Journal of Machine Learning Research 24 (2023) 1-28 Submitted 7/21; Revised 9/23; Published 12/23

A Novel Integer Linear Programming Approach for Global ℓ_0 Minimization

Diego Delle Donne DELLEDONNE@ESSEC.EDU
Information Systems, Decision Sciences and Statistics department
ESSEC Business School
95021 Cergy-Pontoise Cedex, France

Matthieu Kowalski MATTHIEU.KOWALSKI@UNIVERSITE-PARIS-SACLAY.FR
Inria Saclay-Île-de-France
Université Paris-Saclay, CNRS, Laboratoire Interdisciplinaire des Sciences du Numériques
91192 Gif-sur-Yvette Cedex, France

Leo Liberti LIBERTI@LIX.POLYTECHNIQUE.FR
LIX CNRS, Ecole Polytechnique
Institut Polytechnique de Paris
91128 Palaiseau, France

Editor: Silvia Villa

Abstract

Given a vector $y \in \mathbb{R}^n$ and a matrix $H \in \mathbb{R}^{n\times m}$, the sparse approximation problem $\mathcal{P}_{0/p}$ asks for a point x such that $\|y - Hx\|_p \leq \alpha$, for a given scalar α, minimizing the size of the support $\|x\|_0 := \#\{j \mid x_j \neq 0\}$. Existing convex mixed-integer programming formulations for $\mathcal{P}_{0/p}$ are of a kind referred to as "big-M", meaning that they involve the use of a bound M on the values of x. When a proper value for M is not known beforehand, these formulations are not exact, in the sense that they may fail to recover the wanted global minimizer. In this work, we study the polytopes arising from these formulations and derive valid inequalities for them. We first use these inequalities to design a branch-and-cut algorithm for these models. Additionally, we prove that these inequalities are sufficient to describe the set of feasible supports for $\mathcal{P}_{0/p}$. Based on this result, we introduce a new (and the first to our knowledge) M-independent integer linear programming formulation for $\mathcal{P}_{0/p}$, which guarantees the recovery of the global minimizer. We propose a practical approach to tackle this formulation, which has exponentially many constraints. The proposed methods are then compared in computational experimentation to test their potential practical contribution.

Keywords: Sparse Approximation, ℓ_0 minimization, Integer Programming.

1 Introduction and state of the art

The Sparse Representation of a vector $y \in \mathbb{R}^n$ in a matrix $H \in \mathbb{R}^{n\times m}$ aims at finding a solution $x \in \mathbb{R}^m$ to the system $Hx = y$, with minimum number of non-zero components, i.e., minimizing the so-called ℓ_0 pseudonorm of x defined by $\|x\|_0 := \#\{j \mid x_j \neq 0\}$. The Sparse Approximation problem also considers noise and model errors, relaxing the equality constraints and aiming instead at minimizing the misfit data measure $\|y - Hx\|$ for a given norm $\|\cdot\|$. More specifically, we address the sparse linear inverse problem, where

the number of measurements may be fewer than the size of the signal to be recovered. This scenario arises in various fields, such as imaging (Ribes and Schmitt, 2008), Astrophysics (Bourguignon et al., 2011) and the deconvolution problem (Soussen et al., 2011) and have benefited from the sparse representation. Recovering the underlying sparse signal accurately and efficiently is a fundamental challenge in these applications. In this context, several optimization problems may be stated as such:

1. minimize $\|x\|_0$ subject to a given threshold $\|y - Hx\| \leq \alpha$,
2. minimize the data misfit $\|y - Hx\|$ subject to a given bound $\|x\|_0 \leq k$, this problem being also known as the Best Subset Selection problem (Hocking and Leslie, 1967).
3. minimize a weighted sum $\lambda_1\|y - Hx\| + \lambda_2\|x\|_0$ for some $\lambda_1, \lambda_2 \in \mathbb{R}$.

One should note that while the three formulations are not equivalent, they correspond to a classical bi-objective minimization problem (Marler and Arora, 2004; Soussen et al., 2015).

Without any assumption on the structure of the matrix H, sparse approximation problems described above are NP-hard (Natarajan, 1995). This fact has led most of the literature on the subject to focus on approximate and heuristic solution approaches. In particular, many articles are devoted to the relaxation approach that replaces the ℓ_0 sparsity measure with the ℓ_1-norm $\|x\|_1 = \sum_j |x_j|$, yielding convex optimization problems such as the Lasso (Tibshirani, 2011) that admit tractable algorithms, see for example Combettes and Pesquet (2008); Bruckstein et al. (2009); Tropp and Wright (2010). One of the most basic heuristic approaches in the literature, usually referenced as *pursuit methods*, resorts to greedy techniques to build a sparse solution by iteratively modifying one or several coefficients of a current estimate for the solution vector x. The well-known *Matching Pursuit* (MP) method, introduced by Mallat and Zhang (1993), consists in identifying at each step the best column to "add" to the solution, and a proper coefficient for this coordinate, until a given stopping criterion is reached. Some interesting variants of MP include the popular orthogonal matching pursuit (Davis et al., 1997; Pati et al., 1993) and the relaxed greedy algorithm (DeVore and Temlyakov, 1996). More sophisticated pursuit methods were further developed, including, for example, the stagewise orthogonal matching pursuit (StOMP) (Donoho et al., 2012), the regularized orthogonal matching pursuit algorithm (Needell and Vershynin, 2009, 2010), compressive sampling matching pursuit (CoSaMP) (Needell and Tropp, 2009) and subspace pursuit (Dai and Milenkovic, 2009). We refer the reader to Tropp and Wright (2010) for an interesting survey on these methods. The so-called Iterative Hard Thresholding Algorithm can easily obtain a local minimizer of the ℓ_0 problem popularized by Blumensath and Davies (2009), and variants (Zhou et al., 2021). Some authors studied nonconvex penalties such as the capped-L1 (Zhang, 2010) or the Cel0 penalty (Soubies et al., 2015). The latter has the wanted properties of sharing local and global minimizers with the original problem (Soubies et al., 2019). Other approaches have been proposed, such as the Difference of Convex Programming Algorithm (Le Thi et al., 2015). One can refer to the reviews on sparse approximation and algorithms done by Bach et al. (2012), Marques et al. (2018), and Bertsimas et al. (2020)

In this work, we study mixed integer programming (MIP) formulations for the problem stated in Item 1 above, which can be used to find a global minimizer. MIP programming for

the sparse representation problem has been popularized by the recent works of Bourguignon et al. (2016) and Bertsimas et al. (2016). Following the notation from Bourguignon et al. (2016), we define these problems as

$$\mathcal{P}_{0/p}(\alpha) : \min_x \|x\|_0 \text{ s.t. } \|y - Hx\|_p \leq \alpha,$$

where the parameter α represents the threshold for the misfit error. In the remaining of this manuscript, we may write just $\mathcal{P}_{0/p}$ whenever α is clear from the context and/or irrelevant. Also, for any natural number t, we may use $[t]$ as a shortcut for the set $\{1, \ldots, t\}$. The chosen formulation is quite convenient, especially in a signal processing context, the parameter α being related to the level of noise and can be set accordingly using Morozov's discrepancy principle (Bonesky, 2008). The first MIP formulations for sparse optimization problems are proposed by Jokar and Pfetsch (2008) for $\mathcal{P}_{0/\infty}$ and some years later in Tosic and Drewes (2014) for $\mathcal{P}_{0/2}$. However, the computational complexity of solving these formulations leads the authors of these articles to solve modified and/or relaxed versions, yielding just approximated solutions for the original problems.

Some MIP formulations for $\mathcal{P}_{0/p}$, with $p \in \{1, 2, \infty\}$, are proposed and empirically compared in Bourguignon et al. (2016). These models use decision variables $x_j \in \mathbb{R}$ for each $j \in [m]$ to determine the solution, and binary *support variables* b_j to decide whether x_j is non-zero. The object of our study is this family of MIP formulations that we call $\mathsf{MIP}_{0/p}$, which models $\mathcal{P}_{0/p}$. By way of example we explicitly show $\mathsf{MIP}_{0/p}$ for $p \in \{1, 2, \infty\}$ here below:

$$[\mathsf{MIP}_{0/p}(M)] \qquad \min \sum_{j \in [m]} b_j \tag{1}$$

$$\forall j \in [m], \qquad -Mb_j \leq x_j \leq Mb_j \tag{2}$$

$$\forall i \in [n], \qquad \begin{cases} -w_i \leq y_i - \sum_{j \in [m]} h_{ij} x_j \leq w_i & \text{if } p \in \{1, 2\} \\ -w \leq y_i - \sum_{j \in [m]} h_{ij} x_j \leq w & \text{if } p = \infty \end{cases} \tag{3}$$

$$\begin{cases} \sum_{i \in [n]} w_i \leq \alpha & \text{if } p = 1 \\ \sum_{i \in [n]} w_i^2 \leq \alpha^2 & \text{if } p = 2 \\ w \leq \alpha & \text{if } p = \infty \end{cases} \tag{4}$$

$$\begin{cases} w \in \mathbb{R}_+^n & \text{if } p \in \{1, 2\} \\ w \in \mathbb{R}_+ & \text{if } p = \infty \end{cases} \tag{5}$$

$$x \in \mathbb{R}^m, b \in \{0, 1\}^m \tag{6}$$

The objective function (1) calculates $\|x\|_0$ in any optimal solution as constraints (2) force b_j to take value 1 whenever x_j is a non-zero component. Here, M shall be a sufficiently large bound for the value of $|x_j|$ when $b_j = 1$ (we further discuss on this issue in Section 2). In the sequel, we may write just $\mathsf{MIP}_{0/p}$ whenever M is clear from the context and/or irrelevant. For $p \in \{1, 2\}$ (resp. $p = \infty$), Constraints (3) ensure that w_i (resp. w) is an upper bound on the data misfit given by row i and Constraint (4) asserts that the ℓ_p norm of these values (i.e., an upper bound on $\|y - Hx\|_p$) does not exceed α. We shall note that Constraints (4) make $\mathsf{MIP}_{0/2}$ a *convex Mixed-Integer Quadratically Constrained Program*

(cMIQCP), in contrast to $\mathsf{MIP}_{0/1}$ and $\mathsf{MIP}_{0/\infty}$ which correspond to *linear* formulations (MILP).

The object of study of this paper is the formulation presented above as $\mathsf{MIP}_{0/p}$, for the practical resolution of $\mathcal{P}_{0/p}$. On the one side, we study the polyhedral structure of these formulations to develop algorithms to improve the efficiency of the solution of these models. On the other side, we address a crucial issue concerning the necessity of these models of using a "big M" in order to relate x and b variables. This kind of formulations, widely known as *big-M formulations*, usually suffers from relaxation slackness (Wolsey, 1998, §1.6) and is consequently detrimental to solver performance. Moreover, the smallest valid values for M are often difficult to compute in practice. To the best of our knowledge, no MIP formulation currently exists in the literature for the $\mathcal{P}_{0/p}$ problem, which eschews the "big M" issue without introducing some nonlinearity in the formulation (Mhenni, 2020). In this paper, we propose such a *linear* "big M"-free formulation for $\mathcal{P}_{0/p}$, by resorting, however, to an exponentially sized constraint set.

In recent years, several approaches have been developed by Xie and Deng (2020); Bertsimas et al. (2020); Hazimeh et al. (2022) to address the limitations of the Big-M reformulation in solving underdetermined linear inverse problems. These methods either ensure a non-zero minimum eigenvalue of the matrix H or introduce a ridge term to avoid the Big-M. While these advancements have shown promise in certain problem settings, they cannot be directly applied to the specific problem we are considering, namely $\mathcal{P}_{0/p}$ for underdetermined linear inverse problems. The unique nature of this problem necessitates a tailored approach and alternative strategies are required to address its specific characteristics.

More precisely, in this work, we study the polytopes arising from $\mathsf{MIP}_{0/p}$ formulations and derive valid inequalities for them, i.e., inequalities satisfied by every point of these polytopes which can be used as cutting planes (see Section 3 for more details). As a first solution approach, we use these inequalities to design a branch-and-cut algorithm for $\mathsf{MIP}_{0/p}$. Additionally, we prove that these inequalities are sufficient to describe the projection of the related polytopes onto the space of the binary variables b_j. Based on this result, we introduce a novel integer linear programming (ILP) formulation for $\mathcal{P}_{0/p}$, which consists in solving a pure combinatorial set covering formulation. To the best of our knowledge, this is the first *exact* (i.e., M-independent) integer *linear* programming formulation for $\mathcal{P}_{0/p}$ (we give further details on this claim in Section 2). Since the proposed set covering formulation may have exponentially many constraints, we propose a practical approach to tackle this integer programming (IP) formulation.

The rest of this work is as follows. In Section 2 we discuss the $\mathsf{MIP}_{0/p}$ formulations and show that current approaches using these "big M" formulations cannot be considered exact. In Section 3 we carry out a polyhedral study of the polytopes arising from $\mathsf{MIP}_{0/p}$. In particular, we derive valid inequalities and propose a heuristic *separation procedure* for these inequalities. Additionally, we propose a *rounding heuristic* which takes a fractional solution of $\mathsf{MIP}_{0/p}$ and attempts to find a feasible solution from it (see Section 3.2 for more details on rounding heuristics). Finally, we develop a branch-and-cut algorithm with these elements to tackle the solution of $\mathsf{MIP}_{0/p}$. Section 4 is devoted to the main contribution of this paper. First, we prove that the proposed valid inequalities introduced in Section 3 are sufficient to describe the set of feasible supports of $\mathcal{P}_{0/p}$. Based on this result, we introduce an M-independent integer *linear* programming (ILP) formulation for $\mathcal{P}_{0/p}$ and a

practical algorithm to tackle its solution. Section 5 presents the results of computational experimentation comparing the proposed approaches. Section 6 concludes the paper.

In the rest of the paper, we will use the following notations. Given a set of columns $J \subseteq [m]$, we define H^J (resp. x^J) to be the submatrix of H (resp. subvector of x) involving only those columns indexed by J. Additionally, we say that a vector $x \in \mathbb{R}^m$ has support $J \subseteq [m]$, if $x_j = 0$, for all $j \notin J$. Moreover, without loss of generality, we assume that H has normalized columns. We say that a point satisfying every constraint of a MIP formulation is a *feasible solution* for it. We may sometimes omit the term "feasible" and call it just a *solution*. If no other feasible solution exists for the MIP with a better objective function value, then the feasible solution is also *optimal*. If a point satisfies every constraint of a MIP except for one or more integrality constraints, we say that it is a *fractional solution* (here "fractional" refers to non-integer values on the variables). When integrality constraints on a MIP are relaxed, the resulting formulation is known as its *continuous relaxation*.

2 Dealing with $\mathsf{MIP}_{0/p}$

Formulation $\mathsf{MIP}_{0/p}$ requires a bound M for the value of $|x_j|$, in order to impose the *binding constraints* (2) between variables x_j and b_j. Models of these kind are widely known as *big-M formulations*. It is well-known that they usually suffer from relaxation slackness (Wolsey, 1998, §1.6), which leads to degraded solver performance, in particular when the value of M is high (hence, setting $b_j = \varepsilon \ll 1$ when solving the continuous relaxation would allow x_j to be almost unrestricted, while b_j is almost zero). Using a different value M_j for each coordinate may help reduce this relaxation slackness – however, the problem of finding proper values for M_j remains. Moreover, the smallest valid values for M are often difficult to compute in practice. Obtaining this value may be, in general, NP-hard by itself (Kleinert et al., 2020).

The non-zero coordinates of the global solution of the $\mathsf{MIP}_{0/2}$ problems can be written as $x^J = \left(H^{J^T} H^J\right)^{-1} H^{J^T} y$, where J is an optimal subset. The smallest M can then be upper bounded using the inverse of the smallest eigenvalue of $H^{J^T} H^J$ for any subset $J \subset [m]$. Cauchy's interlace Theorem (Hwang, 2004) implies that the smallest eigenvalue of $H^T H$ lower bounds this eigenvalue. However, such a result is useless in the underdetermined setting ($m > n$) and not tight enough as soon as the matrix $H^T H$ is too poorly conditioned in the over-determined setting ($m \leq n$).

In Bourguignon et al. (2016), an iterative procedure is proposed to tackle this issue within the solution of $\mathsf{MIP}_{0/p}$. The algorithm starts by setting M to an initial value of $1.1x^1_{\max}$, where $x^1_{\max} = \|H^T y\|_\infty$ corresponds to the maximum amplitude of 1-sparse solutions estimated by least-squares (the columns of H being normalized). Then $\mathsf{MIP}_{0/p}(M)$ is solved. If the obtained solution *is tight* on the given M (i.e., $|x_j| = M$ for some column $j \in [m]$), then M is increased by a certain proportion (authors use 10%) and the process is repeated. Otherwise, the solution is considered optimal, and the process is ended.

Unfortunately, no proof of optimality can be given for the obtained solution. Moreover, there exist examples in which the proposed algorithm fails to find an optimal solution. We illustrate this possibility with the following simple examples:

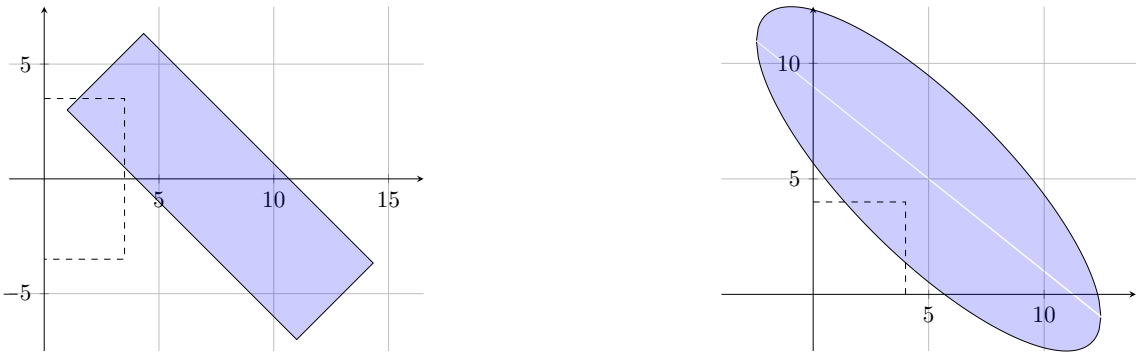

Figure 1: Some simple examples illustrate the incremental algorithm's potentially suboptimal behavior to solve $\mathsf{MIP}_{0/p}$.

- Example 1: $H = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$, $y = \begin{pmatrix} 7 \\ 15 \end{pmatrix}$, $\alpha = 10$, $p = 1$, initial $M = 3.5$
- Example 2: $H = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$, $y = \begin{pmatrix} 15 \\ 15 \end{pmatrix}$, $\alpha = 10$, $p = 2$, initial $M = 4$

The set of feasible solutions for Examples 1 and 2 are depicted in Figure 1 (left and right, respectively), where the bound given by the initial M is represented with a dotted line. Optimal solutions lie on the intersection of the feasible region with the coordinate axis (i.e., where $\|x\|_0 = 1$). However, when restricted to the initial value for M, every feasible solution has $\|x\|_0 = 2$; hence all these solutions are equally optimal for such M. Therefore, it is possible to obtain a solution that is slack w.r.t. M (that is, for which the procedure would end) but suboptimal for the original problem. Furthermore, we note that these examples may be extended to be valid for any initial value of M by replacing vector y by $y + \lambda(1, 2)^T$, for a proper $\lambda > 0$ (which is equivalent to "moving" the feasible region λ units in the direction of the positive horizontal axis).

We shall note that when the feasible set for x is bounded, then a proper value for M may be obtained beforehand by solving a sequence of linear programs (two for each coordinate x_j). Nevertheless, it is not hard to find examples, in which the set of feasible solutions is unbounded, e.g., as soon as the system is underdetermined. One of these examples is depicted in Figure 2, where the set of feasible solutions corresponds to the instance in which $H = \begin{pmatrix} 1 & 1 \end{pmatrix}$, $y = 10$, $\alpha = 1$ and $p \in \{1, 2, \infty\}$ (and $M = 8$). As in the previous cases, all optimal solutions lie on the axis, however the process may end with suboptimal solutions (for the same reasons as before).

The presented examples show that the procedure stated in Bourguignon et al. (2016) could not be considered to be an *exact* method to solve $\mathcal{P}_{0/p}$ unless proof is given that the initial value for M includes an optimal solution (in which case the iterative procedure will not be necessary). Additionally, other simple examples are proposed in Mhenni (2020, §1.4.2), in the context of the problem $\mathcal{P}_{2/0}$, which represents the minimization of the data misfit $\|y - Hx\|_2$ subject to a given bound on $\|x\|_0$. We shall remark that these examples reflect a deficiency of one specific method for calibrating M, that is, the procedure

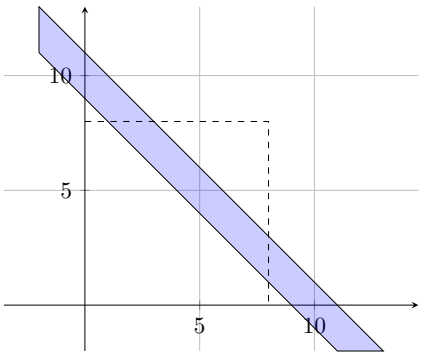

Figure 2: A simple example illustrates the incremental algorithm's potentially suboptimal behavior to solve $\mathsf{MIP}_{0/p}$ when the feasible set of solutions is unbounded.

from Bourguignon et al. (2016), rather than a proper deficiency of big-M formulations *per se*.

We conclude this section with technical remarks on the formulation $\mathsf{MIP}_{0/p}$. For implementation purposes when $p = \infty$, variable w and Constraint (4) can be omitted by just replacing w with α in Constraints (3). In turn, when $p = 2$, variables w_i can also be omitted by replacing constraints (3) and (4) with the following non-linear quadratic (convex) set of constraints:

$$x^T H^T H x - 2y^T H x \leq \alpha^2 - y^T y. \tag{7}$$

After some preliminary experimentation, we could verify that the difference in the computational hardness on solving these variants of $\mathsf{MIP}_{0/p}$ instead of the one presented in Section 1 seems to be negligible. Nevertheless, we shall use these variants for our computational comparison in Section 5, as these are the formulations used in the existing literature. We also remark that the "big M" related issues are still present in these variants of $\mathsf{MIP}_{0/p}$.

3 A branch-and-cut algorithm for $\mathsf{MIP}_{0/p}$

Given a set of points $X \subseteq \mathbb{R}^d$ (e.g., the set of feasible solutions of $\mathsf{MIP}_{0/p}$), a *valid inequality* for X is an inequality satisfied by every point of X. Within the context of a branch-and-bound algorithm to solve a MIP formulation, valid inequalities may be used as *cutting planes* to separate non-feasible solutions (e.g., a fractional solution coming from the continuous relaxation of the MIP) from the set of feasible solutions; this approach is widely called *branch-and-cut*. Additionally, generating feasible solutions during the branch-and-bound process may severely reduce the size of the branching tree by pruning whole branches before even exploring them, when the lower bound for the branch is worse than the current *incumbent* (i.e., the best feasible solution known).

In this section, we introduce a family of valid inequalities for $\mathsf{MIP}_{0/p}$, and we show how these inequalities can be used as cutting planes in a branch-and-cut scheme (by proposing a separation procedure) to improve the performance of the solution of these models. Additionally, we propose a rounding heuristic that helps the branch-and-cut algorithm in finding feasible solutions during the solution process. In the rest of the paper, we may indistinctly

use the concepts of *column* and *column index*, whenever the context is clear, to refer to a column position in the matrix H or a position in the solution vector x.

3.1 Forbidden support inequalities as cutting planes

We first define a *forbidden support*.

Definition 1 (Forbidden support) *A set of columns $J \subseteq [m]$ is a* forbidden support *for $\mathcal{P}_{0/p}(\alpha)$ if there exist no solutions for $\mathcal{P}_{0/p}(\alpha)$ with J as support. Equivalently, if $\min_{x \in \mathbb{R}^m} \{\|y - H^J x^J\|_p\} > \alpha$.*

We note that for $p \in \{1, 2, \infty\}$, we can efficiently determine whether a fixed set of columns $J \subseteq [m]$ is a forbidden support or not, and in the latter case, find a feasible solution $\hat{x}$ having support J. To this end, we need to minimize $\|y - H^J x^J\|_p$ and compare this value with α. When $p \in \{1, \infty\}$, this minimum can be found by solving a linear program (i.e., minimizing the left hand side of (4) subject to (3), for the given columns J) and when $p = 2$, this is simply realized by solving a least squares problem (i.e., minimizing $\|y - H^J x^J\|_2$). We shall remark that in any case, it is not necessary to use a bound M on the values of x, as the set J is fixed.

Remark 2 *For a fixed set of columns $J \subseteq [m]$, we can efficiently determine whether J is a forbidden support for $\mathcal{P}_{0/p}$ or not, in the latter case finding a feasible solution $\hat{x}$ for $\mathcal{P}_{0/p}$ with support J. This decision does not depend on any bound M for the values of x.*

Proposition 3 *If $J \subseteq [m]$ is a forbidden support for $\mathcal{P}_{0/p}$, then the* forbidden support inequality

$$\sum_{j \in [m] \setminus J} b_j \geq 1 \tag{8}$$

is valid for $\mathsf{MIP}_{0/p}$.

Proof If (8) is not satisfied by a feasible solution $(\hat{x}, \hat{w}, \hat{b})$, then $\hat{b}_j = 0$ for each $j \in [m] \setminus J$. Therefore, $\|y - H\hat{x}\|_p = \|y - H^J \hat{x}^J\|_p > \alpha$, as J is a forbidden support, thus contradicting the fact that $(\hat{x}, \hat{w}, \hat{b})$ is a feasible solution. ■

Given a fractional solution $(\hat{x}, \hat{w}, \hat{b})$ of $\mathsf{MIP}_{0/p}$ (e.g., an optimal solution of its linear relaxation with fractional values on $\hat{b}$), the so-called *separation problem* aims at finding a valid inequality for $\mathsf{MIP}_{0/p}$ not satisfied by $(\hat{x}, \hat{w}, \hat{b})$. Depending on the families of valid inequalities to be used as cutting planes, the separation problem is likely to be NP-hard.

We present here a heuristic separation routine for the forbidden support inequalities (8). Our procedure, shown in Algorithm 1, starts by looking for a set of columns $\bar{J} \subseteq [m]$ such that $\sum_{j \in \bar{J}} \hat{b}_j < 1$. We include as many columns as possible in $\bar{J}$ (Lines 2-8). This results in a set $J := [m] \setminus \bar{J}$, as small as possible, aiming to maximize the possibility for J to be a forbidden support, hence yielding a violated forbidden support inequality which may be added as a cutting plane (Line 12). We recall that checking whether a set J is a forbidden

Algorithm 1 Separation routine for forbidden support inequalities

Require: a fractional solution $(\hat{x}, \hat{w}, \hat{b})$.
1: $\bar{J} \leftarrow \{\}$
2: **for all** $\hat{b}_j \in \hat{b}$ in non-decreasing order **do**
3: $\quad \bar{J} \leftarrow \bar{J} \cup \{j\}$
4: $\quad$ **if** $\sum_{j \in \bar{J}} \hat{b}_j \geq 1$ **then**
5: $\quad\quad \bar{J} \leftarrow \bar{J} \setminus \{j\}$
6: $\quad\quad$ **break loop**
7: $\quad$ **end if**
8: **end for**
9: $J \leftarrow [m] \setminus \bar{J}$
10: **if** J is a forbidden support **then**
11: $\quad$ Extend J to a wider forbidden support (using Algorithm 2 on $\bar{J}$).
12: $\quad$ Use forbidden support inequality (8) associated with J to *cut off* $(\hat{x}, \hat{w}, \hat{b})$
13: **end if**

support or not (Line 10) can be done efficiently by checking if there exists a solution $\hat{x}$ such that $\|y - H^J \hat{x}^J\|_p \leq \alpha$ or not (Remark 2).

We observe that the forbidden support J obtained at Line 10 (if any) can be expected to be small (by construction), hence, the forbidden support inequality associated with J may include too many variables b_j (specifically, all the columns in $\bar{J}$). Although this inequality would serve as a cutting plane as such, it may also be strengthened by extending J to a wider (preferably maximal) forbidden support before adding the cut. We do this in Line 11 by applying Algorithm 2, which we describe next.

3.1.1 Extension of a forbidden support.

If a forbidden support $J \subseteq [m]$ is not a maximal one (i.e., $J \cup \{j\}$ is also a forbidden support for some $j \in [m] \setminus J$), then the obtained inequality (8), may be strengthened by adding columns to J, thus removing variables from the associated inequality. This extension of a set J can be done in several ways, resulting in different (maximal) forbidden supports.

In our implementation, shown in Algorithm 2, we consider the complement $\bar{J}$ of the forbidden support J and we iteratively move columns from $\bar{J}$ to J until achieving maximality of J. The computational cost of this procedure is mostly incurred when checking if J is still a forbidden support after adding columns to this set. To reduce the number of times this check is needed, we perform the following steps. First, we copy the elements of $\bar{J}$ into an array A. Then, at each iteration we apply a binary search on A to find the biggest index i such that $J \cup \{A[1], \ldots, A[i]\}$ is still a forbidden support, where $A[j]$ is the j-th element of array A (Line 4). If such index exists, we add elements $A[1], \ldots, A[i]$ to J and remove them from A. Finally, we remove the first element of the current vector A (if it is not yet empty), and we iterate this process until A is empty.

We remark that the elements of $\bar{J}$ are copied into A in the order in which they appear in $\bar{J}$. This detail is important, since different orderings may produce different results. We remark also that when Algorithm 2 is called from Algorithm 1, the elements of $\bar{J}$ are sorted according to the values $\hat{b}_j$ in the corresponding fractional solution, in a non-increasing order.

Algorithm 2 Extension of a forbidden support

Require: an ordered list with the complement $\bar{J}$ of a forbidden support.
1: $J \leftarrow [m] \setminus \bar{J}$ (i.e., the forbidden support associated to $\bar{J}$)
2: $A \leftarrow$ array with elements of $\bar{J}$ (sorted as in $\bar{J}$)
3: **while** A is not empty **do**
4: $i \leftarrow$ biggest index i such that $J \cup \{A[1], \ldots, A[i]\}$ is a forbidden support, or 0 if such i does not exist
5: **if** $i > 0$ **then**
6: $J \leftarrow J \cup \{A[1], \ldots, A[i]\}$
7: Remove $A[1], \ldots, A[i]$ from A
8: **end if**
9: Remove the first element from A, if any.
10: **end while**
11: **return** J

We make this remark, because later, in Section 4.2, we resort again to Algorithm 2 (in a slightly different context) but sorting $\bar{J}$ differently.

3.2 A rounding heuristic

In the context of a branch-and-cut algorithm for a MIP, it is essential to find reasonable, feasible integer solutions as soon as possible, as this helps to cut branches off on the branching tree before even exploring them. To this end, a classical technique called *rounding heuristic* consists in taking the optimal solution of the continuous relaxation of the model (usually a fractional solution) and trying to obtain a feasible integer solution from it by rounding the fractional values up or down in some "intelligent" way (which usually depends on the problem being solved). Note that this heuristic can be applied to every subproblem of the branch-and-bound tree whenever a fractional solution is found on the continuous relaxation.

We developed a rounding heuristic that takes a fractional solution $(\hat{x}, \hat{w}, \hat{b})$ and attempts to build from it a better integer solution than the incumbent. The idea of the procedure lies in the presumption that a higher value of variable b_j indicates a stronger influence of column j in the solution. Hence, the algorithm sorts the elements of $\hat{b}$ in a non-increasing order and takes the first k columns for which a feasible solution can be found. As far as the selection of these k columns is concerned, even when we can efficiently test whether a given set of columns can yield a feasible solution or not (by Remark 2), this testing may be required several times in order to find these columns. To minimize the number of these tests, we consider a lower bound lb and an upper bound ub for the support of the desired solution, and we perform a binary search to find k in the interval $[\mathsf{lb}, \mathsf{ub}]$, thus achieving our goal with a logarithmic number of required tests. The lower bound is given by the lower bound of the current branch in the branching tree, while the upper bound is given by the size of the incumbent.

4 An ILP formulation for $\mathcal{P}_{0/p}$

Section 3 shows that forbidden support inequalities (8) can be used as cutting planes to help in the solution of $\mathsf{MIP}_{0/p}$. In this section, we explore further these valid inequalities, and we show that they suffice to describe the projection of the feasible solutions of $\mathsf{MIP}_{0/p}$ onto the space of the binary variables b. Based on this fact, we introduce a novel integer linear programming formulation for $\mathcal{P}_{0/p}$, with exponentially many constraints, and we propose a practical algorithm to tackle its solution. We conclude the section with a discussion on the combinatorial structure of the proposed new formulation and its implications for some potential improvements for both this formulation and $\mathsf{MIP}_{0/p}$.

4.1 On the projection of $\mathsf{MIP}_{0/p}$on the binary variables space

The projection of the feasible solutions of $\mathsf{MIP}_{0/p}$ onto the space of variables b is the set of all vectors $b \in \{0,1\}^m$ characterizing *feasible supports*, i.e., all those supports for which a feasible solution of $\mathsf{MIP}_{0/p}$ exists. To be precise, as $\mathsf{MIP}_{0/p}$ depends on the value for the bound M, we define $\mathcal{S}(M) := \{b \in \{0,1\}^m \mid (x,b,w)$ is a feasible solution to $\mathsf{MIP}_{0/p}(M)$ for some $(x,w)\}$. It is trivial to see that $\mathcal{S}(M_1) \subseteq \mathcal{S}(M_2)$ whenever $M_1 < M_2$, and that for a sufficiently large value for M, the set $\mathcal{S}(M)$ is the set of all feasible supports for $\mathcal{P}_{0/p}$. Therefore, we get the following.

Remark 4 *The set $\mathcal{S} := \lim_{M\to\infty} \mathcal{S}(M)$ is the set of all feasible supports for $\mathcal{P}_{0/p}$.*

In light of Remarks 2 and 4, we can find an optimal solution for $\mathcal{P}_{0/p}$ by finding a support $b \in \mathcal{S}$ of minimum size and then obtaining a proper solution x with support b. However, to find such vector b, we need a proper characterization of $\mathcal{S}$. The following theorem shows that such characterization can be obtained by using the forbidden support inequalities (8).

Theorem 5 *For a sufficiently large value of M, the projection on the variables b_j of all feasible solutions of formulation $\mathsf{MIP}_{0/p}(M)$ is described by the forbidden support inequalities* (8)*. Formally, the set*

$$\mathcal{S} := \lim_{M\to\infty} \mathcal{S}(M)$$

is equal to the set

$$\hat{\mathcal{S}} := \{b \in \{0,1\}^m \mid b \text{ satisfies } (8) \text{ for each forbidden support } J \subseteq [m]\}.$$

Proof The fact that $\mathcal{S} \subseteq \hat{\mathcal{S}}$ is clearly given by Proposition 3. In order to prove the converse, we take an arbitrary solution $\hat{b} \in \hat{\mathcal{S}}$ and its support $J = \{j \in [m] \mid \hat{b}_j \neq 0\}$. Clearly, J is not a forbidden support for $\mathcal{P}_{0/p}$ as $\sum_{j\in[m]\setminus J} \hat{b}_j = 0$ and $\hat{b}$ satisfies (8) for each forbidden support. Therefore, by taking $\hat{x} := \arg\min_{x\in\mathbb{R}^m}\{\|y - H^J x^J\|_p\}$ (with $\hat{x}_j = 0$ for $j \notin J$) we get a feasible solution of $\mathsf{MIP}_{0/p}(\|\hat{x}\|_\infty)$, hence $\hat{b} \in \mathcal{S}(\|\hat{x}\|_\infty)$, thus proving that $\hat{b} \in \mathcal{S}$. ■

Theorem 5 lets us state the following formulation to obtain a minimum support of a solution to $\mathcal{P}_{0/p}$.

$$[\mathsf{IP}^{cov}_{0/p}] \qquad \min \sum_{j\in[m]} b_j \tag{9}$$

$$\sum_{j\in[m]\setminus J} b_j \geq 1 \qquad \forall \text{ forbidden support } J \subseteq [m] \tag{10}$$

$$b_j \in \{0,1\} \qquad \forall j \in [m] \tag{11}$$

We recall that by solving $\mathsf{IP}^{cov}_{0/p}$, we do not obtain a solution for $\mathcal{P}_{0/p}$ but just an optimal support $J \subseteq [m]$. However, by Remark 2, a solution $\hat{x}$ for this support can be efficiently obtained afterward. A remarkable characteristic of this proposed method is that there is no need to use the (usually artificial) big-M bounds for $\hat{x}$ (as the support J is already fixed for this last step), and no non-linearities are introduced either. This fact gives $\mathsf{IP}^{cov}_{0/p}$ an important advantage against formulation $\mathsf{MIP}_{0/p}$, as it does not require any bound M on the values of x. Therefore, it always finds a proper optimal solution for $\mathcal{P}_{0/p}$ (in contrast with $\mathsf{MIP}_{0/p}$ which cannot be considered an exact method for $\mathcal{P}_{0/p}$, as it was discussed in Section 2).

As the reader may have already noticed, a potential drawback towards the computational solution of $\mathsf{IP}^{cov}_{0/p}$ is that the formulation may have exponentially many constraints (10). However, in Section 4.2 we propose a practical approach to tackle this issue in the solution of $\mathsf{IP}^{cov}_{0/p}$, and in Section 5 we show how this approach may obtain exciting results in practice.

4.2 A tractable approach for the solution of $\mathsf{IP}^{cov}_{0/p}$

The main challenge for solving $\mathsf{IP}^{cov}_{0/p}$ is the exponential constraint set (10) present in the formulation. However, only a small (but a priori unknown) subset of these constraints is likely necessary to obtain an optimal solution. Moreover, due to Theorem 5, we can determine if a vector $b \in \{0,1\}^m$ satisfies all these constraints without the need of enumerating them; namely, a vector $b \in \{0,1\}^m$ satisfies constraints (10) if and only if $b \in \mathcal{S}$, i.e., if the support described by b is not a forbidden support, which by Remark 2 can be efficiently verified. Precisely, for $p \in \{1,\infty\}$, this can be determined by solving a linear program, and for $p = 2$, the problem reduces to solve a least-squares problem. In this section, we propose a procedure that exploits all these characteristics to solve $\mathsf{IP}^{cov}_{0/p}$ by dynamically adding constraints (10) whenever an integral infeasible solution is found.

The procedure, shown in Algorithm 3, starts with a relaxed version of $\mathsf{IP}^{cov}_{0/p}$ with an (initially empty) subset $\mathcal{J}$ of constraints (10), i.e., a *combinatorial relaxation*. At each iteration, this relaxation is solved to optimality, and a support $J \subseteq [m]$ is obtained[1]. This support is then tested (in Line 4) to verify if there exists a feasible solution $\hat{x}$ associated with it. If such $\hat{x}$ exists, the algorithm stops with this optimal solution. Otherwise, J is a forbidden support, hence an inequality (10) can be added to the formulation, as in Line 9, thus "cutting out" the previous (infeasible) solution J. Note that the termination of the algorithm is guaranteed as the set of forbidden supports is finite.

1. In our implementation, we solve this IP by using the commercial solver CPLEX. More details are given in Section 5.

Algorithm 3 Solution procedure for $\mathsf{IP}^{cov}_{0/p}$

1: $\mathcal{J} \leftarrow \{\}$ *// Forbidden supports $J \in \mathcal{J}$ will be used as constraints* (10)
2: **repeat**
3: Solve $\mathsf{IP}^{cov}_{0/p}$ **only with constraints** (10) associated to $\mathcal{J}$ to obtain an optimal support $J \subseteq [m]$
4: **if** there exists a point $\hat{x} \in \mathbb{R}^m$ with support J and $\|y - H^J \hat{x}^J\|_p \leq \alpha$ **then**
5: End the procedure with optimal solution $\hat{x}$
6: **else**
7: *// J is a forbidden support for $\mathcal{P}_{0/p}$*
8: Extend J to a wider forbidden support (using Algorithm 2).
9: $\mathcal{J} \leftarrow \mathcal{J} \cup \{J\}$
10: **end if**
11: **until** a solution is found

We note that the forbidden support J obtained in Line 3 is usually expected to be small, as it is the minimum support of the relaxed version of $\mathsf{IP}^{cov}_{0/p}$. Hence, the forbidden support inequality associated with J may include too many variables b_j (specifically, those in the complement of J). Although this inequality would serve the goal of the algorithm, if we restrict ourselves to using just these forbidden supports, one can easily show that the number of iterations to reach an optimal solution with support size k would be in the order of $\binom{m}{k}$, which is of course prohibitive in practice. To avoid this issue, the forbidden supports obtained in Line 3 are extended to wider (maximal) forbidden supports before adding them to $\mathcal{J}$. We do this in Line 8 by resorting to Algorithm 2 (described in Section 3.1.1). As we mention earlier, the results of Algorithm 2 are highly sensitive to the ordering on the list $\bar{J}$. In this case, we sort these elements by their *frequency* in previous optimal solutions, i.e., a column which appeared less often in optimal solutions is prioritized to be included in the resulting forbidden support J. In any case, we remark that our Algorithm returns always maximal forbidden supports, as inequalities with non-maximal supports can always be strengthened. In Section 4.4 we give further theoretical details about the strength of these inequalities

We shall note that Algorithm 3 can be implemented in practice by employing the so-called *lazy constraints call-back* mechanism, usually provided by general MIP solvers. This technique allows us to solve a formulation with a subset of its constraints (e.g., the set $\mathcal{J}$ above), and whenever an integer solution is found, a call-back method is executed in order to find a violated constraint to add to the current formulation (e.g., lines 4–10). By these means, the solution of the model in one iteration may take advantage of the work done in previous iterations, thus drastically reducing the solution times.

4.3 Solving the relaxations of $\mathsf{IP}^{cov}_{0/p}$

During the execution of Algorithm 3, several relaxed versions of $\mathsf{IP}^{cov}_{0/p}$ need to be solved (i.e., for different sets $\mathcal{J}$ of forbidden supports inequalities). In order to help in the solution of these formulations, a general IP solver can be aided by the addition of several elements, some of them already discussed in Section 3 for the solution of $\mathsf{MIP}_{0/p}$; namely, cutting planes

and rounding heuristics. These same elements may also be implemented in a branch-and-cut scheme to help in the solution of the mentioned relaxed versions of $\mathsf{IP}^{cov}_{0/p}$. Indeed, both Algorithm 1 (to generate cutting planes) and the rounding heuristic described in Section 3.2, work on the basis of variables b from $\mathsf{MIP}_{0/p}$ (without the need of variables x). Therefore, we implemented the same approaches to improve the performance of Algorithm 3 when solving the successive relaxed versions of $\mathsf{IP}^{cov}_{0/p}$ (Line 3).

4.4 On the combinatorial structure of $\mathsf{IP}^{cov}_{0/p}$

The equivalence given by Theorem 5 has additional (potentially useful) implications. In particular, we note that $\mathsf{IP}^{cov}_{0/p}$ represents a *minimum set covering* problem, and this kind of problems has been widely studied in the literature both in the polyhedral and in the combinatorial aspects (Balas and Ng, 1989a,b; Borndörfer, 1998; Cornuéjols and Sassano, 1989; Laurent, 1989; Nobili and Sassano, 1989; Sánchez-García et al., 1998; Sassano, 1989). A direct implication of this fact is that any valid inequality for these set covering polytopes is a valid inequality for both $\mathsf{IP}^{cov}_{0/p}$ and $\mathsf{MIP}_{0/p}$. Thus it can be used as cutting planes in a branch-and-cut algorithm. We give next an example in which we present a family of valid inequalities for $\mathsf{IP}^{cov}_{0/p}$ and $\mathsf{MIP}_{0/p}$ obtained from a known family of set covering (facet-defining) valid inequalities (Balas and Ng, 1989a).

Proposition 6 *Let $\mathcal{J} \subseteq 2^{[m]}$ be a family of forbidden supports for $\mathcal{P}_{0/p}$, and define $J_0 := [m] \setminus \bigcup_{J\in\mathcal{J}} J$, and $J_* := [m] \setminus (J_0 \cup \bigcap_{J\in\mathcal{J}} J)$. Then the* forbidden support family inequality

$$\sum_{j\in J_0} 2b_j + \sum_{j\in J_*} b_j \geq 2 \tag{12}$$

is valid for $\mathsf{MIP}_{0/p}$ and for $\mathsf{IP}^{cov}_{0/p}$.

Proof The validity is implied by 5 and the fact that (12) is a valid inequality for the set covering polytope (Balas and Ng, 1989a). ■

We may remark that when $|\mathcal{J}| = 1$, then (12) is a forbidden support inequality (8), multiplied by 2. Also, we know from the literature (Balas and Ng, 1989a) that (12) are the only facet-defining inequalities for (the convex hull of feasible solutions of) $\mathsf{IP}^{cov}_{0/p}$ with coefficients in $\{0, 1, 2\}$ and a right-hand side equal to 2. Adversely, it is not clear if variables x from formulation $\mathsf{MIP}_{0/p}$ may be added to (12) in order to obtain a stronger (e.g., facet-defining) inequality for this formulation. Another interesting result, derived from the literature, relating to the strength of forbidden support inequalities with respect to $\mathsf{IP}^{cov}_{0/p}$ is given in the following proposition.

Proposition 7 *The constraint* (8) *associated to a forbidden support J defines a facet of $\mathsf{IP}^{cov}_{0/p}$ if and only if:*

(i) J is a maximal forbidden support; and

(ii) for each $j \in J$, there exists $k \in [m] \setminus J$ such that $k \notin J'$ for every forbidden support J' which contains $J \setminus j$.

Proof Given a set covering polytope $P_I = \text{conv}(\{x \in \{0,1\}^m \mid Ax \geq 1\})$ with $A := \{(a_{ij})\} \in \{0,1\}^{n \times m}$, Balas and Ng (1989a) prove that row i of A, with $M^i := \{k \in [m] \mid a_{ij} = 1\}$, defines a facet of P_I if and only if:

(a) there exist no $i' \in [n]$ such that $M^{i'} \subset M^i$, and

(b) for each column $j \in [m] \setminus M^i$, there exists another column $k(j) \in M^i$ such that $a_{hk(j)} = 1$ for all $h \in [n]$ for which $a_{hj} = 1$ and $a_{hk} = 0, \forall k \in [m] \setminus M^i \cup \{j\}$.

Considering that $\mathsf{IP}_{0/p}^{cov}$ is a set covering polytope of the mentioned structure, it is easy to check that *(a)* and *(b)* imply *(i)* and *(ii)*, respectively. ■

In Section 4.2 we proposed an algorithm to tackle the fact that $\mathsf{IP}_{0/p}^{cov}$ may have an exponential number of constraints. The algorithm consists in finding forbidden supports iteratively as needed. Proposition 7 gives some insight on which characteristics should be prioritized when looking for the latter, since using only facet-defining constraints is enough to solve the problem. Our implementation aims at finding forbidden supports satisfying property *(i)* of the proposition, however it does not focus on property *(ii)*, as it is not clear how to include this characteristic without significantly increasing the computational bargain of the process.

We conclude this section by noting the fact that there exist several families of known valid inequalities for the set covering polytope, such as, e.g., the *generalized web* (Sassano, 1989) and *antiweb* (Laurent, 1989; Sassano, 1989) inequalities. Facet-generating procedures were also introduced in the set covering literature, leading to new families of valid inequalities obtained by the composition or other routines (Nobili and Sassano, 1989). All these elements may represent valuable tools to enhance the solution of both $\mathsf{IP}_{0/p}^{cov}$ and $\mathsf{MIP}_{0/p}$, as any of these inequalities is valid for these formulations. We also note that by means of a simple variable substitution (by defining $\bar{b}_j := 1 - b_j$), the model $\mathsf{IP}_{0/p}^{cov}$ can be transformed into a *general maximum set packing problem*, widely studied in the literature also. We should remark however, that this transformation leads to a packing problem in which the right-hand-side of the constraints depends on the size of the associated forbidden set, contrarily to our original set covering formulation in which every right-hand-side is 1. Hence, it is not clear if the mentioned re-formulation may be profitable or not, and so we prefer to keep the set covering formulation instead. An excellent compendium on the set covering (and the *set packing*) problem can be found in (Borndörfer, 1998). We note that the implementation of separation routines for (12) and/or the latter mentioned inequalities escapes the scope of this work.

5 Computational results

In this section, we conduct computational experimentations intending to test the potential of the approaches presented in this work to tackle problem $\mathcal{P}_{0/p}$. In particular, we compare the performance of solving $\mathcal{P}_{0/p}$ by the three following approaches:

1. By iteratively solving formulation $\mathsf{MIP}_{0/p}$ with the algorithm proposed in Bourguignon et al. (2016) (explained in Section 2). We denote this approach simply as $\mathsf{MIP}_{0/p}$.

2. By using the above algorithm but solving the successive formulations of $\mathsf{MIP}_{0/p}$ with the branch-and-cut algorithm proposed in Section 3. We denote this approach as $\mathsf{BC}_{0/p}$.

3. By solving $\mathsf{IP}^{cov}_{0/p}$ with Algorithm 3 (introduced in Section 4.2) and applying the branch-and-cut depicted in Section 4.3 to solve the successive relaxed versions of $\mathsf{IP}^{cov}_{0/p}$. We denote this approach simply as $\mathsf{IP}^{cov}_{0/p}$.

All the above approaches are meant to solve $\mathcal{P}_{0/p}$ for $p \in \{1, 2, \infty\}$. However, we shall remark two characteristics which could significantly change the performance of the methods whether $p = 2$ or $p \in \{1, \infty\}$. On one hand, the formulation solved in $\mathsf{MIP}_{0/p}$ and $\mathsf{BC}_{0/p}$ is *linear* when $p \in \{1, \infty\}$ and *non-linear* when $p = 2$. Hence, these methods are expected to be favored when $p \in \{1, \infty\}$. On the other hand, we note that the task of deciding whether a set of columns $J \in [m]$ represents a feasible support or not, is tackled by *linear programming* when $p \in \{1, \infty\}$ and by a simple *least squares problem* when $p = 2$. This task is a crucial element for $\mathsf{IP}^{cov}_{0/p}$, hence the performance of this method may be significantly degraded when $p \in \{1, \infty\}$.

The above approaches were implemented in Java using the general-purpose solver CPLEX (version 12.6) through its Java API to solve the MIP/IP formulations. The cutting planes and the rounding heuristics for the branch-and-cut approaches are incorporated by resorting to CPLEX *call-back* functions (provided in its Java API). For the solution of $\mathsf{IP}^{cov}_{0/p}$, we implemented Algorithm 3 by resorting to the *lazy constraint call-back* mechanism, also provided by this API (as explained at the end of Section 4.2). We shall remark that CPLEX modifies some of its default parameters when these call-back functions are implemented. We use this same configuration when solving $\mathsf{MIP}_{0/p}$ for a fair comparison. The computational framework is an Intel© Xeon© E5-2620 processor clocked at 2.1GHz and limited to 8 GB of RAM.

Our goal is to compare both the running times and the obtained supports by each method. Nonetheless, we should remark that the comparison of running times of $\mathsf{IP}^{cov}_{0/p}$ against the ones of $\mathsf{MIP}_{0/p}$ and $\mathsf{BC}_{0/p}$ is not a fair one, since $\mathsf{IP}^{cov}_{0/p}$ is an exact algorithm whereas $\mathsf{MIP}_{0/p}$ and $\mathsf{BC}_{0/p}$ may yield suboptimal solutions. Indeed, we found several instances in which $\mathsf{MIP}_{0/p}$ and $\mathsf{BC}_{0/p}$ finish the process (within the time limit) with suboptimal solutions. In light of this remark, we base our computational experimentation on the two following sets of instances (which we describe in detail afterward):

- The first test set is based on Bourguignon et al. (2016) and is a set of instances in which $\mathsf{MIP}_{0/p}$ finds almost always optimal solutions. Here, the goal is to compare the performance in the solution times taken by the proposed approaches.

- The second test set resorts to a "pathological" example from the literature, namely the *adversarial strategy* introduced by Mairal and Yu (2012), and it aims to show how the solutions obtained by $\mathsf{MIP}_{0/p}$ can be far from being optimal.

5.1 Deconvolution problems

As a first test-bed, we resort to the sparse deconvolution instances used in Bourguignon et al. (2016). In these instances, x is a K-sparse sequence in $\mathbb{R}^{100}$ with uniformly distributed

			$\mathsf{MIP}_{0/2}$				$\mathsf{BC}_{0/2}$				$\mathsf{IP}^{cov}_{0/2}$			
SNR	K	# ins	solv	time	uns	supp	solv	time	uns	supp	solv	time	uns	supp
10	5	50	49	352	1	5.0	46	274	4	5.8	50	26	0	-
	7	48	8	949	40	7.9	4	612	44	7.9	37	576	11	7.6
	9	16	0	-	16	10.2	0	-	16	10.3	2	898	14	11.1
20	5	50	50	105	0	-	50	24	0	-	50	4	0	-
	7	49	29	700	20	7.9	48	190	1	9.0	49	19	0	-
	9	41	4	673	37	10.6	22	729	19	11.2	41	99	0	-
30	5	50	50	62	0	-	50	9	0	-	50	2	0	-
	7	50	48	529	2	15.0	49	31	1	23.0	50	5	0	-
	9	50	12	1119	38	10.3	50	235	0	-	50	22	0	-
Instances solved...														
...by all:		242		360 sec.				104 sec.				12 sec.		
...by none:		25		9.04				9.36				9.56		

Table 1: Performance of the compared approaches for $\mathcal{P}_{0/2}$ in the deconvolution instances with a time limit of 1800 seconds.

spike locations, H is a discrete convolution matrix corresponding to a 21-sample impulse response, and y is a 120-sample signal (we point the reader to Bourguignon et al. (2016) for more details on these instances). White noise is added with a variable signal-to-noise ratio (SNR). The authors created 50 instances for each considered SNR and each different value $K \in \{5, 7, 9\}$. Unfortunately, we have had trouble recovering correctly the totality of these instances (as we encountered missing values for some of them). Hence we report results only on the ones we could recover (404 out of the 450 instances).

5.1.1 Results for $\mathcal{P}_{0/2}$

We start by showing results for problem $\mathcal{P}_{0/2}$, i.e., when the Euclidean ℓ_2 norm is used to measure the misfit error. Table 1 shows the performance of the compared approaches in the set of instances described above. For each group of instances, column "# ins" shows the number of instances available for the group, and the remaining columns show the results obtained by each method. Specifically, columns "solv" and "time" indicate the number of instances solved to optimality (i.e., the algorithm ends before reaching a time limit of 1800 seconds) and the average time taken (in seconds) to solve these instances, respectively. On the other hand, columns "uns" and "supp" indicate the number of unsolved instances and the average support sizes of the best solutions obtained for these instances, respectively. It is worth noting that the average time and support size are calculated on the sets of solved and unsolved instances, respectively, for each method, and these sets may differ from method to method. Therefore, a "fair" comparison is not easy to depict. In this direction, the last two lines of Table 1 summarize the average running times in the set of instances solved by all methods and the average obtained support sizes for the set of instances not solved by any of them.

As evidenced by Table 1, the branch-and-cut approach $\mathsf{BC}_{0/2}$ represents an interesting improvement of the classical approach $\mathsf{MIP}_{0/2}$. In most of the groups, it solves to optimality more instances than $\mathsf{MIP}_{0/2}$, sometimes having a very high difference, as in the group of SNR $= 20$ and $K = 9$, where $\mathsf{BC}_{0/2}$ solves 22 instances against 4 solved by $\mathsf{MIP}_{0/2}$. The running times are also significantly improved, sometimes reducing them in more than one order of magnitude (as in the groups with SNR $= 30$) even when these times consider all solved instances by each method (which may benefit the method solving fewer instances). When considering the average time taken by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ over the 242 instances solved by all three methods, the difference is remarkable (360 seconds against 104). As far as the unsolved instances are concerned, the differences in the average obtained solution support seem not to be significant for all the groups but one, namely SNR $= 30$ and $K = 7$, in which $\mathsf{BC}_{0/2}$ obtains average support of 23.0 while the average obtained by $\mathsf{MIP}_{0/2}$ is 15.0. However, this latter value is the average of two unsolved instances for $\mathsf{MIP}_{0/2}$, with obtained support sizes 9 and 21, and just one of them for $\mathsf{BC}_{0/2}$, with support size 23 (as the other instance is solved to optimality with the proper support size of 7), hence the comparison, in this case, is not considerable. Indeed, when considering the average support obtained by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ over the 25 instances not solved by any of the three methods, the difference is not too big (i.e., around 3.5%).

As far as the results of $\mathsf{IP}^{cov}_{0/2}$ are concerned, Table 1 shows a clear dominance over the other two methods. In particular, $\mathsf{IP}^{cov}_{0/2}$ is able to solve to proven optimality all instances with SNR $\in \{20, 30\}$ and 89 out of the 114 instances with SNR $= 10$. In total, $\mathsf{IP}^{cov}_{0/2}$ solves to proven optimality 379 out of the whole set of 404 instances, while $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ solve only 250 and 319 instances, respectively. In terms of running times, the difference in favor of $\mathsf{IP}^{cov}_{0/2}$ is also remarkable, in particular in the group of instances with SNR $= 30$, in which this difference is approximate of 2 orders of magnitude, with respect to $\mathsf{MIP}_{0/2}$. When considering the average time taken by $\mathsf{IP}^{cov}_{0/2}$ over the 242 instances solved by all three methods, we also can note a remarkable difference (12 seconds against 360 incurred by $\mathsf{MIP}_{0/2}$). The results of $\mathsf{IP}^{cov}_{0/2}$ concerning unsolved instances are not easy to measure as we can only analyze the groups with SNR $= 10$ and $K \in \{7, 9\}$, in which the sets of unsolved instances by $\mathsf{IP}^{cov}_{0/2}$ and the other two methods are considerably different. As an example, when $K = 7$, method $\mathsf{IP}^{cov}_{0/2}$ obtains an average support size of 7.6 for its set of 11 unsolved instances, which seems to be slightly worse than the 7.9 obtained by $\mathsf{BC}_{0/2}$ in 44 unsolved instances, however, if all instances in this group are taken into consideration we can see in our experiments that $\mathsf{IP}^{cov}_{0/2}$ obtains an average support size of 7.01. In contrast, $\mathsf{BC}_{0/2}$ achieves an average of 7.85 (these individual values cannot be deduced from Table 1). Indeed, when considering the 25 instances not solved by any of the three methods, we can see that the average support obtained by $\mathsf{IP}^{cov}_{0/2}$ is again not too far from the ones obtained by the other methods.

5.1.2 Results for $\mathcal{P}_{0/2}$ on underdetermined cases

Our primary focus is not on achieving the fastest computation, but rather on ensuring the optimality of the solution, irrespective of the specific problem scenario, whether it be overdetermined or underdetermined. In this regard, we introduce an underdetermined deconvolution problem using the 'valid convolution' instead of the previously employed 'full

			$\mathsf{MIP}_{0/2}$				$\mathsf{BC}_{0/2}$				$\mathsf{IP}^{cov}_{0/2}$			
SNR	K	# ins	solv	time	uns	supp	solv	time	uns	supp	solv	time	uns	supp
10	5	50	50	54	0	-	50	115	0	-	50	16	0	-
	7	48	46	305	2	7.0	38	274	10	7.8	48	68	0	-
	9	16	12	743	4	8.5	4	684	12	10.2	15	323	1	11.0
20	5	50	50	69	0	-	50	174	0	-	50	25	0	-
	7	49	48	302	1	7.0	35	271	14	7.8	47	52	2	8.0
	9	41	25	572	16	9.1	14	507	27	10.4	33	255	8	10.8
30	5	50	50	39	0	-	50	99	0	-	50	25	0	-
	7	50	48	324	2	7.0	37	353	13	7.8	49	75	1	9.0
	9	50	31	570	19	9.3	17	536	33	10.3	43	327	7	11.7
Instances solved...														
...by all:		292		150 sec.				233 sec.				49 sec.		
...by none:		15		9.06				11.13				11.07		

Table 2: Performance of the compared approaches for $\mathcal{P}_{0/2}$ on underdetermined deconvolution instances with a time limit of 1800 seconds.

convolution' as detailed in the previous section. To illustrate this, we refer back to the instances introduced in Bourguignon et al. (2016). For each of these instances, we generate a new one by following the same procedure, but this time employing the transpose of the matrix H (with dimensions of 100×120).

The results are shown in Table 2 (which follows the same format as Table 1). As far as the effectiveness of the three methods is concerned, it is evident that $\mathsf{BC}_{0/2}$ does not outperform $\mathsf{MIP}_{0/2}$ in these new (underdetermined) instances, as the number of solved instances of the former is worst or equal in every group than the latter, and the running times are always higher. On the other side, it can be seen that $\mathsf{IP}^{cov}_{0/2}$ obtains optimal solutions in more instances than $\mathsf{MIP}_{0/2}$ for most of the groups, as it was the case in the previous experimentation (Table 1), although the gap between these two methods is not as large as for the overdetermined cases. A similar analysis can be done regarding the running times, which for $\mathsf{IP}^{cov}_{0/2}$ are always lower than for $\mathsf{MIP}_{0/2}$. The summary lines at the bottom of the table show again a big difference between $\mathsf{IP}^{cov}_{0/2}$ and $\mathsf{MIP}_{0/2}$, since the former uses on average one third of the time required by the latter. We can see again that in the few instances not solved to optimality, the support provided by $\mathsf{MIP}_{0/2}$ results to be slightly better than the ones provided by $\mathsf{IP}^{cov}_{0/2}$.

5.1.3 Results for $\mathcal{P}_{0/\infty}$

We now present the results obtained for $\mathcal{P}_{0/\infty}$, i.e., when the ℓ_∞ norm is used to measure the misfit error. In this case, we modified the instances replacing H and y by $H^T H$ and $H^T y$, respectively, to obtain instances corresponding to the *Dantzig Selector* (Candes et al., 2007) which is well studied in the literature (see Asif and Romberg 2010 and references therein for more details and the relationship between the Lasso and the Dantzig Selector)

			$\mathsf{MIP}_{0/\infty}$				$\mathsf{BC}_{0/\infty}$				$\mathsf{IP}^{cov}_{0/\infty}$			
SNR	K	# ins	solv	time	uns	supp	solv	time	uns	supp	solv	time	uns	supp
10	5	50	33	340	17	7.4	40	375	10	7.7	32	1012	18	8.4
	7	48	5	1033	43	8.3	10	941	38	8.7	2	1465	46	11.2
	9	16	0	-	16	10.0	0	-	16	10.2	0	-	16	15.1
20	5	50	38	331	12	7.3	45	356	5	7.4	41	964	9	9.7
	7	49	11	717	38	7.9	30	878	19	8.7	11	1340	38	11.6
	9	41	1	1144	40	9.6	2	1095	39	9.9	0	-	41	15.2
30	5	50	44	330	6	7.3	48	290	2	9.0	45	810	5	9.2
	7	50	17	591	33	8.0	37	758	13	8.7	26	1343	24	13.4
	9	50	1	1142	49	9.7	9	1110	41	10.0	3	1443	47	16.2
Instances solved...														
...by all:		129		366 sec.				290 sec.				954 sec.		
...by none:		179		9.04				9.32				13.57		

Table 3: Performance of the compared approaches for $\mathcal{P}_{0/\infty}$ in the deconvolution instances from (Bourguignon et al., 2016) (premultiplied by H^T) with a time limit of 1800 seconds.

That is, the considered problem is

$$\min \|x\|_0 \quad \text{s.t.} \quad \|H^T(y - Hx)\|_\infty \leq \alpha \tag{13}$$

with the same H and y as before. We show these results in Table 3, with the same format as in Table 1.

Table 3 verifies that $\mathsf{BC}_{0/\infty}$ represents an improvement of $\mathsf{MIP}_{0/\infty}$. In this case, the difference is more notorious in the number of solved instances rather than in the running times. In every group but one, $\mathsf{BC}_{0/\infty}$ solves to optimality more instances than $\mathsf{MIP}_{0/\infty}$. There seems to be no significant difference in the running times when considering all solved instances by each method. However, when considering the average time taken by $\mathsf{MIP}_{0/\infty}$ and $\mathsf{BC}_{0/\infty}$ over the 129 instances solved by all three methods, the difference is slightly in favor of the latter (366 seconds against 290). The average obtained solution supports for unsolved instances seem to have an insignificant difference.

As far as the results of $\mathsf{IP}^{cov}_{0/\infty}$ are concerned, Table 3 shows that this method does not perform better than $\mathsf{MIP}_{0/\infty}$. Even when the number of solved instances is similar in almost every group, we can see that the running times are considerably increased when using $\mathsf{IP}^{cov}_{0/\infty}$. Moreover, the average solution support obtained by $\mathsf{IP}^{cov}_{0/\infty}$ is always worse than the one obtained by $\mathsf{MIP}_{0/\infty}$ (due to time-limits). Indeed, when considering the 179 instances not solved by any of the three methods, we can see that the average support obtained by $\mathsf{IP}^{cov}_{0/\infty}$ is almost 50% greater than the one obtained by the other two methods. These results seem to verify the hypothesis stating that when $p \in \{1, \infty\}$, the performance of $\mathsf{IP}^{cov}_{0/p}$ gets degraded while the performance of $\mathsf{MIP}_{0/p}$ and $\mathsf{BC}_{0/p}$ is favored. Nevertheless, we should remark that we are comparing here an exact approach against heuristic ones (i.e., $\mathsf{IP}^{cov}_{0/p}$ against $\mathsf{MIP}_{0/p}$ and $\mathsf{BC}_{0/p}$). We omit the results for $p = 1$ as they follow the same trends as the results for $p = \infty$ presented in Table 3.

5.2 A pathological case study

Intending to test the potential difference in the solutions obtained by $\mathsf{IP}^{cov}_{0/p}$ against $\mathsf{MIP}_{0/p}$ and $\mathsf{BC}_{0/p}$, we resort to a "pathological" example from the literature, namely the *adversarial strategy* introduced in (Mairal and Yu, 2012). Given an instance with t variables, such a strategy consists of building a new instance with $t+1$ variables increasing the complexity by a multiplicative factor. It results in explicit, surprisingly simple, pathological examples, unlike other classical pathological examples such as those for the simplex algorithm or SVMs (we point the reader to (Mairal and Yu, 2012) for more details on this strategy). In the end, these examples have a particularly simple shape:

$$y = (1, 1, \ldots, 1)^T \quad \text{and} \quad H = \begin{pmatrix} \alpha_1 & 2\alpha_2 & 2\alpha_3 & \ldots & 2\alpha_m \\ 0 & \alpha_2 & 2\alpha_3 & \ldots & 2\alpha_m \\ 0 & 0 & \alpha_3 & \ldots & 2\alpha_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \alpha_m \end{pmatrix}$$

with α_k being positive numbers satisfying mild conditions, for $k \in [m]$.

We extract random submatrices from this adversarial strategy matrix as a test-bed. For each generated overcomplete ($m \leq n$) submatrix H, we randomly generate a sparse solution x with a given support size K and then calculate the vector $y := Hx + \varepsilon$, where ε represents a gaussian noise. We set afterwards $\alpha := \|y - Hx\|_p$. We generate 10 instances for each size of H in $\{20 \times 40, 30 \times 60, 40 \times 80\}$ and each value of K in $\{4, 6, 8\}$. We remark that by these means, we know that a solution of support size K always exists for these instances. However, an optimal solution with a support size smaller than K may also exist. Indeed, as seen in the results below, this is the case for some instances with $K = 6$. In the following experiments, we set a time limit of 1800 seconds.

Based on the results shown in Section 5.1.3, we focus the experimentation on problem $\mathcal{P}_{0/2}$, i.e., when the ℓ_2 norm is used to measure the misfit error. As we shall show below, in most of the tested instances, $\mathsf{MIP}_{0/2}$ ends the process before reaching the time limit (i.e., it ends normally), but yet the solution provided is far from being optimal. Our intention with this set of instances is to assess the potential optimality gap of $\mathsf{MIP}_{0/2}$, even when the algorithm ends normally. In this setting, comparing computational times of $\mathsf{MIP}_{0/2}$ and $\mathsf{IP}^{cov}_{0/2}$ makes no sense since it is much more likely for $\mathsf{MIP}_{0/2}$ to have shorter times than $\mathsf{IP}^{cov}_{0/2}$ (as it may end with a solution far from an optimal one). However, we shall report whether the methods reached a time limit in an instance or not, as this fact is essential to the analysis of the results.

Table 4 summarizes the results obtained by $\mathsf{MIP}_{0/2}$, $\mathsf{BC}_{0/2}$ and $\mathsf{IP}^{cov}_{0/2}$ in the test set described above, within a time limit of 1800 seconds. For each group of instances and each method, columns "tl" and "supp" report the number of instances in which the method reached the time limit and the average support size obtained. For each method, column "top" reports the number of instances in which the solution provided by the method is not worse than the solutions provided by the other two methods, and column "best" counts the cases in which the solution is strictly better than the other methods.

The first element we shall remark from the results reported in Table 4 is the difference in the support sizes obtained by $\mathsf{IP}^{cov}_{0/2}$ in comparison with the obtained by the other two

		$\mathsf{MIP}_{0/2}$				$\mathsf{BC}_{0/2}$				$\mathsf{IP}^{cov}_{0/2}$			
Size	K	tl	supp	top	best	tl	supp	top	best	tl	supp	top	best
20×40	4	0	16.2	2	0	0	15.5	2	0	0	4.0	10	8
	6	0	14.7	2	0	1	16.2	2	0	0	5.8	10	8
	8	0	20.5	1	0	1	19.5	1	0	9	8.7	9	9
30×60	4	0	9.8	4	0	2	9.2	5	0	0	4.0	10	5
	6	2	15.8	3	0	4	16.6	3	0	0	5.7	10	7
	8	2	11.9	6	1	5	12.1	5	0	10	9.8	7	4
40×80	4	5	11.0	4	0	5	11.1	4	0	0	4.0	10	6
	6	2	14.6	6	0	5	16.4	6	0	7	7.6	8	4
	8	4	17.2	6	0	6	19.1	5	0	9	10.1	8	4

Table 4: Performance of the compared approaches for $\mathcal{P}_{0/2}$ in the pathological instances extracted from the adversarial strategy (Mairal and Yu, 2012), with a time limit of 1800 seconds.

methods. More precisely, we can see that the support sizes obtained by $\mathsf{IP}^{cov}_{0/2}$ are, on average, around 50% of the support sizes obtained by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$. Additionally, when we focus on the first group (size 20×40), we can see that $\mathsf{MIP}_{0/2}$ does not reach the time limit in any of these instances, meaning that the algorithm ends normally in these cases. However, the obtained supports are far from being optimal, as we can see that the obtained support sizes are more than double the sizes of those solutions found by $\mathsf{IP}^{cov}_{0/2}$. Furthermore, this also holds for the subgroup of $K = 8$, in which $\mathsf{IP}^{cov}_{0/2}$ incurs a time limit in 9 out of the 10 instances of the group, and yet the average support size for $\mathsf{IP}^{cov}_{0/2}$ is 8.7, which is less than half of the one obtained by $\mathsf{MIP}_{0/2}$ (i.e., 20.5). As depicted in Table 4, $\mathsf{IP}^{cov}_{0/2}$ obtains the best solution among the three approaches in 82 out of the 90 instances while $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ achieve this in 34 and 33 instances, respectively. Furthermore, $\mathsf{IP}^{cov}_{0/2}$ obtains strictly better solutions than the other two methods (columns "best") in 55 instances while $\mathsf{MIP}_{0/2}$ does it only in 1 of them (and $\mathsf{BC}_{0/2}$ in none).

To further analyze the gap between the solutions obtained by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ against those obtained by $\mathsf{IP}^{cov}_{0/2}$, we disaggregate in Table 5 the data corresponding to the 40 instances in which the three methods usually finish (i.e., before reaching the time or memory limit). For each individual instance, we report the (optimal) support size obtained by $\mathsf{IP}^{cov}_{0/2}$ and the support sizes of the solutions obtained by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ (columns "supp"). Additionally, for $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ we report the optimality gap computed as $100 \times (s - s_{opt})/s_{opt}$, where s is the corresponding support size and s_{opt} is the optimal support size found by $\mathsf{IP}^{cov}_{0/2}$. In the table, an instance named `n_m_k_i` corresponds to the $\mathtt{i}^{th}$ instance of the group of size $\mathtt{n} \times \mathtt{m}$ with $K = \mathtt{k}$.

As we can see in Table 5, the optimality gap of the solutions obtained by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ has a notable variance. Even when the average gap in these 40 instances is approximately 175%, we can see that it may reach very high values, as is the case of $\mathsf{MIP}_{0/2}$ on instance `30_60_4_9` where the gap reaches 900%, meaning that the obtained solution is 10 times greater than the optimal support.

	$\mathsf{IP}^{cov}_{0/2}$	$\mathsf{MIP}_{0/2}$		$\mathsf{BC}_{0/2}$	
Instance	supp	supp	gap	supp	gap
20_40_4_1	4	13	225%	12	200%
20_40_4_2	4	4	0%	4	0%
20_40_4_3	4	26	550%	26	550%
20_40_4_4	4	29	625%	29	625%
20_40_4_5	4	7	75%	6	50%
20_40_4_6	4	13	225%	12	200%
20_40_4_7	4	18	350%	16	300%
20_40_4_8	4	30	650%	30	650%
20_40_4_9	4	18	350%	16	300%
20_40_4_10	4	4	0%	4	0%
20_40_6_1	6	6	0%	6	0%
20_40_6_2	6	11	83%	11	83%
20_40_6_3	6	31	417%	31	417%
20_40_6_4	6	17	183%	31	417%
20_40_6_5	6	21	250%	24	300%
20_40_6_6	6	17	183%	17	183%
20_40_6_8	5	10	100%	10	100%
20_40_6_9	5	14	180%	12	140%
20_40_6_10	6	6	0%	6	0%
20_40_8_8	7	34	386%	34	386%

	$\mathsf{IP}^{cov}_{0/2}$	$\mathsf{MIP}_{0/2}$		$\mathsf{BC}_{0/2}$	
Instance	supp	supp	gap	supp	gap
30_60_4_1	4	5	25%	4	0%
30_60_4_2	4	5	25%	5	25%
30_60_4_4	4	4	0%	4	0%
30_60_4_5	4	8	100%	8	100%
30_60_4_6	4	4	0%	4	0%
30_60_4_7	4	4	0%	4	0%
30_60_4_8	4	4	0%	4	0%
30_60_4_9	4	40	900%	31	675%
30_60_6_1	6	6	0%	6	0%
30_60_6_2	5	33	560%	42	740%
30_60_6_7	6	32	433%	28	367%
30_60_6_8	6	6	0%	6	0%
30_60_6_9	5	5	0%	5	0%
30_60_6_10	5	10	100%	10	100%
40_80_4_1	4	4	0%	4	0%
40_80_4_2	4	4	0%	4	0%
40_80_4_5	4	4	0%	4	0%
40_80_4_6	4	6	50%	6	50%
40_80_4_10	4	4	0%	4	0%
40_80_6_3	6	6	0%	6	0%

Table 5: Support sizes were obtained for the 40 instances in which the three methods finish normally (i.e., before reaching the time or memory limit).

The reader may have noticed from Table 5 that support sizes obtained by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ may be different for the same instance. This behavior may seem odd at first, as these methods work with the same MIP formulation. However, we shall remark that $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ implement the incremental algorithm from (Bourguignon et al., 2016) (explained in Section 2) in which the MIP formulation is solved several times depending on the values of the successively obtained solutions (i.e., if these values are tight on M, then this bound is increased and the model is solved again). Consequently, $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ may find different (although equivalent) initial solutions, leading to different numbers of iterations on the algorithm, hence reaching different values for the bound M, thus finally yielding different support sizes for the final solution. Indeed, we could verify this is precisely the case for the above instances.

6 Final remarks and future work

Problem $\mathcal{P}_{0/p}$ is a challenging Sparse Approximation problem, combining both global minimization aspects within a combinatorial structure. Existing convex Mixed Integer

Programming approaches for these problems (denoted in this article as $\mathsf{MIP}_{0/p}$) use "big-M" formulations but fails to provide an exact algorithm to tackle the problem of finding a proper value for M. In order to fairly compare exact ℓ_0 minimization with other sparse approaches such as the Lasso, as discussed in Hastie et al. (2020), it appears essential to have an algorithm that warranty to reach such a global minimizer.

In this work, we studied the polytopes arising from the mentioned formulations and derived valid inequalities to improve the performance of the existing methods. In particular, we introduce the family of *forbidden supports* inequalities. As a first solution approach, we used these inequalities to design a branch-and-cut algorithm for these models, which we called here $\mathsf{BC}_{0/p}$. On the other hand, the main contribution of this paper relies on an attractive characteristic of this family of inequalities; we proved that they are sufficient to describe the set of *feasible supports* of $\mathcal{P}_{0/p}$. Based on this result, we introduced a novel integer linear programming formulation for $\mathcal{P}_{0/p}$ which consists in solving a pure combinatorial set covering formulation and which we denote $\mathsf{IP}^{cov}_{0/p}$ here. As far as we know, $\mathsf{IP}^{cov}_{0/p}$ is the first exact integer *linear* programming formulation for $\mathcal{P}_{0/p}$. Since the proposed formulation may have exponentially many constraints, we developed a practical approach to tackle its solution, which starts from a combinatorial relaxation of $\mathsf{IP}^{cov}_{0/p}$ with few (or no) constraints and it dynamically adds constraints as needed during the process. We also strengthen the solution of $\mathsf{IP}^{cov}_{0/p}$ by the addition of cutting planes and rounding heuristics, similarly to the branch-and-cut algorithm $\mathsf{BC}_{0/p}$.

The computational experimentation conducted to assess the potential of the newly developed methods for $\mathcal{P}_{0/p}$ shows interesting results when $p = 2$, i.e., when the ℓ_2 norm is used to measure the misfit error. In this case, our results show a significant improvement by $\mathsf{IP}^{cov}_{0/2}$ and $\mathsf{BC}_{0/2}$ concerning the existing approach $\mathsf{MIP}_{0/p}$, and in particular $\mathsf{IP}^{cov}_{0/2}$ seems to outperform $\mathsf{BC}_{0/2}$ in general. In addition to the fact that $\mathsf{IP}^{cov}_{0/2}$ solves more instances in less time than the other methods, the most crucial characteristic is maybe the fact that it is an exact approach, while the others can only be considered as heuristic ones (unless some bound on the solution values is known beforehand). As evidenced by the computational experimentation with the pathological examples, the gap between the (optimal) support obtained by $\mathsf{IP}^{cov}_{0/2}$ and the ones obtained by $\mathsf{MIP}_{0/2}$ and $\mathsf{BC}_{0/2}$ can be huge, even when the processes end normally (i.e., before reaching time or memory limits). Unfortunately, when using $p \in \{1, \infty\}$, the performance of $\mathsf{IP}^{cov}_{0/p}$ gets seriously degraded (verifying the hypothesis stated in Section 5), and the results in these cases do not seem to improve the existing approaches. However, we could see that the newly developed branch-and-cut algorithm $\mathsf{BC}_{0/p}$ does indeed obtain better results than $\mathsf{MIP}_{0/p}$ in these cases, thus representing an improvement of this approach.

We conclude this paper with a remark about the combinatorial structure of $\mathsf{IP}^{cov}_{0/p}$ (i.e., the minimum set covering structure). We strongly think this structure shall be further exploited to improve the performance of the proposed approaches. Many articles have been written about this problem, both about the polyhedral and the combinatorial aspects. Many families of valid inequalities are known for the set covering polytopes, which may be used as cutting planes within the solution of $\mathsf{IP}^{cov}_{0/p}$ and $\mathsf{MIP}_{0/p}$. Also, there is an appreciable amount of work in algorithms and heuristics to solve this kind of problem, which can be used to improve our newly developed approaches. Nevertheless, we leave these ideas as a future line of further work.

Acknowledgments and Disclosure of Funding

This research was partially supported by Labex DigiCosme (project ANR-11-LABEX-0045-DIGICOSME) operated by ANR as part of the program "Investissement d'Avenir" Idex Paris-Saclay (ANR-11-IDEX-0003-02). Most of this work was carried out while Diego Delle Donne was affiliated to LIX CNRS, Ecole Polytechnique.

References

M. Salman Asif and Justin Romberg. On the lasso and dantzig selector equivalence. In *2010 44th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6, March 2010.

Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1): 1–106, 2012.

Egon Balas and Shu Ming Ng. On the set covering polytope: I. All the facets with coefficients in $\{0, 1, 2\}$. *Mathematical Programming*, 43:57–69, 1989a.

Egon Balas and Shu Ming Ng. On the set covering polytope: II. All the facets with coefficients in $\{0, 1, 2\}$. *Mathematical Programming*, 45:1–20, 1989b.

Dimitris Bertsimas, Angela King, Rahul Mazumder, et al. Best subset selection via a modern optimization lens. *Annals of statistics*, 44(2):813–852, 2016.

Dimitris Bertsimas, Jean Pauphilet, and Bart Van Parys. Sparse regression: Scalable algorithms and empirical performance. *Statistical Science*, 35(4):555–578, 2020.

Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, 27(3):265–274, 2009.

Thomas Bonesky. Morozov's discrepancy principle and tikhonov-type functionals. *Inverse Problems*, 25(1):015015, 2008.

Ralf Borndörfer. *Aspects of Set Packing, Partitioning, and Covering.* PhD thesis, TU Berlin, 1998.

Sébastien Bourguignon, David Mary, and Éric Slezak. Restoration of astrophysical spectra with sparsity constraints: Models and algorithms. *IEEE Journal of Selected Topics in Signal Processing*, 5(5):1002–1013, 2011.

Sébastien Bourguignon, Jordan Ninin, Hervé Carfantan, and Marcel Mongeau. Exact Sparse Approximation Problems via Mixed-Integer Programming: Formulations and Computational Performance. *IEEE Transactions on Signal Processing*, 64(6):1405–1419, March 2016.

Alfred M. Bruckstein, David L. Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1):34–81, 2009.

Emmanuel Candes, Terence Tao, et al. The dantzig selector: Statistical estimation when p is much larger than n. *Annals of statistics*, 35(6):2313–2351, 2007.

Patrick L Combettes and Jean-Christophe Pesquet. A proximal decomposition method for solving convex variational inverse problems. *Inverse problems*, 24(6):065014, 2008.

Gérard Cornuéjols and Antonio Sassano. On the 0, 1 facets of the set covering polytope. *Mathematical Programming*, 43:45–55, 1989.

Wei Dai and Olgica Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE transactions on Information Theory*, 55(5):2230–2249, 2009.

Geoff Davis, Stephane Mallat, and Marco Avellaneda. Adaptive greedy approximations. *Constructive approximation*, 13(1):57–98, 1997.

Ronald A. DeVore and Vladimir N. Temlyakov. Some remarks on greedy algorithm. *Advances in Computational Mathematics*, 5:173–187, 12 1996.

David Donoho, Yaakov Tsaig, Iddo Drori, and Jean-Luc Starck. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 58:1094–1121, 02 2012.

Trevor Hastie, Robert Tibshirani, Ryan Tibshirani, et al. Best subset, forward stepwise or lasso? analysis and recommendations based on extensive comparisons. *Statistical Science*, 35(4):579–592, 2020.

Hussein Hazimeh, Rahul Mazumder, and Ali Saab. Sparse regression at scale: Branch-and-bound rooted in first-order optimization. *Mathematical Programming*, 196(1-2):347–388, 2022.

Ronald R Hocking and RN Leslie. Selection of the best subset in regression analysis. *Technometrics*, 9(4):531–540, 1967.

Suk-Geun Hwang. Cauchy's interlace theorem for eigenvalues of hermitian matrices. *The American mathematical monthly*, 111(2):157–159, 2004.

Sadegh Jokar and Marc Pfetsch. Exact and approximate sparse solutions of underdetermined linear equations. *SIAM Journal on Scientific Computing*, 31:23–44, 01 2008.

Thomas Kleinert, Martine Labbé, Fr¨ ank Plein, and Martin Schmidt. Thereâ€™s no free lunch: on the hardness of choosing a correct big-m in bilevel optimization. *Operations research*, 68(6):1716–1721, 2020.

Monique Laurent. A Generalization of Antiwebs to Independence Systems and Their Canonical Facets. *Mathematical Programming*, 45:97–108, 1989.

Hoai An Le Thi, T Pham Dinh, Hoai Minh Le, and Xuan Thanh Vo. Dc approximation approaches for sparse optimization. *European Journal of Operational Research*, 244(1): 26–46, 2015.

Julien Mairal and Bin Yu. Complexity analysis of the lasso regularization path. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML 12, pages 1835–1842, Madison, WI, USA, 2012. Omnipress.

Stéphane G. Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

R Timothy Marler and Jasbir S Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395, 2004.

Elaine Crespo Marques, Nilson Maciel, Lirida Naviner, Hao Cai, and Jun Yang. A review of sparse recovery algorithms. *IEEE access*, 7:1300–1322, 2018.

Ramzi B. Mhenni. *Méthodes de programmation en nombres mixtes pour l´optimisation parcimonieuse en traitement du signal (in French)*. Phd thesis, École Central Nantes, 2020.

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

Deanna Needell and Joel A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301 – 321, 2009.

Deanna Needell and Roman Vershynin. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Foundations of computational mathematics*, 9(3):317–334, 2009.

Deanna Needell and Roman Vershynin. Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *IEEE Journal of Selected Topics in Signal Processing*, 4:310 – 316, 05 2010.

Paolo Nobili and Antonio Sassano. Facets and lifting procedures for the set covering polytope. *Mathematical Programming*, 45:111–137, 1989.

Yagyensh Chandra Pati, Ramin Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 40–44, 1993.

Alejandro Ribes and Francis Schmitt. Linear inverse problems in imaging. *IEEE Signal Processing Magazine*, 25(4):84–99, 2008.

M. Sánchez-García, M.I. Sobrón, and B. Vitoriano. On the set covering polytope: Facets with coefficients in $\{0, 1, 2, 3\}$. *Annals of Operations Research*, 81:343–356, 1998.

Antonio Sassano. On the facial structure of the set covering polytope. *Mathematical Programming*, 44:181–202, 1989.

Emmanuel Soubies, Laure Blanc-Féraud, and Gilles Aubert. A continuous exact ℓ_0 penalty (cel0) for least squares regularized problem. *SIAM Journal on Imaging Sciences*, 8(3): 1607–1639, 2015.

Emmanuel Soubies, Laure Blanc-Féraud, and Gilles Aubert. New insights on the optimality conditions of the $\ell_2 - \ell_0$ minimization problem. *Journal of Mathematical Imaging and Vision*, pages 1–17, 2019.

Charles Soussen, Jérôme Idier, David Brie, and Junbo Duan. From bernoulli–gaussian deconvolution to sparse signal restoration. *IEEE Transactions on Signal Processing*, 59 (10):4572–4584, 2011.

Charles Soussen, Jérome Idier, Junbo Duan, and David Brie. Homotopy based algorithms for ℓ_0-regularized least-squares. *IEEE Transactions on Signal Processing*, 63(13):3301–3316, 2015.

Robert Tibshirani. Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(3):273–282, 2011.

Ivana Tosic and Sarah Drewes. Learning joint intensity-depth sparse representations. *IEEE Transactions on Image Processing*, 23:2122–32, 05 2014.

Joel A. Tropp and Stephen J. Wright. Computational methods for sparse solution of linear inverse problems. *Proceedings of the IEEE*, 98(6):948–958, 2010.

Laurence A. Wolsey. *Integer Programming*. Wiley, New York, 1998.

Weijun Xie and Xinwei Deng. Scalable algorithms for the sparse ridge regression. *SIAM Journal on Optimization*, 30(4):3359–3386, 2020.

Tong Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11(3), 2010.

Shenglong Zhou, Naihua Xiu, and Hou-Duo Qi. Global and quadratic convergence of newton hard-thresholding pursuit. *Journal of Machine Learning Research*, 22(12):1–45, 2021.