}^p$ are restricted: the regression coefficients are forced to be on one side of a hyperplane in the hyperplane arrangement induced by $\{\boldsymbol{x_i}^\top \boldsymbol{\beta} = y_i \pm \sqrt{2\mu}\}_{i=1}^n$. Every time a branching decision is incompatible with a region of the hyperplane arrangement, then the branch-and-bound node is pruned due to being infeasible for Problem (9). In contrast, with formulation (5), no branch-and-bound nodes are infeasible, and pruning only occurs due to bounding or by finding integer solutions. In Proposition 5 we show that, thanks to the possibility of fathoming by infeasibility, branch-and-bound algorithms run in time polynomial in n.

PROPOSITION 5. *If formulation* (9) *is solved via the branch-and-bound method where:*

- *all convex interval relaxations are solved to optimality at each node of the branch-and-bound tree,*
- *variable dichotomy is used for branching, that is, branching is performed only via disjunctions "$z_i^{\pm} \leq 0$ or $z_i^{\pm} \geq 1$", and*
- *only fractional variables are selected for branching,*

then the branch and bound tree has at most $\mathcal{O}\left(\min\{4^n, n^{p+1}\}\right)$ nodes.

Proof The computational cost of $\mathcal{O}\left(4^n\right)$ is obtained since there are $2n$ binary variables, and a binary tree of depth $2n$ has at most $2 \cdot 2^{2n} - 1$ nodes.

To prove the complexity of $\mathcal{O}\left(n^{p+1}\right)$, assume the worst-case scenario that the branch-and-bound algorithm never fathoms nodes by bounding, and pruning only occurs at integer or infeasible solutions; these integer or infeasible nodes are thus the only leaves of the trees. Observe that there is a one-to-one correspondence between feasible integer solutions of Porblem (9) and regions of the hyperplane arrangement induced by $\{\boldsymbol{x_i}^\top \boldsymbol{\beta} = y_i \pm \sqrt{2\mu}\}_{i=1}^n$. Thus, from Proposition 2 we find that there are at most $\mathcal{O}(n^p)$ leaves corresponding to integer feasible solutions.

We now count the leaves corresponding to infeasible solutions. Consider an arbitrary leaf node that was pruned by infeasibility, which we refer to as the *infeasible leaf*. Since we assumed the convex relaxations are solved to optimality, pruning by infeasibility is identified immediately, as soon as a subproblem is infeasible. In other words, the parent of the infeasible leaf has a feasible

continuous relaxation, that is, there exist $\bar{\boldsymbol{\beta}} \in \mathbb{R}^p$ satisfying all constraints (9a)-(9b) and additional branching constraints added to reach that node. Therefore, this parent has a feasible descendant; in particular, there is a (unique) descendant obtained by branching at each step according to $\bar{\boldsymbol{\beta}}$ until an integer solution is found. We say that this unique feasible descendant and the original infeasible node are *relatives* to each other. Finally, note that an integer node at depth q of the tree can have at most $q-1$ infeasible relatives, where the upper bound is attained in the case shown in Figure 3. Since the maximum depth is $2n$ and there are at most $\mathcal{O}(n^p)$ integer nodes, it follows that there are at most $\mathcal{O}(2n^{p+1})$ infeasible nodes, at most $\mathcal{O}(n^p + 2n^{p+1})$ leaves and twice as many nodes in a binary tree, proving the result. □

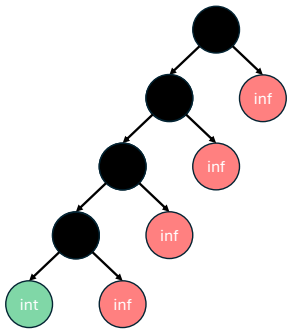

Figure 3 A depth 5 integer nodes with four relatives.

REMARK 1. Proposition 5 is stated in terms of the proposed formulation (9) for simplicity. However, we note that the same proof can be used for any correct formulation including the logic (7). In particular, using the big-M formulation (3) and big-M constraints (8) would have the same worst-case complexity (provided an adequate value of M can be found). The second and third assumption in Proposition 5 are imposed for simplicity, but the polynomial time complexity holds for most reasonable branching and variable selection strategies. The first assumption—relaxations being solved to optimality—on the other hand is crucial as the proof requires infeasible nodes to be identified immediately. ■

Proposition 5 suggests that formulation (9) is particularly effective in low-dimensional settings, even if the number of datapoints (and binary variables) is large. Moreover, Proposition 5 is concerned with a worst-case performance when no bounding occurs at all. In practice, branch-and-bound algorithms perform much better than their worst-case complexity, especially when using strong formulations. Thus, even if computational costs of the order $\mathcal{O}(n^{p+1})$ may appear

prohibitively large for $p \approx 3$ and moderately large values of n, we show in our computational experiments that the proposed method explores far fewer branch-and-bound nodes in practice and exhibits good computational performance in practice.

Formulation (9) can be used with most off-the-shelf MIO solvers by reformulating the perspective terms $\left(w_i^{\pm}\right)^2 / z_i^{\pm}$ as rotated cone constraints (e.g., see Aktürk et al. 2009, Günlük and Linderoth 2010) and linearizing the absolute value constraint (9b). We found in our computations that this direct approach (with Gurobi as the MIO solver) works quite well provided that n is small (in the low hundreds). However, we also observe that the direct approach can struggle if $n \geq 1000$, even if p is very small. Indeed, formulation (9) requires creating $4n$ variables and sets of linear and conic constraints. Solving a large number of convex optimization problems of this form requires a substantial computational effort. In Section 4 we describe a custom branch-and-bound framework to address this issue. Rather than dealing with additional $4n$ variables, we handle the node relaxation problems only in the space of the regression coefficients $\beta \in \mathbb{R}^p$, which is substantially smaller in dimension. This tailored approach avoids the computational overhead incurred by the auxiliary variables introduced in formulation (9), yielding significant practical efficiency gains, as demonstrated in Section 5.

REMARK 2. Huchette and Vielma (2019) propose, in a different context, MIO formulations for settings defined by hyperplane arrangements. However, their approach is not comparable with ours, nor can it be directly applied to the setting we consider in a practical manner. At a high level, they propose to consider every region defined by the arrangement and construct an ideal formulation for the complete problem. Such an approach would require enumeration of every region *a priori* (which is computationally demanding for $p \geq 3$), and may lead to a prohibitively large number of regions: for example, in a dataset we consider in our experiments with $n = 86$ and $p = 7$, the bound from Proposition 2 suggests we may need to enumerate 10^9 regions and then solve a convex problem with billions of decision variables. In contrast, formulation (9) does not lead to a strengthening of the continuous relaxation (Proposition 4), but instead ensures that enumeration of the regions of the arrangement occurs implicitly during branching. Thanks to the ability to bound, the number of regions considered is substantially less than the worst case scenario: for example, in the same instance with $n = 86$ and $p = 7$, Gurobi terminates after only $15,000$ nodes. ■

4. A Custom Branch-and-Bound Algorithm based on First Order Methods

In this section, we present a custom nonlinear branch-and-bound (BnB) framework for Problem (9), in which the the primal and dual bounds of the node relaxations are obtained via specialized first-order methods. Our work draws inspiration from the work of Hazimeh et al. (2021) for sparse linear models, but has important differences: Hazimeh et al. (2021) study ridge regularized least squares problem with a penalty on the number of nonzero regression coefficients. They employ standard binary branching on the binary variables (indicating whether a variable is zero or not), whereas we consider a nonconvex optimization problem expressed directly in terms of $\boldsymbol{\beta} \in \mathbb{R}^p$ — having projected out all other variables — and adopt a three-way branching scheme tailored to the structure of our formulation. For additional details on the general BnB framework, we refer the reader to Wolsey (2020, Chapter 7); details on the specific components of our customized procedure are discussed in Section 4.2. Formally, defining function $f : \mathbb{R}^p \to \mathbb{R}$ as

$$\begin{aligned} f(\boldsymbol{\beta}) \stackrel{\text{def}}{=} \min_{\substack{\boldsymbol{w},\boldsymbol{z} \\ \boldsymbol{w}^{\pm},\boldsymbol{z}^{\pm}}} \; & \frac{1}{2}\|\boldsymbol{y}\|_2^2 - \boldsymbol{y}^\top(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{w}) + \frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}\begin{pmatrix}\boldsymbol{\beta} \\ \boldsymbol{w}\end{pmatrix} + \sum_{i=1}^n \left(\mu z_i + \frac{d_i (w_i^+)^2}{z_i^+} + \frac{d_i (w_i^-)^2}{z_i^-}\right) \\ & \text{s.t. (9a), (9b), (9c)} \\ & \boldsymbol{w}, \boldsymbol{w}^-, \boldsymbol{w}^+ \in \mathbb{R}^n_+, \boldsymbol{z}, \boldsymbol{z}^-, \boldsymbol{z}^+ \in \{0,1\}^n, \end{aligned}$$

we observe that problem (9) is equivalent to $\min_{\boldsymbol{\beta}\in\mathbb{R}^p} f(\boldsymbol{\beta})$. Function $\boldsymbol{\beta} \mapsto f(\boldsymbol{\beta})$ is nonconvex, and our branch-and-bound procedure needs to compute, at each node, a lower bound on f restricted to the subregion defined by the node's branching decisions. In particular, these branching decisions can be described in terms of bounds $\boldsymbol{\ell}^{\pm}, \boldsymbol{u}^{\pm} \in \{0,1\}^n$ with $\boldsymbol{\ell}^{\pm} \leq \boldsymbol{u}^{\pm}$ on the binary variables $\boldsymbol{z}^{\pm}$. Given such bounds, we consider lower bounding functions inspired by the interval relaxation

$$\begin{aligned} f_R(\boldsymbol{\beta}; \boldsymbol{\ell}^{\pm}, \boldsymbol{u}^{\pm}) \stackrel{\text{def}}{=} \min_{\substack{\boldsymbol{w},\boldsymbol{z} \\ \boldsymbol{w}^{\pm},\boldsymbol{z}^{\pm}}} \; & \frac{1}{2}\|\boldsymbol{y}\|_2^2 - \boldsymbol{y}^\top(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{w}) + \frac{1}{2}\begin{pmatrix}\boldsymbol{\beta}^\top & \boldsymbol{w}^\top\end{pmatrix}\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{d}}\begin{pmatrix}\boldsymbol{\beta} \\ \boldsymbol{w}\end{pmatrix} + \sum_{i=1}^n \left(\mu z_i + \frac{d_i (w_i^+)^2}{z_i^+} + \frac{d_i (w_i^-)^2}{z_i^-}\right) \\ & \text{s.t. (9a), (9b), (9c)} \\ & \boldsymbol{\ell}^- \leq \boldsymbol{z}^- \leq \boldsymbol{u}^-, \boldsymbol{\ell}^+ \leq \boldsymbol{z}^+ \leq \boldsymbol{u}^+ \\ & \boldsymbol{w}, \boldsymbol{w}^-, \boldsymbol{w}^+ \in \mathbb{R}^n_+, \boldsymbol{z}, \boldsymbol{z}^-, \boldsymbol{z}^+ \in [0,1]^n. \end{aligned} \tag{11}$$

First, in Section 4.1, we describe explicit forms and relaxations of $f_R(\boldsymbol{\beta}; \boldsymbol{\ell}^{\pm}, \boldsymbol{u}^{\pm})$ based on augmented Lagrangians. Next, in Section 4.2, we describe the overall branching scheme, pruning conditions and other details for the BnB algorithm. In Section 4.3, we discuss how to solve node relaxations and efficiently compute dual bounds. Finally, in Section 4.4 we discuss additional details and computational improvements of the BnB algorithm.

4.1. Closed form expressions and differentiable lower bounds for f_R

We begin by showing that f_R admits a separable closed-form representation in terms of one-dimensional functions, which forms the basis of our subsequent algorithmic developments.

PROPOSITION 6. *For any $\boldsymbol{\beta} \in \mathbb{R}^p$ and any bounds $\boldsymbol{\ell}^{\pm}, \boldsymbol{u}^{\pm} \in \{0,1\}^n$ with $\boldsymbol{\ell}^{\pm} \leq \boldsymbol{u}^{\pm}$, the relaxation f_R defined in* (11) *can be written as*

$$f_R(\boldsymbol{\beta}; \boldsymbol{\ell}^{\pm}, \boldsymbol{u}^{\pm}) \;=\; \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 + \sum_{i=1}^{n} \phi\big(y_i - \boldsymbol{x_i}^{\top}\boldsymbol{\beta};\, \ell_i^-, \ell_i^+, u_i^-, u_i^+,\, d_i\big)\,, \tag{12}$$

where, for each $r \in \mathbb{R}$ and $d > 0$, the one-dimensional function ϕ is defined as

$$\phi(r; \ell^-, \ell^+, u^-, u^+, d) \;=\; \min_{\substack{w^-, w^+ \in \mathbb{R}_+ \\ z^-, z^+ \in \mathbb{R}_+}} \tfrac{1}{2}(r - w^+ + w^-)^2 + \mu(z^- + z^+) + \tfrac{d(w^+)^2}{z^+} + \tfrac{d(w^-)^2}{z^-} - d(w^+ - w^-)^2 \tag{13a}$$

$$\text{s.t. } w^- \geq \sqrt{2\mu}\, z^-, \quad w^+ \geq \sqrt{2\mu}\, z^+, \tag{13b}$$

$$|r - w^+ + w^-| \leq \sqrt{2\mu}\,(1 - z^- - z^+), \tag{13c}$$

$$\ell^- \leq z^- \leq u^-, \quad \ell^+ \leq z^+ \leq u^+, \quad z^- + z^+ \leq 1. \tag{13d}$$

Moreover, provided that $\mathbf{d}$ satisfies the criteria of Section 2.1.2, the function $\boldsymbol{\beta} \mapsto f_R(\boldsymbol{\beta}; \boldsymbol{\ell}^{\pm}, \boldsymbol{u}^{\pm})$ is convex.

The representation (12) follows from the observation that, for fixed $\boldsymbol{\beta}$, the inner minimization in (11) is separable across the per-datapoint variable groups $(w_i, z_i, w_i^-, w_i^+, z_i^-, z_i^+)$: constraints (9a)–(9c) and the bounds on $\boldsymbol{z}^{\pm}$ all decouple across i; adding and subtracting $\boldsymbol{w}^{\top}\mathrm{Diag}(\boldsymbol{d})\boldsymbol{w}$ to the objective and minimizing out w_i in closed form yields the per-datapoint subproblems (13) after the substitution $r = y_i - \boldsymbol{x_i}^{\top}\boldsymbol{\beta}$. Convexity of f_R is a direct consequence of the discussion in Section 2.1.2.

Proposition 7 provides a closed form expression of function ϕ when $\boldsymbol{\ell}^{\pm} = \mathbf{0}$ and $\boldsymbol{u}^{\pm} = \mathbf{1}$.

PROPOSITION 7. *If $0 < d < 1/2$, $\boldsymbol{\ell}^{\pm} = \mathbf{0}$ and $\boldsymbol{u}^{\pm} = \mathbf{1}$, then ϕ defined in* (13) *is given by:*

$$\phi(r; 0, 0, 1, 1, d) = \begin{cases} \dfrac{1}{2}r^2, & |r| \leq 2\sqrt{\mu d} \qquad (z^- + z^+ = 0) \\ \dfrac{-dr^2 + 2\sqrt{\mu d}|r| - 2\mu d}{1 - 2d}, & 2\sqrt{\mu d} < |r| < \sqrt{\dfrac{\mu}{d}}, \ (0 < z^- + z^+ < 1) \\ \mu, & |r| \geq \sqrt{\dfrac{\mu}{d}}, \qquad (z^- + z^+ = 1) \end{cases} \tag{14}$$

where we indicate in parenthesis the optimal values of variables (z^-, z^+). *Moreover, the result holds if constraints* (13c) *are removed from* (13).

Interestingly, the closed form solution reveals new interpretations of the convex relaxations of the perspective formulation (5) and proposed formulation (9): they are equivalent to using a special family of nonconvex loss functions that underestimate the capped quadratic loss ϕ_{cap}.

To simplify the notation, we will use the shorthand:

$$\bar{\phi}(r; d) \stackrel{\text{def}}{=} \phi(r; 0, 0, 1, 1, d).$$

Observe that the second statement of the proposition, stating that constraints (13c) are redundant, imply Proposition 4: at the root node, when all lower bounds are zero and upper bounds are one, the hyperplane arrangement constraints do not improve the strength of the perspective relaxation. Figure 4 illustrates function $r \mapsto \bar{\phi}(r; d)$ for different values of d. We observe that function $r \mapsto \bar{\phi}(r; d)$ underestimates the capped loss ϕ_{cap} and becomes a better approximation of ϕ_{cap} as $d \to 1/2$, and the two functions coincide for $|r| < 2\sqrt{\mu d}$ and $|r| \geq \sqrt{\mu/d}$. The function $r \mapsto \bar{\phi}(r; d)$ is differentiable: we provide explicit forms of its derivative in Appendix A.2. While function $r \mapsto \bar{\phi}(r; d)$ is nonconvex, we point out that f_R is convex as stated in Proposition 6 because the presence of the strongly convex term $\frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2$ offsets the nonconvexities introduced by $\bar{\phi}$.

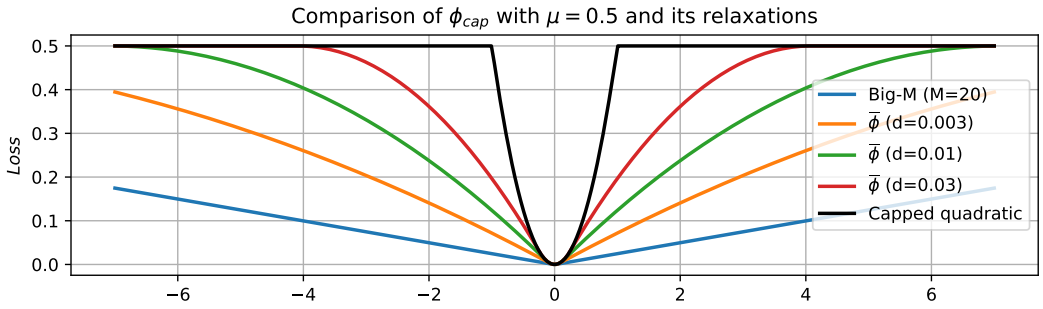

Figure 4 Comparison of loss functions for a single residual r: the original least trimmed loss ϕ_{cap} (before relaxation), the perspective relaxation $\bar{\phi}(r; d)$, and the analogous one-dimensional function obtained by applying the same projection argument to the standard Big-M relaxation of (9) in place of the perspective relaxation. The perspective relaxation provides a tighter convex approximation than the Big-M relaxation.

We conclude this section by deriving explicit closed-form expressions for ϕ at the remaining bound configurations. We also construct differentiable lower bounds for ϕ: as the closed-form expressions involve hard indicator constraints on the residual r, which effectively turn the minimization of relaxation (12) into a constrained optimization problem, preventing direct application

of gradient descent. To avoid this, we employ the augmented Lagrangian construction (Bertsekas 1982). Concretely, for a constrained problem of the form $\min_{r\in\mathbb{R}} g(r)$ subject to $c_j(r) \le 0$ for $j = 1,\dots,m$, the corresponding augmented Lagrangian with multipliers $\boldsymbol{\nu} \in \mathbb{R}^m_+$ and penalty parameter $\rho \ge 0$ is $\mathcal{L}_\rho(r;\boldsymbol{\nu}) = g(r) + \sum_{j=1}^m \nu_j c_j(r) + \frac{\rho}{2}\sum_{j=1}^m [c_j(r)]_+^2$, where $[\cdot]_+ \stackrel{\text{def}}{=} \max\{\cdot, 0\}$. Whenever g and the c_j's are differentiable, so is $\mathcal{L}_\rho(\cdot;\boldsymbol{\nu})$; moreover, by weak duality, $\min_r \mathcal{L}_\rho(r;\boldsymbol{\nu}) \le \min_r\{g(r) : c_j(r) \le 0, \forall j\}$ for any $\boldsymbol{\nu} \ge \mathbf{0}$ and $\rho \ge 0$, so that $\mathcal{L}_\rho(\cdot;\boldsymbol{\nu})$ serves as a differentiable lower-bounding function for the constrained problem. Applying this construction to each of the constrained forms of ϕ that arise during branching yields the following proposition.

PROPOSITION 8. *Let $\delta_S(\cdot)$ denote the indicator function of a set S, taking value 0 on S and $+\infty$ elsewhere. For any $d > 0$ and any $\nu, \nu_1, \nu_2, \rho \ge 0$, the function ϕ defined in* (13) *admits the following closed-form expressions, each accompanied by a differentiable lower bound (in r) obtained from the augmented Lagrangian construction described above:*

- $\phi(r;0,0,0,0,d) = \frac{1}{2}r^2 + \delta_{[-\sqrt{2\mu},\sqrt{2\mu}]}(r)$ *and a differentiable lower bound is given by*

$$\bar{\phi}_0(r;\nu_1,\nu_2,\rho) = \frac{1}{2}r^2 + \nu_1\left(-r - \sqrt{2\mu}\right) + \nu_2\left(r - \sqrt{2\mu}\right) + \frac{\rho}{2}\left(\left[-r - \sqrt{2\mu}\right]_+^2 + \left[r - \sqrt{2\mu}\right]_+^2\right).$$

- $\phi(r;1,0,1,0,d) = \mu + \delta_{[-\infty,-\sqrt{2\mu}]}(r)$ *and a differentiable lower bound is given by*

$$\bar{\phi}_-(r;\nu,\rho) = \mu + \nu\left(r + \sqrt{2\mu}\right) + \frac{\rho}{2}\left[r + \sqrt{2\mu}\right]_+^2.$$

- $\phi(r;0,1,0,1,d) = \mu + \delta_{[\sqrt{2\mu},\infty]}(r)$ *and a differentiable lower bound is given by*

$$\bar{\phi}_+(r;\nu,\rho) = \mu + \nu\left(-r + \sqrt{2\mu}\right) + \frac{\rho}{2}\left[-r + \sqrt{2\mu}\right]_+^2.$$

The functions $\bar{\phi}_0, \bar{\phi}_-, \bar{\phi}_+$ play a key role in the remainder of the paper: in Section 4.2, they are used to construct the augmented Lagrangian of the full node relaxation (12), and in Section 4.3 this Lagrangian is in turn the objective minimized by our augmented Lagrangian method (ALM) for computing dual bounds at each BnB node.

4.2. Description of the BnB method

We now describe the main components of our custom branch-and-bound algorithm.

Root node The BnB process begins by solving the interval relaxation of problem (9) at the root node. From (12) we find that the relaxation is given by

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 + \sum_{i=1}^n \bar{\phi}(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; d_i), \tag{15}$$

where function $\bar{\phi}$ is explictly described in Proposition 7. All functions involved are differentiable, thus (15) can be solved via standard first-order methods. In our approach, we solve it approximately via gradient descent with line search. We describe our specific implementation and the computation of valid dual bounds from approximate solutions in Section 4.3.

Branching variable selection Once a root (or node) relaxation has been solved, typical branch-and-bound algorithms for mixed-integer optimization would then select a fractional integer variable to branch on. In our case, the binary variables are not defined explicitly, but appear implicitly in the definition of function ϕ in (13). In particular, function $\bar{\phi}(\cdot; d)$ underestimates the capped loss $\phi_{\text{cap}}(\cdot)$, and is a strict underestimator if and only if the binary variables appearing in (13) are fractional. Thus, given an optimal solution $\bar{\boldsymbol{\beta}} \in \mathbb{R}^p$ to Problem (15), the algorithm selects an index $i \in [n]$ such that $\bar{\phi}(y_i - \boldsymbol{x_i}^\top \bar{\boldsymbol{\beta}}; d_i) < \phi_{\text{cap}}(y_i - \boldsymbol{x_i}^\top \bar{\boldsymbol{\beta}})$.

Branching scheme The definition of ϕ in Problem (13) involves two binary variables (z^- and z^+) and three feasible configurations ($z^- = z^+ = 0$; $z^- = 1$, $z^+ = 0$; and $z^- = 0$, $z^+ = 1$; note that $z^- = z^+ = 1$ is infeasible). Thus, instead of using the standard variable dichotomy for branching, we use three-way branching for our custom branch-and-bound. After fixing the branching variables, the child node's relaxation objective is obtained from the parent's by replacing $\bar{\phi}(y_i - \boldsymbol{x_i}^\top \bar{\boldsymbol{\beta}}; d_i)$ in the parent node's objective with the appropriate differentiable lower bounds described in Proposition 8, as summarized in Table 1.

Disjunction	Replace $\bar{\phi}(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}; d_i)$ with...
$z_i^- = 0, z_i^+ = 0$	$\bar{\phi}_0(\cdot; \nu_1, \nu_2, \rho)$
$z_i^- = 1, z_i^+ = 0$	$\bar{\phi}_-(\cdot; \nu, \rho)$
$z_i^- = 0, z_i^+ = 1$	$\bar{\phi}_+(\cdot; \nu, \rho)$

Table 1: Subproblem construction

Recursion The algorithm proceeds recursively, applying the same branching scheme to each newly created node. To describe the relaxation solved at a generic node, let $\mathcal{F}_0, \mathcal{F}_-, \mathcal{F}_+ \subseteq [n]$ denote the disjoint sets of indices recording the branching decisions made along the path from the root to the current node: $i \in \mathcal{F}_0$ if $z_i^- = z_i^+ = 0$ has been fixed, $i \in \mathcal{F}_-$ if $z_i^- = 1$ has been fixed, and $i \in \mathcal{F}_+$ if $z_i^+ = 1$ has been fixed. Following the construction summarized in Table 1, the term $\bar{\phi}(\cdot; d_i)$ associated with each branched index i is replaced by the corresponding differentiable lower bound from Proposition 8, parameterized by dual multipliers $\boldsymbol{\nu}$ and a penalty parameter $\rho \geq 0$. This yields

the node objective

$$L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 + \sum_{i\in\mathcal{F}_0} \bar{\phi}_0(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; (\nu_i)_1, (\nu_i)_2, \rho) + \sum_{i\in\mathcal{F}_-} \bar{\phi}_-(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; \nu_i, \rho) + \sum_{i\in\mathcal{F}_+} \bar{\phi}_+(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; \nu_i, \rho) + \sum_{i\notin(\mathcal{F}_0\cup\mathcal{F}_-\cup\mathcal{F}_+)} \bar{\phi}(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; d_i). \tag{16}$$

where $\boldsymbol{\nu} \in \mathbb{R}_+^{2|\mathcal{F}_0|+|\mathcal{F}_-|+|\mathcal{F}_+|}$ collects all dual multipliers associated with the branched indices. By Proposition 8, each function $\bar{\phi}_0, \bar{\phi}_-, \bar{\phi}_+$ is precisely the augmented Lagrangian of the corresponding constrained function ϕ. Consequently, $L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu})$ is the augmented Lagrangian of (12) in which the bounds $(\boldsymbol{\ell}^\pm, \boldsymbol{u}^\pm)$ encode the branching decisions in $\mathcal{F}_0, \mathcal{F}_-, \mathcal{F}_+$. Since the objective in (12) is convex (in $\boldsymbol{\beta}$) and all constraints are linear, standard augmented Lagrangian duality (Bertsekas 1982, Rockafellar 1976) guarantees that

$$\bar{\zeta} = \max_{\boldsymbol{\nu}\geq\mathbf{0}} \min_{\boldsymbol{\beta}} L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu}), \tag{17}$$

where $\bar{\zeta}$ denotes the optimal value of the node relaxation (12). Solving the node thus reduces to (approximately) solving this max-min problem; the procedure is described in detail in the next subsection, and produces a primal iterate $\bar{\boldsymbol{\beta}}$ together with a valid lower bound $\bar{\zeta}'$ on $\bar{\zeta}$. The node is then processed according to the standard BnB rules: if $\bar{\zeta}'$ exceeds the current incumbent, the node is pruned by bounding; if $\bar{\phi}(y_i - \boldsymbol{x_i}^\top\bar{\boldsymbol{\beta}}; d_i) = \phi_{\text{cap}}(y_i - \boldsymbol{x_i}^\top\bar{\boldsymbol{\beta}})$ for all $i \in [n] \setminus (\mathcal{F}_0 \cup \mathcal{F}_- \cup \mathcal{F}_+)$, the node is pruned by integrality and the incumbent is updated (in our implementation, we adopt a best-bound strategy, so this case arises only at the final node explored, if at all); otherwise, an index $i \in [n] \setminus (\mathcal{F}_0 \cup \mathcal{F}_- \cup \mathcal{F}_+)$ with $\bar{\phi}(y_i - \boldsymbol{x_i}^\top\bar{\boldsymbol{\beta}}; d_i) < \phi_{\text{cap}}(y_i - \boldsymbol{x_i}^\top\bar{\boldsymbol{\beta}})$ is selected for branching, and the recursion continues.

REMARK 3. Since our BnB implementation does not solve the node relaxation problem to optimality (due to the nature of the first order methods which can be slow to obtain high accuracy solutions), the first condition of Proposition 5 is no longer satisfied. As a consequence, the proposed algorithm is no longer guaranteed to explore at most $\mathcal{O}(n^{p+1})$ nodes. In practice however, when optimally solving the convex interval relaxation at a node becomes expensive, our approach is typically still able to prune the node by bounding, so the total number of nodes explored remains largely unaffected. This is confirmed empirically in Tables 8–9 of Section 5, which compare the node counts of our solver against those of Gurobi applied directly to Problem (9).

4.3. Node relaxations and dual bounds

At each node of the branch-and-bound algorithm, we compute dual bounds for Problem (17). Observe that for any fixed $\bar{\boldsymbol{\nu}}$

$$\bar{\zeta}(\bar{\boldsymbol{\nu}}) = \min_{\boldsymbol{\beta}} L_\rho(\boldsymbol{\beta}; \bar{\boldsymbol{\nu}}) \tag{18}$$

is a lower bound on $\bar{\zeta}$. Function $L_\rho(\boldsymbol{\beta}; \boldsymbol{\nu})$ is differentiable as a function of $\boldsymbol{\beta}$, and the gradient is

$$\begin{aligned}\nabla_{\boldsymbol{\beta}} L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu}) = \lambda\boldsymbol{\beta} - \sum_{i\in\mathcal{F}_0} \boldsymbol{x_i}\bar{\phi}_0'(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; (\nu_i)_1, (\nu_i)_2, \rho) - \sum_{i\in\mathcal{F}_-} \boldsymbol{x_i}\bar{\phi}_-'(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; \nu_i, \rho) \\ - \sum_{i\in\mathcal{F}_+} \boldsymbol{x_i}\bar{\phi}_+'(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; \nu_i, \rho) - \sum_{i\notin(\mathcal{F}_0\cup\mathcal{F}_-\cup\mathcal{F}_+)} \boldsymbol{x_i}\bar{\phi}'(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}; d_i)\end{aligned} \tag{19}$$

where functions $\bar{\phi}'$ $\bar{\phi}_0'$, $\bar{\phi}_-'$, $\bar{\phi}_+'$ denote the derivatives, which admit a closed-form expression and are continuous with respect to $\boldsymbol{\beta}$ (see Appendix A.2). In the proposed approach, we obtain a solution to (18) via gradient descent with line search. Note that $\bar{\zeta}(\bar{\boldsymbol{\nu}})$ is a lower bound only if the associated problem is solved to optimality. However, first order methods can be slow to converge to high-precision solutions. Therefore, we need to consider methods to get reliable lower bounds from an approximate solution, as we discuss next.

We recall that for any $\tilde{\lambda}$-strongly convex function $\boldsymbol{\beta} \mapsto \Phi(\boldsymbol{\beta})$ where $\Phi(\boldsymbol{\beta})$ is differentiable, the optimal value satisfies

$$\min_{\boldsymbol{\beta}} \Phi(\boldsymbol{\beta}) \geq \Phi(\bar{\boldsymbol{\beta}}) - \frac{1}{2\tilde{\lambda}}\|\nabla\Phi(\bar{\boldsymbol{\beta}})\|_2^2 \tag{20}$$

for any point $\bar{\boldsymbol{\beta}}$. As mentioned in Section 2.1.2, the perspective reformulation and relaxations can be ensured to be $\tilde{\lambda}$-strongly convex for any $\tilde{\lambda} \in (0, \lambda)$, see (6) (note that larger values of the strong convexity parameter $\tilde{\lambda}$ come at the expense of weaker convex relaxations and thus less effective pruning of the BnB algorithm). Using this observation, we obtain the following proposition.

PROPOSITION 9. *Let $(\bar{\boldsymbol{\beta}}, \bar{\boldsymbol{\nu}})$ be any primal-dual pair of Problem* (17)*, then the inequality*

$$\bar{\zeta}(\bar{\boldsymbol{\nu}}) \geq L_\rho(\bar{\boldsymbol{\beta}}, \bar{\boldsymbol{\nu}}) - \frac{\|\nabla_{\boldsymbol{\beta}} L_\rho(\bar{\boldsymbol{\beta}}, \bar{\boldsymbol{\nu}})\|_2^2}{2\tilde{\lambda}} \tag{21}$$

holds true.

The proof of Proposition 9 is provided in Appendix A.3. Using Proposition 9, we can compute dual bounds for the convex relaxations in the BnB algorithm efficiently from any iterate of a first-order method and its gradient, even when the relaxation is solved approximately.

Building on this result, we solve Problem (17) via the augmented Lagrangian method, which alternates between minimizing $L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu})$ over the primal variables $\boldsymbol{\beta}$ with multipliers $\boldsymbol{\nu}$ fixed, and updating $\boldsymbol{\nu}$. In the primal step, we minimize $L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu})$ over $\boldsymbol{\beta}$ using gradient descent with an Armijo backtracking line search, as detailed in Algorithm 1. The complete augmented Lagrangian procedure for solving (17) is summarized in Algorithm 2. Here, Proposition 9 is applied at each iteration of both the inner and outer loops to obtain a valid dual bound from the current iterate. As the multipliers $\boldsymbol{\nu}$ are progressively updated through the outer loop, the quality of the resulting dual bounds improves.

Algorithm 1 Gradient descent with Armijo backtracking for minimizing $L_\rho(\cdot, \boldsymbol{\nu})$

Input: Initial point $\boldsymbol{\beta}$, dual values $\boldsymbol{\nu}$, initial step size t_0, Armijo parameter $\alpha \in (0, 1)$, backtracking factor $\gamma \in (0, 1)$, tolerance ϵ

$t \leftarrow t_0$
repeat
  Compute $\boldsymbol{g} = \nabla_{\boldsymbol{\beta}} L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu})$
  while $L_\rho(\boldsymbol{\beta} - t\boldsymbol{g}, \boldsymbol{\nu}) > L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu}) - \alpha t \|\boldsymbol{g}\|_2^2$ **do**
    $t \leftarrow \gamma t$
  end while
  $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} - t\boldsymbol{g}$
  $t \leftarrow t/\gamma$ ▷ warm start for next iteration
  Compute dual bound D using Proposition 9
until $(L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu}) - D)/L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu}) \leq \epsilon$
return $\boldsymbol{\beta}$

4.4. Implementation Details

We discuss several practical techniques to make our proposed BnB approach computationally efficient. Additionally, we discuss how our work differs from L0BnB (Hazimeh et al. 2021), who also use a custom BnB framework with fist order methods to solve the node relaxations for the sparse linear regression which is different from the robust statistics problem we consider here.

Algorithm 2 Augmented Lagrangian Method for solving the relaxation problem

Input: Initial solution $\boldsymbol{\beta}^{(0)}$, multipliers $\boldsymbol{\nu}^{(0)} \geq \mathbf{0}$, penalty $\rho > 0$, tolerances $\epsilon_{\text{in}}, \epsilon_{\text{out}}$

for $k = 0, 1, 2, \ldots$ **do**
  $\boldsymbol{\beta}^{(k+1)} \leftarrow$ minimize $L_\rho(\boldsymbol{\beta}, \boldsymbol{\nu}^{(k)})$ using Algorithm 1 with tolerance ϵ_{in}, starting from $\boldsymbol{\beta}^{(k)}$
  Update dual multipliers: $\nu_j^{(k+1)} = \max\{0, \nu_j^{(k)} + \rho\, c_j(\boldsymbol{\beta}^{(k+1)})\}$ for all $j \in \mathcal{C}$
    $\nu_i^{(k+1)} = \max\left\{0, \nu_i^{(k)} + \rho\left(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(k+1)} + \sqrt{2\mu}\right)\right\}$ for all $i \in \mathcal{F}_-$
    $\nu_i^{(k+1)} = \max\left\{0, \nu_i^{(k)} + \rho\left(-y_i + \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(k+1)} + \sqrt{2\mu}\right)\right\}$ for all $i \in \mathcal{F}_+$
    $(\nu_i)_1^{(k+1)} = \max\left\{0, (\nu_i)_1^{(k)} + \rho\left(-y_i + \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(k+1)} - \sqrt{2\mu}\right)\right\}$ for all $i \in \mathcal{F}_0$
    $(\nu_i)_2^{(k+1)} = \max\left\{0, (\nu_i)_2^{(k)} + \rho\left(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(k+1)} - \sqrt{2\mu}\right)\right\}$ for all $i \in \mathcal{F}_0$
  Compute dual bound $D^{(k+1)}$ using Proposition 9
  if $(D^{(k+1)} - D^{(k)})/D^{(k)} \leq \epsilon_{\text{out}}$ **then**
    break
  end if
end for
return $\boldsymbol{\beta}^{(k+1)}$, dual bound $D^{(k+1)}$

BnB configuration For node selection, we employ a best-bound search strategy (Linderoth and Savelsbergh 1999), which selects the open node with the smallest lower bound as the next node to explore. For selecting the branching variable, we branch on the most fractional variable. Note that the integer variables $\boldsymbol{z}^-, \boldsymbol{z}^+$ are not explicitly defined, but can be computed implicitly: choosing the index i whose $z_i^- + z_i^+$ is closest to $1/2$ is equivalent to choosing

$$i^\star \in \arg\min_i \left|\, |r_i| - r_0 \,\right|, \quad \text{where } r_0 := \left(\tfrac{1}{2} + d\right)\sqrt{\frac{\mu}{d}}, \quad r_i = y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}.$$

Equivalently, since $\phi(r)$ is strictly increasing in $|r|$ when $|r| \in [0, \sqrt{\mu/d}]$, we find

$$i^\star \in \arg\min_i \left|\, \phi(r_i) - \phi_0 \,\right|, \qquad \phi_0 := \phi(r_0) = \frac{\mu\left(\frac{3}{4} - d - d^2\right)}{1 - 2d}.$$

The BnB procedure terminates when the gap between bounds falls below a predetermined tolerance (1% in our experiments), or when all nodes have been explored or pruned.

Hyperparameter selection In our implementation, we set the quadratic penalty term ρ to 100 if $n \geq 300$ and equal to 5 otherwise. The strong-convexity parameter is set to $\tilde{\lambda} = 0.1\lambda$. We use Armijo type line search with initial step size $t_0 = 1$, sufficient-decrease parameter $\alpha = 10^{-4}$, and backtracking factor $\gamma = 0.5$. We set tolerances $\epsilon_{\text{in}} = 10^{-4}$ and $\epsilon_{\text{out}} = 10^{-4}$ in Algorithm 2.

We found the method to be fairly robust to the choice of hyperparameters: ρ was selected from $\{1, 5, 10, 50, 100, 500\}$ on 10 synthetic datasets of various sizes; the remaining hyperparameters were not tuned.

Warm starting Rather than solving the node relaxation (17) from scratch, we warm start Algorithm 2 using the parent node's solution. Specifically, $\boldsymbol{\beta}^{(0)}$ is initialized to the final primal iterate of the parent node. For each dual multiplier $\nu_i^{(0)}$, if the corresponding constraint is also present in the parent node, we inherit its final dual value; otherwise, we set $\nu_i^{(0)} = 0$. Since a child node differs from its parent only by fixing a single binary variable z_i, the parent's solution typically provides a high-quality starting point, substantially reducing the number of iterations required for convergence in practice.

Selective upper bound computation At any node of the BnB tree, we can use heuristics similar to those discussed in Section 2.3 based on alternating minimization to compute upper bounds for Problem (9). We provide a precise description of the heuristics in Appendix A.4. As computing incumbent solutions at every node is expensive, we compute these upper bounds only at the root node and nodes whose depth is a multiple of a constant (10 in our experiments). This reduces computational overhead and cost while maintaining effective pruning.

Early pruning. Since dual bounds via Proposition 9 are already computed at every iteration of Algorithms 1 and 2, we can monitor them on the fly: as soon as the dual bound exceeds the current incumbent, we terminate the inner solver and prune the node immediately. This step is critical for the efficiency of our BnB procedure, as the original relaxation (12) at a given node may be infeasible, in which case Algorithm 2 produces a sequence of dual bounds $D^{(k)}$ diverging to infinity. Without early pruning, the algorithm would expend substantial effort driving $D^{(k)}$ toward infinity on a node that can already be safely discarded. We note that the use of heuristics at the root node guarantees that an incumbent solution is always available for this comparison.

Accelerated gradient computation We use Numba (Lam et al. 2015), a just-in-time compiler for Python, to accelerate the gradient evaluation in (19).

Relation to L0BnB At a high level, our proposed BnB algorithm extends L0BnB (Hazimeh et al. 2021)—a BnB framework originally designed for ℓ_0-penalized least squares regression—to the outlier detection problem (1). Several key modifications are needed to address the distinct challenges posed by (1). First, we build our BnB on an enhanced MIO formulation (9) and derive a new form of the continuous relaxation (Section 4.1). Second, we develop a three-way branching scheme that efficiently encodes the status of each observation—inlier or outlier, with positive or

negative residual (Section 4.2). Third, the node relaxations appearing in L0BnB are in the composite form (without constraints) and coordinate descent procedures are used to compute solutions to these problems. In this work, the relaxation subproblems are more complicated as they have constraints, and we use augmented Lagrangian approaches and also introduce a new procedure for extracting valid dual bounds (Section 4.3).

5. Experiments

We evaluate our proposed BnB solver against commercial MIO solvers on both synthetic and real datasets. Our experiments focus on three aspects: computational efficiency compared to existing methods, sensitivity to problem parameters, and the value of exact optimization over heuristic approaches.

5.1. Experimental Setup

Solver settings: We implement our BnB solver in Python with gradient computations accelerated using Numba (Lam et al. 2015). We compare against Gurobi (Gurobi Optimization, LLC 2022) and Mosek (ApS 2022) applied to the Big-M formulation (3) (M), the perspective formulation (5) (P), and the strengthened perspective formulation with hyperplane-arrangement (9) (H). Accordingly, BnB (P) and BnB (H) denotes our solver on perspective formulation and hyperplane-arrangement formulation, respectively. And Gurobi/Mosek (M, P, H) denotes the commercial solvers on the same formulations. For all solvers, we define the relative optimality gap as $(UB - LB)/UB$, where UB is the incumbent objective value and LB is the best dual bound. We terminate when the gap falls below 1% or when the runtime exceeds 1 hour. All experiments are conducted on a computing cluster, utilizing an AMD EPYC 9474F machine with 10 CPU cores and 40GB RAM.

Synthetic data generation: Following Hazimeh and Mazumder (2020), we generate the data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ by drawing each row from a multivariate Gaussian $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_p)$. The true coefficient vector $\boldsymbol{\beta}^\dagger \in \mathbb{R}^p$ is generated with entries drawn uniformly from $[0, 1]$. We generate clean responses as $\boldsymbol{y}_{\text{clean}} = \boldsymbol{X}\boldsymbol{\beta}^\dagger + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}_n)$ and σ is chosen to achieve a target signal-to-noise ratio $\mathrm{SNR} = \mathrm{Var}(\boldsymbol{X}\boldsymbol{\beta}^\dagger)/\sigma^2$.

To introduce outliers, we contaminate a subset of observations by adding heavy-tailed noise to their responses. Specifically, for each contaminated observation i, we set $y_i = y_{\text{clean},i} + \delta \cdot \sigma_y \cdot Z_i$, where $\sigma_y = \mathrm{std}(\boldsymbol{y}_{\text{clean}})$, δ controls the outlier magnitude, and Z_i is drawn from a t-distribution with 3 degrees of freedom. Unless otherwise specified, we set $\mathrm{SNR} = 50$, $\delta = 10$, and introduce 10 outliers in each dataset.

Real datasets. We evaluate solver performance on 13 benchmark regression datasets: 7 smaller datasets that have been used in prior work on the LTS problem (Gómez and Neto 2025) and 6 larger datasets obtained from the OpenML repository (Vanschoren et al. 2014). The dimensions of each dataset are summarized in the headers of the corresponding results tables.

Parameter selection: We set the ridge regularization parameter as $\lambda = \lambda_0 \cdot \mathrm{Mean}(\mathrm{Diag}(\boldsymbol{X}^\top \boldsymbol{X}))$, where $\lambda_0 = 0.01$ for synthetic datasets and $\lambda_0 = 0.2$ for real datasets, unless otherwise specified. The outlier penalty μ is chosen so that the model identifies exactly the number of outliers present in the synthetic dataset (or the specified number for real datasets).

5.2. Comparison on Synthetic Datasets

5.2.1. Varying dataset size Table 2 compares runtime performance across different numbers of samples $n \in \{1000, 2000, 5000\}$ and features $p \in \{10, 20, 50\}$. The results reveal several key findings.

Method	$n = 1000$			$n = 2000$			$n = 5000$		
	p=10	p=20	p=50	p=10	p=20	p=50	p=10	p=20	p=50
*BnB (P)	(10.0%, 0)	(8.8%, 1)	(7.0%, 2)	(14.6%, 0)	(14.3%, 0)	(13.8%, 0)	(17.5%, 0)	(17.7%, 0)	(18.4%, 0)
*BnB (H)	2.5	3.6	12.9	6.3	10.0	53.4	35.7	49.2	750.6
Gurobi (M)	(33.3%, 0)	(31.5%, 0)	(25.6%, 0)	(53.8%, 0)	(49.9%, 0)	(51.0%, 0)	(65.0%, 0)	(64.6%, 0)	(64.2%, 0)
Gurobi (P)	(5.3%, 2)	(4.9%, 3)	(4.5%, 6)	(13.7%, 0)	(14.1%, 0)	(15.3%, 0)	(19.2%, 0)	(18.8%, 0)	(19.4%, 0)
*Gurobi (H)	358.5	414.0	766.4	(1.5%, 8)	2301.2	(6.6%, 5)	(14.0%, 0)	(16.1%, 0)	(19.6%, 0)
Mosek (M)	(32.3%, 0)	(30.1%, 0)	(16.9%, 0)	(47.4%, 0)	(47.0%, 0)	(45.4%, 0)	(58.9%, 0)	(58.6%, 0)	(58.8%, 0)
Mosek (P)	(7.5%, 0)	(7.5%, 0)	(4.7%, 1)	(16.1%, 0)	(15.4%, 0)	(15.0%, 0)	(19.4%, 0)	(20.0%, 0)	(20.9%, 0)
*Mosek (H)	(8.6%, 0)	(1.0%, 2)	(1.8%, 0)	(12.5%, 0)	(11.0%, 0)	(18.9%, 0)	(20.1%, 0)	(20.1%, 0)	(21.3%, 0)

Table 2 Runtime performance on synthetic datasets, averaged over 10 random seeds. Methods marked with * are proposed in this work. Methods: (M) Big-M, (P) perspective, (H) hyperplane arrangement formulation. Numeric values indicate solve time in seconds when all 10 instances are solved within 1 hour. Otherwise, we report (gap, count) where gap is the average optimality gap and count is the number of instances solved.

Benchmarking methods the Big-M formulation (M) yields the weakest relaxation bounds, with optimality gaps exceeding 30% for smaller instances and over 60% for larger ones. The perspective formulation (P) significantly improves upon Big-M, reducing gaps to 5–20% depending on problem size – these results are consisting with those reported by Gómez and Neto (2025).

Hyperplane arrangement with off-the-shelf solvers The hyperplane arrangement formulation (H) allows for more aggresive pruning of the branch-and-bound tree. When used with Gurobi, problems with $n = 1000$ can now be solved to optimality within minutes, while just using the

perspective reformulation results in 5% gaps. Gurobi is also able to solve more instances and reduce optimality gaps in problems with $n = 2,000$, but the benefits seem to disappear in problems with $n = 5,000$. The improvements are far less notable with Mosek, and in many cases the use of hyperplane arrangement actually degrades performance. These findings suggest that Gurobi is better equipped to handling problems with heavy constraints.

Hyperplane arrangement with the proposed BnB solver The proposed BnB (H) consistently outperforms all commercial solvers, achieving up to 100× speedup over Gurobi (H) on comparable instances. This efficiency stems from our reformulation of the relaxation problem in the $\boldsymbol{\beta}$-space, which involves only p variables and has low complexity dependence on n. For problems with $n = 1000$, the method solves instances in seconds while Gurobi requires several minutes. For the largest instances ($n = 5000$, $p = 50$), BnB (H) completes in approximately 12 minutes, whereas competing methods fail to achieve the 1% optimality gap within the time limit. Notably, even without the additional hyperplane arrangement constraints, BnB (P) achieves comparable performance to Gurobi (P) for large n, confirming the method's scalability with sample size.

Additional results with smaller datasets We also examine solver performance on smaller datasets. To maintain problem difficulty and clearly distinguish solver performance, we set $\mathrm{SNR} = n/25$ and $\delta \in \{5, 6, 9\}$ for $n \in \{50, 100, 200\}$, respectively. Table 3 presents the results.

Method	$n=50$			$n=100$			$n=200$		
	p=5	p=10	p=20	p=5	p=10	p=20	p=5	p=10	p=20
Gurobi (P)	2.5	2.2	4.2	21.9	26.3	19.2	447.0	87.5	97.0
*Gurobi (H)	2.0	2.8	7.1	7.5	23.3	33.3	27.2	39.8	77.9
*BnB (H)	2.0	3.9	34.5	10.5	12.5	11.9	9.5	9.0	50.0

Table 3 Runtime performance on small datasets, averaged over 10 random seeds. Methods marked with * are proposed in this work. Numeric values indicate solve time in seconds.

From Table 3, we observe that Gurobi (H) is highly efficient when the dataset scale is very small ($n = 50$), solving instances in 2–7 seconds and outperforming BnB (H). However, Gurobi's runtime increases as n grows. For $n \geq 100$, BnB (H) achieves better performance in most of the settings considered.

5.2.2. Ablation studies We investigate how various problem parameters affect solver performance. We fix $n = 2000$ and $p = 20$, adopt the parameter settings described in Section 5.1, and vary one parameter at a time to conduct the ablation. Table 4 examines the impact of regularization

λ_0	Gurobi (P)	*Gurobi (H)	*BnB (H)
0.0005	(66.3%, 0)	(61.7%, 0)	(32.4%, 0)
0.001	(57.9%, 0)	(53.6%, 0)	(19.1%, 0)
0.002	(46.4%, 0)	(43.4%, 0)	(5.2%, 4)
0.005	(27.8%, 0)	(18.2%, 1)	34.7
0.01	(14.3%, 0)	(1.2%, 8)	10.3
0.02	(4.9%, 0)	1627.2	4.3
0.05	453.3	731.6	0.9

# outliers	Gurobi (P)	*Gurobi (H)	*BnB (H)
5	(9.9%, 3)	(1.1%, 9)	7.3
10	(14.2%, 0)	(1.0%, 9)	9.2
15	(15.5%, 0)	(5.7%, 5)	(2.8%, 9)
20	(18.5%, 0)	(10.3%, 4)	(4.7%, 8)
25	(16.4%, 0)	(8.8%, 5)	(4.2%, 8)
30	(17.5%, 0)	(12.3%, 4)	(4.4%, 6)

Table 4 Impact of regularization strength (left) and number of outliers (right). Numeric values indicate solve time in seconds when all 10 instances are solved within 1 hour. Otherwise, we report (gap, count) where gap is the average optimality gap and count is the number of instances solved.

strength λ_0 (the ridge is defined as $\lambda = \lambda_0 \cdot \mathrm{Mean}(\mathrm{Diag}(\boldsymbol{X}^\top \boldsymbol{X}))$) and the number of outliers, while Table 5 analyzes the effect of signal-to-noise ratio and outlier magnitude δ.

From Table 4, we observe that stronger regularization ($\lambda_0 \geq 0.01$) significantly improves tractability for all methods, while weak regularization ($\lambda_0 \leq 0.002$) makes the problem harder due to weaker relaxations. The table also shows that problem difficulty increases with the number of outliers. The BnB solver handles up to 10 outliers efficiently, solving instances in under 10 seconds. As the outlier count increases beyond 15, performance degrades for all methods, though BnB (H) consistently achieves smaller optimality gaps.

SNR	Gurobi (P)	*Gurobi (H)	*BnB (H)
5.0	(52.7%, 0)	(52.2%, 0)	(45.0%, 0)
10.0	(45.7%, 0)	(44.8%, 0)	(34.8%, 0)
20.0	(32.7%, 0)	(33.6%, 0)	(13.1%, 1)
30.0	(24.1%, 0)	(18.5%, 1)	(1.3%, 8)
40.0	(18.4%, 0)	(1.0%, 9)	18.9
50.0	(14.3%, 0)	(1.2%, 8)	9.4
100.0	(2.3%, 4)	1268.1	3.6

δ	Gurobi (P)	*Gurobi (H)	*BnB (H)
2.0	(40.8%, 0)	(40.7%, 0)	(36.9%, 0)
4.0	(36.9%, 0)	(36.7%, 0)	(31.0%, 0)
6.0	(29.2%, 0)	(29.4%, 0)	(15.5%, 0)
8.0	(22.0%, 0)	(11.3%, 2)	474.4
10.0	(14.3%, 0)	(1.7%, 8)	9.3
15.0	(2.4%, 5)	951.2	3.3

Table 5 Impact of signal-to-noise ratio (left) and outlier magnitude (right). Numeric values indicate solve time in seconds when all 10 instances are solved within 1 hour. Otherwise, we report (gap, count) where gap is the average optimality gap and count is the number of instances solved.

Table 5 demonstrates that both higher SNR and larger outlier magnitudes make outliers easier to identify, thereby improving solver performance. When SNR ≤ 20 or $\delta \leq 6$, distinguishing outliers from inliers becomes difficult, resulting in large optimality gaps for all methods. The proposed solver achieves the best performance across all parameter settings, solving most instances to optimality when SNR ≥ 30 or $\delta \geq 8$.

5.3. Evaluation on Real Datasets

We report solver performance on 13 benchmark regression datasets. For each dataset, we set the outlier penalty μ so that the model identifies exactly 10 outliers. Table 6-7 presents the runtime and optimality gap results, and Table 8-9 reports the number of BnB nodes explored by each solver. The results are summarized in Figure 5.

Method	alcohol $(44, 6)$	education $(50, 4)$	food $(150, 3)$	milk $(86, 7)$	pulpfiber $(62, 7)$	radar $(1573, 4)$	wagner $(63, 6)$
*BnB (P)	(28.6%)	(14.6%)	(7.6%)	(10.9%)	(29.0%)	(6.7%)	(14.7%)
*BnB (H)	**3.4**	118.3	**1.0**	128.9	**2.0**	**23.7**	56.2
Gurobi (M)	11.9	754.6	(18.5%)	1564.5	16.6	(23.5%)	80.0
Gurobi (P)	25.4	38.5	39.2	51.8	7.6	(4.4%)	19.3
*Gurobi (H)	5.4	**13.6**	3.0	**28.8**	10.6	660.2	**12.9**
Mosek (M)	127.7	(8.8%)	(26.0%)	(11.2%)	129.8	(23.6%)	1234.2
Mosek (P)	21.4	235.4	746.2	563.2	8.4	(7.7%)	101.8
*Mosek (H)	(24.8%)	(–%)	(–%)	(12.9%)	(26.2%)	(–%)	(12.9%)

Table 6 Solver performance on smaller real benchmark datasets with 10 outliers. Methods marked with * are proposed in this work. Dataset dimensions (n, p) are shown below each name. Values in parentheses indicate the optimality gap when the solver fails to reach the 1% tolerance within the time limit; numeric values indicate solve time in seconds. Bold entries highlight the best result for each dataset. (–%) indicates solver failure.

Method	pm10 $(500, 7)$	pollen $(3848, 4)$	no $(500, 7)$	fri $(1000, 50)$	rmftsa $(508, 10)$	galaxy $(323, 4)$
*BnB (P)	(23.8%)	(19.2%)	(15.0%)	(11.4%)	(26.7%)	(24.3%)
*BnB (H)	(15.6%)	**(16.7%)**	(4.6%)	**(8.8%)**	(9.4%)	(14.7%)
Gurobi (M)	(31.7%)	(50.2%)	(30.2%)	(37.4%)	(44.7%)	(51.3%)
Gurobi (P)	(20.2%)	(19.6%)	(10.4%)	(10.6%)	(18.7%)	(21.7%)
*Gurobi (H)	**(14.8%)**	(19.6%)	**985.7**	(10.0%)	**1987.8**	**(4.0%)**
Mosek (M)	(33.2%)	(49.1%)	(32.3%)	(36.2%)	(46.6%)	(52.1%)
Mosek (P)	(27.4%)	(19.9%)	(13.5%)	(11.3%)	(21.9%)	(22.6%)
*Mosek (H)	(–%)	(19.9%)	(15.0%)	(10.9%)	(–%)	(23.2%)

Table 7 Solver performance on larger real benchmark datasets with 10 outliers. Methods marked with * are proposed in this work. Dataset dimensions (n, p) are shown below each name. Values in parentheses indicate the optimality gap when the solver fails to reach the 1% tolerance within the time limit; numeric values indicate solve time in seconds. Bold entries highlight the best result for each dataset. (–%) indicates solver failure.

We observe that the hyperplane arrangement formulations (H) with Gurobi or the proposed BnB algorithm consistently outperform the Big-M and perspective formulations across all datasets, demonstrating the value of incorporating threshold constraints (similarly to results with synthetic

Method	alcohol $(44,6)$	education $(50,4)$	food $(150,3)$	milk $(86,7)$	pulpfiber $(62,7)$	radar $(1573,4)$	wagner $(63,6)$
*BnB (P)	111960	106436	88882	97331	99744	105795	100338
*BnB (H)	1161	16069	616	17530	919	6253	12136
Gurobi (M)	80743	6666649	14265644	7271224	80587	1043403	481076
Gurobi (P)	84197	124485	63882	109642	5560	684574	40401
*Gurobi (H)	4665	14080	578	15205	7103	9715	13031
Mosek (M)	131507	4816500	854332	2019163	105449	80660	757731
Mosek (P)	12767	105370	59726	90755	3022	1191	31376
*Mosek (H)	1129695	–	–	398930	582820	–	695613
Prop. 5	2471988	41752	22652	$\sim 10^9$	$\sim 10^8$	$\sim 10^9$	$\sim 10^7$

Table 8 Number of BnB nodes explored by each solver on smaller real benchmark datasets. "–" indicates solver failure. The last row shows the theoretical upper bound from Proposition 5.

Method	pm10 $(500,7)$	pollen $(3848,4)$	no $(500,7)$	fri $(1000,50)$	rmftsa $(508,10)$	galaxy $(323,4)$
*BnB (P)	92461	88564	96312	91594	83874	108458
*BnB (H)	76871	70127	81740	75705	75569	76574
Gurobi (M)	4748094	22293	5370154	365364	4143545	4857546
Gurobi (P)	1525648	75180	1764024	255223	1289427	2548249
*Gurobi (H)	522409	19384	170739	135830	276590	1674007
Mosek (M)	224224	25079	260771	94819	254029	372766
Mosek (P)	9475	47	20571	4377	21400	51221
*Mosek (H)	–	47	21270	3913	–	44744
Prop. 5	$\sim 10^{14}$	$\sim 10^{10}$	$\sim 10^{14}$	$\sim 10^{104}$	$\sim 10^{19}$	$\sim 10^7$

Table 9 Number of BnB nodes explored by each solver on larger real benchmark datasets. "–" indicates solver failure. The last row shows the theoretical upper bound from Proposition 5.

data, hyperplane arrangement formulations with Mosek perform poorly). Neither BnB (H) or Gurobi (H) consistently outperforms the other: from the performance profile in Figure 5, BnB (H) seems to be able to prove optimality faster in easier instances (less than 10 seconds), and Gurobi (H) seems to be more effective in instances requiring more than 1,000 seconds.

Tables 8–9 reveal that the hyperplane arrangement formulation (H) requires fewer nodes to solve to optimality compared to the perspective formulation (P), but each node takes longer to solve. This is evident from instances where all solvers reach the time limit: the perspective formulation explores more nodes than the strengthened formulation. Comparing Gurobi (H) to Gurobi (P), the hyperplane arrangement formulation explores only 30–60% as many nodes. In contrast, BnB (H) explores approximately 80% of the nodes that BnB (P) does. In other words, while the time required to solve the continuous relaxations increases with the hyperplane arrangement considerations, the proposed BnB method is comparatively less impacted, showcasing the benefit of using first order methods to solve the continuous relaxations.

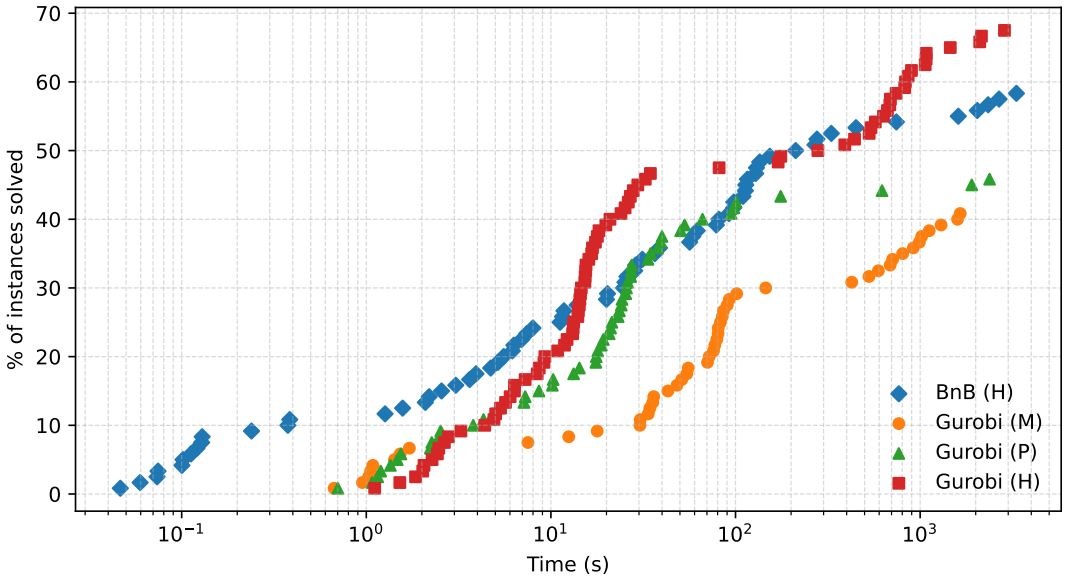

Figure 5 Performance profile on the real datasets. For each time t, the curve reports the fraction of instances solved within time t (to the 1% optimality-gap tolerance).

We observe that, in general, Gurobi (H) requires fewer branch-and-bound nodes than BnB (H) to prove optimality, although: *(i)* there are notable exceptions, see for example results with the "pulpfiber" dataset; *(ii)* the differences in number of nodes are typically small. In other words, we observe that the proposed BnB method does not seem to be substantially hampered from not solving the constrained problems to optimality. The last row of each table shows the theoretical upper bound on the number of nodes required for the strengthened formulation from Proposition 5. In practice, both BnB (H) and Gurobi (H) require only a small fraction of these nodes, showing that the methods perform much better in practice than what the worst-case theoretical bound would suggest.

Overall, as summarized in Figure 5, BnB (H) attains the highest solve rate at short-to-moderate runtimes, while Gurobi (H) remains competitive and becomes advantageous on a subset of harder instances. Both substantially outperform the remaining formulations, confirming that using hyperplane arrangement (H) is the dominant driver of performance on these benchmarks.

5.4. Value of Optimal Solutions

We assess the practical benefit of computing optimal solutions compared to heuristic approaches. For each real dataset, we vary the number of outliers from 5 to 20 and compare the objective values obtained by the popularly used alternating minimization heuristic (Algorithm 3) against the

solutions computed by the BnB solver. Figure 6 presents the relative suboptimality gap $(f_{\text{heuristic}} - f_{\text{optimal}})/f_{\text{optimal}}$ as a box plot.

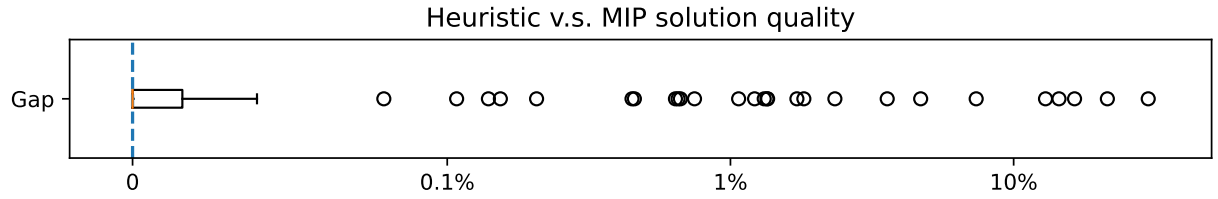

Figure 6 Box plot of relative suboptimality of heuristic solutions (Algorithm 3) compared to optimal solutions from BnB (H) on 13 real datasets, with outlier counts from 5 to 20.

The figure demonstrates the value of exact optimization. By construction, the optimal method always achieves an objective value at least as good as the heuristic (since the heuristic is called at the root node). In practice, the optimal method achieves a strictly better objective in more than 50% of cases. Furthermore, the gap exceeds 1% in more than 10% of cases, with the largest gap reaching approximately 30%. These findings indicate that while heuristics provide reasonable solutions in many cases, exact methods can yield substantially better solutions for a significant fraction of problem instances.

Acknowledgments

This research was supported by AFOSR Grant No. FA9550-24-1-0086 and NSF Grants No. 2346058 and 2112533. RM acknowledges research funding from ONR N00014-25-1-2504.

References

Agulló J (2001a) New algorithms for computing the least trimmed squares regression estimator. *Computational Statistics & Data Analysis* 36(4):425–439.

Agulló J (2001b) New algorithms for computing the least trimmed squares regression estimator. *Computational statistics & data analysis* 36(4):425–439.

Aktürk MS, Atamtürk A, Gürel S (2009) A strong conic quadratic reformulation for machine-job assignment with controllable processing times. *Operations Research Letters* 37(3):187–191.

ApS M (2022) *The MOSEK optimization toolbox for Python manual. Version 9.3.* URL `https://docs.mosek.com/latest/pythonfusion/index.html`.

Bernholt T (2006) Robust estimators are hard to compute. Technical Report Technical Report No. 2005,52, Universität Dortmund, SFB 475.

Bertsekas DP (1982) *Constrained Optimization and Lagrange Multiplier Methods* (Academic Press).

Bertsimas D, King A, Mazumder R (2016) Best subset selection via a modern optimization lens. *The annals of statistics* 44(2):813–852.

Bertsimas D, Mazumder R (2014) Least quantile regression via modern optimization. *The Annals of Statistics* 42(6):2494–2525.

Bhatia K, Jain P, Kar P (2015) Robust regression via hard thresholding. *Advances in neural information processing systems* 28.

Ceria S, Soares J (1999) Convex programming for disjunctive convex optimization. *Mathematical Programming* 86(3):595–614.

Černỳ M, Hladík M, Rada M (2019) Walks on hyperplane arrangements and optimization of piecewise linear functions. *arXiv preprint arXiv:1912.12750* .

Chang LC, Jones DK, Pierpaoli C (2012) Restore: robust estimation of tensors by outlier rejection. *Magnetic Resonance in Medicine* 68(2):538–552, URL `http://dx.doi.org/10.1002/mrm.23268`.

Cover TM (1965) Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers* (3):326–334.

Edelsbrunner H, Souvaine DL (1990) Computing least median of squares regression lines and guided topological sweep. *Journal of the American Statistical Association* 85(409):115–119.

Frangioni A, Gentile C (2006) Perspective cuts for a class of convex 0–1 mixed integer programs. *Mathematical Programming* 106:225–236.

Giloni A, Padberg M (2002) Least trimmed squares regression, least median squares regression, and mathematical programming. *Mathematical and Computer Modelling* 35(9-10):1043–1060.

Gómez A (2021) Outlier detection in time series via mixed-integer conic quadratic optimization. *SIAM Journal on Optimization* 31(3):1897–1925.

Gómez A, Neto J (2025) Outlier detection in regression: conic quadratic formulations. *INFORMS Journal on Computing* .

Günlük O, Linderoth J (2010) Perspective reformulations of mixed integer nonlinear programs with indicator variables. *Mathematical programming* 124:183–205.

Gurobi Optimization, LLC (2022) Gurobi Optimizer Reference Manual. URL `https://www.gurobi.com`.

Hawkins DM (1994) The feasible solution algorithm for least trimmed squares regression. *Computational Statistics & Data Analysis* 17(2):185–196.

Hawkins DM, Olive DJ (1999) Improved feasible solution algorithms for high breakdown estimation. *Computational Statistics & Data Analysis* 30(1):1–11.

Hazimeh H, Mazumder R (2020) Fast best subset selection: Coordinate descent and local combinatorial optimization algorithms. *Operations Research* 68(5):1517–1537.

Hazimeh H, Mazumder R, Saab A (2021) Sparse regression at scale: Branch-and-bound rooted in first-order optimization. *Mathematical Programming* 1–42.

Hofmann M, Gatu C, Kontoghiorghes EJ (2010) An exact least trimmed squares algorithm for a range of coverage values. *Journal of Computational and Graphical Statistics* 19(1):191–204.

Holland PW, Welsch RE (1977) Robust regression using iteratively reweighted least-squares. *Communications in Statistics - Theory and Methods* 6(9):813–827, URL `http://dx.doi.org/10.1080/03610927708827533`.

Hössjer O (1995) Exact computation of the least trimmed squares estimate in simple linear regression. *Computational Statistics & Data Analysis* 19(3):265–282.

Huber PJ (1964) Robust estimation of a location parameter. *The Annals of Mathematical Statistics* 35(1):73–101, URL `http://dx.doi.org/10.1214/aoms/1177703732`.

Huber PJ (1981) Robust statistics. *Wiley Series in Probability and Mathematical Statistics* .

Huchette J, Vielma JP (2019) A geometric way to build strong mixed-integer programming formulations. *Operations Research Letters* 47(6):601–606.

Insolia L, Kenney A, Chiaromonte F, Felici G (2022) Simultaneous feature selection and outlier detection with optimality guarantees. *Biometrics* 78(4):1592–1603.

Klouda K (2015) An exact polynomial time algorithm for computing the least trimmed squares estimate. *Computational Statistics & Data Analysis* 84:27–40.

Lam SK, Pitrou A, Seibert S (2015) Numba: A llvm-based python jit compiler. *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1–6.

Linderoth JT, Savelsbergh MW (1999) A computational study of search strategies for mixed integer programming. *INFORMS Journal on Computing* 11(2):173–187.

Maronna RA, Martin RD, Yohai VJ (2006) *Robust Statistics: Theory and Methods* (Chichester: John Wiley & Sons).

Motulsky HJ, Brown RE (2006) Detecting outliers when fitting data with nonlinear regression: a new method based on robust nonlinear regression and the false discovery rate. *BMC Bioinformatics* 7:123, URL `http://dx.doi.org/10.1186/1471-2105-7-123`.

Rockafellar RT (1976) Augmented lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research* 1(2):97–116.

Rousseeuw PJ (1984) Least median of squares regression. *Journal of the American Statistical Association* 79(388):871–880.

Rousseeuw PJ, Leroy AM (1987) *Robust Regression and Outlier Detection* (New York: John Wiley & Sons).

Rousseeuw PJ, Leroy AM (2003) *Robust regression and outlier detection* (John wiley & sons).

Rousseeuw PJ, Van Driessen K (2006a) Computing lts regression for large data sets. *Data Mining and Knowledge Discovery* 12(1):29–45.

Rousseeuw PJ, Van Driessen K (2006b) Computing lts regression for large data sets. *Data mining and knowledge discovery* 12(1):29–45.

Shen Y, Sanghavi S (2019a) Iterative least trimmed squares for mixed linear regression. *Advances in Neural Information Processing Systems* 32.

Shen Y, Sanghavi S (2019b) Learning with bad training data via iterative trimmed loss minimization. *International conference on machine learning*, 5739–5748 (PMLR).

Sun Q, Mao R, Zhou WX (2021) Adaptive capped least squares. *arXiv preprint* URL `http://dx.doi.org/10.48550/arXiv.2107.00109`.

Vanschoren J, Van Rijn JN, Bischl B, Torgo L (2014) Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter* 15(2):49–60.

Wolsey LA (2020) Branch and bound. *Integer Programming*, chapter 7 (John Wiley & Sons), 2 edition.

Zaslavsky T (1975) *Facing up to arrangements: Face-count formulas for partitions of space by hyperplanes*, volume 154 (American Mathematical Soc.).

Zioutas G, Avramidis A (2005a) Deleting outliers in robust regression with mixed integer programming. *Acta Mathematicae Applicatae Sinica, English Series* 21(2):323–334.

Zioutas G, Avramidis A (2005b) Deleting outliers in robust regression with mixed integer programming. *Acta Mathematicae Applicatae Sinica* 21(2):323–334.

Zioutas G, Pitsoulis L, Avramidis A (2009) Quadratic mixed integer programming and support vectors for deleting outliers in robust regression. *Annals of Operations Research* 166(1):339–353.

Appendix A: Additional Technical Details

A.1. Proof of Proposition 3

In this section we derive the convex hull of set $Z_{\text{HA}}(b)$, which we repeat for convenience.

$$Z_{\text{HA}} = \left\{(z,w,r,t) \in \{0,1\} \times \mathbb{R}^3 : t \geq w^2,\ w(1-z) = 0,\ |r|(1-z) \leq b(1-z),\ |r|z \geq bz,\ w = rz\right\};$$

we omit the explicit dependence on parameter $b \in \mathbb{R}_+$ for convenience, as it is fixed for the purposes of this section.

To prove the result, we use disjunctive programming (Ceria and Soares 1999). Indeed, set $Z_{\text{HA}} = Z_- \cup Z_+ \cup Z_=$ where

$$\begin{aligned}
Z_- &\stackrel{\text{def}}{=} \left\{(z,w,r,t) \in \mathbb{R}^4 : t \geq w^2, z = 1, r \leq -b, w = r\right\} \\
Z_+ &\stackrel{\text{def}}{=} \left\{(z,w,r,t) \in \mathbb{R}^4 : t \geq w^2, z = 1, r \geq b, w = r\right\} \\
Z_= &\stackrel{\text{def}}{=} \left\{(z,w,r,t) \in \mathbb{R}^4 : t \geq w^2, z = 0, |r| \leq b, w = 0\right\}.
\end{aligned}$$

Using standard disjunctive programming reformulations, we find that $(\bar{z},\bar{w},\bar{r},\bar{t}) \in \text{cl conv}(Z_{\text{HA}})$ if and only if there exists additional variables $(z_-,w_-,r_-,t_-,\alpha_-)$, $(z_+,w_+,r_+,t_+,\alpha_+)$ and $(z_=,w_=,r_=,t_=,\alpha_=)$ such that the system

$$\begin{aligned}
&\alpha_- + \alpha_+ + \alpha_= = 1,\ \alpha_- \geq 0,\ \alpha_+ \geq 0,\ \alpha_= \geq 0 \\
&\bar{z} = z_- + z_+ + z_=,\ \bar{w} = w_- + w_+ + w_=,\ \bar{r} = r_- + r_+ + r_=,\ \bar{t} = t_- + t_+ + t_= \\
&t_-\alpha_- \geq w_-^2,\ z_- = \alpha_-,\ r_- \leq -b\alpha_-,\ w_- = r_- \\
&t_+\alpha_+ \geq w_+^2,\ z_+ = \alpha_+,\ r_+ \geq b\alpha_+,\ w_+ = r_+ \\
&t_=\alpha_= \geq w_=^2,\ z_= = 0,\ |r_=| \leq b\alpha_=,\ w_= = 0.
\end{aligned}$$

To obtain the result, we project out the additional variables. First, we remove auxiliary variables $(\boldsymbol{z}, \boldsymbol{w})$ by using the equality constraints, resulting in the simplified system

$$\begin{aligned}
&\alpha_- + \alpha_+ + \alpha_= = 1,\ \alpha_- \geq 0,\ \alpha_+ \geq 0,\ \alpha_= \geq 0\\
&\bar{z} = \alpha_- + \alpha_+,\ \bar{w} = r_- + r_+,\ \bar{r} = r_- + r_+ + r_=,\ \bar{t} = t_- + t_+ + t_=\\
&t_-\alpha_- \geq r_-^2,\ r_- \leq -b\alpha_-\\
&t_+\alpha_+ \geq r_+^2,\ r_+ \geq b\alpha_+\\
&t_= \geq 0,\ |r_=| \leq b\alpha_=.
\end{aligned}$$

Next we can project out auxiliary variables $\boldsymbol{t}$, leading inequality $\bar{t} \geq r_-^2/\alpha_- + r_+^2/\alpha_+$, and auxiliary variable $\alpha_=$ using the first equality constraint. The simplified system reads

$$\begin{aligned}
&\alpha_- \geq 0,\ \alpha_+ \geq 0,\ \alpha_- + \alpha_+ \leq 1\\
&\bar{z} = \alpha_- + \alpha_+,\ \bar{w} = r_- + r_+,\ \bar{r} = r_- + r_+ + r_=\\
&\bar{t} \geq r_-^2/\alpha_- + r_+^2/\alpha_+\\
&r_- \leq -b\alpha_-,\ r_+ \geq b\alpha_+,\ |r_=| \leq b(1 - \alpha_- - \alpha_+).
\end{aligned}$$

Finally, we project out $r_=$ since $\bar{r} = r_- + r_+ + r_= \Leftrightarrow \bar{r} = \bar{w} + r_=$, and thus $r_=$ satisfying constraints exists if and only $|\bar{r} - \bar{w}| \leq b(1 - \alpha_- - \alpha_+)$. The result of Proposition 3 then follows by renaming variables as $\alpha_- \leftrightarrow z_-$, $\alpha_+ \leftrightarrow z_+$, $r_- \leftrightarrow -w^-$ and $r_+ \leftrightarrow w^+$.

A.2. Proof of Proposition 7 and Proposition 8

We first establish a key lemma about the structure of optimal solutions of the relaxation problem.

LEMMA 1. *Any optimal solution of* (11) *with any given* $\boldsymbol{\ell}^-, \boldsymbol{\ell}^+, \boldsymbol{u}^-, \boldsymbol{u}^+$ *satisfies* $w_i^+ w_i^- = 0$ *and* $z_i^+ z_i^- = 0$ *for all* $i \in [n]$.

Proof. Given an index i, we discuss the following three cases.

Case 1: $u_i^- = u_i^+ = 0$**.** This implies $z_i^+ = z_i^- = 0$. The perspective terms $\frac{d_i(w_i^+)^2}{z_i^+}$ and $\frac{d_i(w_i^-)^2}{z_i^-}$ then force $w_i^+ = w_i^- = 0$. Thus $w_i^+ w_i^- = z_i^+ z_i^- = 0$.

Case 2: $\ell_i^- = 1$ **or** $\ell_i^+ = 1$**.** This implies $z_i^+ + z_i^- = 1$. Constraint (9b) becomes $|r_i - w_i| \leq 0$, which forces $w_i = r_i$. For a fixed residual r_i, we determine the optimal values of the auxiliary variables.

If $\ell_i^+ = 1$ (i.e., $w_i \geq 0$), then $r_i = w_i \geq 0$. If the problem is feasible, the objective is minimized by setting $z_i^+ = 1$, $z_i^- = 0$, $w_i^+ = r_i$, and $w_i^- = 0$, which satisfies $w_i = w_i^+ - w_i^- = r_i$ and minimizes the perspective term.

If $\ell_i^- = 1$ (i.e., $w_i \leq 0$), then $r_i = w_i \leq 0$. If the problem is feasible, the objective is minimized by setting $z_i^+ = 0$, $z_i^- = 1$, $w_i^+ = 0$, and $w_i^- = -r_i$, which satisfies $w_i = w_i^+ - w_i^- = r_i$ and minimizes the perspective term.

In both cases, $w_i^+ w_i^- = z_i^+ z_i^- = 0$.

Case 3: $\ell_i^- = \ell_i^+ = 0$**,** $u_i^- = u_i^+ = 1$**.** We prove by contradiction. Suppose at an optimal solution, both $z_i^+ > 0$ and $z_i^- > 0$ for some i. By constraint (9a), we have $w_i^+ \geq \sqrt{2\mu} z_i^+ > 0$ and $w_i^- \geq \sqrt{2\mu} z_i^- > 0$.

Consider a perturbation: replace $(z_i^+, z_i^-, w_i^+, w_i^-)$ by $(\alpha z_i^+, \beta z_i^-, \alpha w_i^+, \beta w_i^-)$ for some $0 < \alpha, \beta < 1$ chosen such that

$$\alpha w_i^+ - \beta w_i^- = w_i^+ - w_i^-.$$

This equation can be rewritten as $w_i^+(1-\alpha) = w_i^-(1-\beta)$. Since $w_i^+, w_i^- > 0$, we can choose $\alpha, \beta \in (0,1)$ satisfying this constraint.

We verify that all constraints remain satisfied:

- Constraint (9a): $\alpha w_i^+ \geq \sqrt{2\mu}\alpha z_i^+$ and $\beta w_i^- \geq \sqrt{2\mu}\beta z_i^-$ hold since the original constraints hold.
- Constraint (9b): The left-hand side $|r_i - w_i| = |r_i - (w_i^+ - w_i^-)|$ is unchanged since $\alpha w_i^+ - \beta w_i^- = w_i^+ - w_i^-$. The right-hand side $\sqrt{2\mu}(1 - z_i)$ increases since the new $z_i = \alpha z_i^+ + \beta z_i^- < z_i^+ + z_i^-$. Thus the constraint remains satisfied.
- The new $z_i = \alpha z_i^+ + \beta z_i^- < z_i^+ + z_i^-$, so the constraint $z_i \in [0,1]$ remains satisfied.

The objective changes only in the perspective terms. The new contribution is

$$\frac{d_i(\alpha w_i^+)^2}{\alpha z_i^+} + \frac{d_i(\beta w_i^-)^2}{\beta z_i^-} = \alpha \frac{d_i (w_i^+)^2}{z_i^+} + \beta \frac{d_i (w_i^-)^2}{z_i^-} < \frac{d_i (w_i^+)^2}{z_i^+} + \frac{d_i (w_i^-)^2}{z_i^-},$$

since $\alpha, \beta < 1$. This contradicts the optimality of the original solution. Therefore, at any optimal solution, $z_i^+ z_i^- = 0$. By constraint (9a), this implies $w_i^+ w_i^- = 0$. $\square$

Lemma 1 allows us to project out the auxiliary variables $z_i^+, z_i^-, w_i^+, w_i^-$ by writing

$$\frac{d_i (w_i^+)^2}{z_i^+} + \frac{d_i (w_i^-)^2}{z_i^-} = \frac{d_i w_i^2}{z_i},$$

where $z_i = z_i^+ + z_i^-$ and $w_i = w_i^+ - w_i^-$. Note that $\ell_i^- = 1$ implies $z_i^- = 1$, $z_i^+ = 0$ and therefore $w_i^+ = 0$, $w \le 0$. Similarly, $\ell_i^+ = 1$ implies $w \ge 0$. We can rewrite (13) as

$$\phi(r;\ell^-,\ell^+,u^-,u^+,d) = \min_{w\in\mathbb{R},z\in[0,1]} \frac{1}{2}(w-r)^2 + \mu z + d\left(\frac{w^2}{z} - w^2\right) \tag{22a}$$

$$\text{s.t.} \quad |w| \ge \sqrt{2\mu}z, \qquad |r-w| \le \sqrt{2\mu}(1-z) \tag{22b}$$

$$z \le u^+ + u^-,\ w \ge 0 \text{ if } \ell^+ = 1,\ w \le 0 \text{ if } \ell^- = 1. \tag{22c}$$

We now discuss the following four Cases, which gives us the closed form expression of ϕ in Proposition 7 and Proposition 8.

Case 1: $u^- = u^+ = 0$**.** We have $z = 0$. The constraint $|r - w| \le \sqrt{2\mu}$ combined with $w = 0$ (forced by the perspective term) yields $|r| \le \sqrt{2\mu}$, which gives

$$\phi(r;0,0,0,0,d) = \frac{1}{2}r^2 + \delta_{[-\sqrt{2\mu},\sqrt{2\mu}]}(r).$$

Case 2: $l^+ = 1$**.** we have $z = 1$ and $w \ge 0$. The constraint $|r - w| \le 0$ forces $w = r$. Combined with $|w| \ge \sqrt{2\mu}$, we obtain $r \ge \sqrt{2\mu}$, , which gives

$$\phi(r;0,1,0,1,d) = \mu + \delta_{[\sqrt{2\mu},\infty]}(r).$$

Case 3: $l^- = 1$**.** We have $z = 1$ and $w \le 0$. Similarly, $w = r$ and $w \le -\sqrt{2\mu}$ yield $r \le -\sqrt{2\mu}$, which gives

$$\phi(r;1,0,1,0,d) = \mu + \delta_{[-\infty,-\sqrt{2\mu}]}(r).$$

Case 4: $\ell^- = \ell^+ = 0$**,** $u^- = u^+ = 1$**.** We first ignore the constraints in (22b) and derive the closed-form solution for the optimization problem

$$\phi(r) = \min_{w\in\mathbb{R},z\in[0,1]} \frac{1}{2}(w-r)^2 + \mu z + d\left(\frac{w^2}{z} - w^2\right), \tag{23}$$

where $\mu > 0$ and $0 < d < 1/2$.

Step 1: Minimization over w **for fixed** $z \in (0,1]$**.** For any $z > 0$, the objective is strictly convex in w. The coefficient of w^2 is

$$a(z) := \frac{1}{2} + d\left(\frac{1}{z} - 1\right) = \frac{1}{2} - d + \frac{d}{z} > 0.$$

The unique minimizer is

$$w^*(z) = \frac{r}{1 + 2d\left(\frac{1}{z} - 1\right)} = \frac{r}{1 - 2d + \frac{2d}{z}}. \tag{24}$$

Substituting back and completing the square, the minimal value over w is

$$g(z) := \min_w \phi(w; z) = \mu z + \frac{1}{2}r^2 - \frac{r^2}{2\left(1 - 2d + \frac{2d}{z}\right)}. \tag{25}$$

When $z = 0$, feasibility forces $w = 0$, yielding $g(0) = \frac{1}{2}r^2$. At $z = 1$, we have $w^*(1) = r$ and $g(1) = \mu$. The original problem thus reduces to

$$\phi(r) = \min_{z \in [0,1]} g(z).$$

Step 2: Stationarity in z and the interior solution. Define $D(z) := 1 - 2d + \frac{2d}{z} = \frac{(1-2d)z+2d}{z}$ for $z \in (0, 1]$. Differentiating (25) gives

$$g'(z) = \mu - \frac{r^2 d}{z^2 D(z)^2}.$$

Setting $g'(z) = 0$ and noting that $z^2 D(z)^2 = N(z)^2$ where $N(z) := (1 - 2d)z + 2d$, we obtain

$$N(z) = |r|\sqrt{\frac{d}{\mu}}.$$

Solving for z yields the candidate interior minimizer

$$z^*(r) = \frac{|r|\sqrt{\frac{d}{\mu}} - 2d}{1 - 2d}. \tag{26}$$

This is feasible (i.e., $0 \le z^* \le 1$) precisely when $2\sqrt{\mu d} \le |r| \le \sqrt{\mu/d}$.

Step 3: Evaluation at the optimizer and boundary regimes. When $2\sqrt{\mu d} \le |r| \le \sqrt{\mu/d}$, we have $z^*(r) \in (0, 1)$ and $g'(z^*) = 0$. Using $N(z^*) = |r|\sqrt{d/\mu}$ and $D(z^*) = N(z^*)/z^*$ in (25):

$$\phi(r) = g(z^*) = \mu z^* + \frac{1}{2}r^2 - \frac{r^2 z^*}{2N(z^*)}.$$

Substituting z^* from (26) and simplifying yields

$$\phi(r) = \frac{-dr^2 + 2\sqrt{\mu d}|r| - 2\mu d}{1 - 2d}.$$

The optimal w^* in this regime is obtained from (24):

$$w^*(r) = \frac{r}{D(z^*)} = \frac{rz^*}{N(z^*)} = \text{sign}(r)\sqrt{\frac{\mu}{d}}z^*(r).$$

When $|r| < 2\sqrt{\mu d}$, the stationary z^* in (26) is negative and infeasible. Since g is convex on $[0, 1]$, the minimum occurs at $z = 0$, which forces $w = 0$. Thus $\phi(r) = \frac{1}{2}r^2$.

When $|r| > \sqrt{\mu/d}$, we have $z^* > 1$, which is infeasible. The minimum occurs at $z = 1$, where $w^*(1) = r$ and $\phi(r) = \mu$.

Summary. Combining the three cases:

$$\phi(r) = \begin{cases} \dfrac{1}{2}r^2, & |r| \le 2\sqrt{\mu d}, \\ \dfrac{-dr^2 + 2\sqrt{\mu d}|r| - 2\mu d}{1-2d}, & 2\sqrt{\mu d} \le |r| \le \sqrt{\dfrac{\mu}{d}}, \\ \mu, & |r| \ge \sqrt{\dfrac{\mu}{d}}, \end{cases}$$

with optimal (w^*, z^*) given by

$$z^*(r) = \begin{cases} 0, & |r| \le 2\sqrt{\mu d}, \\ \dfrac{|r|\sqrt{\frac{d}{\mu}} - 2d}{1-2d}, & 2\sqrt{\mu d} \le |r| \le \sqrt{\dfrac{\mu}{d}}, \\ 1, & |r| \ge \sqrt{\dfrac{\mu}{d}}, \end{cases}$$

$$w^*(r) = \begin{cases} 0, & |r| \le 2\sqrt{\mu d}, \\ \operatorname{sign}(r)\sqrt{\dfrac{\mu}{d}}z^*(r), & 2\sqrt{\mu d} \le |r| \le \sqrt{\dfrac{\mu}{d}}, \\ r, & |r| \ge \sqrt{\dfrac{\mu}{d}}. \end{cases}$$

Derivative of $\phi(r)$. Differentiating each piece yields:

$$\phi'(r) = \begin{cases} r, & |r| < 2\sqrt{\mu d}, \\ \dfrac{-2dr + 2\sqrt{\mu d}\operatorname{sign}(r)}{1-2d}, & 2\sqrt{\mu d} < |r| < \sqrt{\dfrac{\mu}{d}}, \\ 0, & |r| > \sqrt{\dfrac{\mu}{d}}. \end{cases}$$

At the transition points, continuity can be verified directly. At $|r| = 2\sqrt{\mu d}$: the left derivative is $\operatorname{sign}(r) \cdot 2\sqrt{\mu d}$, and the right derivative is $\frac{-2d \cdot 2\sqrt{\mu d}\operatorname{sign}(r) + 2\sqrt{\mu d}\operatorname{sign}(r)}{1-2d} = \operatorname{sign}(r) \cdot 2\sqrt{\mu d}$. At $|r| = \sqrt{\mu/d}$: the left derivative is $\frac{-2d\sqrt{\mu/d}\operatorname{sign}(r) + 2\sqrt{\mu d}\operatorname{sign}(r)}{1-2d} = 0$. Thus $\phi'(r)$ is continuous everywhere.

Checking feasibility in (22b)**.**

- When $|r| \le 2\sqrt{\mu d}$: We have $z^* = w^* = 0$. Both constraints reduce to $0 \ge 0$ and $|r| \le \sqrt{2\mu}$. The latter holds since $|r| \le 2\sqrt{\mu d} \le \sqrt{2\mu}$ (using $d \le 1/2$).

- When $2\sqrt{\mu d} \le |r| \le \sqrt{\mu/d}$: We have $w^* = \text{sign}(r)\sqrt{\mu/d}z^*$. The first constraint becomes $\sqrt{\mu/d}z^* \ge \sqrt{2\mu}z^*$, which holds since $d \le 1/2$. For the second constraint, we compute $|r - w^*| = |r| - |w^*| = |r| - \sqrt{\mu/d}z^*$. Substituting z^* and simplifying shows that this equals $\sqrt{2\mu}(1 - z^*)$.
- When $|r| \ge \sqrt{\mu/d}$: We have $z^* = 1$ and $w^* = r$. The constraints become $|r| \ge \sqrt{2\mu}$ and $|r - r| = 0 \le 0$, both of which hold.

Since the optimal (w^*, z^*) satisfies constraints (22b), they are also the optimal solutions to $\phi(r; 0, 0, 1, 1, d)$. Moreover, the constraints (13c) can be removed.

A.3. Proof of Proposition 9

We first establish a preliminary lemma.

LEMMA 2 **(Gradient-suboptimality inequality)**. *If $\Phi : \mathbb{R}^p \to \mathbb{R}$ is $\tilde{\lambda}$-strongly convex and differentiable, then for any $\boldsymbol{x} \in \mathbb{R}^p$,*

$$\min_{\boldsymbol{u} \in \mathbb{R}^p} \Phi(\boldsymbol{u}) \ge \Phi(\boldsymbol{x}) - \frac{1}{2\tilde{\lambda}} \|\nabla \Phi(\boldsymbol{x})\|_2^2.$$

Proof. Strong convexity yields for all $\boldsymbol{u}$: $\Phi(\boldsymbol{u}) \ge \Phi(\boldsymbol{x}) + \langle \nabla \Phi(\boldsymbol{x}), \boldsymbol{u} - \boldsymbol{x} \rangle + \frac{\tilde{\lambda}}{2} \|\boldsymbol{u} - \boldsymbol{x}\|_2^2$. The right side is minimized in $\boldsymbol{u}$ at $\boldsymbol{u} = \boldsymbol{x} - \tilde{\lambda}^{-1} \nabla \Phi(\boldsymbol{x})$, giving the result. □

As mentioned in Section 2.1.2, the perspective relaxation (12) is designed to be $\tilde{\lambda}$-strongly convex. Since the augmented Lagrangian (16) is obtained by removing some constraints in (12) and adding some piecewise-linear/quadratic functions, it is also $\tilde{\lambda}$-strongly convex. Therefore,

$$\bar{\zeta}(\bar{\boldsymbol{\nu}}) = \min_{\boldsymbol{\beta}} L_\rho(\boldsymbol{\beta}; \bar{\boldsymbol{\nu}}) \ge L_\rho(\bar{\boldsymbol{\beta}}, \bar{\boldsymbol{\nu}}) - \frac{\|\nabla_{\boldsymbol{\beta}} L_\rho(\bar{\boldsymbol{\beta}}, \bar{\boldsymbol{\nu}})\|_2^2}{2\tilde{\lambda}}$$

where the right-hand side provides a lower bound of the relaxation problem. □

A.4. Computing Incumbent Solutions

We obtain incumbent solutions in the BnB procedure by solving an ℓ_2-regularized estimation problem with the outlier set $\mathcal{S} \subset [n]$ fixed. Specifically, for a given support $\mathcal{S}$, we solve the original problem (1) with $z_i = 1$ for $i \in \mathcal{S}$ and $z_i = 0$ otherwise:

$$F^*(\mathcal{S}) := \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2} \sum_{i \notin \mathcal{S}} (y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta})^2 + \frac{\lambda}{2} \|\boldsymbol{\beta}\|_2^2 + \mu |\mathcal{S}|. \tag{27}$$

This is a weighted ridge regression problem with a closed-form solution. Let $\boldsymbol{W} = \text{Diag}(1 - \boldsymbol{z})$ be the diagonal weight matrix. The optimal coefficient vector is

$$\boldsymbol{\beta}^*(\mathcal{S}) = (\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{y}. \tag{28}$$

Before constructing the BnB tree, following (Rousseeuw and Van Driessen 2006b), we compute an initial incumbent solution using a heuristic based on alternating minimization. The method alternates between optimizing over $\boldsymbol{\beta}$ for fixed $\boldsymbol{z}$ and optimizing over $\boldsymbol{z}$ for fixed $\boldsymbol{\beta}$. Starting with $\boldsymbol{z} = \mathbf{0}$ (i.e., no outliers), we iterate as follows:

1. **Solve for $\boldsymbol{\beta}$**: Given the current $\boldsymbol{z}$, compute $\boldsymbol{\beta}$ using (28).
2. **Update $\boldsymbol{z}$**: Given the current $\boldsymbol{\beta}$, compute residuals $r_i = y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}$ and update each z_i by comparing the squared loss to the outlier penalty:

$$z_i = \begin{cases} 1, & \text{if } \frac{1}{2} r_i^2 > \mu, \\ 0, & \text{otherwise.} \end{cases} \tag{29}$$

The update rule (29) follows from the optimality condition: for fixed $\boldsymbol{\beta}$, observation i should be an outlier if and only if the penalty μ is less than the squared loss $\frac{1}{2} r_i^2$. This alternating procedure continues until the outlier set $\mathcal{S} = \{i : z_i = 1\}$ stabilizes, which is guaranteed since the objective decreases monotonically. Algorithm 3 summarizes the method.

Algorithm 3 Computing initial incumbent via alternating minimization

Input: Data $(\boldsymbol{X}, \boldsymbol{y})$, parameters λ, μ

Initialize $\boldsymbol{z} = \mathbf{0}$
repeat
  Compute $\boldsymbol{\beta} = (\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{y}$ where $\boldsymbol{W} = \text{Diag}(1 - \boldsymbol{z})$
  Compute residuals $r_i = y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}$ for all $i \in [n]$
  Update $z_i = \mathbf{1}\{\frac{1}{2} r_i^2 > \mu\}$ for all $i \in [n]$
until the outlier set $\mathcal{S} = \{i : z_i = 1\}$ does not change
return $\boldsymbol{\beta}$, outlier set $\mathcal{S}$

At each node in the BnB tree, we further refine the upper bound by solving problem (27) with the support set determined by rounding the relaxed integer variables: $\mathcal{S} = \{i : z_i^- + z_i^+ \geq 0.5\}$. Note that variables $z_i^- + z_i^+$ are only implicitly computed, and this is equivalent to choosing

$$\mathcal{S} = \left\{ i : |y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}| \geq \left(\tfrac{1}{2} + d\right) \sqrt{\frac{\mu}{d}} \right\} \quad \text{or} \quad \mathcal{S} = \left\{ i : \phi(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}) \geq \frac{\mu \left(\frac{3}{4} - d - d^2\right)}{1 - 2d} \right\}.$$