European Journal of Operational Research 316 (2024) 1070–1084

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Analytics, Computational Intelligence and Information Management

Mathematical programming for simultaneous feature selection and outlier detection under l1 norm

Michele Barbato *, Alberto Ceselli

Department of Computer Science, Università degli Studi di Milano, 18, via Celoria, 20133, Milano, Italy

ARTICLE INFO

Keywords:
Data science
Outlier detection
Feature selection
Least absolute deviation
Mathematical programming

ABSTRACT

The goal of simultaneous feature selection and outlier detection is to determine a sparse linear regression vector by fitting a dataset possibly affected by the presence of outliers.

The problem is well-known in the literature. In its basic version it covers a wide range of tasks in data analysis. Simultaneously performing feature selection and outlier detection strongly improves the application potential of regression models in more general settings, where data governance is a concern. To trigger this potential, flexible training models are needed, with more parameters under control of decision makers.

The use of mathematical programming, although pertinent, is scarce in this context and mostly focusing on the least-squares setting. Instead we consider the least absolute deviation criterion, proposing two mixed-integer linear programs, one adapted from existing studies, and the other obtained from a disjunctive programming argument. We show theoretically and computationally that the disjunctive-based formulation is better in terms of both continuous relaxation quality and integer optimality convergence.

We experimentally benchmark against existing methodologies from the literature. We identify the characteristics of contamination patterns, in which mathematical programming is better than state-of-the-art algorithms in combining prediction quality, sparsity and robustness against outliers. Additionally, the mathematical programming approaches allow the decision maker to directly control parameters like the number of features or outliers to tolerate, those based on least absolute deviations performing best. On real world datasets, where privacy is a concern, our approach compares well to state-of-the-art methods in terms of accuracy, being at the same time more flexible.

1. Introduction

In the last two decades huge advances in data collection and digitization techniques have led high-dimensional problems becoming relevant in several fields, such as chemometrics (Bertsimas, Kitane, Azami, & Doucet, 2020), biomedical research (Insolia, Kenney, Chiaromonte, & Felici, 2021), genomics and pattern recognition. Classical references from the literature cover this domain (Greenshtein, 2006 and references therein).

A key application is *linear regression*. Given is a dataset of points in a $(d+1)$-dimension space: d dimensions represent features, and the last one represents a response.

Intuitively, each point in the dataset is assumed to represent a linearly related input–response pair describing a certain phenomenon, whose measurement is affected by noise. With a geometric analogy, this linear relation can be modeled by a hyperplane in a $(d+1)$-dimension space: the regression problem consists in finding the slope and intercept which optimize a predefined notion of proximity between the hyperplane and the points. That is, searching for an optimal linear fitting of points.

Such a fitting hyperplane provides an approximation of the phenomenon measured by the dataset, and can therefore be used either as a descriptive model for it, or more commonly to perform *predictions* of the same phenomenon on new observations.

Linear regression has been studied for decades in statistical data analysis. However, the increase in the dimensionality and size of datasets has recently led to important methodological consequences. First, applications require an accurate feature selection. Such a need is not only computational. For instance, the interpretability of a model produced by fitting is preserved only if a restricted set of relevant explanatory variables are included in it. In the extreme case of *underdetermined regimes* (having more features than points) classical solution methods for linear regression problems do not even apply directly (Filzmoser & Nordhausen, 2021). Other situations in which an accurate feature selection is desirable arise, *e.g.*, in applications where privacy must be, at least partially, preserved. As a paradigmatic example,

* Corresponding author.
E-mail addresses: michele.barbato@unimi.it (M. Barbato), alberto.ceselli@unimi.it (A. Ceselli).

https://doi.org/10.1016/j.ejor.2024.03.035
Received 13 January 2023; Accepted 24 March 2024
Available online 26 March 2024

consider datasets containing sensitive attributes that, if appropriately combined, might reveal identity or personal data. In this case it is desirable to forbid such combinations or to use a very limited amount of attributes to perform predictions from such datasets. Giving to the *decision maker* a direct, *a priori* control on parameters like the number of features to select is central in this type of applications. Encoding these portions of logic in a mathematical program best suits this need.

Another consequence in the increase of dataset storage complexity is that contamination chances are greater. Structurally, classical contamination types are the *vertical outliers* and the *bad leverage points*, corresponding to large deviations in the response or explanatory variables, respectively (James et al., 2021, pages 97-99). The presence of a few *outliers*, that is, points not conforming to the trends of the dataset as a whole, may greatly affect the fitting quality. Outliers may share statistical properties or not, depending on the application. For example, in distributed dataset storage, data corruption may be due to the deliberated action of an adversary (not necessarily malicious) in one of the storage nodes. That is, very general contamination types may occur, making it desirable to remain *agnostic* with respect to the contamination type. Furthermore, providing to the decision maker the direct management of parameters like the number of outliers to tolerate, in the form of data in a mathematical program, yields invaluable additional flexibility.

While outlier detection and feature selection have traditionally been treated separately, there is a very recent research trend, attempting to handle them together (Bottmer, Croux, & Wilms, 2022). At the same time, a seminal paper by Bertsimas, King, and Mazumder (2016) opened up new perspectives, not only in the Operations Research community, showing mathematical programming to be a potential breakthrough in these applications. In fact, mixed-integer programming (MIP) formalizations of the regression with *simultaneous feature selection and outlier detection* (SFSOD) problem have been considered independently in a few studies (Insolia et al., 2021; Jammal, 2020; Jammal, Canu, & Abdallah, 2020, 2021; Thompson, 2022). All these works focus on a common assumption: using the least-squares (LS) criterion to measure the proximity of a point to the hyperplane. This is not a coincidence: while measures based on least absolute deviations are common in machine learning, having differentiable proximity measures allows more freedom in the choice of optimization techniques.

In this paper we instead focus on the option of solving the linear regression with SFSOD using a least absolute deviation (LAD) optimization criterion.

Motivations and contributions. Two facts motivate a research interest on the LAD criterion. The first fact is straightforward: a LAD regression problem can be efficiently re-formulated as a mixed-integer linear program and solved with specific state-of-the-art optimization packages. The second fact comes from the literature: LAD models tend to produce better solutions in terms of robustness (Arslan, 2012; Dodge, 1997; Wang, Li, & Jiang, 2007; Wang & Zhu, 2017; Xu & Ying, 2010). They are also easier to be handled when exploiting dual formulations for training, as in Support Vector Regression models (Hastie, Tibshirani, & Friedman, 2009, pp. 434–437). A research question naturally arises, whether using mathematical programming models with LAD optimization criteria helps in combining these two effects, or not. Such a research question triggers other ones, for instance how different types of outliers affect the behavior of the regression methods which use mathematical programming.

As first contribution we propose two MIP models of the linear regression with SFSOD using the LAD optimization criterion. One model is the natural adaptation to the LAD setting of the model used in Insolia et al. (2021), Jammal et al. (2020, 2021) and Thompson (2022) for the LS setting; it exploits auxiliary variables to cancel the residuals of the fitting hyperplane over the selected outliers. The other one elaborates over a canonical disjunctive linearization of the bilinear terms used to eliminate the residuals directly in the objective function. We provide a theoretical analysis of this latter approach, which proves to grant structural advantages, such as provably stronger LP bounds, with respect to the method adapted from the literature. All these MIP approaches involve the use of big-M values bounding the residuals of the fitting hyperplane. We adapt and integrate procedures from the literature in a heuristic to compute them. We therefore analyze theoretically and computationally the relationship between their magnitude and the quality of their dual bounds. We also consider both comparison and integration of principles borrowed from the leveraged least-trimmed absolute deviation method of Sudermann-Merx and Rebennack (2021).

Our second contribution is application-oriented. We conduct an extensive experimental analysis on synthetic and real-world datasets, comparing the MIP-based models, considering also benchmarks from the literature. Our study takes into account (a) computational efficiency of the fitting optimization process, (b) quality of the results, when the corresponding regression solutions are used for prediction, not only in terms of regression error, but also of accuracy in selecting features.

Our paper is organized as follows. In Section 2 we introduce our notation and review relevant concepts and related approaches from the literature. In Section 3 we describe in detail the MIP models for the linear regression with SFSOD with LAD objective including our heuristic procedure for computing and refining big-M values. In Section 4 and Section 5 we present our computational analysis on synthetic and real world data, respectively, and in Section 6 we collect a few conclusions.

2. Notation and background

Given a dataset of the form $\mathcal{P} = \{p^i = (a^i, r_i) \in \mathbb{R}^{d+1}, i = 1 \dots k\}$, the traditional *linear regression* problem is to find a vector $\omega \in \mathbb{R}^d$ of *coefficients* and an *intercept* value $\zeta \in \mathbb{R}$ such that a given proximity measure between the set $\mathcal{P}$ and the hyperplane $\mathcal{H} = \{(x, y) \in \mathbb{R}^{d+1} : y = \omega x + \zeta\}$ is optimized. The process of determining the pair (ω, ζ) is also called *fitting*.

The linear regression paradigm is useful when it is reasonable to assume that a linear relation exists between the *input* values a^i's (belonging to a d-dimensional space of *features*) and the corresponding scalar *response* values r_i's. In this context, the dataset $\mathcal{P}$ represents a set of k input–response measurements, possibly affected by noise or errors, and the hyperplane $\mathcal{H}$ represents a reliable estimate of the actual underlying mechanism generating the data in $\mathcal{P}$. In particular, $\mathcal{H}$ can be used to forecast the responses that would be obtained from new input data through the same mechanism that generated the dataset $\mathcal{P}$.

As discussed in the introduction, the SFSOD approach enhances the traditional linear regression by combining it with (a) *feature selection*, that is, drop dimensions which are irrelevant and (b) *outlier detection*, that is, drop points which represent measurements strongly affected by errors.

Elaborating on the most general form provided in Insolia et al. (2021), we consider a *loss function* $\ell : \mathbb{R} \to \mathbb{R}_{\geq 0}$ and express the linear regression with SFSOD as the following MIP:

$$\min \sum_{i=1}^{k} \frac{1}{k} s_i \ell(r_i - a^i w - z) \tag{1}$$

$$||w||_0 \leq d_0 \tag{2}$$

$$||s||_0 \geq k - k_0 \tag{3}$$

$$w \in \mathbb{R}^d, z \in \mathbb{R} \tag{4}$$

$$s \in \{0, 1\}^k \tag{5}$$

where $||\cdot||_0$ is the 0-norm (counting the nonzero entries of its argument) and d_0 and k_0 are given "budget" values respectively for the number of features that can be selected in the final solution and for the number of points that can be marked as outliers. The feature and outlier budget constraints (2) and (3) are put as inequalities for flexibility and consistency with the literature. In fact, optimal solutions always exist in which constraint (3) is tight, and constraint (2) is likely to

be tight due to numerical reasons. For the sake of generality, function $\ell(\cdot)$ is not given explicitly. A typical choice in the literature is to take $\ell(x) = x^2$, under which (1) corresponds to optimize the LS criterion. We call the corresponding problem *LS-SFSOD*. Another common criterion is the LAD, corresponding to take $\ell(x) = |x|$. We call the corresponding problem *LAD-SFSOD*. Variables $s_1, s_2, \dots, s_k$ are called *switches* hereafter: given a solution $(\hat{w}, \hat{z}, \hat{s})$ to (1)–(5) point $p^i = (a^i, r_i)$ is selected as an *authentic* point (a non-outlier) if and only if $s_i = 1$.

Sparse and robust regression. Sparsity in regression problems consists in determining faithful predictors relying on small sets of features. The prototype of sparse regression problems is the *best subset selection* (BSS), obtained from (1)–(5) by taking $\ell(x) = x^2$ and $k_0 = 0$. BSS is **NP**-hard (Natarajan, 1995) and it has been considered computationally intractable for decades: an effective MIP-based resolution algorithm for the BSS has been proposed only recently (Bertsimas et al., 2016). Previously, sparse regression has been mainly performed through relaxations of BSS. An efficiently solvable convex relaxation of the BSS is the *LASSO* (Tibshirani, 1996), obtained by replacing the 0-norm constraint with its 1-norm counter-part, that is, $||w||_1 \leq d_0$. For a suitably chosen $\lambda \geq 0$, the LASSO can be expressed in the equivalent "penalized ℓ_1" form

$$\min\{||r - Aw - z\mathbf{1}||_2^2 + \lambda||w||_1 : w \in \mathbb{R}^d, z \in \mathbb{R}\},$$

highlighting that sparsity is incentivized through the penalization of the term $||w||_1$. Similar penalization approaches are used in the non-convex estimators *SCAD* (Fan & Li, 2001) and *MC+* (Zhang, 2010); variations of the standard LASSO method also include the usage of multiple penalization coefficients to control separately the feature selection and the feature coefficient shrinkage (Meinshausen, 2007), and adaptive penalization coefficients, where the λ used in the LASSO varies *e.g.*, with the number of training points (Zou, 2006). The *elastic net LASSO* approach (Zou & Hastie, 2005) replaces the penalization term of the LASSO with a combination of 1-norm and 2-norm penalization obtaining reliable sparse predictors in presence of highly correlated variables. The *Dantzig estimator* (Candes & Tao, 2007) relies on the resolution of a linear program whose objective function is equivalent to the 1-norm of the feature vector and whose domain controls the size of correlated residual vectors.

The above-mentioned BSS and LASSO-like methods are suitable for sparse regression in both the underdetermined ($d < k$) and the overdetermined ($d \geq k$) regimes but they are sensitive to outliers. The most intuitive way to deal with outliers is to recognize and remove them before training any learning model. Among the methods for *apriori* outlier recognition we first mention those based on statistical preprocessing. They describe the data at hand through a (known or unknown) stochastic distribution and identify outliers as those not conforming to that distribution model. For the univariate case, a simple example of this approach is provided in Laurikkala et al. (2000): a point is considered an outlier whenever it exceeds the upper or the lower distribution extremes by 1.5× the interquartile range of the dataset values. *Parametric* statistical methods assume a prior distribution of the data (*e.g.*, a Gaussian model) and first estimate the distribution model parameters, then run some statistical test to assess the conformity of the points. Examples of this approach are the *maximum volume ellipsoid estimation* and the *convex peeling* described in Rousseeuw and Leroy (2005) as well as the method presented in Yang, Latecki, and Pokrajac (2009) which fits a Gaussian model using an expectation maximization algorithm. *Non-parametric* methods do not assume prior knowledge on the data distribution. Among them the *kernel density estimation* uses kernel functions to identify outliers as those points whose local density is not aligned to the local densities of the neighbors (Pavlidou & Zioutas, 2014). In a similar fashion, *density-based* approaches leveraging the use of nearest neighbors algorithms instead of kernel-based estimations, identify outliers as those points having low-density neighborhood, see *e.g.*, the *local outlier factor* algorithm (Breunig, Kriegel, Ng, & Sander, 2000). Several other outlier detection methods are available in the literature. An exhaustive review is beyond the scope of this section, hence we refer the reader to the extensive surveys provided in Hodge and Austin (2004) and Wang, Bah, and Hammad (2019).

Other methods exploit robust loss functions. Regression performed with the LS criterion can be robustified by adopting the *least-trimmed squares* (LTS) paradigm (Rousseeuw, 1984), obtained from (1)–(5) by setting $\ell(x) = x^2$ and $d_0 = d$. The LTS is satisfactorily solvable through fast heuristic algorithms, see *e.g.*, the R `Studio` package `fastLTS` implementing the approach of Rousseeuw and Van Driessen (2006). The *penalized trimmed square* (PTS) approach (Zioutas, Pitsoulis, & Avramidis, 2009) is obtained from the LTS by relaxing the outlier budget constraint and augmenting the objective function with a penalization term for each point, so that points producing residuals larger than the penalization are discarded. Alternative approaches do not involve direct control on the outlier selection and adopt robust loss functions. Among them we mention *Huber's loss function* and the *Tukey's biweight loss function* which are both convex, see the literature survey in Thompson (2022) for these and additional robust loss functions. A similar approach is to replace the LS loss function with the LAD loss function, which is generally more robust as it places less weight on large deviations (Hastie et al., 2009, pp. 349–350). Moreover, under mild conditions, the LAD criterion is *provably* immune to vertical outliers (Dodge, 1997). Based on this latter observation, the appealing idea of the *leveraged least-trimmed absolute deviation* (LLTA) method (Sudermann-Merx & Rebennack, 2021) is to solve problem (1)–(5) in which $d_0 = d$ (there is no feature selection), $\ell(x) = |x|$ (a LAD loss function is considered) and only a subset of switch variables s are used, making the resolution of the corresponding MIP faster. The used switch variables correspond to potential bad leverage points, as detected by a statistical preprocessing step. The LLTA method is described in greater detail in Section 4.3, because we compare our methods with a modification of the LLTA method embedding the feature selection task.

Finally, we review methods attempting to obtain simultaneously sparsity and robustness. One approach is to combine the LASSO idea (for sparsity) with the LAD criterion (for robustness), thus obtaining the so-called *LAD-LASSO* method (Wang et al., 2007). Variations of the LAD-LASSO method include the use of adaptive penalization coefficients in the feature shrinkage term (Xu & Ying, 2010) and the so-called *weighted* LAD-LASSO (WLAD) method (Arslan, 2012) where the residuals summed in the LAD loss function of the LAD-LASSO objective are weighted distinctly to obtain resistance against bad leverage points. A combination of the WLAD method with the SCAD penalization term mentioned above is studied in Wang and Zhu (2017).

Analogously, the *sparseLTS* method (Alfons, Croux, & Gelper, 2013) embeds the LTS idea within the LASSO framework. It is equivalent to the following optimization problem:

$$\min\{\sum_{i=1}^{k}(1 - s_i)\ell(r_i - a^i w - z) + k_0\lambda||w||_1 : ||s||_0 \leq k_0, s \in \{0,1\}^k, w \in \mathbb{R}^d, z \in \mathbb{R}, \}$$

for a suitably chosen $\lambda \geq 0$. A sparseLTS implementation is available in package `robustHD` (Alfons, 2021) of R `Studio` (RStudio Team, 2020). A similar LTS embedding in the elastic net LASSO has been studied in Kurnaz, Hoffmann, and Filzmoser (2018a) and is available in the R `Studio` package `enetLTS` (Kurnaz, Hoffmann, & Filzmoser, 2018b). Alternative approaches from the literature to get sparsity and robustness against outliers often work in a 2-stage fashion. For example, *robust principal components regression* and *robust partial least-squares regression* methods combine a robust dimension reduction with a subsequent robust regression on the set of selected explanatory variables (Filzmoser & Nordhausen, 2021, Sect. 3.2-3.3). All above-mentioned robust sparse estimators lack direct control on either the selection of the features (*e.g.*, LASSO-based methods) or the selection of the outliers (*e.g.,* BSS-derived methods).

In the present paper we elaborate on the linear regression with SFSOD defined by (1)–(5), which can be seen as a fully robustified BSS method. Formalizations of the linear regression with SFSOD as the optimization problem (1)–(5) are present in the literature (Chen, Caramanis, & Mannor, 2013), but only very recently some works have appeared in which such problem has been formalized *and solved* as a MIP. To our knowledge, such an approach to tackle the LS-SFSOD has been proposed in Jammal et al. (2020, 2021). In Jammal et al. (2020) a LASSO version is considered, that is, a 1-norm penalization term is included in the objective function. Instead, in Jammal et al. (2021) the LS-SFSOD is tackled by solving exactly a mixed-integer quadratic program (MIQP); to improve convergence, the latter is supplied with a warm-start heuristic solution obtained through a *proximal alternating minimization* (PALM) algorithm. The results of Jammal et al. (2020, 2021), together with similar approaches in the domain of support vector machine, are also reported in Jammal (2020). The MIQP used in Jammal et al. (2020, 2021) is independently found and used in two other works (Insolia et al., 2021; Thompson, 2022). Both those studies provide an analysis of the breakdown point of the proposed LS-SFSOD models; moreover, (Thompson, 2022) provides a primal heuristic algorithm alternative to the above-mentioned PALM algorithm, which is based on a gradient descent technique.

The computational outcome of these works is consistent: when compared to classical methods in sparse or robust regression, solving the LS-SFSOD as a MIP provides solutions of high quality in terms of sparsity and robustness, at the price of a greater computational effort.

We also stress that all the above-mentioned works dealing with the SFSOD consider the LS setting for their experimental evaluation. There is therefore a clear uncovered research ground on the LAD setting, which in turn is the subject of investigation in our paper.

On handling bilinear terms in MIPs. When considering the SFSOD approach, a key role is taken by the handling of the product of terms in the objective function (1). In the LAD setting, in fact, they become bilinear terms: linearization techniques are known in the literature to work well in this context. One of them relies on McCormick's envelopes (McCormick, 1976). It replaces each single quadratic term $z = xy$ by a set of linear inequalities allowing the resolution of the initial MIP through standard mixed-integer linear programming (MILP) solvers. Such approach typically needs bounds on the variables appearing in the product hence it introduces big-M values in the resulting MILP. The *spatial branch-and-bound* paradigm tightens the McCormick's envelopes locally to each branch-and-bound node, by exploiting bounds implied by the branching steps. In the context of generic bilinear MIPs a regular spatial branch-and-bound scheme can be improved by means of non-standard branching rules and by separating valid intersection cuts (Fischetti & Monaci, 2020). For the cases of bilinear terms involving the product of one continuous and one integer variable a binary expansion method of the integer variables produces an exact integer linear reformulation which proves to be superior to McCormick's one both theoretically and computationally (Gupte, Ahmed, Cheon, & Dey, 2013). State-of-the-art solvers typically offer built-in functionalities to solve optimization problems with specific bilinear terms without relying on explicit linearizations of such terms. For example, Gurobi 10.1 (Gurobi Optimization, 2023) has native support for bilinear terms in the objective function. Moreover, bilinear terms with one binary and one generic bounded variable can be reformulated in Gurobi as well as CPLEX (IBM ILOG, 2020) through the so-called *Special Ordered Sets of type 1* (SOS-1), that is, expressions of the type $(x, y) \in$ SOS-1 imposing that at most one of variables x and y can have a nonzero value in the solution (Bertsimas & Weismantel, 2005). Indeed the identity $z = xy$ with $x \in \{0, 1\}$ and $y \in \mathbb{R}$ is equivalent to the pair of SOS-1 constraints $(x, z-y) \in$ SOS-1 and $(1-x, z) \in$ SOS-1. Such constraints are managed by the supporting solvers through specific branching rules thus avoiding big-M terms. We also recall that SOS-1 constraints can be used to model 0-norm constraints as (2) and (3), see *e.g.,* (Bertsimas et al., 2016). In particular the LAD version of MIP (1)–(5) can be modeled in both CPLEX and Gurobi using SOS-1 constraints without using big-M values.

In the same setting of products between one binary and one generic bounded variable another reformulation approach is through the so-called *indicator constraints*: these latter enforce conditions linked to the value of the binary variable in an "on/off" fashion, so that the relation $z = xy$ with $x \in \{0, 1\}$ amounts to specify the two indicator constraints $(x = 0 \Rightarrow z = 0)$ and $(x = 1 \Rightarrow z = y)$. When, as in this case, the conditions amount to satisfy linear constraints, solvers internally reformulate indicator constraints in an extended space and linearize them following McCormick's technique, and the involved variable bounds are automatically tightened during branch-and-bound algorithms (Belotti et al., 2016).

Finally, we report that the use of disjunctive programming (Balas, 1998) to linearize specific indicator constraints in the space of natural variables, also when the underlying condition is nonlinear, showed great potential in the literature (Bonami, Lodi, Tramontani, & Wiese, 2015; Hijazi, Bonami, Cornuéjols, & Ouorou, 2012).

3. Formulations and properties

Formally, the LAD-SFSOD problem is obtained by taking $\ell(x) = |x|$ in (1) and thus it amounts to solve

$$\min \left\{ \frac{1}{k} \sum_{i=1}^{k} s_i |r_i - a^i w - z| : s, w, z \text{ satisfy (2)–(5)} \right\} \qquad \text{(LAD-SFSOD)}$$

Given points $p^1 = (a^1, r_1), p^2 = (a^2, r_2), \dots, p^k = (a^k, r_k)$ as in Section 2, we define A as the *design matrix* (whose rows are $a^1, a^2, \dots, a^k$), and $r = (r_1, r_2, \dots, r_k)$ in $\mathbb{R}^k$ as the (column) *response vector*.

Let $(\hat{s}, \hat{w}, \hat{z})$ be a feasible solution to (LAD-SFSOD) and let $\hat{A}$ and $\hat{r}$ be obtained from A and r by removing all k_0 rows indexed by $h \in \{1, 2, \dots, k\}$ such that $\hat{s}_h = 0$. Then the value of $(\hat{s}, \hat{w}, \hat{z})$ in (LAD-SFSOD) corresponds to $||\hat{r} - \hat{A}w - \hat{z}\mathbf{1}||_1$ where $||\cdot||_1$ is the 1-norm and $\mathbf{1}$ is the all 1's column vector of the conforming dimension. For every $i \in \{1, 2, \dots, k\}$, a point $p^i = (a^i, r_i)$ is labeled as an outlier when $s_i = 0$ while it is labeled as authentic if $s_i = 1$. With this nomenclature we conclude that an optimal solution to (LAD-SFSOD) amounts to selecting, among all possible combinations, at most k_0 outliers such that the hyperplane $\hat{\mathcal{H}} = \{(x, y) \in \mathbb{R}^{d+1} : y = \hat{w}x + \hat{z}\}$ provides the minimum least absolute deviation from the set of authentic points. Moreover, the hyperplane $\hat{\mathcal{H}}$ must be sparse in the sense that $\hat{w}$ must contain at most d_0 nonzero entries, corresponding to features which are *retained*. The remaining features are *discarded*.

In order to solve (LAD-SFSOD) with commercial MILP solvers it is convenient to linearize its objective function and the 0-norm constraints (2) and (3). We proceed in two steps. We first linearize the 0-norm constraints and the absolute value terms in the objective function using the same approach considered in Insolia et al. (2021), Jammal et al. (2021) and Thompson (2022); next, we focus on the linearization of the products in the objective function. To perform the first step we assume to know upper- and lower-bound vectors $W^U = (W_1^U, W_2^U, \dots, W_d^U)$ and $W^L = (W_1^L, W_2^L, \dots, W_d^L)$ such that $W_j^L \le \hat{w}_j \le W_j^U$ for every $j = 1, 2, \dots, d$ and for every optimal solution $(\hat{s}, \hat{w}, \hat{z})$ to (LAD-SFSOD).

Then, we introduce binary variables $f \in \{0, 1\}^d$, each f_j taking value 1 if feature j is retained, 0 if it is discarded, and the continuous variables $p \ge \mathbf{0}$, each p_i upper-bounding the value of the residual on point p^i. Consider the following MIP:

$$(\mathcal{F}) \quad \min \frac{1}{k} \sum_{i=1}^{k} s_i p_i \tag{6}$$

$$p_i \ge r_i - a^i w - z \qquad \forall i = 1, 2, \dots, k \tag{7}$$

$$p_i \ge z + a^i w - r_i \qquad \forall i = 1, 2, \dots, k \tag{8}$$

$$w_j \ge W_j^L f_j \qquad \forall j = 1, 2, \dots, d \tag{9}$$

$$w_j \le W_j^U f_j \qquad \forall j = 1,2,\dots,d \tag{10}$$

$$\sum_{j=1}^{d} f_j \le d_0 \tag{11}$$

$$\sum_{i=1}^{k} s_i \ge k - k_0 \tag{12}$$

$$s \in \{0,1\}^k \tag{13}$$

$$f \in \{0,1\}^d \tag{14}$$

$$p \ge \mathbf{0}, w \in \mathbb{R}^d, z \in \mathbb{R} \tag{15}$$

Proposition 1. *The MIP (F) is equivalent to (LAD-SFSOD).*

Indeed, let $(p^\star, f^\star, s^\star, w^\star, z^\star)$ be a solution to (6)–(15). Since it is a minimization problem, constraints (7)–(8) imply that $p_i^\star = |r_i - w^\star a^i - z^\star|$ for every $i = 1,2,\dots,k$ such that $s_i^\star = 1$; moreover, by (9)–(10) we get that $w_j^\star \neq 0$ only if $f_j^\star = 1$, hence (11) guarantees that $w^\star$ has at most d_0 nonzero entries, that is, $||w^\star||_0 \le d_0$. Since $s^\star \in \{0,1\}^k$ we also immediately get $||s^\star||_0 \ge k - k_0$ from (12). Therefore $(s^\star, w^\star, z^\star)$ is a feasible solution to (LAD-SFSOD) of value $\sum_{i=1}^k s_i^\star |r_i - w^\star a^i - z^\star|$; analogously, if $(\hat{s}, \hat{w}, \hat{z})$ is a solution to (LAD-SFSOD) we can define $\hat{f}_j = 0$ if and only if $\hat{w}_j = 0$ for every $j = 1,2,\dots,d$ and $\hat{p}_i = |r_i - \hat{w}a^i - \hat{z}|$ for every $i = 1,2,\dots,k$. Since W^L and W^U provide valid lower- and upper-bounds on the entries of $\hat{w}$ we get that $(\hat{p}, \hat{f}, \hat{s}, \hat{w}, \hat{z})$ satisfies (9)–(15), showing the equivalence.

Observation 1. *We point out that, without additional information on the structure of features (e.g., mandatory/forbidden simultaneous selection of feature subsets), the difficulty of solving (F) mostly depends on the big-M values used in constraints (9) and (10). Indeed, even slightly loose big-M values in (9)–(10) allow solutions $(\tilde{w}, \tilde{z})$ of the continuous relaxation of (F) with all nonzero $\tilde{w}_j$'s. This is opposed to the outlier detection part, which benefits from the presence of the switch variables in the objective function (so that it is always convenient to set a switch variable to 0 while the budget constraint is satisfied).*

3.1. Objective function linearizations and theoretical analysis

All constraints of the MIP (F) are linear. However, its objective function is bilinear and, in general, non-convex. Therefore in this section we consider two linearizations for the bilinear terms in (1). Both linearizations involve the use of big-M values. For notational convenience, we define $\text{res}_i(w,z) = |r_i - wa^i - z|$ for every $i = 1,2,\dots,k$; that is, $\text{res}_i(w,z)$ is the residual of hyperplane $\mathcal{H} = \{(x,y) \in \mathbb{R}^{d+1} : y = wz + z\}$ with respect to point p^i. We assume to know a vector $R^U = (R_1^U, R_2^U, \dots, R_k^U)$ of *valid* upper bounds on the residuals of an optimal solution $(s^\star, w^\star, z^\star)$ to (LAD-SFSOD), that is, $\text{res}_i(w^\star, z^\star) \le R_i^U$ for every $i = 1,2,\dots,k$.

The first linearization of (6) we consider is a refinement of a standard technique based on disjunctive programming. Let us fix $i \in \{1,2,\dots,k\}$ and consider an auxiliary variable $x_i \ge 0$ defined by $x_i = s_i p_i$; since $s_i \in \{0,1\}$ we may restrict to consider the solutions $(\hat{x}, \hat{p}, \hat{f}, \hat{s}, \hat{w}, \hat{z})$ such that one of the two cases holds: either $\hat{s}_i = 1$ and hence $0 \le \hat{x}_i = \hat{p}_i \le R_i^U$; or $\hat{s}_i = 0$ and hence $\hat{x}_i = 0$ and $\hat{p}_i = R_i^U$. The latter holds because when $\hat{s}_i = \hat{x}_i = 0$ defining $\hat{p}_i = R_i^U$ necessarily satisfies all constraints (7)–(15) and does not have any impact on the objective function. Equivalently, this disjunction is expressed by $(\hat{x}_i, \hat{p}_i, \hat{s}_i) \in P_0^i \cup P_1^i$ where $P_0^i = \{(0, R_i^U, 0)\}$ and $P_1^i = \{(\pi, \pi, 1) : 0 \le \pi \le R_i^U\}$ for every $i = 1,2,\dots,k$; since P_0^i is a single point in $\mathbb{R}^3$, while P_1^i is a segment, the convex hull of their union is easily determined as:

$$\begin{aligned}&\text{conv}(P_0^i \cup P_1^i)\\ &= \{(x_i, p_i, s_i) : x_i = p_i - R_i^U(1 - s_i), 0 \le x_i \le p_i \le R_i^U, 0 \le s_i \le 1\}.\end{aligned}$$

Projecting out the x variables we get the following disjunctive-based linearization of (LAD-SFSOD):

$$(D)\ \min \frac{1}{k}\sum_{i=1}^{k}(p_i - (1 - s_i)R_i^U) \tag{16}$$

$$p_i \ge R_i^U(1 - s_i) \qquad \forall i = 1,2,\dots,k \tag{17}$$

$$p_i \le R_i^U \qquad \forall i = 1,2,\dots,k \tag{18}$$

$$(p, f, s, w, z) \text{ satisfy (7)–(15).} \tag{19}$$

Observation 2. *Since (D) is a minimization problem, an optimal solution will automatically satisfy constraints (18). Therefore these latter may be omitted from (D) without affecting the correctness of the model. We include them since they appear explicitly in the description of $\text{conv}(P_0^i \cup P_1^i)$ given above.*

We point out the following relations to exist between the disjunctive-based formulation (D) and the linearization of (6) via McCormick's envelopes.

Observation 3. *The following results hold true:*

- *the polyhedron corresponding to the continuous relaxation of (D) is a face of that corresponding to the continuous relaxation of McCormick's reformulation of (F);*
- *the optimal continuous relaxation value of (D) and that of McCormick's reformulation coincide, even if the set of feasible solutions does not;*
- *by fixing to 0 a switch variable s_i for some $i \in \{1,2\dots,k\}$ formulation (D) automatically implies the constraint $p_i = R_i^U$, which does not hold for McCormick's reformulation (for which, in general $\mathbf{0} \le p \le R^U$ independently of the value of the s variables).*

For the sake of conciseness, here we do not report the McCormick's reformulation of (F) nor the formal derivation of the first two items in Observation 3. They can be found in the online Appendix A.

Since formulation (D) is of smaller size than McCormick's reformulation and the fixing in the last item of Observation 3 is a typical operation in several MIP resolution techniques (*e.g.*, branching in branch-and-bound methods), the disjunctive-based approach offers computational advantages *a priori*. In fact, as reported in Vielma (2015, Sect. 2.2), effective MIP formulations combine small size with strong LP bounds and with good behavior under branching. In this regard the disjunctive-based reformulation of LAD-SFSOD is superior to McCormick's one in all these aspects, hence we will focus on (D) from now on.

We now move to the second linearization of the objective function of (F). It arises from an adaptation to the LAD setting of the LS-SFSOD considered independently in Insolia et al. (2021), Jammal et al. (2021) and Thompson (2022). All these three works introduce variables which cancel the residuals in correspondence of outliers. We are not aware of alternative exact models of the LS-SFSOD in the literature. For the sake of structural comparison with the literature, we consider the corresponding LAD version, which can be obtained by using the loss function $\ell(x) = |x|$ instead of $\ell(x) = x^2$ in the model of Insolia et al. (2021), (Jammal et al., 2021) and Thompson (2022). This leads to the following linearization of (LAD-SFSOD):

$$(\mathcal{L})\ \min \frac{1}{k}\sum_{i=1}^{k} q_i \tag{20}$$

$$q_i + r_i - wa^i - z - t_i \ge 0 \qquad \forall i = 1,2,\dots,k \tag{21}$$

$$r_i - wa^i - z - t_i - q_i \le 0 \qquad \forall i = 1,2,\dots,k \tag{22}$$

$$t_i \ge -(1 - s_i)R_i^U \qquad \forall i = 1,2,\dots,k \tag{23}$$

$$t_i \le (1 - s_i)R_i^U \qquad \forall i = 1,2,\dots,k \tag{24}$$

$$(f, s, w, z) \text{ satisfy (9)–(15)} \tag{25}$$

$$q \geq \mathbf{0}, t \in \mathbb{R}^k. \tag{26}$$

Variables t, appearing in (20)–(26), are used to cancel the residual in correspondence of outliers, which in turn are selected by the switch variables s, while in $(\mathcal{D})$ this is done directly with the binary switch variables.

Our finding is that the disjunctive-based linearization is inherently different from the linearization $(\mathcal{L})$ proposed in the literature. Namely, we are able to theoretically prove that the continuous relaxation value of the disjunctive-based formulation is at least as strong as the continuous relaxation value of the literature MILP; next, we derive sufficient conditions on the bound vector R^U for the two continuous relaxation values to coincide; finally we experimentally show that by violating such a condition we can construct instances on which the continuous relaxation of the disjunctive-based formulation is strictly stronger. To fix notation let $\bar{\mathcal{D}}_{R^U}$ and $\bar{\mathcal{L}}_{R^U}$ be the polytopes arising from the continuous relaxations of $(\mathcal{D})$ and $(\mathcal{L})$ respectively and let $v(\bar{\mathcal{D}}_{R^U})$ and $v(\bar{\mathcal{L}}_{R^U})$ indicate the corresponding continuous relaxation values. The following result holds for every big-M vector R^U used in the above formulations (also non-valid ones).

Proposition 2. *$v(\bar{\mathcal{D}}_{R^U}) \geq v(\bar{\mathcal{L}}_{R^U})$ for every $R^U \geq \mathbf{0}$.*

Proof. Let $(\tilde{p}, \tilde{f}, \tilde{s}, \tilde{w}, \tilde{z}) \in \bar{\mathcal{D}}_{R^U}$; we determine $\tilde{q}, \tilde{t}$ such that $(\tilde{q}, \tilde{t}, \tilde{f}, \tilde{s}, \tilde{w}, \tilde{z}) \in \bar{\mathcal{L}}_{R^U}$ and $\sum_{i=1}^k (\tilde{p}_i - (1-\tilde{s}_i)R_i^U) = \sum_{i=1}^k \tilde{q}_i$, which therefore proves the claim.

To this end, we define $\tilde{q}_i = \tilde{p}_i - (1-\tilde{s}_i)R_i^U$. Then the equivalence of the objective functions is immediate. Moreover, from (17) it follows that $\tilde{q}_i \geq 0$. Finally, let $I_i = [r_i - \tilde{w}a^i - \tilde{z} - \tilde{q}_i, r_i - \tilde{w}a^i - \tilde{z} + \tilde{q}_i] \cap [-(1-\tilde{s}_i)R_i^U, (1-\tilde{s}_i)R_i^U]$ for every $i = 1, 2, \dots, k$. Note that $I \neq \emptyset$: from (7) we get $r_i - \tilde{w}a^i - \tilde{z} - \tilde{q}_i \leq (1-\tilde{s}_i)R_i^U$ while (8) implies $r_i - \tilde{w}a^i - \tilde{z} + \tilde{q}_i \geq -(1-\tilde{s}_i)R_i^U$. Let us pick $\tilde{t}_i \in I_i$ for every $i = 1, 2, \dots, k$. Then $(\tilde{q}, \tilde{t}, \tilde{f}, \tilde{s}, \tilde{w}, \tilde{z}) \in \bar{\mathcal{L}}_{R^U}$. □

We now present a sufficient condition for $v(\bar{\mathcal{D}}_{R^U}) = v(\bar{\mathcal{L}}_{R^U})$.

Proposition 3. *If $R_i^U \geq \max\{|r_i - wa^i - z| : (q, t, f, s, w, z)$ attains $v(\bar{\mathcal{L}}_{R^U})\}$ for $i = 1, 2, \dots, k$ then $v(\bar{\mathcal{D}}_{R^U}) = v(\bar{\mathcal{L}}_{R^U})$.*

Proof. In view of Proposition 2 we only need to prove that $v(\bar{\mathcal{D}}_{R^U}) \leq v(\bar{\mathcal{L}}_{R^U})$ when R^U is as in statement of Proposition 3. Let $(\bar{q}, \bar{t}, \bar{f}, \bar{s}, \bar{w}, \bar{z}) \in \bar{\mathcal{L}}_{R^U}$ attaining value $v(\bar{\mathcal{L}}_{R^U})$. Then $\bar{q}_j \leq \bar{s}_j R_j^U$ for every $j = 1, 2, \dots, k$. Indeed, suppose by contradiction that $\bar{q}_i > \bar{s}_i R_i^U$ for some $i \in \{1, 2, \dots, k\}$. Since $(\bar{q}, \bar{t}, \bar{f}, \bar{s}, \bar{w}, \bar{z})$ attains the minimum of (20) on $\bar{\mathcal{L}}_{R^U}$, constraints (21) and (22) imply $\bar{q}_i = |r_i - \bar{w}a^i - \bar{z} - \bar{t}_i|$; then $|r_i - \bar{w}a^i - \bar{z} - \bar{t}_i| > s_i R_i^U$. Let first $r_i - \bar{w}a^i - \bar{z} - \bar{t}_i \geq 0$. Then $R_i^U \geq r_i - \bar{w}a^i - \bar{z} > s_i R_i^U + \bar{t}_i$ since R_i^U is a valid upper bound on $|r_i - \bar{w}a^i - \bar{z}|$ by the hypothesis. Then $\bar{t}_i < (1-\bar{s}_i)R_i^U$. Let $\varepsilon = \min\{(1-\bar{s}_i)R_i^U - \bar{t}_i, \bar{q}_i - \bar{s}_i R_i^U\}$. We define $q_i' := \bar{q}_i - \varepsilon$, $t_i' := \bar{t}_i + \varepsilon$ and $q_j' = \bar{q}_j$, $t_j' = \bar{t}_j$ for every $j \neq i$. By the definitions, $\bar{q}_i > q_i' \geq 0$ and $r_i - \bar{w}a^i - \bar{z} - t_i' = q_i'$. Hence $(q', t', \bar{f}, \bar{s}, \bar{w}, \bar{z}) \in \bar{\mathcal{L}}_{R^U}$ and its value is strictly less than $v(\bar{\mathcal{L}}_{R^U})$, a contradiction. The case $r_i - \bar{w}a^i - \bar{z} - \bar{t}_i < 0$ is analogous. To conclude we define $\bar{p}_j = \bar{q}_j + (1-\bar{s}_j)R_j^U$ for every $j = 1, 2, \dots, k$. The fact that $\bar{q}_j \leq \bar{s}_j R_j^U$ for every $j = 1, 2, \dots, k$ implies that $\bar{p}_j \leq R_i^U$. All other constraints of (17)–(19) are easily seen to be satisfied by $(\bar{p}, \bar{s}, \bar{w}, \bar{z})$ and the latter obviously has value $v(\bar{\mathcal{L}}_{R^U})$. Thus $v(\bar{\mathcal{D}}_{R^U}) \leq v(\bar{\mathcal{L}}_{R^U})$. □

We recall the following.

Observation 4. *When R^U is a vector of valid upper bounds on the residuals of an optimal solution of (LAD-SFSOD), MILPs $(\mathcal{D})$ and $(\mathcal{L})$ are both valid reformulations of (LAD-SFSOD).*

In fact, if for some $i \in \{1, 2, \dots, k\}$ the value R_i^U does not satisfy the assumption in Proposition 3, inequality $q_i \leq s_i R_i^U$ is not implied by the constraints defining $\bar{\mathcal{L}}_{R^U}$. Therefore using them may lead to $v(\bar{\mathcal{L}}_{R^U}) < v(\bar{\mathcal{D}}_{R^U})$. In particular, this is the case of the bounds vectors R^U as those considered in Observation 4. We have actually observed experimentally this phenomenon to occur frequently, see Section 4.1.

3.2. Big-M computation

As reported in Thompson (2022) it "remains an open research question how to estimate provably correct parameters in the absence of any assumptions on" the design matrix. In fact, in the SFSOD context, it is nontrivial to determine big-M values (even loose ones) which do not cut off optimal solutions. As a consequence, the works adopting the SFSOD approach resort to the following heuristic method: first a primal solution $(\omega^\star, \zeta^\star)$ and the corresponding residuals $\text{res}_1^\star, \text{res}_2^\star, \dots, \text{res}_k^\star$ are computed; then (arbitrary) multipliers $m_1, m_2 \geq 1$ are selected, allowing to set $W_j^L = -m_1|\omega_j^\star|$ and $W_j^U = m_1|\omega_j^\star|$ for every $j = 1, 2, \dots, d$ and $R_i^U = m_2 \text{res}_i^\star$ for every $i = 1, 2, \dots, k$. A refinement of this technique exploiting multiple primal solutions is given in Insolia et al. (2021).

The approach just described is not exact since it could cut off the global minimum of (LAD-SFSOD) (or its LS counter-part), although it maximally improves the starting primal solutions without worsening its residuals nor changing its sparsity: if $w_j^\star = 0$ for some $j = 1, 2, \dots, d$ then also $W_j^L = W_j^U = 0$. More flexibility on the sparsity of the resulting vector is obtained in Jammal et al. (2021) and Thompson (2022) by setting $W_j^L = -m_1||w^\star||_\infty$ and $W_j^U = m_1||w^\star||_\infty$, but in this case all entries of W^L, as well as those of W^U, coincide.

Building on alternative ideas from the literature, and in particular on the heuristic approach presented in Bertsimas et al. (2016) to calculate W^L, W^U for the BSS in the overdetermined regime ($k > d$ with at least d points in general position), we proceed as follows. Let $(w^\star, z^\star)$ be hyperplane coefficients of a feasible solution to (LAD-SFSOD) and let $v^\star = ||r - Aw^\star - z^\star \mathbf{1}||_2^2$; for every $j = 1, 2, \dots, d$ we get W_j^U and W_j^L by solving the following pair of convex optimization programs:

$$W_j^L = \min w_j \quad ||r - Aw - z\mathbf{1}||_2^2 \leq v^\star \quad w \in \mathbb{R}^d, z \in \mathbb{R},$$

$$W_j^U = \max w_j \quad ||r - Aw - z\mathbf{1}||_2^2 \leq v^\star \quad w \in \mathbb{R}^d, z \in \mathbb{R}.$$

Since the domain of the above problems is a compact set of $\mathbb{R}^{d+1}$, they both admit finite optimal values. These latter can be computed analytically exploiting the convexity of the problem domain, see the formulas in the supplementary material of Bertsimas et al. (2016).

We remark that while the approach of Bertsimas et al. (2016) to the BSS allows an exact computation of valid big-M values, our adaptation to the LAD-SFSOD turns out to be heuristic: the 2-norm of the total residual of an optimal solution to the (LAD-SFSOD) could be larger than the one of a sub-optimal solution. Indeed, in an optimal solution to (LAD-SFSOD), large deviations can be cancelled by setting the corresponding switch variables to 0. We point out that, with this approach, $w_j^\star = 0$ does not necessarily imply $W_j^U = 0$ nor $W_j^L = 0$ for any $j \in \{1, 2, \dots, d\}$ and the bounds are not necessarily coinciding. We also point out that, when computing the big-M vectors W^L and W^U used in the (LAD-SFSOD) linearization, an alternative could be to replace the convex constraint in the above optimization problems with its 1-norm counter-part $||r - Aw - z\mathbf{1}||_1 \leq \bar{v}^\star$, where $\bar{v}^\star = ||r - Aw^\star - z^\star \mathbf{1}||$. This yields an alternative optimization problem that can be cast to a linear program. In our experiments we adopt the convex optimization-based approach for two reasons: first, since it admits an analytical solution, the computation of W^L and W^U takes constant time; second the big-M values obtained from the convex programs can be used for both the LAD- and LS-SFSOD reformulations with which we experiment in the subsequent sections.

We heuristically compute a bound vector R^U on the residuals in a similar manner. Namely, by first solving the following convex optimization programs for every $i = 1, 2, \dots, k$:

$$S_i^L = \min a^i w + z \qquad S_i^U = \max a^i w + z$$

Algorithm 1 Bisection algorithm for the refinement of big-M values.

Require:

- an incumbent solution $(w^\star, z^\star)$
- a starting big-M vector R^U
- design matrix and response vector A and y
- an iteration limit L
- a nonnegative real number $\mu \geq 1$
- an outlier budget k_0

$newR^U \leftarrow R^U/2$
for $\ell < L$ **do**
  if $|r_i - w^\star a^i - z^\star| \leq newR^U_i$ for $i = 1, 2, \dots, k$ **then**
    $R^U \leftarrow newR^U$
    $newR^U \leftarrow R^U/2$
  else
    $newR^U \leftarrow newR^U + (R^U - newR^U)/2$
  end if
  $\ell \leftarrow \ell + 1$
end for
Sort R^U by non-increasing value
for $i = 1, 2, \dots k_0$ **do**
  $R^U_i \leftarrow \mu R^U_i$
end for
reset the entries of R^U to the original positions
return R^U

$$||r - Aw - z\mathbf{1}||_2^2 \leq v^\star \qquad ||r - Aw - z\mathbf{1}||_2^2 \leq v^\star$$
$$w \in \mathbb{R}^d, z \in \mathbb{R}, \qquad w \in \mathbb{R}^d, z \in \mathbb{R}.$$

and then using $|r^i - a^i w - z| \leq |r^i| + \max\{|S_i^L|, |S_i^U|\} =: R_i^U$; in this case the convex programs defining S_i^L and S_i^U admit analytical solutions in both the underdetermined and overdetermined regimes - see the supplementary material of Bertsimas et al. (2016). Once we have computed the vector R^U we refine the values of its entries by using a bisection algorithm: the current values are halved if the incumbent solution remains feasible and are increased otherwise, until an iteration limit is reached; next, the k_0 largest values obtained in R^U in this way are multiplied by a large factor. See Algorithm 1 for the pseudo-code.

The above convex programming-based approaches to compute big-M values for our reformulations of the linear regression with SFSOD are useful when no information on the magnitude of the prediction errors and/or hyperplane coefficients of an optimal solution is available *a priori*. This is the case of the real-world dataset considered in Section 5.

4. Computational results on synthetic instances

In this section we evaluate branch-and-bound algorithms based on the formulations presented in Section 3. The comparison is performed from two standpoints. On one hand we consider an integer programming perspective: we measure the quality of the continuous relaxations of the proposed formulations as well as the computational effort in solving them to optimality. We compare such formulations with each other, and with an enhancement of the LLTA method (Sudermann-Merx & Rebennack, 2021) which includes feature selection. On the other hand we consider the prediction quality of the regression hyperplanes found as solutions to (LAD-SFSOD) by means of the formulations of Section 3. We compare such solutions with state-of-the-art methods for the robust sparse regression. In order to control the parameters affecting the performance of the considered algorithms we generated synthetic instances to perform the experiments in this section.

Instances. Let d and k be fixed. Each synthetic instance is a pair of one training and one testing dataset, each containing k points in $\mathbb{R}^{d+1}$. Five instances are created for each value of $k \in \{100, 150, 200\}$; even if they are not enough to provide a ultimate statistical proof of dominance between methods, we found them to offer a reasonable trade off between the CPU time required to perform our experiments and the insights coming from their results. Moreover, each instance depends on a coefficient feature vector $\omega \in \mathbb{R}^d$, on a signal-to-noise ratio (SNR) value α, on an outlier ratio $0 < \pi < 1/2$ and on response and design error means μ_r and μ_A. The design matrices of both datasets are initially constructed row-wise, drawing each row from a multi-variate normal distribution $\mathcal{N}(0, \mathbb{I})$ with $\mathbb{I}$ being the identity matrix. We observe that this specific generation criterion yields normalized data. Therefore no further standardization technique to increase numerical stability and improve bounds computation is needed. Given also $\sigma^2 = ||\omega||_2^2/\alpha$ the response vector is defined by $r_i = \omega a^i + \zeta + \varepsilon_i$ for every $i = 1, 2, \dots, k$ with ζ extracted from a $\mathcal{N}(0, 1)$ distribution and ε_i extracted from a $\mathcal{N}(0, \sigma^2)$ distribution. The ε_i's simulate the presence of noise in the measurements yielding the synthetic datasets. In a subsequent step we consider one of the two corruption schemes, only for the training dataset:

1. corruption only in the responses (*vertical outliers*);
2. corruption in both design matrix and responses (*bad leverage points*).

In both schemes, each point is independently chosen as outlier with a given probability π. For outlier points, the corresponding response entry is corrupted, changing it by a value extracted from a $\mathcal{N}(\mu_r, 1)$ distribution. Only in scheme 2, also the design matrix is corrupted by changing the entries in the rows corresponding to outliers by a value extracted from a $\mathcal{N}(\mu_A, 1)$ distribution. Only entries corresponding to nonzero coefficients in the feature vector are corrupted in this way. For both corruption schemes, we consider the hyperplane coefficient vector $\omega \in \mathbb{R}^d$ having its first 5 entries equal to 1, and all others equal to 0 and the parameters given in Table 1. Such parameters are essentially those used in Insolia et al. (2021) for the tests on synthetic instances. The only differences are in the definition of ω, whose first 5 entries are set to 2 in Insolia et al. (2021), and in the fact that here we focus on the overdetermined regime (having $k > d$). The following theorem sheds light on the robustness of the LAD fitting with respect to vertical outliers:

Table 1
Parameters used to generate our synthetic instances.

Parameter name	d	α	μ_r	μ_A	π
Parameter value	49	5	−10	10	0.1

Proposition 4 (*Dodge, 1997*). *If* $(\hat{w}, \hat{z})$ *is a solution to* $\min_{w \in \mathbb{R}^d, z \in \mathbb{R}} \sum_{i=1}^k |r_i - a^i w - z|$ *then it is also a solution to* $\min_{w \in \mathbb{R}^d, z \in \mathbb{R}} \sum_{i=1}^k |r'_i - a^i w - z|$ *provided that* $r'_i \geq a^i \hat{w} + \hat{z}$ *(resp.* $r'_i \leq a^i \hat{w} + \hat{z}$*) whenever* $r_i > a^i \hat{w} + \hat{z}$ *(resp.* $r_i < a^i \hat{w} + \hat{z}$*) for every* $i = 1, 2, \dots, k$.

In other words, the LAD fitting is automatically unaffected by vertical outliers, when the corruption noise on the response has the same sign of the measurement noise. However, our models do not rely on this assumption, allowing us to handle arbitrary adversarial corruptions as well. Our training instances, in fact, cover also this general case (including, in particular, cases where μ_r and the ε_i's have opposite sign). The synthetic instances used in this section are available on the online repository associated with this paper (Barbato & Ceselli, 2023).

Computation of W^L, W^U and R^U. Vectors W^L and W^U are computed via the resolution of convex optimization programs reported in Section 3.2. The feasible solution needed for the computation of W^L and W^U is obtained by the PALM algorithm of Jammal et al. (2020). The PALM algorithm was designed to produce a primal solution for the LS-SFSOD problem but such a solution is obviously also feasible for (LAD-SFSOD). Pseudo-code and convergence guarantees

Table 2
LP bound values of formulations of Section 3 and relative increase of $v(D)$ over $v(\mathcal{L})$. We consider LAD-SFSOD on instances corrupted with bad leverage points. Results are averaged over the 5 dataset replications of each dataset size k.

k	k_0 (%)	$\phi = 1$			$\phi = 1.2$			$\phi = 1.5$		
		$v(\mathcal{L})$	$v(D)$	Incr. (%)	$v(\mathcal{L})$	$v(D)$	Incr. (%)	$v(\mathcal{L})$	$v(D)$	Incr. (%)
100	10	37.45	38.09	1.69	30.58	31.04	1.49	23.81	23.86	0.19
	13	26.78	27.73	3.56	20.73	21.11	1.84	13.09	13.17	0.60
	17	20.91	21.75	4.03	14.34	14.74	2.78	7.83	7.90	0.84
	20	15.55	16.33	5.06	9.61	9.96	3.67	3.33	3.40	1.98
	Avg.			3.58			2.45			0.90
150	10	130.78	131.18	0.30	80.13	80.54	0.50	65.79	66.04	0.39
	13	76.29	77.44	1.51	63.59	64.26	1.05	49.16	49.51	0.70
	17	61.22	62.96	2.85	47.92	48.72	1.66	31.24	31.58	1.07
	20	52.39	54.24	3.54	38.32	39.12	2.07	23.01	23.41	1.72
	Avg.			2.05			1.32			0.97
200	10	119.55	121.46	1.60	105.27	105.97	0.67	90.91	90.92	0.01
	13	99.77	102.78	3.02	87.00	87.76	0.87	69.91	69.92	0.02
	17	85.34	88.36	3.53	70.20	70.96	1.08	49.24	49.26	0.05
	20	73.34	76.36	4.11	55.80	56.57	1.38	36.97	37.09	0.32
	Avg.			3.06			1.00			0.10

of such an algorithm can be found in Jammal et al. (2020). The convex programming-based approach for getting R^U turned out to produce very large bounds, resulting in numerical instability and poor performance of the branch-and-bound algorithms used in this section. Since we are interested in studying the strength of the formulations of Section 3 depending on the magnitude of the bounds in R^U, we proceed differently as follows. For each instance replication we define $E_i = |r_i - \omega a^i - \zeta|$ for every $i = 1, 2, \dots, k$, where r_i and a^i are the response and design row of the ith point in the instance after applying the corruption step. Vector $E = (E_1, E_2, \dots, E_k)$ represents the smallest bounds on the residuals of the authentic hyperplane on the training dataset points. In our experiments we will set $R^U = \phi E$ with $\phi \geq 1$ being a parameter allowing us to control the tightness of the big-M used in the MIP formulations of Section 3. Depending on the experiment, the values in R^U are refined using Algorithm 1 with iteration limit $L = 10$ and multiplier $\mu = 20$. We point out that using such big-M values could cut off the global optimum of (LAD-SFSOD) but ensure its feasibility.

Implementation details. The instance generation script is implemented in R. The MIP formulations presented in Section 3 are implemented using the C++ API of Gurobi 10.1 and solved with the branch-and-bound framework offered by the same optimization suite. Unlike formulations (D) and ($\mathcal{L}$) given above, in our implementation we omit the term $1/k$ in the objective function, since in preliminary tests we observed that it did not improve the computational performance. The redundant constraints (18) discussed in Observation 2 are implemented as upper bounds of variables p_i's so that Gurobi manages them automatically. Each run is performed in parallel mode on a Linux machine (Ubuntu 20.04.4) equipped with 32 GB of RAM and with a 6 cores 4.10 GHz CPU (model Intel(R) Xeon(R) W-1250P). The whole test session is performed in parallel using the `parallel` package of R (we parallelize over the feature budgets). We use the default setting of Gurobi except for the parameters `Cuts`, `CutPasses` and `MIPFocus` that are set to 3, 1000 and 2, respectively, after a side experimental investigation (Barbato, Bertoncini, & Ceselli, 2023). The former two parameters have the effect of producing a high number of generic cuts at the root node while the latter parameter drives the solver to produce strong dual bounds during the branch-and-bound execution. The primal solution produced by the PALM algorithm in the computation of the bound vectors W^L and W^U is provided to Gurobi as a warm-start. Since we consider increasing sequences of the feature budget, when possible, the model is reoptimized by exploiting the optimization performed in the previous step: we update the feature budget constraint of each model and provide Gurobi with the optimal solution of the previous step. In this way we avoid to reoptimize similar models from scratch.

Comparison to general purpose solvers. In a round of experiments, we have considered the straight use of built-in functionalities of MIP solvers. In details, in the online Appendix B we compare algorithm (D) with several alternative algorithms relying on built-in functionalities of CPLEX 20.1 and Gurobi 10.1 to model the non-convex terms in LAD-SFSOD, without explicitly linearizing them. In particular, this allows to express the LAD-SFSOD MIP without relying on the use of big-M values. In the datasets with 100 points and bad leverage outliers, the plain use of CPLEX solves 45 instances to proven optimality within a time limit of $1800s$, while our algorithm (D) solves 91 of them. Gurobi was not able to close any of them, neither in a configuration with bilinear objective and 0-norm constraints, nor in a configuration where 0-norm constraints have been linearized, nor in a configuration using indicator constraints. That is, we found such a straight use of MIP solvers to be far from effective.

4.1. Comparison of continuous relaxations

In this section we experimentally compare the continuous relaxation of formulations (D) and ($\mathcal{L}$). Proposition 2 states the former to never be weaker than the latter. Our main finding is the following.

Experimental Observation 1. *The continuous relaxation of the disjunctive-based formulation (D) is strictly stronger than the continuous relaxation of ($\mathcal{L}$), unless other assumptions on data are made (in which case they might be equal).*

We recall that according to Proposition 3 a necessary condition to guarantee such a result is that the value of R^U_i is small enough for at least one $i \in \{1, 2, \dots, k\}$. Experimentally, we compare the relative increase of the LP bound obtained by the disjunctive-based approach with respect to the literature approach, by considering sequences of vectors R^U having increasing entries. Let $v(D_{R^U})$ and $v(\mathcal{L}_{R^U})$ be the LP bound values of (D) and ($\mathcal{L}$) respectively. The relative increase of the LP bound is computed as $100 \cdot \frac{v(D_{R^U}) - v(\mathcal{L}_{R^U})}{v(\mathcal{L}_{R^U})}$. In this experiment we use $R^U = \phi E$ with $\phi = 1, 1.2, 1.5$; that is, we consider the vector with the smallest bounds on the residuals of the authentic hyperplane and two additional bound vectors whose entries are respectively increased by 20% and 50% with respect to the smallest ones. We consider datasets polluted with bad leverage points, that is, following scheme 2. For each instance replication we solve the continuous relaxations of both formulations (D) and ($\mathcal{L}$), where we consider all combinations of $d_0 = 5, 6, \dots, 10$ and $k_0 = \lfloor 0.1k \rfloor, \lfloor 0.13k \rfloor, \lfloor 0.17k \rfloor, \lfloor 0.2k \rfloor$ (corresponding to

about 10%, 13%, 17% and 20% of points selectable as outliers). This gives a grid of 24 parameter combinations for each considered formulation; since we have 5 dataset replications for each dataset of size k, we get a total of 120 tests for each value of $k \in \{100, 150, 200\}$. Averaging over all the tests executed in this section, the continuous relaxation of each formulation is computed in less than 0.06 CPU seconds.

The averaged computational results are provided in Table 2, while the detailed results are available on the online repository (Barbato & Ceselli, 2023). Interestingly, the continuous relaxation values do not depend on d_0 for any formulation. Therefore in Table 2 we report results according to the dataset size (column "k") and to the percentage of points selectable as outliers (column "k_0 (%)"). The remaining three columns give the LP bound values and the relative increase depending on the value of ϕ. An additional line ("Avg".) reports the average of the relative increases for each dataset size. We immediately observe that the continuous relaxation values of the disjunctive-based formulation are strictly larger than those of the literature formulation and that larger relative increases correspond to smaller values of ϕ. This is consistent with the theoretical findings of Section 3.1. The table also clearly shows that the relative increase improves when k_0 is larger. In general, the average relative increase worsens by increasing the dataset size, although some exceptions are present ($k = 200$ with $\phi = 1$ and $k = 150$ with $\phi = 1.5$).

4.2. LAD-SFSOD models: Comparison of MILP performances

In this section we analyze the computational performance of the branch-and-bound algorithms relying on formulations $(\mathcal{D})$ and $(\mathcal{L})$ of Section 3 (for short, we will indicate the algorithms by $(\mathcal{D})$ and $(\mathcal{L})$ as well). We use the same grid of combinations for k_0 and d_0 employed in Section 4.1. That is, for each corruption scheme, each algorithm is executed over a total of 360 runs (120 for each dataset size). Each run has a CPU time limit of 1800 s. We define $R^U = 2E$ for both MILPs and we refine the values in R^U by using Algorithm 1 with $L = 10$ and $\mu = 20$.

The results are presented here in a concise and qualitative way. Detailed results are provided in spreadsheets available on the online repository (Barbato & Ceselli, 2023). We consider the CPU time as the performance indicator of the algorithms. In Fig. 1 we provide performance profiles of $(\mathcal{D})$ (blue line) and $(\mathcal{L})$ (orange line) computed on the instances with vertical outliers and bad leverage points, respectively. A performance profile reports the number of solved tests (y-axis) within a given time bound (x-axis): an upper curve corresponds to a better performance. Then Fig. 1 leads to the following observation.

Experimental Observation 2. *Regardless of the corruption scheme, algorithm $(\mathcal{D})$ reaches optimality in more tests than algorithm $(\mathcal{L})$ within the same time bound.*

The above is in line with the theoretical observations made in Section 3.

Additional observations can be drawn from Fig. 1. The first one is expected: larger datasets yield LAD-SFSOD instances whose optimization is more time-consuming. Second, comparing the profiles obtained on the instances with vertical outliers with those obtained on the instances with bad leverage points, we deduce the following.

Experimental Observation 3. *Regardless of the LAD-SFSOD MIP, the instances with vertical outliers are more time-consuming and difficult to optimize than those with bad leverage points.*

In fact, we report that, within the time limit, algorithm $(\mathcal{D})$ solves to optimality 124 tests on instances with vertical outliers and 196 tests on instances with bad leverage points (out of 360 for each corruption scheme); algorithm $(\mathcal{L})$ solves to optimality 110 instances with vertical outliers and 158 with bad leverage points. As expected from Observation 1, another parameter affecting the difficulty of the instances is the feature budget. More precisely, we have the following.

Experimental Observation 4. *Regardless of the corruption scheme, instances with larger feature budget d_0 are more time-consuming.*

This is shown in Fig. 2. For every value d_0 of the feature budget and every corruption scheme it contains two boxplots, corresponding to algorithms $(\mathcal{D})$ and $(\mathcal{L})$. The boxplots describe the distribution of the CPU times employed by each algorithm to reach optimality on the 101 (resp. 154) tests on instances with vertical outliers (resp. bad leverage points) solved by *both* algorithms. The points overlapping with the boxplots are outcomes on individual tests. Fig. 2 clearly shows that for $5 \le d_0 \le 7$ the number of points overlapping with the boxplots is larger than for $8 \le d_0 \le 10$. Also, regardless of the corruption scheme, the median CPU time of $(\mathcal{D})$ increases as d_0 does; the same behavior is observed on the median CPU time of $(\mathcal{L})$, although, for this algorithm, the trend is more evident on instances with bad leverage points.

4.3. Comparison with the LLTA method

We also consider an extension of the LLTA method (Sudermann-Merx & Rebennack, 2021) to the feature selection problem. The LLTA method relies on the assumption that the LAD loss function is unaffected by vertical outliers and that the potential bad leverage points can be detected during a preprocessing phase. Given a multiplier $m > 1$ and a point p^i in the dataset, the preprocessing phase considers each of the features p^i_j $(j = 1, 2, \dots, d)$ and checks whether it exceeds by m times the extrema of the interquartile interval of the same feature in the dataset (see Sudermann-Merx & Rebennack, 2021 for details). If that is the case, p^i is a *potential* bad leverage point. Let $\mathcal{B}$ be the set of indices of the potential bad leverage points detected in that way. The LLTA method consists in solving model (LAD-SFSOD) in which $s_i = 1$ for every $i \notin \mathcal{B}$ and $d_0 = d$, that is, switch variables are assigned only to potential bad leverage points and there is no feature selection.

In Sudermann-Merx and Rebennack (2021), the LLTA model is implemented in Gurobi exploiting some its features to avoid the use of big-M values. In particular, the bilinear terms in the objective function of (LAD-SFSOD) are maintained and Gurobi deals with them automatically. Following the same spirit, we embed the feature selection task within the LLTA model by exploiting the possibility of specifying 0-norm constraints in that solver. The resulting formulation, corresponding to our actual implementation, is denoted (FS-LLTA) and reads:

$$\text{(FS-LLTA)} \quad \min \frac{1}{k} \sum_{i \in \mathcal{B}} s_i p_i + \sum_{i \notin \mathcal{B}} p_i \tag{27}$$

$$p_i \ge r_i - a^i w - z \qquad \forall i = 1, 2, \dots, k \tag{28}$$

$$p_i \ge z + a^i w - r_i \qquad \forall i = 1, 2, \dots, k \tag{29}$$

$$f = ||w||_0 \tag{30}$$

$$f \le d_0 \tag{31}$$

$$\sum_{i \in \mathcal{B}} s_i \ge k - k_0 \tag{32}$$

$$s \in \{0, 1\}^{|\mathcal{B}|} \tag{33}$$

$$f \in \mathbb{Z}_+ \tag{34}$$

$$p \ge \mathbf{0}, w \in \mathbb{R}^d, z \in \mathbb{R} \tag{35}$$

In our experiments with (FS-LLTA) we use $m = 1.5$ as in Sudermann-Merx and Rebennack (2021) but we point out that the multiplier should be tuned as changing its value can affect the performance of the method as well as the prediction quality of its solutions. Indeed, small sets $\mathcal{B}$ imply a small number of switch variables, hence, in general, a more easily solvable model (FS-LLTA), but also a more inaccurate outlier

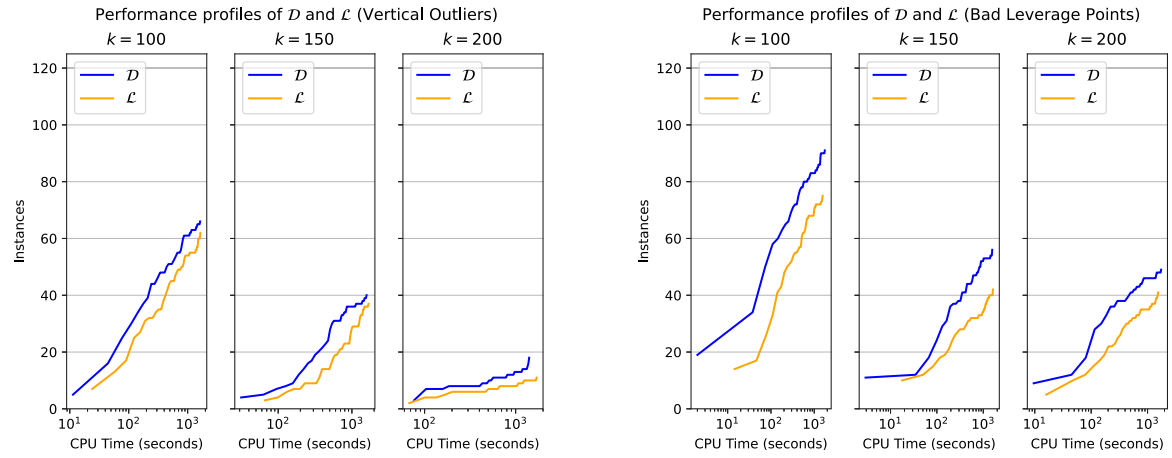

Fig. 1. Performance profiles of algorithms $(\mathcal{D})$ (blue line) and $(\mathcal{L})$ (orange line): the y-axis of each plot reports the number of solved tests on instances with vertical outliers (left) and bad leverage points (right) within a given CPU time (reported on the x-axis of each plot). For a given corruption scheme, each plot corresponds to the performance of the algorithms on the instances with k points ($k \in \{100, 150, 200\}$). In each plot all instances with feature budget $d_0 \in \{5, 6, 7, 8, 9, 10\}$ and outlier budget $k_0 \in \{10\%, 13\%, 17\%, 20\%\}$ are considered. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

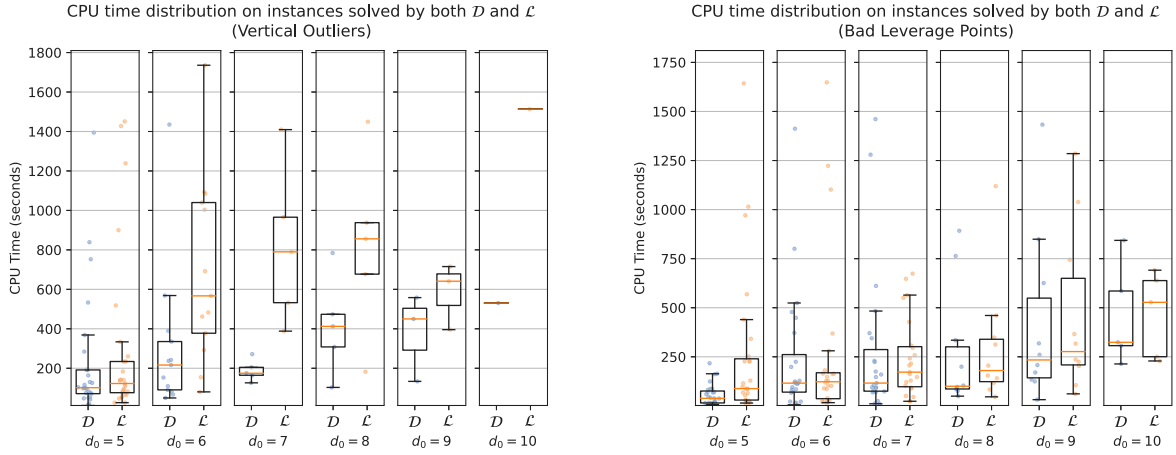

Fig. 2. Boxplots of the CPU times employed by algorithms $(\mathcal{D})$ and $(\mathcal{L})$ to solve tests on instances with vertical outliers (left) and bad leverage points (right).

detection. This is especially true when the interquartile range criterion used in preprocessing is not able to capture the authentic structure of the outliers, as it can happen in presence of specific "adversarial" outliers. Since this is the case in both corruption types affecting our synthetic instances, we test (FS-LLTA) on both sets of instances with vertical outliers and bad leverage points. The performance profiles of Fig. 3 show a qualitative comparison of the computational performance of (FS-LLTA) and $(\mathcal{D})$. The detailed computational results of (FS-LLTA) are available in spreadsheets on the online repository (Barbato & Ceselli, 2023). We see that (FS-LLTA) is able to solve a few tests within very short time bounds (during which $(\mathcal{D})$ does not solve any); when larger time bounds are considered $(\mathcal{D})$ exhibits a better performance. As it is clear from the discrepancy between the profiles of the two algorithms, $(\mathcal{D})$ solves more tests than (FS-LLTA), regardless of the corruption scheme. We explain this outcome as follows. In our experiments we also measured how many potential outliers are left by LLTA preprocessing. We found that after preprocessing, between 40 and 80 binary variables are left in the models, independently on the type of outliers affecting the instance. This leads to a high number of binary variables in model (FS-LLTA), whose resolution is made more difficult than that of $(\mathcal{D})$ by the presence of non-convex objective and constraints in the model.

The principles behind LLTA remain appealing. We further experiment the potential of them, also by means of an embedding of the LLTA preprocessing within our algorithm $(\mathcal{D})$. This is described and experimentally evaluated in Section 5 on a real-world dataset.

4.4. Comparison of prediction quality and solution sparsity

In this section we study the quality of the solutions produced by the algorithms described in the previous sections. We follow to a large extent the evaluation setting proposed in Insolia et al. (2021). We compare the solutions obtained from formulations of Section 3 with those produced by the R packages `enetLTS` (v. 1.1.0) and `sparseLTS` (v. 0.7.2) mentioned in Section 2 (the complete parameter specification of these methods is provided in the online Appendix D). Both are much faster than MIP approaches (see Appendix D for details). For completeness, we include in the comparison method $(\mathcal{L})_2$ which is obtained from $(\mathcal{L})$ by using the LS loss as objective function. That is, $(\mathcal{L})_2$ coincides with the LS-SFSOD model used in Insolia et al. (2021). The detailed computational results of $(\mathcal{L})_2$ are available in spreadsheets on the online repository (Barbato & Ceselli, 2023). We also consider the prediction made by the authentic hyperplane with the correct outlier detection. In the comparison table the corresponding solution is denoted *Authentic*. Its values are in italic and serve as baseline to evaluate the performance of the other algorithms. The comparison is made evaluating the quality of the solutions produced by the considered methods. Since we deal with *simultaneous* feature and outlier detection, three measures are relevant: the quality of the prediction

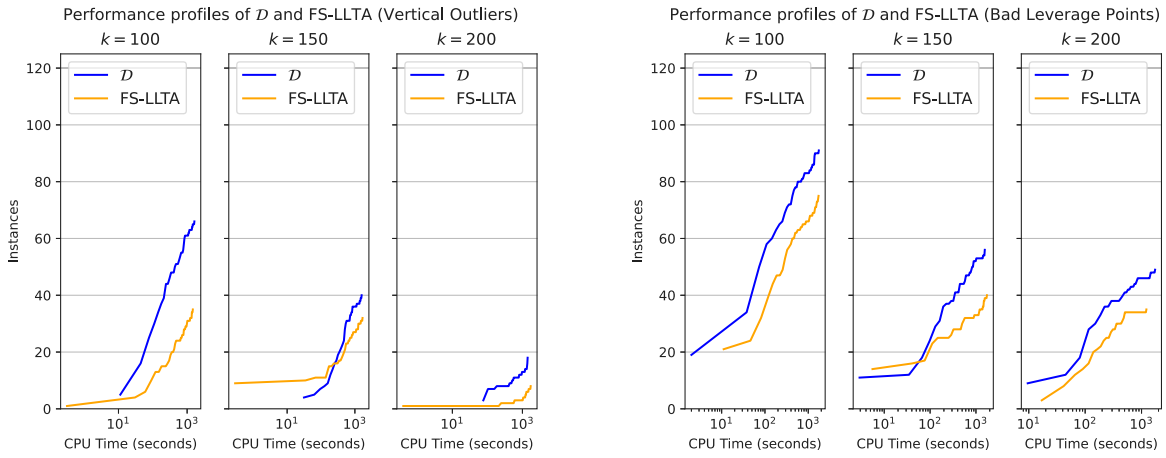

Fig. 3. Performance profiles of algorithms ($\mathcal{D}$) (blue line) and (FS-LLTA) (orange line): number of solved instances with vertical outliers (left) and bad leverage points (right) within a given CPU time.

over the testing dataset (which, we recall, is not corrupted), the sparsity level of the fitted hyperplanes and the accuracy of the coefficients of the fitted hyperplanes. Let $p^1 = (a^1, r_1), p^2 = (a^2, r_2), \dots, p^k = (a^k, r_k)$ be the points in a testing dataset and let $(\hat{\omega}, \hat{\xi})$ be a solution obtained from one of the above-mentioned methods. The first measures we define are:

- *rooted mean squared prediction error* (RMSPE) defined by $\text{RMSPE} = \left(\frac{\sum_{i=1}^{k}(r_i - \hat{\omega}a^i - \hat{\xi})^2}{k}\right)^{1/2}$
- *mean absolute error* (MAE) defined by $\text{MAE} = \frac{\sum_{i=1}^{k}|r_i - \hat{\omega}a^i - \hat{\xi}|}{k}$
- *false positive ratio* (FPR) and *false negative ratio* (FNR) for the feature selection defined by $\text{FPR} = \frac{|\{j=1,2,\dots,d : \hat{\omega}\neq 0 \text{ and } \omega = 0\}|}{|\{j=1,2,\dots,d : \omega = 0\}|}$ and $\text{FNR} = \frac{|\{j=1,2,\dots,d : \hat{\omega}= 0 \text{ and } \omega\neq 0\}|}{|\{j=1,2,\dots,d : \omega\neq 0\}|}$ where ω is the authentic vector of hyperplane coefficients.
- *F1-score* for the feature selection: $\text{F1} = (1-\text{FNR})/(1-\text{FNR}+(\text{FPR}+\text{FNR})/2)$

For each dataset size, let also $(\hat{\omega}^1, \hat{\xi}^1), \dots, (\hat{\omega}^5, \hat{\xi}^5)$ be the solutions obtained on the 5 corresponding replications. Their average is defined by $(\bar{\omega}, \bar{\xi}) = \sum_{h=1}^{5}(\hat{\omega}^h, \hat{\xi}^h)/5$. Given the authentic hyperplane ω, in the tables we additionally report the *mean squared error* (MSE) subdivided in *MSE bias* defined by $\frac{\sum_{j=1}^{d}(\omega_j - \bar{\omega}_j)^2}{d}$ and *MSE variance* defined by $\frac{\sum_{j=1}^{d}\sum_{h=1}^{5}(\hat{\omega}_j^h - \bar{\omega}_j)^2}{5d}$.

Before presenting the results, we recall that our branch-and-bound algorithms for the LAD- and LS-SFSOD are run on each training dataset by varying d_0 and k_0 over a grid of parameters, thus producing a grid of solutions; then we select only one of them which is subsequently used to perform the prediction on the corresponding testing dataset. To this end we use a method based on a "robustified" Akaike information criterion (Akaike, 1974), taken from the implementation available in Insolia et al. (2021). A candidate solution $(w^\star, z^\star, s^\star)$ is selected if it minimizes the quantity $\kappa\delta + \kappa\log(L)$, where κ and δ denote the number of non-outliers and of nonzero features in the solution (including the intercept, so that for non-degenerate instances $\kappa = k_0$ and $\delta = d_0+1$), A' and r' are the restrictions of the design matrix and response vector to the κ rows corresponding to the non-outliers and $L = ||r' - w^\star A' - z^\star \mathbf{1}||_2^2/\kappa$ is the average prediction error (under the LS criterion) on the non-outlier training points. A similar approach is used to select one solution of `sparseLTS` and `enetLTS` per dataset replication. In our experience, the robustified Akaike information criterion (RAIC from now on) privileges sparse solutions with good predictive power more than other information criteria that we have tested.

In Table 3 we present the results in both settings involving vertical outliers (left part) and bad leverage points (right part). For each approach, the values reported in the table are averages of the quality measures above. The averages are computed over the 5 solutions (one per training dataset replication) selected via the RAIC, each evaluated on the corresponding testing dataset. For each quality measure and each dataset size we highlight in boldface the best corresponding result. A high prediction quality is related to low values of RMSPE and MAE. The results in Table 3 show that, in terms of prediction quality, all MIP-based methods except (FS-LLTA) yield similar results and better than those of `enetLTS` and `sparseLTS`. Among MIP-based methods, (FS-LLTA) yields the worst RMSPE values for $k = 100, 150$. This is due to two combined factors: first, the difficulty in solving (FS-LLTA) to optimality, as already discussed in Section 4.3; second, the lack of full control on the outlier detection, resulting in a less accurate estimation of the feature coefficients, especially when the number of points is low.

High accuracy in recovering the coefficients of the authentic hyperplane corresponds to low values of FPR and FNR. The FNR is identically 0 for all methods, meaning that the authentic nonzero features are identified correctly by all methods. At the same time, only the MIP-based approaches and `sparseLTS` yield solutions with a FPR close or equal to 0, while `enetLTS` reports a FPR of at least 30%. Since the FPR relates to features wrongly assigned with a nonzero coefficient, this means that the predicting power of `enetLTS` comes at the price of producing non-sparse hyperplanes (we recall that the authentic hyperplane coefficients has only 5 nonzero entries and 44 zero entries).

An analogous analysis is made in the right part of Table 3 for datasets corrupted with bad leverage points. In terms of prediction quality the MIP-based approaches clearly outperform `enetLTS` and `sparseLTS`. Within the MIP-based approaches, (FS-LLTA) has the worst RMSPE for all dataset sizes. Again, this is explained by the less accurate outlier selection as well as by the reduced optimization potential of (FS-LLTA) model which, we recall, is not linearized. Similar conclusions are obtained when considering the values of the FPR and FNR. Therefore we have the following.

Experimental Observation 5. *Regardless of the corruption scheme, the MIP-based approaches with direct control of the feature selection and outlier detection are the best methods in combining prediction and feature selection quality.*

Discussion. Given the above analysis, as well as the considerations on the computational performance of the MIP-based approaches, we draw the following conclusions: in the case of low computational resources, `enetLTS` and `sparseLTS` are viable, although they may yield inaccuracies in the hyperplane reconstruction and, as a consequence, higher prediction errors than the MIP-based methods. When higher computational resources are available, the MIP-based approaches with direct control on the feature selection and outlier detection should be

Table 3
Prediction and feature selection quality of several SFSOD-based algorithms. The values reported in each column are averaged over 5 replications.

k	Method	Vertical outliers							Bad leverage points						
		RMSPE	MAE	FPR	FNR	F1	MSE bias	MSE variance	RMSPE	MAE	FPR	FNR	F1	MSE bias	MSE variance
100	LAD-SFSOD (D)	**1.15**	**0.92**	**0.00**	0.00	**1.00**	**0.00**	0.01	1.04	0.84	**0.00**	**0.00**	**1.00**	**0.00**	**0.03**
	LAD-SFSOD $(\mathcal{L})$	1.18	0.95	**0.00**	0.00	**1.00**	**0.00**	0.01	1.06	0.85	**0.00**	**0.00**	**1.00**	**0.00**	**0.03**
	LS-SFSOD $(\mathcal{L})_2$	1.16	0.93	**0.00**	0.00	**1.00**	**0.00**	0.01	**1.03**	**0.83**	**0.00**	**0.00**	**1.00**	**0.00**	**0.03**
	(FS-LLTA)	1.17	0.94	0.01	0.00	**1.00**	**0.00**	0.01	1.15	0.93	0.05	**0.00**	0.97	**0.00**	0.04
	`enetLTS`	1.21	0.97	0.30	0.00	0.87	**0.00**	0.01	2.97	2.38	0.30	0.33	0.66	0.14	0.05
	`sparseLTS`	1.31	1.05	0.02	0.00	0.99	0.01	0.01	1.65	1.32	0.07	0.10	0.90	0.03	0.06
	Authentic	*1.05*	*0.83*	*0.00*	*0.00*	*1.00*	*0.00*	*0.00*	*0.98*	*0.78*	*0.00*	*0.00*	*1.00*	*0.00*	*0.00*
150	LAD-SFSOD (D)	**1.03**	**0.82**	**0.00**	0.00	**1.00**	**0.00**	0.02	**1.07**	**0.87**	**0.00**	**0.00**	**1.00**	**0.00**	**0.01**
	LAD-SFSOD $(\mathcal{L})$	1.04	0.83	**0.00**	0.00	**1.00**	**0.00**	0.02	1.09	0.89	**0.00**	**0.00**	**1.00**	**0.00**	**0.01**
	LS-SFSOD $(\mathcal{L})_2$	**1.03**	**0.82**	**0.00**	0.00	**1.00**	**0.00**	0.02	**1.07**	**0.87**	**0.00**	**0.00**	**1.00**	**0.00**	**0.01**
	(FS-LLTA)	1.13	0.90	0.01	0.00	**1.00**	**0.00**	0.02	1.11	0.90	0.02	**0.00**	0.99	**0.00**	**0.01**
	`enetLTS`	1.09	0.87	0.31	0.00	0.87	**0.00**	0.02	2.68	2.16	0.29	0.27	0.68	0.13	0.02
	`sparseLTS`	1.23	0.99	**0.00**	0.00	**1.00**	0.01	0.02	1.77	1.45	0.05	0.07	0.94	0.04	0.03
	Authentic	*0.98*	*0.78*	*0.00*	*0.00*	*1.00*	*0.00*	*0.00*	*1.03*	*0.84*	*0.00*	*0.00*	*1.00*	*0.00*	*0.00*
200	LAD-SFSOD (D)	**1.00**	**0.81**	**0.00**	0.00	**1.00**	**0.00**	**0.00**	1.04	**0.82**	**0.00**	**0.00**	**1.00**	**0.00**	0.02
	LAD-SFSOD $(\mathcal{L})$	**1.00**	**0.81**	**0.00**	0.00	**1.00**	**0.00**	**0.00**	1.05	0.83	**0.00**	**0.00**	**1.00**	**0.00**	0.02
	LS-SFSOD $(\mathcal{L})_2$	**1.00**	**0.81**	**0.00**	0.00	**1.00**	**0.00**	**0.00**	**1.03**	**0.82**	**0.00**	**0.00**	**1.00**	**0.00**	**0.01**
	(FS-LLTA)	1.05	0.85	0.01	0.00	0.99	**0.00**	0.01	1.07	0.86	0.01	**0.00**	0.99	**0.00**	0.02
	`enetLTS`	1.05	0.85	0.41	0.00	0.84	**0.00**	**0.00**	2.81	2.25	0.25	0.27	0.70	0.13	0.02
	`sparseLTS`	1.27	1.02	**0.00**	0.00	**1.00**	0.01	**0.00**	1.85	1.48	0.02	0.27	0.78	0.04	0.03
	Authentic	*0.98*	*0.78*	*0.00*	*0.00*	*1.00*	*0.00*	*0.00*	*1.01*	*0.80*	*0.00*	*0.00*	*1.00*	*0.00*	*0.00*

preferred; method (FS-LLTA) also yields good performance, provided that its preprocessing phase reliably captures the outlier structure of the training datasets. The above conclusions should be read in light of a few remarks. First, the results of Table 3 may (also substantially) change if other criteria are used to select the solutions. In the online Appendix C we report the results of our experiments with the standard Akaike and Bayesian information criterion, as well as with a "robustified" Bayesian information criterion: the main outcome is that only the disjunctive-based model (D) yields stable RMSPE values regardless of the used criterion, the number k of dataset points and outlier scheme; the selected solutions of the other LAD- and LS-SFSOD MIPs may be of worse quality.

Second, the results of each MIP-based approach in Table 3 correspond to the averaged performances of *one* solution per dataset replication. In particular, Table 3 do not show substantial differences between the LAD- and the LS-SFSOD models. Hence a natural question is whether additional insights can be obtained by analyzing the whole solution pool produced from the resolution of (D) and $(\mathcal{L})_2$. A detailed comparison between these two algorithms is reported in the online Appendix E. The main outcome is that on instances with vertical outliers, both D and $\mathcal{L}_2$ exhibit almost identical performances in terms of prediction error and outlier detection; on instances with bad leverage points, the performances of the two algorithms is similar but $\mathcal{L}_2$ is more unstable: in several cases it reports high RMSPEs and fails to detect correctly the authentic outliers.

5. Computational results on the student performance dataset

In this section we demonstrate the usage of the LAD-SFSOD approach in a social science application with privacy protection. We start from a clean dataset relating the performance in Mathematics of 395 Portuguese high-school students with several attributes (Cortez, 2014). Most of attributes concern the students' private life, while a few others cover the students' grades and diligence. The goal is to perform regression so to accurately predict the final grades of the students. Given the presence of sensitive data we impose that a small amount of features is used by the predictor, that is, we require high sparsity level.

Moreover, we assume that attributes of a small amount of students have been altered by an adversary: the final grade of these students will be unnaturally high despite their 1st semester grades are low. We require our predictor to perform accurate regression despite the presence of such outliers, that is, we demand robustness against these outliers. In our experiments we consider the LAD-SFSOD models (D) and (FS-LLTA) as well as a hybrid approach where model (D) is combined with the preprocessing phase of the LLTA method, so that several of its switch variables are set to 1.

Instances. Given the original Math student dataset (Cortez, 2014), we generate a new *corrupted* Math student dataset (CMSD, from now on) as follows: (a) remove the 4 categorical attributes concerning the mothers' and fathers' jobs (attributes `Mjob` and `Fjob`), the reason behind the school choice (attribute `reason`) and the students' guardians (attribute `guardian`); additionally remove the 2nd semester grade of each student (attribute `G2`); (b) lower to 2 the 1st semester grade (attribute `G1`) of the 15 students with the highest final grade (rows 1-15 in the dataset non-increasingly ordered by final grade); additionally lower to 10 the 1st semester grade of the subsequent 15 students with the highest final grade (rows 16-30 in the dataset non-increasingly ordered by final grade).

The removal of the categorical attributes allows us to perform a numerical regression eliminating since the beginning some of sensitive attributes; the removal of attribute `G2` makes the accurate regression task more difficult, because there is a strict relation between such attribute and the final grade. After the attribute removals we obtain a dataset with 27 attributes and 1 response value (the original Math student dataset contains 32 attributes). Finally, the modification of the attribute `G1` described above introduces outliers: they correspond to students with expected low performance who are "helped" by the evaluators and obtain a high final grade. The outliers introduced by lowering to 2 the `G1` value will be called *heavy outliers*, those introduced by lowering the `G1` to 10 will be called *mild outliers*. The CMSD is available at the online repository (Barbato & Ceselli, 2023).

We train and evaluate the MIP-based LAD-SFSOD models using a 10-fold cross-validation approach. From the CMSD we obtain 10 pairs of training-testing datasets, where each training dataset contains about the 90% of the CMSD rows and all 30 outliers. The training-testing pairs are available at the online repository (Barbato & Ceselli, 2023).

Implementation details. We compare 5 LAD-SFSOD algorithms. The first one is (D), relying on the disjunctive-based formulation of Section 3. The other ones use two distinct preprocessing phases inherited from LLTA (we call them *LLTA-like* algorithms): a *full preprocessing* which applies the interquartile criterion to all attributes (hence it corresponds to the original preprocessing procedure described in Sudermann-Merx

and Rebennack (2021)); a *weak preprocessing* which only applies the interquartile criterion to non-binary attributes. The rationale behind the weak preprocessing is that the interquartile criterion on binary attributes labels most of entries in the CMSD training dataset as potential bad leverage points, causing the LLTA-like methods to lose their potential computational advantage.

Applying one preprocessing type before solving (*D*) yields two algorithms *D*-fp (using the full preprocessing) and *D*-wp (using the weak preprocessing). In *D*-fp and *D*-wp we only include the switch variables associated with the potential bad leverage points detected by the corresponding preprocessing phase. Similarly, algorithms FS-LLTA-fp and FS-LLTA-wp correspond to solve (FS-LLTA) after applying the full and the weak preprocessing, respectively.

All algorithms are solved in Gurobi 10.1, using the same solver parameters of Section 4. In particular, each test is run with a time limit of 1800 s. In this case we use feature budgets $d_0 \in \{3, 5, 7\}$ and outlier budgets $k_0 \in \{\lfloor 0.05k \rfloor, \lfloor 0.075k \rfloor, \lfloor 0.1k \rfloor\}$ (corresponding to about 5%, 7.5% and 10% of the instance size). This gives a total of 90 tests (9 per training dataset). The budget values are chosen so that the final predictors are very sparse and under the assumption that the CMSD does not contain more than 10% of outliers. We solve the LAD-SFSOD formulation (*D*) by providing the incumbent solution computed via the PALM algorithm as a warm-start. In each training repetition we compute the big-M values used in the LAD-SFSOD formulation (*D*) as follows: we first apply the convex programming approach described in Section 3.2; next we apply the refining procedure of Algorithm 1 with $L = 10$ and $\mu = 20$. As in Section 4, when increasing the feature budget, we exploit the solution corresponding to the last feature budget test, thus avoiding a resolution from scratch.

Computational results. The detailed computational results of the SFSOD-based algorithms of this section are available on the online repository (Barbato & Ceselli, 2023). Here we provide a qualitative analysis. Regardless of the feature budget, the tests on the CMSD instances turn out to be very hard to solve, as the algorithms never reach optimality. We point out here that the final incumbent reported at reaching the time limit could be an optimal or a near-optimal solution even in the case integer optimality has not been *proved* by the branch-and-bound process. In view of such remark, we have compared the incumbent values (primal bounds) of the tested algorithms. The results are reported in Appendix F. The main outcome is summarized in the following:

Experimental Observation 6. *In the majority of tests, algorithm (D) yields better incumbents than the algorithms based on the full-preprocessing (D-wp and FS-LLTA-wp); moreover, it always produces solutions of better quality than the algorithms relying on the weak-preprocessing phase (D-wp and FS-LLTA-wp).*

In Appendix F we also analyze the quality of the relative optimality gaps. It shows that (a) algorithm (*D*) with no preprocessing produces smaller optimality gaps than FS-LLTA-fp (b) solutions obtained by weak preprocessing can hardly be further improved by optimization.

Prediction quality and feature selection. Finally we study the quality of the solutions produced by the tested algorithms in terms of prediction quality and feature selection. We include in the comparison the results of `enetLTS` and `sparseLTS` which are trained and tested on the same 10 training-testing pairs of CMSD instances described above. The parameters used to train `enetLTS` and `sparseLTS` on the CMDS instances are given in the online Appendix D.

Given a CMSD training instance, each MIP-based algorithm produces a grid of solutions obtained from all combinations of feature and outlier budgets $d_0 \in \{3, 5, 7\}$ and $k_0 \in \{5\%, 7.5\%, 10\%\}$. We select one solution per training instance via the rAIC, as done for the synthetic instances. The selected solution is then evaluated on the corresponding testing CMSD instance in terms of RMSPE, MAE, number of nonzero features and several metrics concerning the detection of outliers. The values of each quality measure are averaged over the 10 testing CMSD instances. The same procedure is used to select and evaluate one solution among those produced by `enetLTS` and `sparseLTS` from each training CMSD instance. The averaged results are presented in Table 4.

The best RMSPE value is obtained by `enetLTS` which, however, produced less sparse hyperplanes (using more than 9 nonzero features in average). All other algorithms yield solutions with similar RMSPE, which is between 2.90 (*D*-wp and FS-LLTA-wp) and 3.00 (*D*-fp and FS-LLTA-fp). The MAE is similar for all algorithms, with the MIP-based approach yielding slightly better results, between 1.86 (*D*-fp) and 1.88 (*D*-wp and FS-LLTA-wp). The results on the feature selection can be seen in column "Selected features" of Table 4.

The rAIC always selects solutions of the MIP-based approaches with 3 nonzero features (the smallest feature budget). This is in line with the results of `sparseLTS` which selects solutions having between 2 and 3 nonzero features (we recall that in `sparseLTS` there is no direct control on the number of features). In our tests with the other information criteria described in Section 4 and Appendix C this result does not change except for the rBIC, which seldom selects hyperplanes with 5 nonzero features. Putting together these results on the MAE and on the feature selection we deduce that the real final grade can be predicted with a shift of at most 2 units by using a very limited amount of features.

A closer look to the actual nonzero features that are used for the prediction yields that the most recurring ones are: `G1` (1st semester grade), always used by all methods; `age` (students' age), always used by all solutions of the MIP-based approaches, and by the solutions of `sparseLTS` with 3 nonzero features; `failures` (number of previous class failures), used by the `sparseLTS` solutions and by the solutions of the MIP-based approaches with weak preprocessing; `absences` (number of students' absences), used by the MIP-based approaches with full preprocessing.

As seen in column "Detected (%)" the selected solutions of the MIP-based approaches detect a number of outliers which is the 9.76% of the total number of points in the training instances regardless of the preprocessing phase (the real value is around 8.5% in each training instance). The value coincides for all MIP-based approach because the outlier budget constraints is always saturated. Algorithms `sparseLTS` and `enetLTS` detect 13.02% and 11.46% outliers respectively. Another interesting observation concerns the capability of the tested algorithms in performing an accurate outlier detection. To analyze it we rely on standard true positive (TP), true negative (TN), false positive (FP) and false negative (FN) values of the outlier detection. Then the quality of the outlier detection performed by each algorithm is evaluated by using the following metrics: the percentage of outliers over the total number of dataset points (column "Detected (%)" in Table 4); the true positive ratio (TPR) of the outlier detection (column "TPR"); the TPR restricted to the detection of the hard outliers (column "Hard outliers TPR"); the precision (column "Precision") computed as $TP/(TP+FP)$; the accuracy (column "Accuracy") computed as $(TP+TN)/(TP+TN+FP+FN)$ and the F1-score (column "F1") computed as in Section 4.

For each considered metric, `enetLTS` yields the best results. Restricting ourselves to the methods also yielding very sparse solutions, we see that only (*D*) and `sparseLTS` detect all hard outliers correctly, while the algorithms based on the preprocessing phase do not. Algorithm `sparseLTS` yields a higher TPR than (*D*). This is the consequence of a higher number of outliers used in the `sparseLTS` solution, as indicated in column "Detected (%)"; as a consequence, (*D*) is 4% more precise and 1% more accurate than `sparseLTS`.

Discussion. We draw the following conclusions from the above experiments: the results obtained by algorithm (*D*) are in line with those of state-of-the-art algorithms from the literature in terms of prediction quality, feature selection and outlier detection. The other LAD-SFSOD

Table 4
Prediction, feature selection and outlier detection quality of the solutions produced by several algorithms on the corrupted Math student dataset. The values reported in each column are averaged over 10 testing replications.

	RMSPE	MAE	Selected features	Outlier detection					
				Detected (%)	TPR	Hard outliers TPR	Precision	Accuracy	F1
D	2.94	1.87	3.00	9.76	0.56	**1.00**	0.48	**0.91**	0.52
D-fp	2.99	**1.86**	3.00	9.76	0.39	0.61	0.34	0.88	0.36
D-wp	2.90	1.88	3.00	9.76	0.27	0.28	0.23	0.86	0.25
FS-LLTA-fp	3.00	1.87	3.00	9.76	0.39	0.61	0.34	0.88	0.36
FS-LLTA-wp	2.90	1.88	3.00	9.76	0.27	0.28	0.23	0.86	0.25
`sparseLTS`	2.93	1.90	2.40	13.02	0.69	**1.00**	0.44	0.90	0.54
`enetLTS`	**2.83**	1.87	9.10	11.46	**0.72**	**1.00**	**0.49**	**0.91**	**0.58**

MIPs relying on the preprocessing phase introduced in Sudermann-Merx and Rebennack (2021) are not accurate in the outlier detection, hence this aspect should be kept in mind when applying them to problems where the outlier detection enters the decision process. While solving the LAD-SFSOD MIPs is computationally more demanding than using `sparseLTS` and `enetLTS` it allows a full control on the number/type of features to be selected. Moreover, the good prediction results of the solutions of (*D*) hold even when the corresponding MIP resolution does not finish with a proven integer optimality, suggesting that the produced incumbents are of good quality.

6. Conclusions

Our paper revolves around the idea of using a least absolute deviation (LAD) criterion for the simultaneous feature selection and outlier detection (SFSOD), instead of the least-squares (LS) criterion which is more commonly used in the literature. We rely on a mathematical programming approach: we model and solve the LAD-SFSOD by means of two MILP formulations, one adapted from the literature dealing with the LS-SFSOD and the other one based on a disjunctive argument. Our theoretical analysis proves the disjunctive-based formulation to offer, in terms of polyhedral structure, advantages with respect to MILP adapted from the literature. These yield computational advantages. For instance, the quality of the continuous relaxation of the disjunctive-based formulation is never worse than those of models adapted from the literature, and it becomes strictly better when a proper choice of model parameters can be made.

For what concerns the specific application to the SFSOD approach, we are able to obtain several insights. First, the performance of LAD methods are strongly affected by the type of outliers in the dataset. When vertical outliers are involved both the MIP-based approaches and state-of-the-art least-trimmed square (LTS) heuristics provide good balancing of prediction error and sparsity, LTS heuristics being faster. However, when bad leverage points pollute the dataset, our experiments show that the mathematical programming approaches using LAD-SFSOD models perform best (and especially the disjunctive-based formulation). In particular, in this case, LTS methods are not able to combine sparsity and robustness satisfactorily.

We have also tested MIP-based LAD-SFSOD approaches and state-of-the-art LTS approaches on real-world datasets describing students' performance. Our results indicate that the disjunctive-based formulation produces solutions similar to the LTS methods, while MIP-based counterparts relying on statistical preprocessing to identify potential outliers may lead to worse outlier detection. In these experiments, the quality of LTS and MIP-based solutions are similar. However, the MIP-based approach has a fundamental advantage: it allows a direct control on the feature selection process, granting more potential in applications where this flexibility is central. For instance, the budget on the number of features may be used either as a hyperparameter, whose tuning can improve regression quality, *or as a user parameter* whose fixing is under the direct control of the decision maker.

Finally, we found it interesting to note that our results were obtained by changing the classical approach in computational statistics, in favor of one which is mathematical programming oriented. This highlights the potential impact that Operations Research techniques can still bring in computational statistics. A fundamental preliminary step toward this direction is to improve the scalability of the proposed mathematical programming approaches making them able to deal with datasets of orders of magnitudes larger than those considered in our paper and in the recent literature. In this spirit, a first direction for future investigation is the design of dedicated algorithms for optimizing the LAD-SFSOD MILPs of this paper. To this end decomposition approaches are promising to improve computational performances, as shown in related applications of MILPs to computational statistics (see *e.g.*, Warwicker & Rebennack, 2022 for a Benders' decomposition algorithm for the resolution of robust piecewise linear regression based on a previous work of Rebennack & Krasko, 2020). Another direction is more application-oriented: investigate if the application of kernels to the dual of stronger formulations (as the disjunctive-based one) would improve the performance of classic methods.

Funding

The work was partially funded by Università degli Studi di Milano, Piano Sostegno alla Ricerca (PSR) and partially supported by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data and code availability

The instances and the detailed computational results of this study are publicly available at the online repository of Barbato and Ceselli (2023). The code implemented for this study is available from the corresponding author upon request.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ejor.2024.03.035.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*(6), 716–723.

Alfons, A. (2021). `robustHD` v. 0.7.2 robust methods for high-dimensional data. https://cran.r-project.org/web/packages/robustHD/robustHD.pdf. (Accessed 31 March 2022).

Alfons, A., Croux, C., & Gelper, S. (2013). Sparse least trimmed squares regression for analyzing high-dimensional large data sets. *The Annals of Applied Statistics*, 226–248.

Arslan, O. (2012). Weighted LAD-LASSO method for robust parameter estimation and variable selection in regression. *Computational Statistics & Data Analysis*, *56*(6), 1952–1965.

Balas, E. (1998). Disjunctive programming: Properties of the convex hull of feasible points. *Discrete Applied Mathematics*, *89*(1–3), 3–44.

Barbato, M., Bertoncini, A., & Ceselli, A. (2023). Enhancing sparse regression models with cutting planes. In *Talk at optimization and decision science conference*.

Barbato, M., & Ceselli, A. (2023). Instances and computational results of: Mathematical programming for simultaneous feature selection and outlier detection under l1 norm. http://dx.doi.org/10.13130/RD_UNIMI/LZA4F8.

Belotti, P., Bonami, P., Fischetti, M., Lodi, A., Monaci, M., Nogales-Gómez, A., et al. (2016). On handling indicator constraints in mixed integer programming. *Computational Optimization and Applications*, *65*(3), 545–566.

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., Kitane, D. L., Azami, N., & Doucet, F. (2020). Novel mixed integer optimization sparse regression approach in chemometrics. *Analytica Chimica Acta*, *1137*, 115–124.

Bertsimas, D., & Weismantel, R. (2005). *Optimization over integers*: *vol. 13*, Belmont: Dynamic Ideas.

Bonami, P., Lodi, A., Tramontani, A., & Wiese, S. (2015). On mathematical programming with indicator constraints. *Mathematical Programming*, *151*(1), 191–223.

Bottmer, L., Croux, C., & Wilms, I. (2022). Sparse regression for large data sets with outliers. *European Journal of Operational Research*, *297*(2), 782–794.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on management of data* (pp. 93–104).

Candes, E., & Tao, T. (2007). The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, *35*(6), 2313–2351.

Chen, Y., Caramanis, C., & Mannor, S. (2013). Robust sparse regression under adversarial corruption. In *International conference on machine learning* (pp. 774–782). PMLR.

Cortez, P. (2014). Student performance. In *UCI machine learning repository*. http://dx.doi.org/10.24432/C5TG7T.

Dodge, Y. (1997). LAD regression for detecting outliers in response and explanatory variables. *Journal of Multivariate Analysis*, *61*(1), 144–158.

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, *96*(456), 1348–1360.

Filzmoser, P., & Nordhausen, K. (2021). Robust linear regression for high-dimensional data: An overview. *Wiley Interdisciplinary Reviews: Computational Statistics*, *13*(4), Article e1524.

Fischetti, M., & Monaci, M. (2020). A branch-and-cut algorithm for mixed-integer bilinear programming. *European Journal of Operational Research*, *282*(2), 506–514.

Greenshtein, E. (2006). Best subset selection, persistence in high-dimensional statistical learning and optimization under $l1$ constraint. *The Annals of Statistics*, *34*(5), 2367–2386.

Gupte, A., Ahmed, S., Cheon, M. S., & Dey, S. (2013). Solving mixed integer bilinear problems using MILP formulations. *SIAM Journal on Optimization*, *23*(2), 721–744.

Gurobi Optimization (2023). Gurobi 10.1 optimizer reference manual. https://www.gurobi.com/documentation/10.0/refman/index.html. (Accessed 27 March 2024).

Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (2nd ed.). New York: Springer.

Hijazi, H., Bonami, P., Cornuéjols, G., & Ouorou, A. (2012). Mixed-integer nonlinear programs featuring on/off constraints. *Computational Optimization and Applications*, *52*(2), 537–558.

Hodge, V., & Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review*, *22*, 85–126.

IBM ILOG (2020). CPLEX optimization studio version 20 release 1.0. https://www.ibm.com/docs/en/icos/20.1.0. (Accessed 26 January 2024).

Insolia, L., Kenney, A., Chiaromonte, F., & Felici, G. (2021). Simultaneous feature selection and outlier detection with optimality guarantees. *Biometrics*.

James, G., Witten, D., Hastie, T., Tibshirani, R., et al. (2021). *An introduction to statistical learning, vol. 112* (2nd ed.). Springer.

Jammal, M. (2020). *Variable selection and outlier detection via mixed integer programming* (Ph.D. thesis), Normandie Université; Université Libanaise.

Jammal, M., Canu, S., & Abdallah, M. (2020). ℓ_1 Regularized robust and sparse linear modeling using discrete optimization. In *International conference on machine learning, optimization, and data science* (pp. 645–661). Springer.

Jammal, M., Canu, S., & Abdallah, M. (2021). Joint outlier detection and variable selection using discrete optimization. *SORT-Statistics and Operations Research Transactions*, 47–66.

Kurnaz, F. S., Hoffmann, I., & Filzmoser, P. (2018a). Robust and sparse estimation methods for high-dimensional linear and logistic regression. *Chemometrics and Intelligent Laboratory Systems*, *172*, 211–222.

Kurnaz, F. S., Hoffmann, I., & Filzmoser, P. (2018b). `enetLTS` v. 0.1.1 robust and sparse methods for high dimensional linear and logistic regression. https://cran.r-project.org/web/packages/enetLTS/enetLTS.pdf. (Accessed 31 March 2022).

Laurikkala, J., Juhola, M., Kentala, E., Lavrac, N., Miksch, S., & Kavsek, B. (2000). Informal identification of outliers in medical data. In *Fifth international workshop on intelligent data analysis in medicine and pharmacology, vol. 1* (pp. 20–24).

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part I – Convex underestimating problems. *Mathematical Programming*, *10*(1), 147–175.

Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics & Data Analysis*, *52*(1), 374–393.

Natarajan, B. K. (1995). Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, *24*(2), 227–234.

Pavlidou, M., & Zioutas, G. (2014). Kernel density outlier detector. In *Topics in nonparametric statistics: proceedings of the first conference of the international society for nonparametric statistics* (pp. 241–250). Springer.

Rebennack, S., & Krasko, V. (2020). Piecewise linear function fitting via mixed-integer linear programming. *INFORMS Journal on Computing*, *32*(2), 507–530.

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

Rousseeuw, P. J., & Leroy, A. M. (2005). *Robust regression and outlier detection*. John wiley & sons.

Rousseeuw, P. J., & Van Driessen, K. (2006). Computing LTS regression for large data sets. *Data Mining and Knowledge Discovery*, *12*(1), 29–45.

RStudio Team (2020). Rstudio: integrated development environment for R.

Sudermann-Merx, N., & Rebennack, S. (2021). Leveraged least trimmed absolute deviations. *OR Spectrum*, *43*(3), 809–834.

Thompson, R. (2022). Robust subset selection. *Computational Statistics & Data Analysis*, *169*, Article 107415.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *58*(1), 267–288.

Vielma, J. P. (2015). Mixed integer linear programming formulation techniques. *Siam Review*, *57*(1), 3–57.

Wang, H., Bah, M. J., & Hammad, M. (2019). Progress in outlier detection techniques: A survey. *IEEE Access*, *7*, 107964–108000.

Wang, H., Li, G., & Jiang, G. (2007). Robust regression shrinkage and consistent variable selection through the LAD-Lasso. *Journal of Business & Economic Statistics*, *25*(3), 347–355.

Wang, Y., & Zhu, L. (2017). Variable selection and parameter estimation via wlad–scad with a diverging number of parameters. *Journal of the Korean Statistical Society*, *46*(3), 390–403.

Warwicker, J. A., & Rebennack, S. (2022). Generating optimal robust continuous piecewise linear regression with outliers through combinatorial benders decomposition. *IISE Transactions*, 1–13.

Xu, J., & Ying, Z. (2010). Simultaneous estimation and variable selection in median regression using lasso-type penalty. *Annals of the Institute of Statistical Mathematics*, *62*(3), 487–514.

Yang, X., Latecki, L. J., & Pokrajac, D. (2009). Outlier detection with globally optimal exemplar-based gmm. In *Proceedings of the 2009 SIAM international conference on data mining* (pp. 145–154). SIAM.

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, *38*(2), 894–942.

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*, 339–353.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, *101*(476), 1418–1429.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *67*(2), 301–320.