4OR (2021) 19:531–548
https://doi.org/10.1007/s10288-020-00459-6

RESEARCH PAPER

Scanning integer points with lex-inequalities: a finite cutting plane algorithm for integer programming with linear objective

Michele Conforti[1] · Marianna De Santis[2] · Marco Di Summa[1] · Francesco Rinaldi[1]

Received: 5 June 2020 / Accepted: 18 September 2020 / Published online: 23 December 2020

Abstract
We consider the integer points in a unimodular cone K ordered by a lexicographic rule defined by a lattice basis. To each integer point x in K we associate a family of inequalities (lex-inequalities) that define the convex hull of the integer points in K that are not lexicographically smaller than x. The family of lex-inequalities contains the Chvátal–Gomory cuts, but does not contain and is not contained in the family of split cuts. This provides a finite cutting plane method to solve the integer program $\min\{cx : x \in S \cap \mathbb{Z}^n\}$, where $S \subset \mathbb{R}^n$ is a compact set and $c \in \mathbb{Z}^n$. We analyze the number of iterations of our algorithm.

Keywords Nonlinear integer programming · Valid inequalities · Cutting plane method

Mathematics Subject Classification 90C10 · 90C57

M. Conforti, M. Di Summa and F. Rinaldi were supported by the Grant 2015B5F27W of the Italian Ministry of Education, University and Research (MIUR) and by a Grant "SID 2016" of the University of Padova. M. De Santis acknowledges support within the Project Number RP11715C7D8537BA, which has received funding from Sapienza, University of Rome.

✉ Marianna De Santis
marianna.desantis@uniroma1.it

Michele Conforti
conforti@math.unipd.it

Marco Di Summa
disumma@math.unipd.it

Francesco Rinaldi
rinaldi@math.unipd.it

[1] Dipartimento di Matematica "Tullio Levi-Civita", Università degli Studi di Padova, Padua, Italy

[2] Dipartimento di Ingegneria Informatica Automatica e Gestionale, Sapienza Università di Roma, Rome, Italy

1 Introduction

The area of integer nonlinear programming is rich in applications but quite challenging from a computational point of view. We refer to the articles (Belotti et al. 2013b; Burer and Letchford 2012) for comprehensive surveys on these topics. The tools that are mainly used are sophisticated techniques that exploit relaxations, constraint enforcement (e.g., cutting planes) and convexification of the feasible set. Reformulations in an extended space and cutting planes for integer nonlinear programs have been investigated and proposed for some time, see e.g. Ceria and Soares (1999), Frangioni and Gentile (2006), Stubbs and Mehrotra (1999). This line of research mostly provides a nontrivial extension of the theory of disjunctive programming to the nonlinear case. To the best of our knowledge, these results are obtained under some restrictive conditions: Typically, the feasible set is assumed to be convex or to contain 0/1 points only (these cases cover some important areas of application).

In this paper we present a finite cutting plane algorithm for problems of the form

$$\min\{cx : x \in S \cap \mathbb{Z}^n\}, \tag{1.1}$$

where S is a compact subset of $\mathbb{R}^n$ (not necessarily convex or connected) and $c \in \mathbb{Z}^n$. Our algorithm uses a new family of cutting planes which do not make any use of a description of the set S. The cutting planes employed in our algorithm are obtained as follows. We consider the integer points in a unimodular cone K, ordered by a lexicographic rule, associated with a lattice basis. To each integer point x in K, we associate a family of inequalities (lex-inequalities) that, in a sense, is best possible, as it defines the convex hull of the integer points in K that are not lexicographically smaller than x. Our family of cuts includes the Chvátal–Gomory cuts, but it does not contain, nor is it contained in, the family of split cuts.

Our algorithm recursively solves optimization problems of the form $\min\{cx : x \in S \cap P\}$, where P is a polyhedron, and we assume that an algorithm for problems of this type is available as a black box. We remark that as long as this black box is available, no assumption on S other than compactness is required by our algorithm. To the best of our knowledge, this is in contrast to the rest of the literature where convexity (or even polyhedrality) is a common assumption.

The cuts we introduce are *linear* inequalites. As the convex hull of the integer points in a bounded subset of $\mathbb{R}^n$ is a polytope, a finite number of linear inequalities suffices for its characterization, and only n such inequalities determine an optimal point.

Furthermore a finite number of linear inequalities suffices to describe some relevant relaxations of the convex hull of the integer points in a bounded set: Most notably, Dadush et al. (2014) proved that the Chvátal–Gomory closure of a compact convex set is a polytope (whereas this is not the case for the split closure of the set).

However, nonlinear inequalities have also been successfully used to provide elegant convex hull characterizations. We mention the work by Andersen and Jensen (2013) where a formula to describe the convex hull of a split disjunction applied to a second-order cone is provided. Their work is related to the paper by Modaresi et al. (2016), where the authors derive split cuts for convex sets described by a single conic quadratic inequality and extend general intersection cuts to a wide variety of quadratic

sets. Belotti et al. (2013a, 2017) introduce the so called disjunctive conic cuts studying families of quadratic surfaces intersected with two given hyperplanes. Burer and Kılınç-Karzan (2017) extend the works cited above and show that the convex hull of the intersection of a second-order-cone representable set and a single homogeneous quadratic inequality can be described by adding a single nonlinear inequality, defining an additional second-order-cone representable set.

From an algorithmic perspective, deriving a finite cutting plane procedure that uses a well defined family of inequalities does not seem to be straightforward. The oldest and most notable example is Gomory's finite cutting plane algorithm for integer linear programming over bounded sets based on fractional cuts (Gomory 1958, 1963). Other finite cutting plane algorithms (again for bounded sets) can be found in Armstrong et al. (1979), Bell (1973), Bowman and Nemhauser (1970), He and Lee (2017) for integer linear programming and in Lee and Wiegele (2017) for mixed integer linear programming.

While in all the papers cited above the correctness of the algorithms is based on a specific procedure for solving the continuous relaxation, there are methods that only assume that an optimal solution of the continuous relaxation is given by a black box. This is the case for the lift-and-project method of Balas et al. (1993) for mixed 0/1 linear problems, the procedure described by Orlin (1985) for 0/1 integer linear programming, and the algorithm presented by Neto (2012) for integer linear programming over bounded sets.

The family of cuts used in Neto's algorithm is related to ours. As it will be clarified later, the inequalities introduced in Neto (2012) are weaker than the lex-inequalities and, in particular, they are derived under the assumption that a box containing the set S is known.

We notice that a common feature of the above papers is the (explicit or implicit) use of some lexicographic rule for the choice of an optimal solution of the continuous relaxation or the selection of the cut. This seems to be a key tool to prove finite convergence of this type of algorithms.

The paper is organized as follows. In Sect. 2, we introduce the lex-inequalities with their properties. In Sect. 3, we present the cutting plane algorithm and we show that it terminates in a finite number of iterations. In Sect. 4, an instance where the algorithm stops after an exponential number of iterations is provided. Furthermore, we compare the performance of our algorithm with a natural enumeration approach. A comparison with Chvátal–Gomory cuts and split cuts is presented in Sect. 5. Section 6 concludes the paper.

2 Lexicographic orderings and lex-inequalities

A *lattice basis* of $\mathbb{Z}^n$ is a set of n linearly independent vectors $c^1, \ldots, c^n \in \mathbb{Z}^n$ such that for every $v \in \mathbb{Z}^n$ we have that $\lambda_1, \ldots, \lambda_n \in \mathbb{Z}$ in the unique expression $v = \sum_{i=1}^{n} \lambda_i c^i$.

The lex-inequalities that we introduce in this paper are defined for a given lattice basis of $\mathbb{Z}^n$. To simplify the presentation, we first work with the standard basis and then extend the results to general lattice bases.

We will use standard notions in the theory of polyhedra, for which we refer the reader to Schrijver (1986).

2.1 Standard basis

We consider the *lexicographic ordering* $\prec$ associated with the standard basis $e^1, \ldots, e^n$: Given $x^1, x^2 \in \mathbb{R}^n$, $x^1 \prec x^2$ if and only if $x^1 \neq x^2$ and $x^1_i < x^2_i$, where i is the smallest index for which $x^1_i \neq x^2_i$. We use $\preceq, \succ, \succeq$ with the obvious meaning.

We consider the cone $K = \mathbb{R}^n_+ = \{x \in \mathbb{R}^n : x_i \geq 0, i = 1, \ldots, n\}$. Given $\bar{x} \in K \cap \mathbb{Z}^n$, we define

$$Q(\bar{x}) := \text{conv}\{x \in K \cap \mathbb{Z}^n : x \succeq \bar{x}\},$$

where "conv" denotes the convex hull operator.

Given $\bar{x} \in K \setminus \{\mathbf{0}\}$, we define the *leading index* $\hat{\imath}(\bar{x})$ as the largest index i such that $\bar{x}_i > 0$.

Lemma 1 *Fix $\bar{x} \in K \cap \mathbb{Z}^n$. The set $Q(\bar{x})$ is a full-dimensional polyhedron. Its vertices are precisely the following points $v^1, \ldots, v^{\hat{\imath}(\bar{x})}$: For $k = 1, \ldots, \hat{\imath}(\bar{x}) - 1$, v^k has entries*

$$\begin{aligned} v^k_i &= \bar{x}_i, && i = 1, \ldots, k-1 \\ v^k_k &= \bar{x}_k + 1 \\ v^k_i &= 0, && i = k+1, \ldots, n, \end{aligned}$$

and $v^{\hat{\imath}(\bar{x})} = \bar{x}$. Furthermore, the recession cone of $Q(\bar{x})$ is K.

Proof Define $X := \{x \in K \cap \mathbb{Z}^n : x \succeq \bar{x}\}$ and $X_i = v^i + (K \cap \mathbb{Z}^n)$ for every $i \in \{1, \ldots, \hat{\imath}(\bar{x})\}$. Note that $X = \bigcup_{i=1}^{\hat{\imath}(\bar{x})} X_i$ and therefore $\text{conv}(X) = \text{conv}\left(\bigcup_{i=1}^{\hat{\imath}(\bar{x})} \text{conv}(X_i)\right)$. As any rational polyhedral cone is an integral polyhedron, we have $\text{conv}(K \cap \mathbb{Z}^n) = K$. As $v^i \in \mathbb{Z}^n$, this implies $\text{conv}(X_i) = v^i + K$ for every i, and thus these sets are integer translates of K. Therefore $\text{conv}(X) = \text{conv}\{v^1, \ldots, v^{\hat{\imath}(\bar{x})}\} + K$. This shows that $\text{conv}(X)$ is a full-dimensional polyhedron with recession cone K and its vertices are contained in $\{v^1, \ldots, v^{\hat{\imath}(\bar{x})}\}$. It is easy to verify that $v^1, \ldots, v^{\hat{\imath}(\bar{x})}$ are actually all vertices of $\text{conv}(X)$. □

Let $\bar{x} \in K$ be given. Our aim is to derive the inequalities that fully describe the convex hull of the integer points in K not lexicographically smaller than $\bar{x}$.

For every $k \in \{1, \ldots, n\}$ and $i \in \{1, \ldots, k\}$ we define

$$d^k_i := \begin{cases} 1 & \text{if } i = k \\ \bar{x}_k & \text{if } i = k-1, \\ \bar{x}_k \prod_{j=i+1}^{k-1} (\bar{x}_j + 1), & \text{if } i \leq k-2. \end{cases}$$

(Note that the d_i^k's depend on the choice of $\bar{x}$, but we omit the dependence on $\bar{x}$ to keep notation simpler: This will never generate any ambiguity.)

For every $k \in \{1, \ldots, n\}$, the *k-th lex-inequality* associated with $\bar{x}$ is the inequality

$$\sum_{i=1}^{k} d_i^k x_i \geq \sum_{i=1}^{k} d_i^k \bar{x}_i. \tag{2.1}$$

Note that when $\bar{x}_k = 0$, (2.1) is the inequality $x_k \geq 0$.

Theorem 2 *If $\bar{x} \in K \cap \mathbb{Z}^n$, then the lex-inequalities (2.1) for $k = 1, \ldots, n$ and the inequalities $x_i \geq 0$ for $i = 1, \ldots, n$ provide a description of the polyhedron $Q(\bar{x})$.*

Proof As K is the recession cone of $Q(\bar{x})$ (Lemma 1) and $Q(\bar{x}) \subseteq K$, it follows that every facet inducing inequality for $Q(\bar{x})$ (indeed every valid inequality) is of the type

$$\sum_{i=1}^{n} a_i x_i \geq a_0 \tag{2.2}$$

where $a_i \geq 0$, $i = 0, \ldots, n$.

Given $k \in \{1, \ldots, n\}$, we let $Q_k(\bar{x}) \subseteq \mathbb{R}^k$ denote the orthogonal projection of $Q(\bar{x})$ onto the first k variables, and we define $\bar{x}_{[k]} := (\bar{x}_1, \ldots, \bar{x}_k)$. It follows from the definition of lexicographic ordering that $Q_k(\bar{x}) = Q(\bar{x}_{[k]})$.

Therefore the facet inducing inequalities of $Q_k(\bar{x})$ are the facet inducing inequalities of $Q(\bar{x})$ such that $a_j = 0$ for $j = k+1, \ldots, n$. (This can be seen, e.g., as a consequence of the method of Fourier–Motzkin to compute projections.)

As the theorem trivially holds for $Q_1(\bar{x})$, to prove the result by induction on n it suffices to characterize the facets with $a_n > 0$. As the only facet inducing inequality with $a_n > 0$ and $a_0 = 0$ is $x_n \geq 0$, from now on we consider a facet inducing inequality (2.2) with $a_n > 0$ and $a_0 > 0$.

Assume first that $\bar{x}_n = 0$. Then by Lemma 1 we have that $Q(\bar{x}) = Q_{n-1}(\bar{x}) \times \{x_n \in \mathbb{R} : x_n \geq 0\}$ and we are done by induction. Therefore we assume $\bar{x}_n > 0$. Recall that, by Lemma 1, $Q(\bar{x})$ has n vertices, $v^1, \ldots, v^n = \bar{x}$.

Claim 1 *$\bar{x}$ satisfies (2.2) at equality.*

Since $v_n^k = 0$ for $k = 1, \ldots, n-1$, if $\bar{x}$ does not satisfy (2.2) at equality, the inequality

$$\sum_{i=1}^{n-1} a_i x_i + (a_n - \varepsilon) x_n \geq a_0$$

is valid for $Q(\bar{x})$ for some $\varepsilon > 0$. Since (2.2) is the sum of $\varepsilon x_n \geq 0$ and the above inequality, and these inequalities are not multiples of each other as $a_0 > 0$, (2.2) does not induce a facet of $Q(\bar{x})$. This proves Claim 1.

Claim 2 *$a_k > 0$ for $k = 1, \ldots, n$.*

By Claim 1 we have that

$$\sum_{i=1}^{n} a_i \bar{x}_i = a_0.$$

Pick $k \in \{1, \ldots, n-1\}$. Since v^k satisfies (2.2), we have that

$$\sum_{i=1}^{k} a_i \bar{x}_i + a_k \geq a_0.$$

Subtracting the above equation from this inequality, we obtain

$$a_k \geq \sum_{i=k+1}^{n} a_i \bar{x}_i > 0,$$

where the strict inequality follows because $a_i \geq 0$ for $i = 1, \ldots, n$ and $a_n \bar{x}_n > 0$. This proves Claim 2.

Claim 2 shows that if $x'' \neq x'$, $x'' \geq x'$ (componentwise) and x' satisfies (2.2), then x'' cannot satisfy (2.2) at equality. In particular, if x' satisfies (2.2) at equality and r is a nonzero ray of $Q(\bar{x})$ then $x' + r$ cannot satisfy (2.2) at equality. Note that $Q(\bar{x})$ is a full dimensional polyhedron with exactly n vertices and (2.2) induces a facet containing no nonzero rays. Therefore, inequality (2.2) must be satisfied at equality by $v^1, \ldots, v^n$. By imposing these n equations and considering the expression of $v^1, \ldots, v^n$ (see Lemma 1), one obtains a linear system in $a_0, \ldots, a_n$ whose solution is, up to scaling, $a_i = d_i^n$, for all $i = 1, \ldots, n$ and $a_0 = \sum_{i=1}^{n} d_i^n \bar{x}_i$. This implies that (2.2) is

$$\sum_{i=1}^{n} d_i^n x_i \geq \sum_{i=1}^{n} d_i^n \bar{x}_i$$

and the theorem is proven. □

Remark 3 In the description given by Theorem 2, for every k such that $\bar{x}_k = 0$ the k-th lex-inequality is redundant, as it is the inequality $x_k \geq 0$. Furthermore, if $\bar{x}_1 > 0$ then also the inequality $x_1 \geq 0$ is redundant, as it is dominated by the first lex-inequality (which is $x_1 \geq \bar{x}_1$). It can be verified that the remaining inequalities provide an irredundant description of $Q(\bar{x})$.

2.2 General lattice bases

Let $\{c^1, \ldots, c^n\}$ be a lattice basis of $\mathbb{Z}^n$. Then the $n \times n$ matrix C whose rows are $c^1, \ldots, c^n$ is unimodular, i.e., it is an integer matrix with determinant 1 or -1. The unimodular transformation $x \mapsto Cx$ and its inverse map integer points to integer points. By applying the transformation $x \mapsto Cx$, the results of the previous subsection can be immediately extended to the lattice basis $\{c^1, \ldots, c^n\}$.

In particular, the lexicographic ordering defined by the lattice basis is as follows: Given $x^1,\ x^2 \in \mathbb{R}^n$, we have $x^1 \prec x^2$ if and only if $x_1 \neq x_2$ and $c^i x^1 < c^i x^2$, where i is the smallest index for which $c^i x^1 \neq c^i x^2$.

The unimodular cone K is defined as $K := \{x \in \mathbb{R}^n : c^i x \geq 0,\ i = 1, \ldots, n\}$ and, for $\bar{x} \in K \cap \mathbb{Z}^n$, $Q(\bar{x}) := \text{conv}\{x \in K \cap \mathbb{Z}^n : x \succeq \bar{x}\}$.

The leading index $\hat{\imath}(\bar{x})$, for $\bar{x} \in K \setminus \{\mathbf{0}\}$, is the largest index i such that $c^i \bar{x} > 0$. Lemma 1 now reads as follows:

Lemma 4 *Fix $\bar{x} \in K \cap \mathbb{Z}^n$. The convex set $Q(\bar{x})$ is a full-dimensional polyhedron. Its vertices are precisely the following points $v^1, \ldots, v^{\hat{\imath}(\bar{x})}$: for $k = 1, \ldots, \hat{\imath}(\bar{x}) - 1$, v^k is the unique point satisfying*

$$\begin{aligned} c^i v^k &= c^i \bar{x}, && i = 1, \ldots, k-1 \\ c^k v^k &= c^k \bar{x} + 1 \\ c^i v^k &= 0, && i = k+1, \ldots, n, \end{aligned}$$

and $v^{\hat{\imath}(\bar{x})} = \bar{x}$. Furthermore, the recession cone of $Q(\bar{x})$ is K.

For $\bar{x} \in K$, $k \in \{1, \ldots, n\}$ and $i \in \{1, \ldots, k\}$, the definition of the d_i^k's is as follows:

$$d_i^k := \begin{cases} 1 & \text{if } i = k \\ c^k \bar{x} & \text{if } i = k-1, \\ c^k \bar{x} \prod_{j=i+1}^{k-1} (c^j \bar{x} + 1), & \text{if } i \leq k-2. \end{cases} \tag{2.3}$$

The *k-th lex-inequality* associated with $\bar{x}$ is the following:

$$\sum_{i=1}^{k} d_i^k c^i x \geq \sum_{i=1}^{k} d_i^k c^i \bar{x}. \tag{2.4}$$

Theorem 2 now reads as follows:

Proposition 5 *If $\bar{x} \in K \cap \mathbb{Z}^n$, then the lex-inequalities (2.4) for $k = 1, \ldots, n$ and the inequalities $c^i x \geq 0$ for $i = 1, \ldots, n$ provide a description of the polyhedron $Q(\bar{x})$.*

Neto (2012) describes a family of inequalities that, although presented in a different setting, can be seen to be valid for $Q(\bar{x})$ when the lattice basis $\{c^1, \ldots, c^n\}$ is the standard basis. However, those inequalities in general do not induce facets of $Q(\bar{x})$ and are therefore weaker than the lex-inequalities. In particular, the inequalities in Neto (2012) are derived under the assumption that a box containing the continuous set S is known, and their coefficients depend on the size of the box. In contrast, our inequalities only depend on the current fractional solution $\bar{x}$. As a consequence, we obtain inequalities with smaller dynamism (i.e., with smaller ratio between the largest and the smallest absolute value of the coefficients), which is a desirable property in practice.

In order to compare Neto's inequalities with ours, let $n = 2, \bar{x} = (1, 1)$, and consider the box $[0, 3] \times [0, 3]$. Neto's inequalities are in this case $x_1 \geq 1$ and $3x_1 + x_2 \geq 4$,[1]

[1] We note that Neto presents his inequalities in a different form, as he considers the integer points that are lexicographically *smaller* than $\bar{x}$.

while the lex-inequalities are $x_1 \geq 1$ and $x_1 + x_2 \geq 2$. Since the inequality $3x_1 + x_2 \geq 4$ is a proper conic combination of the two lex-inequalities, it cannot be facet inducing for $Q(\bar{x})$.

It should also be noted that in Neto (2012) the inequalities are described only for the case in which the continuous set S is a bounded polyhedron, although it is not difficult to extend them to the case of a compact set.

Furthermore, we remark that a linear-inequality description of the set $Q(\bar{x})$ can be inferred from a result proved by Gupte (2016, Theorem 2) in the context of super-increasing knapsack problems: One needs to apply a change of variables and observe that by removing the lower bounds appearing in Gupte (2016) the remaining facet inducing inequalities are unaffected.

3 The cutting plane algorithm

Let $\mathcal{S}$ be a family of compact (not necessarily connected or convex) subsets of $\mathbb{R}^n$ with the following property:

$$If\ S \in \mathcal{S}\ and\ H\ is\ a\ closed\ half space\ in\ \mathbb{R}^n,\ then\ S \cap H \in \mathcal{S}.$$

Linear optimization over $\mathcal{S}$ is the following problem: Given $S \in \mathcal{S}$ and $c \in \mathbb{Z}^n$, determine an optimal solution to the problem $\min\{cx : x \in S\}$ or certify that $S = \emptyset$. (Since S is compact, either $S = \emptyset$ or the minimum is well defined.)

Integer linear optimization over $\mathcal{S}$ is defined similarly, but the feasible region is $S \cap \mathbb{Z}^n$, the set of integer points in S.

We prove that an oracle for solving linear optimization over $\mathcal{S}$ suffices to design a finite cutting plane algorithm that solves integer linear optimization over $\mathcal{S}$.

We now make this statement more precise. Given a compact subset S of $\mathbb{R}^n$ and $c \in \mathbb{Z}^n$, let $\bar{x} \in S$ be an optimal solution of the program $\min\{cx : x \in S\}$. A *cutting plane* is a linear inequality that is valid for $S \cap \mathbb{Z}^n$ and is violated by $\bar{x}$. Note that a cutting plane exists if and only if $\bar{x} \notin \text{conv}(S \cap \mathbb{Z}^n)$. In particular, this is certainly the case if $\bar{x}$ is a non-integral extreme point of $\text{conv}(S)$.

A (pure) *cutting plane algorithm* for integer linear optimization over $\mathcal{S}$ is an iterative procedure of the following type:

- Let $S \in \mathcal{S}$ and $c \in \mathbb{Z}^n$ be given.
- If $S = \emptyset$, then $S \cap \mathbb{Z}^n = \emptyset$. Otherwise, find an optimal solution $\bar{x}$ of $\min\{cx : x \in S\}$.
- If $\bar{x} \in S \cap \mathbb{Z}^n$, stop: $\bar{x}$ is an optimal solution to $\min\{cx : x \in S \cap \mathbb{Z}^n\}$. Otherwise, detect a cutting plane and let H denote the corresponding half-space. Replace S with $S \cap H$ and iterate.

Assume without loss of generality that the objective function vector c is nonzero and has relatively prime entries. Then there exists a lattice basis $\{c^1, \ldots, c^n\}$ of $\mathbb{Z}^n$ such that $c^1 = c$. The optimal solution $\bar{x}$ of $\min\{cx : x \in S\}$ found by our algorithm

will be a *lexicographically minimum* or *lex-min* solution in S with respect to the lattice basis: i.e., $\bar{x} \prec x$ for every $x \in S \setminus \{\bar{x}\}$. The lex-min vector $\bar{x}$ in S satisfies the following conditions:

- $c^1\bar{x} = \min\{c^1 x : x \in S\}$;
- $c^2\bar{x} = \min\{c^2 x : x \in S,\ c^1 x = c^1\bar{x}\}$;
- $c^3\bar{x} = \min\{c^3 x : x \in S,\ c^1 x = c^1\bar{x},\ c^2 x = c^2\bar{x}\}$;
- …
- $c^n\bar{x} = \min\{c^n x : x \in S,\ c^1 x = c^1\bar{x}, \ldots,\ c^{n-1}x = c^{n-1}\bar{x}\}$.

Since S is nonempty and compact, the above minima are well-defined and can be computed by applying the oracle n times. Furthermore these conditions uniquely define $\bar{x}$. One verifies that $\bar{x}$ is an extreme point of $\mathrm{conv}(S)$.

Algorithm 1: Resolution of integer linear optimization over $\mathcal{S}$

Input: $S \in \mathcal{S}$ with $S \neq \emptyset$, $c \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$ with relatively prime entries, and a lattice basis $\{c^1, \ldots, c^n\}$ of $\mathbb{Z}^n$ with $c^1 = c$.

Output: an optimal integer solution $\bar{x}$ for the problem $\min\{cx : x \in S\}$ or a certificate that $S \cap \mathbb{Z}^n = \emptyset$.

1 Compute $\ell_i^* := \min\{c^i x : x \in S\}$ and $\ell_i := \lceil \ell_i^* \rceil$ for $1 \le i \le n$, and apply a translation so that $\ell_i = 0$ for $1 \le i \le n$. Let $K := \{x \in \mathbb{R}^n : c^i x \ge 0,\ i = 1, \ldots, n\}$ and replace S with $S \cap K$.

2 If $S = \emptyset$, stop: The given problem is infeasible.

3 Else, compute the lex-min solution $\bar{x}$ in S with respect to $\{c^1, \ldots, c^n\}$.

4 If $\bar{x} \in \mathbb{Z}^n$, return $\bar{x}$.

5 Else, let k be the smallest index such that $c^k\bar{x} \notin \mathbb{Z}$ and compute

$$d_i^k := \begin{cases} 1 & \text{if } i = k \\ \left\lceil c^k\bar{x} \right\rceil & \text{if } i = k-1, \\ \left\lceil c^k\bar{x} \right\rceil \prod_{j=i+1}^{k-1} (c^j\bar{x} + 1), & \text{if } i \le k-2. \end{cases}$$

Replace S with $S \cap H$, where H is the halfspace defined by the inequality (3.2)

$$\sum_{i=1}^{k} d_i^k c^i x \ge \sum_{i=1}^{k-1} d_i^k c^i \bar{x} + d_k^k \left\lceil c^k\bar{x} \right\rceil$$

and go to step 2.

Algorithm 1 describes the procedure in detail. Note that since S is compact, numbers $\ell_1^*, \ldots, \ell_n^*$ (as defined in Algorithm 1) exist and can be determined by querying the linear optimization oracle n times. Moreover, as $\{c^1, \ldots, c^n\}$ is a lattice basis of $\mathbb{Z}^n$, an index k as in step 5 always exists when $\bar{x} \notin \mathbb{Z}^n$.

Given $x \in K$, let $x^\uparrow$ be the lex-min vector in $K \cap \mathbb{Z}^n$ such that $x \preceq x^\uparrow$. Obviously $x = x^\uparrow$ if and only if $x \in \mathbb{Z}^n$. If $x \notin \mathbb{Z}^n$, let k be the smallest index such that $c^k x \notin \mathbb{Z}$. It is easy to see that $x^\uparrow$ is the unique point satisfying the following conditions:

$$c^i x^\uparrow = c^i x,\ i < k; \quad c^k x^\uparrow = \left\lceil c^k x \right\rceil; \quad c^i x^\uparrow = 0,\ i > k. \tag{3.1}$$

Definition 6 Let $\bar{x} \notin S \cap \mathbb{Z}^n$ and let k be the smallest index such that $c^k \bar{x} \notin \mathbb{Z}$.

The *k-th lex-cut* is the k-th lex-inequality associated with $\bar{x}^\uparrow$:

$$\sum_{i=1}^{k} d_i^k c^i x \geq \sum_{i=1}^{k-1} d_i^k c^i \bar{x} + d_k^k \left\lceil c^k \bar{x} \right\rceil \tag{3.2}$$

(This is the cut introduced at step 5 of Algorithm 1).

Proposition 7 *Inequality (3.2) defines a cutting plane. Algorithm 1 terminates after a finite number of iterations.*

Proof Since, after the preprocessing of step 1, $S \subseteq K$ and $\bar{x}$ is the lex-min point in S, $\bar{x} \preceq \bar{x}^\uparrow \prec x'$ for every $x' \in S \cap \mathbb{Z}^n \setminus \{\bar{x}^\uparrow\}$. Thus $S \cap \mathbb{Z}^n \subseteq Q(\bar{x}^\uparrow)$ and by Proposition 5 inequality (3.2) is valid for $S \cap \mathbb{Z}^n$. As $c^k \bar{x} \notin \mathbb{Z}$ and $d_k^k > 0$, the inequality is violated by $\bar{x}$. This shows that (3.2) defines a cutting plane.

As different iterations of the algorithm use cuts (3.2) associated with lexicographically increasing vectors in $S \cap \mathbb{Z}^n$, and S is bounded, the number of iterations of the algorithm is finite. □

We mention that Akshay Gupte (personal communication) has elaborated an algorithm to solve $\min\{cx : x \in S\}$, assuming that a box B containing S is given. His algorithm iteratively constructs the convex hull of a set of the form $\{x \in B \cap \mathbb{Z}^n : x \succeq \hat{x}\}$ for some $\hat{x} \in \mathbb{Z}^n$, which can be seen as a truncated version of $Q(\hat{x})$. However, while in Algorithm 1 at each iteration we use $\hat{x} = \bar{x}^\uparrow$, where $\bar{x}$ is the optimal solution of the continuous relaxation, in Gupte's algorithm $\hat{x}$ is obtained by "rounding" the point optimizing an objective function with superincreasing coefficients that is different from the original objective function. As a consequence, in Gupte's algorithm one can have $\hat{x} \prec \bar{x}^\uparrow$, which makes $Q(\hat{x})$ (or its truncated version) weaker.

4 Lexicographic enumeration and the number of iterations

Recall the notation $x^\uparrow$ introduced in (3.1). We extend that definition to sets as follows: Given $S \subseteq \mathbb{R}^n$, let $S^\uparrow := \{x^\uparrow : x \in S\}$. Since S is bounded, $S^\uparrow$ is a finite set, as, given $y \in S^\uparrow$ and $i \in \{1, \ldots, n\}$, $c^i y$ is an integer value satisfying $\min\{c^i x : x \in S\} \leq c^i y \leq \lceil \max\{c^i x : x \in S\} \rceil$.

Observation 8 *Given a nonempty set $S \in \mathcal{S}$, let $(\bar{x})$ be the sequence of points in S computed at step 3 of Algorithm 1. Then the sequence $(\bar{x}^\uparrow)$ is the lex-ordering of some distinct points in $S^\uparrow$.*

Proof If $\bar{x}$ is a point computed at step 3 of Algorithm 1, then clearly $\bar{x}^\uparrow \in S^\uparrow$, as $\bar{x} \in S$. Thus we only have to show that if $\bar{x}$ and $\tilde{x}$ are points computed at step 3 in two consecutive iterations (say iterations q and $q+1$), then $\bar{x}^\uparrow \prec \tilde{x}^\uparrow$. Assume not. Then $\bar{x}^\uparrow = \tilde{x}^\uparrow$ and therefore the cuts introduced at these two iterations would be exactly the same. But then the cut generated at iteration q would already cut off $\tilde{x}$, contradicting the fact that at iteration $q+1$ the point computed at step 3 is $\tilde{x}$. □

Corollary 9 *$|S^{\uparrow}|$ is an upper bound on the number of cuts produced by Algorithm 1.*

We next construct a convex body containing no integer points for which the bound $|S^{\uparrow}|$ on the number of cuts is exponential and tight.

Proposition 10 *For every $n \in \mathbb{N}$, there is a convex subset S of $[0,1]^n$ (described by a single convex constraint plus variable bounds) on which Algorithm 1 computes $|S^{\uparrow}| = 2^n - 1$ cuts.*

Proof We choose the standard basis $\{e^1, \ldots, e^n\}$ as lattice basis of $\mathbb{Z}^n$. Let $\mathbf{1}$ be the point in $\mathbb{R}^n$ with all entries equal to 1, and let $\|\cdot\|$ denote the Euclidean norm. Define

$$S := \left\{ x \in [0,1]^n : \left\| x - \frac{\mathbf{1}}{2} \right\|^2 \leq \frac{n}{4} - \frac{3}{16} \right\}.$$

Note that $S \cap \mathbb{Z}^n = \emptyset$ and $\ell_i = 0$ for $i = 1, \ldots, n$. Furthermore, for every $x \in \{0,1\}^n \setminus \{\mathbf{1}\}$, S contains the point $z(x)$ obtained from x by setting to $\frac{1}{4}$ the entry with largest index that is 0. As $S \subseteq [0,1]^n$, this shows that $S^{\uparrow} = \{0,1\}^n \setminus \{\mathbf{0}\}$, and thus $|S^{\uparrow}| = 2^n - 1$.

We now show that every point in $S^{\uparrow}$ is of the form $\bar{x}^{\uparrow}$ for some point $\bar{x}$ found in step 3. Let $\bar{x}$ be the point computed at some iteration of step 3 and assume $\bar{x}^{\uparrow} \neq \mathbf{1}$. By Theorem 2, the lex-cut associated with $\bar{x}^{\uparrow}$ is satisfied by all $x \in \{0,1\}^n$ such that $x \succeq \bar{x}^{\uparrow}$. As the lex-cut associated with $\bar{x}^{\uparrow}$ is an inequality with nonnegative coefficients, it is also satisfied by the point $z(\bar{x}^{\uparrow})$. This implies that, if we denote by $\tilde{x}$ the point computed in step 3 at the next iteration, $\tilde{x}^{\uparrow}$ is the lex-min point in $\{0,1\}^n$ that is lexicographically larger than $\bar{x}^{\uparrow}$. Thus every point in $S^{\uparrow}$ is of the form $\bar{x}^{\uparrow}$ for some point $\bar{x}$ found in step 3. Together with Observation 8, this shows that precisely $|S^{\uparrow}|$ cuts are needed to discover that S contains no integer points, which happens at step 2 immediately after the iteration in which $\bar{x}^{\uparrow} = \mathbf{1}$. □

Corollary 9 gives a guarantee on the maximum number of iterations of Algorithm 1 and Proposition 10 shows that there are instances for which the performance of this algorithm is of the same order as that of an enumerative algorithm.

In order to better understand how Algorithm 1 compares to simple enumeration, in the following we define Algorithm 2, which we think is the best candidate for an enumerative algorithm.

In Algorithm 2, for ease of description S is assumed to be contained in the nonnegative orthant. The variables $\alpha_1, \ldots, \alpha_n$ and i^* are used to impose restrictions on S, as now illustrated. At the first iteration, $\alpha_1 = \cdots = \alpha_n = 0$ and $i^* = 1$, and therefore $S^* = S$ (step 2). This implies that (assuming $S \neq \emptyset$) $\bar{x}$ is the lex-min point in S (step 7). Clearly, if $\bar{x}^{\uparrow} \in S^* = S$ then $\bar{x}^{\uparrow}$ is the lex-min point in $S \cap \mathbb{Z}^n$ and in this case the algorithm stops (step 8). Otherwise, we know that the lex-min point in $S \cap \mathbb{Z}^n$ is lexicographically larger than $\bar{x}^{\uparrow}$. Thus, we call the continuous optimization oracle to find the lex-min point in S that is lexicographically larger than $\bar{x}^{\uparrow}$: this is done by redefining S^* with the restrictions $c^i x = c^i \bar{x}^{\uparrow}$ for $i = 1, \ldots, n-1$ and $c^n x \geq c^n \bar{x}^{\uparrow}$ (step 9 and step 2 of the next iteration, with $i^* = n$). We proceed this way until S^* becomes empty. When this happens, in order to find the lex-min point in S that is

Algorithm 2: Resolution of integer linear optimization over $\mathcal{S}$ via lex-enumeration

Input: $S \in \mathcal{S}$, $c \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$ with relatively prime entries and a lattice basis $\{c^1, \ldots, c^n\}$ of $\mathbb{Z}^n$, with $c^1 = c$.
Output: an optimal integer solution $\bar{x}$ for the problem $\min\{cx : x \in S\}$ or a certificate that $S \cap \mathbb{Z}^n = \emptyset$.

1 Translate S so that $S \subseteq \{x \in \mathbb{R}^n : c^i x \geq 0,\ i = 1, \ldots, n\}$. Set $\alpha_1 := \cdots := \alpha_n := 0$ and $i^* := 1$.
2 Let $S^* := S \cap \{x \in \mathbb{R}^n : c^i x = \alpha_i,\ i < i^*;\ c^i x \geq \alpha_i,\ i \geq i^*\}$.
3 If $S^* = \emptyset$:
4 If $i^* = 1$, stop: $S \cap \mathbb{Z}^n = \emptyset$.
5 Else update $i^* := i^* - 1$, $\alpha_{i^*} := \alpha_{i^*} + 1$, $\alpha_i := 0$ for $i > i^*$, and go to step 2.
6 Else
7 Let $\bar{x}$ be the lex-min point in S^*.
8 If $\bar{x}^\uparrow \in S^*$, stop: $\bar{x}^\uparrow$ is the lex-min point in $S \cap \mathbb{Z}^n$.
9 Else update $i^* := n$, $\alpha_i := c^i \bar{x}^\uparrow$ for $i = 1, \ldots, n$, and go to step 2.

lexicographically larger than $\bar{x}^\uparrow$ we have to increase $c^{n-1}x$ and remove the restricion on $c^n x$ (step 5 with $i^* = n - 1$). If S^* is empty also when $c^{n-1}x$ is increased, we have to increase $c^{n-2}x$ and remove the restrictions on $c^{n-1}x$ and $c^n x$ (again step 5, this time with $i^* = n - 2$). This process continues until a feasible (and thus optimal) $\bar{x}^\uparrow$ is detected or $S^* = \emptyset$ with $i^* = 1$: in the latter case, $S \cap \mathbb{Z}^n = \emptyset$ (step 4).

The above discussion proves the correctness of Algorithm 2. In particular, the following property has been established.

Lemma 11 *In Algorithm 2, if $\bar{x}$ and $\tilde{x}$ denote the points computed at two consecutive executions of line 7, then $\tilde{x}$ is the lex-min point in S that is lexicographically larger than $\bar{x}^\uparrow$.*

To analyze the performance of Algorithm 2, we need the following definitions. Let C be the $n \times n$ matrix whose rows are $c^1, \ldots, c^n$ and let $S \in \mathcal{S}$ be given. For every $\bar{x} \in S^\uparrow$, let $V(\bar{x})$ be the set of the following n vectors $\alpha^1, \ldots, \alpha^n$: For $k = 1, \ldots, n-1$, α^k is defined as

$$\begin{aligned} \alpha^k_i &= c^i \bar{x}, && i = 1, \ldots, k-1 \\ \alpha^k_k &= c^k \bar{x} + 1 \\ \alpha^k_i &= 0, && i = k+1, \ldots, n, \end{aligned}$$

and $\alpha^n = C\bar{x}$. Notice that by Lemma 4, $V(\bar{x})$ contains all vectors of the form Cx where x is a vertex of $Q(\bar{x})$.

Let $V(S) = \bigcup_{\bar{x} \in S^\uparrow} V(\bar{x})$. Notice that, given $\bar{x}, \bar{y} \in S^\uparrow$, the set $V(\bar{x}) \cap V(\bar{y})$ may be nonempty.

Proposition 12 *Given a set $S \in \mathcal{S}$, let (α) be the sequence of vectors used to define the sequence of sets (S^*) in step 2 of Algorithm 2.*

- *If $S \cap \mathbb{Z}^n = \emptyset$, then (α) is the lex-ordering of all points in $V(S) \cup \{\mathbf{0}\}$ with respect to the standard basis.*
- *If $S \cap \mathbb{Z}^n \neq \emptyset$, the sequence is truncated to the lex-min vector α (with respect to the standard basis) such that $C^{-1}\alpha \in S \cap \mathbb{Z}^n = S \cap S^\uparrow$.*

Proof Clearly the sequence (α) starts with $\alpha = \mathbf{0}$ and is lexicographically increasing with respect to the standard basis.

Let $\alpha \neq 0$ be a vector used in step 2 at some iteration $q > 1$ and let $\bar{x}$ be the last point computed at line 7 before iteration q; say that $\bar{x}$ is computed at iteration $q' < q$. If $q' = q - 1$, then $\alpha = C\bar{x}^{\uparrow}$ and therefore $\alpha \in V(S)$. If $q' = q - t$ for some $t > 1$, then line 5 is executed $t - 1$ times between iterations q' and q. In this case, α is the vector defined by $\alpha_i = c^i\bar{x}^{\uparrow}$ for $i \leq n - t$, $\alpha_{n-t+1} = c^{n-t+1}\bar{x}^{\uparrow} + 1$, $\alpha_i = c^i\bar{x}^{\uparrow}$ for $i \geq n - t + 2$, and therefore $\alpha \in V(S)$.

We now show that every point in $V(S)$ is in the sequence (α). By Lemma 11, the sequence (α) contains all points of the form Cx for $x \in S^{\uparrow}$. Let now $\alpha \in V(S)$, where α is not of the form Cx for any $x \in S^{\uparrow}$. Then there exist $\hat{x} \in S^{\uparrow}$ and an index $k < n$ such that $\alpha_i = c^i\hat{x}$ for $i < k$, $\alpha_k x = c^k\hat{x} + 1$, and $\alpha_i = 0$ for $i > k$. Consider the last iteration of line 7 in which $\bar{x}^{\uparrow}$ satisfies $c^i\bar{x}^{\uparrow} = c^i\hat{x}$ for $i \leq k$ (this definition makes sense because, as shown above, $\hat{x} = \bar{x}^{\uparrow}$ at some iteration of line 7). The algorithm now sets $\alpha = C\bar{x}^{\uparrow}$ and executes line 5 k consecutive times. After this, we have $\alpha = Cx$. This shows that every point in $V(S)$ is in the sequence (α). □

We remark that in the definition of α at line 9, we could impose the stronger condition $\alpha_n := c^n\bar{x}^{\uparrow} + 1$. However, this would not change substantially the bounds on the number of iterations shown above. Moreover, when S is convex the number of iterations is precisely the same in both cases.

By Observation 8 and Proposition 12, the number of iterations of Algorithms 1 and 2 is upper-bounded by $|S^{\uparrow}|$ and $|V(S)| + 1$, respectively. Note that these bounds are sharp, in the sense that they can be attained, and that the latter bound is always larger than the former: indeed, by definition of $V(S)$, we have $|S^{\uparrow}| \leq |V(S)| \leq n|S^{\uparrow}|$. In particular, for the example in Proposition 10 we have $|V(S)| = 2^n + 2^{n-1} - 2$, thus in that case Algorithm 2 executes roughly 50% more iterations than Algorithm 1. It is not clear to us whether there are instances in which $|V(S)|$ is close to $n|S^{\uparrow}|$.

We also remark that comparing the two algorithms by counting the number of iterations may not be "fair", as the computational effort varies from iteration to iteration: For instance, the computation of a lex-min solution (line 3 of Algorithm 1 and line 7 of Algorithm 2) requires up to n oracle calls, while the iterations of Algorithm 2 in which S^* is empty only require a single oracle call. Nonetheless the results on the number of iterations at least indicate that, from the theoretical point of view, Algorithm 1 tends to be more efficient than Algorithm 2.

5 Comparison with Gomory and split cuts

Given a set S, a *Chvátal–Gomory inequality* for S is a linear inequality of the form $gx \geq \lceil \gamma \rceil$ for some $g \in \mathbb{Z}^n$ and $\gamma \in \mathbb{R}$ such that the inequality $gx \geq \gamma$ is valid for S. We call $gx \geq \lceil \gamma \rceil$ a *proper* Chvátal–Gomory inequality if $gx \geq \lceil \gamma \rceil$ is violated by at least one point in S.

Proposition 13 *Given $S \in \mathcal{S}$, every proper Chvátal–Gomory inequality for S is a lex-cut for some lattice basis $\{c^1, \ldots, c^n\}$ of $\mathbb{Z}^n$.*

Proof Let $gx \geq \lceil \gamma \rceil$ be a proper Chvátal–Gomory inequality for S. Without loss of generality, we assume that the entries of g are relatively prime integers. Let $\bar{x}$ be the lex-min solution found at the first iteration of Algorithm 1 with respect to some lattice basis $\{c^1, \ldots, c^n\}$, with $c^1 = g$. Since $gx \geq \lceil \gamma \rceil$ is a proper Chvátal–Gomory inequality for S, we have $\gamma \leq g\bar{x} < \lceil \gamma \rceil$. In particular, $g\bar{x} \notin \mathbb{Z}$. Then the corresponding lex-cut is (equivalent to) $gx \geq \lceil g\bar{x} \rceil = \lceil \gamma \rceil$. □

The converse of the above proposition is false; this will follow from a stronger result.

A linear inequality is a *split cut* for S if there exist $\pi \in \mathbb{Z}^n$ and $\pi_0 \in \mathbb{Z}$ such that the inequality is valid for both $\{x \in S : \pi x \leq \pi_0\}$ and $\{x \in S : \pi x \geq \pi_0 + 1\}$. It is known that every Chvátal–Gomory inequality is a split cut but not vice versa (see e.g. Conforti et al. 2014).

The next result shows that our family of cuts is not included in and does not include the family of split cuts. Combined with the previous proposition, this implies that our family of cuts strictly contains the Chvátal–Gomory inequalities.

Proposition 14 *There exist a bounded polyhedron S and a split cut for S that cannot be obtained as (and is not implied by) a lex-cut for any choice of the lattice basis $\{c^1, \ldots, c^n\}$. Conversely, there exist a bounded polyhedron S and a lex-cut that is not a split cut for S.*

Proof Let $S \subseteq \mathbb{R}^2$ be the triangle with vertices $(0, 0)$, $(1, 0)$ and $(1/2, -1)$. (See Fig. 1 to follow the proof.) The inequality $x_2 \geq 0$ is a split cut for S, as it is valid for both sets $\{x \in S : x_1 \leq 0\}$ and $\{x \in S : x_1 \geq 1\}$. Note that after the application of the cut, the continuous relaxation becomes the segment with endpoints $(0, 0)$ and $(1, 0)$, which is the convex hull of the integer points in S.

Assume that the cut $x_2 \geq 0$ can be obtained via an iteration of Algorithm 1 for some lattice basis $\{c^1, c^2\}$ and the corresponding bounds $\ell_1, \ell_2 \in \mathbb{Z}$. In the following, we will write $c^1 = (c^1_1, c^1_2)$ and $c^2 = (c^2_1, c^2_2)$.

Recall that in Algorithm 1 a translation is applied such that $\ell_i = 0$ for every i. However, in this proof it is convenient to work without applying the translation. It is easy to see that in this case the form of the lex-cut is still (3.2), but now the d^k_i are defined as follows:

$$d^k_i := \begin{cases} 1 & \text{if } i = k \\ \lceil c^k \bar{x} - \ell_k \rceil & \text{if } i = k-1, \\ \lceil c^k \bar{x} - \ell_k \rceil \prod_{j=i+1}^{k-1} (c^j \bar{x} + 1 - \ell_j) & \text{if } i \leq k-2. \end{cases}$$

Since the point $(1/2, -1)$ is the only fractional vertex of S, we must have $\bar{x} = (1/2, -1)$, otherwise no cut is generated. Suppose $k = 1$, i.e., $c^1 \bar{x} \notin \mathbb{Z}$ (see step 5 of the algorithm). Then the inequality generated by the algorithm is equivalent to $c^1 x \geq \lceil c^1 \bar{x} \rceil$. Since this inequality must be equivalent to $x_2 \geq 0$ and the entries of c^1 are relatively prime integers, we necessarily have $c^1 = (0, 1)$. But then $c^1 \bar{x} = -1$, a contradiction to the assumption $c^1 \bar{x} \notin \mathbb{Z}$.

Suppose now $k = 2$, i.e., $c^1\bar{x} \in \mathbb{Z}$ and $c^2\bar{x} \notin \mathbb{Z}$. Then the inequality given by the algorithm is

$$d_1^2\left(c^1x - c^1\bar{x}\right) + c^2x - \left\lceil c^2\bar{x}\right\rceil \geq 0. \tag{5.1}$$

We claim that $c_1^1 \neq 0$. If this is not the case, then $c_1^1 = 0$ and $c_1^2 \neq 0$ (as $\{c^1, c^2\}$ is a basis), and inequality (5.1) does not reduce to the desired cut $x_2 \geq 0$, as the coefficient of x_1 is $d_1^2c_1^1 + c_1^2 = c_1^2 \neq 0$. Thus $c_1^1 \neq 0$. This implies that either the point $(0, -1)$ or the point $(1, -1)$ satisfies the strict inequality $c^1x > c^1\bar{x}$. We assume that this holds for $\hat{x} := (0, -1)$ (the other case is similar). Note that $c^1\hat{x} \geq c^1\bar{x} + 1$, as $c^1\bar{x} \in \mathbb{Z}$ and $c^1, \hat{x} \in \mathbb{Z}^2$. Furthermore, the slope of the line defined by the equation $c^1x = c^1\bar{x}$ is positive.

If $c^2\hat{x} \geq \ell_2$, then $\hat{x}$ satisfies inequality (5.1), as $c^1\hat{x} - c^1\bar{x} \geq 1$ and $c^2\hat{x} - c^2\bar{x} \geq \ell_2 - c^2\bar{x} \geq -d_1^2$. Since the point $(1, 0)$ also satisfies (5.1) (as it is an integer point in S), the middle point of $\hat{x}$ and $(1, 0)$ satisfies (5.1). However, the middle point is $(1/2, -1/2)$, which is in S. This shows that in this case (5.1) is not equivalent to $x_2 \geq 0$.

Therefore we assume $c^2\hat{x} < \ell_2$. Since $c^2\bar{x} \geq \ell_2$, the line defined by the equation $c^2x = \ell_2$ intersects the line segments $[\hat{x}, \bar{x}]$ in a point distinct from $\hat{x}$. Then, because $(0, 0)$ satisfies the inequality $c^2x \geq \ell_2$ (as it is in S), the slope of the line defined by the equation $c^2x = \ell_2$ is negative. Furthermore, since $c^2, \hat{x} \in \mathbb{Z}^2$, we have $c^2\hat{x} \leq \lfloor \ell_2 \rfloor$, and thus the line defined by the equation $c^2x = \lfloor \ell_2 \rfloor$ intersects $[\hat{x}, \bar{x}]$ in some point x^*.

Now consider the system $c^1x = c^1\bar{x}$, $c^2x = \lfloor \ell_2 \rfloor$. Since the constraint matrix is unimodular (as $\{c^1, c^2\}$ is a lattice basis of $\mathbb{Z}^2$) and the right-hand sides are integer, the unique solution to this system is an integer point. However, the first equation defines a line with positive slope containing $\bar{x}$ and the second equation defines a line with negative slope containing x^*. From this we see that the intersection of the two lines is a point satisfying $0 < x_1 \leq 1/2$ and therefore cannot be an integer point, a contradiction. This shows that also in this case (5.1) is not equivalent to $x_2 \geq 0$. This concludes the proof that there is a split cut that cannot be obtained via an iteration of Algorithm 1.

For the converse, let $S \subseteq \mathbb{R}^2$ be the triangle with vertices $(0, 3/2)$, $(1/4, 0)$ and $(1, 0)$. If we take c^1, c^2 to be the vectors in the standard basis of $\mathbb{R}^2$, and $\ell_1 = \ell_2 = 0$, then Algorithm 1 yields the cut $2x_1 + x_2 \geq 2$. Note that every point in S other than $(1, 0)$ is cut off by this inequality. Thus, if the inequality $2x_1 + x_2 \geq 2$ is a split cut for S, then there exist $\pi \in \mathbb{Z}^2$ and $\pi_0 \in \mathbb{Z}$ such that S is contained in the "strip" $\{x \in \mathbb{R}^2 : \pi_0 \leq \pi x \leq \pi_0 + 1\}$. Since S contains a horizontal and a vertical segment of length $3/4$, this is possible only if the Euclidean distance between the lines $\{x \in \mathbb{R}^2 : \pi x = \pi_0\}$ and $\{x \in \mathbb{R}^2 : \pi x = \pi_0 + 1\}$ is at least $\frac{3}{4\sqrt{2}}$. Therefore $\|\pi\|^2 \leq \left(\frac{4\sqrt{2}}{3}\right)^2 = \frac{32}{9} < 4$. Since π is an integer vector, we deduce that $\pi_1, \pi_2 \in \{0, 1, -1\}$. It can be verified that if $|\pi_1| = |\pi_2| = 1$ then S is not contained in the strip. Therefore one entry of π is 0 and the other is 1 or -1. It can be checked that the only strip of this type containing S is $\{x \in \mathbb{R}^2 : 0 \leq x_1 \leq 1\}$. However, the inequality $2x_1 + x_2 \geq 2$ is not valid for all the points in $\{x \in S : x_1 \leq 0\} \cup \{x \in S : x_1 \geq 1\}$, as the point $(0, 3/2)$ is in this set but violates the inequality. □

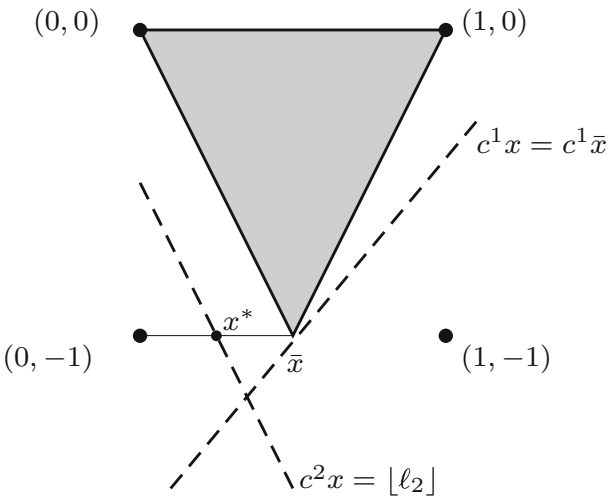

Fig. 1 Illustration of the first part of the proof of Proposition 14. The inequality $x_2 \geq 0$ is a split cut for the shadowed triangle, but is not of the type (3.2).

6 Concluding remarks

An obvious variant of Algorithm 1 is the following: Instead of being computed only once at the beginning of the procedure, the lower bounds ℓ_i can be updated at every iteration or whenever it seems convenient. It can be verified that the bounds of Observation 8 and Proposition 10 also hold for this variant of the algorithm: The proofs are the same.

In view of Observation 8 and Proposition 10, the cardinality of $S^{\uparrow}$ truncated to the lex-min point in $S^{\uparrow} \cap S$ plays a crucial role in the performance of Algorithm 1. This number is dependent on the choice of the lattice basis and its ordering. It is easy to see that different choices of the lattice basis (or different choices of the ordering of the elements of the same lattice basis) may result in a different number of iterations of the algorithm. However, this is not always the case: For instance, in the example in Proposition 10 Algorithm 1 would produce the same number of iterations regardless of the ordering of the standard basis.

A natural question is whether the approach described in this paper can be generalized to the mixed integer case, i.e., to problems of the form $\min\{cx + dy : (x, y) \in S \cap (\mathbb{Z}^n \times \mathbb{R}^p)\}$, where $S \subseteq \mathbb{R}^{n+p}$ is a compact set. However, it does not seem that our algorithm can be easily extended to deal with this case.

Acknowledgements The lex-cuts defined in a previous version of this manuscript were weaker. Giacomo Zambelli suggested to derive stronger lex-cuts via the characterization of the polyhedron $Q(\bar{x})$. Section 4 benefited from remarks due to Stefan Weltge. We thank both of them. We are also grateful to Akshay Gupte for his constructive comments and his pointers to the existing literature. We finally thank two anonimous referees for their constructive comments.

Funding Open access funding provided by Università degli Studi di Roma La Sapienza within the CRUI-CARE Agreement.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Andersen K, Jensen AN (2013) Intersection cuts for mixed integer conic quadratic sets. In: International conference on integer programming and combinatorial optimization, pp 37–48

Armstrong R, Charnes A, Phillips F (1979) Page cuts for integer interval linear programming. Discrete Appl Math 1(1–2):1–14

Balas E, Ceria S, Cornuéjols G (1993) A lift-and-project cutting plane algorithm for mixed 0–1 programs. Math Program 58(1–3):295–324

Bell DE (1973) A cutting plane algorithm for integer programs with an easy proof of convergence. Working paper 73-15, International Institute for Applied Systems Analysis, Laxenburg

Belotti P, Góez JC, Pólik I, Ralphs TK, Terlaky T (2013) On families of quadratic surfaces having fixed intersections with two hyperplanes. Discrete Appl Math 161(16–17):2778–2793

Belotti P, Kirches C, Leyffer S, Linderoth J, Luedtke J, Mahajan A (2013) Mixed-integer nonlinear optimization. Acta Numer 22:1–131

Belotti P, Góez JC, Pólik I, Ralphs TK, Terlaky T (2017) A complete characterization of disjunctive conic cuts for mixed integer second order cone optimization. Discrete Optim 24:3–31

Bowman VJ, Nemhauser GL (1970) A finiteness proof for modified Dantzig cuts in integer programming. Naval Res Logist Q 17(3):309–313

Burer S, Kılınç-Karzan F (2017) How to convexify the intersection of a second order cone and a nonconvex quadratic. Math Program 162:393–429

Burer S, Letchford A (2012) Non-convex mixed-integer nonlinear programming: a survey. Surv Oper Res Manag Sci 17:97–106

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

Conforti M, Cornuéjols G, Zambelli G (2014) Integer programming, vol 271. Springer, Berlin

Dadush D, Dey SS, Vielma JP (2014) On the Chvátal–Gomory closure of a compact convex set. Math Program 145:327–348

Frangioni A, Gentile C (2006) Perspective cuts for a class of convex 0–1 mixed integer programs. Math Program 106(2):225–236

Gomory R (1958) Outline of an algorithm for integer solutions to linear programs. Bull Am Math Soc 64(5):275–278

Gomory R (1963) An algorithm for integer solutions to linear programs. In: Wolfe P, Graves RL (eds) Recent advances in mathematical programming. McGraw-Hill, New York

Gupte A (2016) Convex hulls of superincreasing knapsacks and lexicographic orderings. Discrete Appl Math 201:150–163

He Q, Lee J (2017) Another pedagogy for pure-integer Gomory. RAIRO Oper Res 51(1):189–197

Lee J, Wiegele A (2017) Another pedagogy for mixed-integer Gomory. EURO J Comput Optim 5(4):455–466

Modaresi S, Kılınç MR, Vielma JP (2016) Intersection cuts for nonlinear integer programming: convexification techniques for structured sets. Math Program 155(1–2):575–611

Neto J (2012) A simple finite cutting plane algorithm for integer programs. Oper Res Lett 40(6):578–580

Orlin JB (1985) A finitely converging cutting plane technique. Oper Res Lett 4(1):1–3
Schrijver A (1986) Theory of linear and integer programming. Wiley-Interscience Series in Discrete Mathematics. Wiley, Chichester
Stubbs RA, Mehrotra S (1999) A branch-and-cut method for 0–1 mixed convex programming. Math Program 86(3):515–532