Algorithms for Standard-Form ILP Problems via Komlós' Discrepancy Setting

Dmitry Gribanov[1]
Laboratory LATNA, HSE University, Nizhny Novgorod, Russian Federation
T-Technologies, Moscow, Russian Federation

Tagir Khayaleyev[2]
Moscow Independent Research Institute of Artificial Intelligence (MIRAI), Russian Federation

Mikhail Cherniavskii
T-Technologies, Moscow, Russian Federation

Maxim Klimenko
T-Technologies, Moscow, Russian Federation

Dmitry Malyshev
Moscow Independent Research Institute of Artificial Intelligence (MIRAI), Russian Federation
Lobachevsky State University of Nizhny Novgorod, Russian Federation

Stanislav Moiseev
T-Technologies, Moscow, Russian Federation

Abstract

We study the standard-form ILP problem

$$c^\top x \to \max$$
$$\begin{cases} Ax = b, \\ x \in \mathbb{Z}^n_{\geq 0}, \end{cases}$$

where $A \in \mathbb{Z}^{k \times n}$ has full row rank. We obtain refined FPT algorithms parameterized by k and Δ, the maximum absolute value of a $k \times k$ minor of A. Our approach combines discrepancy-based dynamic programming with matrix discrepancy bounds in Komlós' setting. Let κ_k denote the maximum discrepancy over all matrices with k columns whose columns have Euclidean norm at most 1. Up to polynomial factors in the input size, the optimization problem can be solved in time $O(\kappa_k)^{2k}\Delta^2$, and the corresponding feasibility problem in time $O(\kappa_k)^k\Delta$. Using the best currently known bound $\kappa_k = \widetilde{O}(\log^{1/4} k)$, this yields running times $O(\log k)^{\frac{k}{2}(1+o(1))}\Delta^2$ and $O(\log k)^{\frac{k}{4}(1+o(1))}\Delta$, respectively. Under the Komlós conjecture, the dependence on k in both running times reduces to $2^{O(k)}$.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases Parameterized complexity, FPT algorithms, Integer linear programming, Komlós' conjecture, Discrepancy

Digital Object Identifier 10.4230/LIPIcs.ESA.2026.25

Related Version *Full Version*: `https://arxiv.org/abs/2604.09806`

Funding *Dmitry Gribanov*: Dmitry Gribanov is partially supported in the framework of the Basic Research Program at HSE University (HSE-BR-2025-080).
Dmitry Malyshev: Dmitry Malyshev is partially supported by the Ministry of Science and Higher Education of the Russian Federation (Goszadaniye), project No. FSMG-2024-0025.

[1] Corresponding author
[2] D. Gribanov and T. Khayaleyev are listed first, to reflect that they were the primary contributors to this work; the remaining authors are listed alphabetically.

Acknowledgements The authors would like to thank Nikita Kasyanov, Alexander Rogozin and Demyan Yarmoshik for their participation in discussions and helpful comments.
The initials LD from the definition of *LD-preconditioner* also serve as a quiet tribute to Liza D., who many times made difficult things feel worth doing [T. Khayaleyev].
Additionally, the authors would like to thank the anonymous reviewers for their insightful remarks and helpful suggestions.

1 Introduction

We consider the following ILP problem:

▶ **Definition 1.** *Let* $A \in \mathbb{Z}^{k \times n}$, $\operatorname{rank}(A) = k$*,* $c \in \mathbb{Z}^n$*,* $b \in \mathbb{Z}^k$*.* An ILP problem in standard form of codimension k *is the problem:*

$$c^\top x \to \max$$
$$\begin{cases} Ax = b \\ x \in \mathbb{Z}^n_{\geq 0}\,. \end{cases} \tag{ILP-SF}$$

We study the computational complexity of this problem with respect to k (codimension of the system $Ax = b$) and the absolute values of subdeterminants of A. These values are captured by the following definition:

▶ **Definition 2.** *For a matrix* $A \in \mathbb{Z}^{k \times n}$ *and* $j \in \{1, \ldots, k\}$*, by*

$$\Delta_j(A) = \max\left\{\left|\det(A_{\mathcal{I}\,\mathcal{J}})\right| : \mathcal{I} \subseteq \{1, \ldots, k\}\,, \mathcal{J} \subseteq \{1, \ldots, n\}\,, |\mathcal{I}| = |\mathcal{J}| = j\right\},$$

we denote the maximum absolute value of determinants of all the $j \times j$ *submatrices of* A*, with the convention that*

$$\Delta_j(A) = 0 \qquad \textit{for } j > \min\{k, n\}.$$

We also use the shorthand notation $\Delta(A) := \Delta_{\operatorname{rank}(A)}(A)$ *for the maximum absolute value of a full-rank minor of* A*. A matrix* A *with* $\Delta(A) \leq D$ *is called* D-modular*.*

We will often use the shorthand notation $\Delta := \Delta(A)$ and $\Delta_1 := \Delta_1(A) = \|A\|_{\max}$. To the best of our knowledge, the study of integer linear programming problems of bounded codimension began with the classical work of Papadimitriou [14], which shows that the (ILP-SF) problem can be solved in

$$O\left(n^{2k+2} \cdot \left(k \cdot \max\{\Delta_1, \|b\|_\infty\}\right)^{(k+1)(2k+1)}\right) \tag{1}$$

arithmetic operations. Following the analysis in [15, Section 18.6], this complexity bound can be rewritten in terms of subdeterminants of $A_{\text{ext}} = \begin{pmatrix} A\ b \end{pmatrix}$:

$$n^{O(k)} \cdot D^{O(k)}. \tag{2}$$

Here, D denotes the maximum absolute value of subdeterminants of the extended matrix A_{ext} of all orders up to k.

Four decades later, Eisenbrand & Weismantel [9] improved this bound by showing that (ILP-SF) and its feasibility variant can be solved in time

$$n \cdot O(k\Delta_1)^{2k} \cdot \|b\|_1^2\,, \tag{3}$$
$$n \cdot O(k\Delta_1)^{k} \cdot \|b\|_1\,, \tag{4}$$

respectively. Jansen & Rohwedder [12] later sharpened these bounds and removed dependence on $\|b\|$, obtaining

$$O(k)^k \cdot \Delta_1^{2k}/2^{\Omega(\sqrt{\log \Delta_1})} + T_{\mathrm{LP}}, \tag{5}$$

$$O(k)^{k/2} \cdot \Delta_1^k \cdot (\log \Delta_1)^2 + T_{\mathrm{LP}}, \tag{6}$$

for (ILP-SF) and its feasibility variant, respectively. Here T_{LP} denotes the computational complexity of solving the relaxed LP problem. It was also shown that these complexity bounds are conditionally optimal with respect to Δ_1.

It is natural to investigate the computational complexity of the (ILP-SF) problem with respect to subdeterminants of orders other than 1 in more detail, thereby extending Papadimitriou's bound in (2). In this direction, Gribanov et al. [11, Corollary 9] showed that the (ILP-SF) problem can be solved using

$$O(\log k)^{2k^2+o(k^2)} \cdot \Delta^2 \cdot \log^2(\Delta) + T_{\mathrm{LP}} \tag{7}$$

operations. However, the complexity dependence on k appears overly pessimistic.

Computational complexity bounds with significantly better dependence on k were obtained by Cherniavskii et al. [7] for the (ILP-SF) problem and its feasibility variant:

$$O(\log k)^{2k} \cdot \Delta^2/2^{\Omega(\sqrt{\log \Delta})} + 2^{O(k)} \cdot \mathrm{poly}(\text{input size}), \tag{8}$$

$$O(\log k)^{k} \cdot \Delta \cdot (\log \Delta)^2 + 2^{O(k)} \cdot \mathrm{poly}(\text{input size}). \tag{9}$$

The computational complexity bounds (8) and (9) are also applicable to problems in canonical form with $(n+k)$ inequalities (see ILP-CF and Section 1.2).

▶ Remark 3 (Conditional Lower Bounds). Note that applying Hadamard's inequality to these complexity bounds yields a computational cost that almost matches (5) and (6). The latter fact implies that these complexity bounds are conditionally optimal with respect to Δ (for the discussion, see Section "Conditional Lower Bounds" of [7]).

Additionally, we note that, due to Bock et al. [4], there are no polynomial-time algorithms for the ILP problems with $\Delta = \Omega(n^{\varepsilon})$, for any $\varepsilon > 0$, unless the ETH (the Exponential Time Hypothesis) is false.

The last fact is the reason why we need to use both parameters k and Δ. Indeed, due to [4], the complexity bound $\mathrm{poly}(\Delta, \text{input size})$ is unlikely to exist, while the bounds of the form $2^{\Omega(k)} \cdot \mathrm{poly}(\Delta, \text{input size})$ are provided in our work.

1.1 Our Contribution

Our main result is an algorithm for Δ-modular instances with improved dependence on k, at the cost of only a negligible deterioration in the dependence on Δ. More precisely, the dependence on k is governed by an upper bound for the discrepancy of matrices in the *Komlós* setting. We denote this upper bound by κ_n, and it can be formally defined in the following way. For an arbitrary matrix $A \in \mathbb{R}^{k \times n}$, the *discrepancy* of A is

$$\mathrm{disc}(A) = \min_{x \in \{-1,1\}^n} \|Ax\|_\infty .$$

Let

$$\mathcal{K}_n = \Big\{ A \in \mathbb{R}^{k\times n} \colon k \geq 1, \text{ all columns of } A \text{ have } \ell_2\text{-norm} \leq 1 \Big\},$$

then κ_n is defined by

$$\kappa_n = \max_{A \in \mathcal{K}_n} \mathrm{disc}(A). \tag{10}$$

Now, we can state our main algorithmic result.

▶ **Theorem 4.** *The* (ILP-SF) *problem can be solved in*

$$O(\kappa_k)^{2k} \cdot \Delta^2 \cdot \log \Delta + \operatorname{poly}(\text{input size})$$

arithmetic operations. Its feasibility variant can be solved in

$$O(\kappa_k)^k \cdot \Delta \cdot \operatorname{poly}(\text{input size})$$

bit operations[3].

The proof is given in Section 4.

The well-known Komlós conjecture states that κ_n is bounded by an absolute constant independent of n. A classical bound for κ_n is due to Banaszczyk [2]:

$$\kappa_n = O\big(\sqrt{\log n}\big).$$

The latter was improved by Bansal & Jiang [3], who resolved the Beck-Fiala conjecture for a very wide range of parameters:

$$\kappa_n \leq \sqrt[4]{\log n} \cdot \big(\log \log n\big)^{O(1)}.$$

Consequently, we obtain the following corollaries of Theorem 4.

▶ **Corollary 5.** *The* (ILP-SF) *problem can be solved in*

$$O(\log k)^{\frac{k}{2}(1+o(1))} \cdot \Delta^2 \cdot \log \Delta + \operatorname{poly}(\text{input size})$$

arithmetic operations. Its feasibility variant can be solved in

$$O(\log k)^{\frac{k}{4}(1+o(1))} \cdot \Delta \cdot \operatorname{poly}(\text{input size})$$

bit operations, respectively.

▶ **Corollary 6.** *Assuming that the Komlós conjecture is true, the* (ILP-SF) *problem can be solved in*

$$2^{O(k)} \cdot \Delta^2 \cdot \log \Delta + \operatorname{poly}(\text{input size})$$

arithmetic operations. Its feasibility variant can be solved in

$$2^{O(k)} \cdot \Delta \cdot \operatorname{poly}(\text{input size})$$

bit operations, respectively.

The first corollary (Corollary 5) improves the current dependence on k in complexity bounds from $O(\log k)^{2k}$ and $O(\log k)^k$ to $O(\log k)^{\frac{k}{2}(1+o(1))}$ and $O(\log k)^{\frac{k}{4}(1+o(1))}$, respectively. The second corollary (Corollary 6) states that the validity of the Komlós conjecture implies that both dependencies can be improved to $2^{O(k)}$.

▶ Remark 7 (Randomized vs. deterministic). Strictly speaking, the feasibility part of our algorithm is randomized because of the Boolean convolution subroutine from Section 2.4. More precisely, in Lemma 17 we use the randomized sparse polynomial multiplication algorithm of Giorgi et al. [10]. All remaining ingredients of the feasibility algorithm are deterministic.

This source of randomness is not essential. As explained in Remark 31 from Appendix, one may replace that subroutine by the deterministic sparse nonnegative convolution algorithm of Bringmann, Fischer, and Nakos [6, Theorem 1]. This yields a deterministic implementation of the Boolean convolution routine and therefore a deterministic version of the feasibility algorithm as well. The asymptotic form of the feasibility bounds in Theorem 4 and Corollaries 5 and 6 remains unchanged.

[3] Note that for the feasibility problem we estimate the bit complexity, because it is easier to do so for several reasons. The arithmetic complexity can be analyzed in further work.

1.2 ILP Problems in Canonical Form

It is also natural to consider the ILP problem in canonical form with n variables and $n + k$ constraints.

▶ **Definition 8.** *Let* $A \in \mathbb{Z}^{(n+k)\times n}$, $\operatorname{rank}(A) = n$*,* $c \in \mathbb{Z}^n$*,* $b \in \mathbb{Z}^{n+k}$*.* The ILP problem in canonical form with $n + k$ constraints *is defined as follows:*

$$c^\top x \to \max$$
$$\begin{cases} Ax \leq b \\ x \in \mathbb{Z}^n\,. \end{cases} \tag{ILP-CF}$$

Despite the fact that the (ILP-SF) and (ILP-CF) problems are mutually transformable, all known trivial transformations alter at least one of the key parameters k, Δ or the dimension of the corresponding polyhedra. The existence of a parameter-preserving transformation is a nontrivial question that was resolved positively by Gribanov et al. [11, Lemmas 4 and 5].

As already noted, the complexity bounds (8) and (9) are also applicable to the (ILP-CF) problem and its feasibility version. This relies precisely on the mentioned transformation between the canonical-form problem (ILP-CF) and the standard-form problem (ILP-SF).

The approach described in this manuscript can also be applied to the (ILP-CF) problem, which yields a computational complexity bound with a better dependence on k and a negligible deterioration in Δ.

▶ **Theorem 9.** *The* (ILP-CF) *problem can be solved in*

$$O(\kappa_k)^{2k} \cdot \Delta^2 \cdot \log \Delta + \operatorname{poly}(\text{input size}) \quad \textit{arithmetic operations.}$$

We do not present an accelerated algorithm for the feasibility version of (ILP-CF), since this would require a more complicated generalized Boolean convolution that is not needed for (ILP-SF). Since (ILP-CF) plays only a secondary role in this paper, we also omit the proof of Theorem 9. That proof can be obtained by a minor adaptation of the dynamic program used in Theorem 4, together with the approach of [7] and the reduction of [11, Lemmas 4 and 5]. We plan to resolve these convolution issues and develop algorithms for the (ILP-CF) problem and its feasibility variant in an extended version of our work.

1.3 Complexity Model Assumptions

Some of the algorithms considered in this paper are analyzed in the Word-RAM model. In other words, we assume that additions, subtractions, multiplications, and divisions of rational numbers of the prescribed size (the word size) can be performed in $O(1)$ time. We take the word size to be bounded by a fixed polynomial in $\lceil \log n \rceil + k + \lceil \log \alpha \rceil$, where α is the maximum absolute value of elements of A, b, and c in the problem formulations.

However, sometimes we will explicitly describe the computational complexity of algorithms in terms of the number of bit operations. Depending on which complexity measure we use in our work, we will state this explicitly. That is, we will use the terms “arithmetic operations” (sometimes shortened to “operations”) and “bit operations”.

2 Preliminaries

For $A \in \mathbb{R}^{k \times n}$, we denote by:

- $A_{i\,j}$ the (i, j)-th entry of A;
- $A_{i\,*}$ its i-th row vector;
- $A_{*\,j}$ its j-th column vector;
- $A_{\mathcal{I}\,\mathcal{J}}$ the submatrix consisting of rows and columns of A, indexed by $\mathcal{I}$ and $\mathcal{J}$, respectively;
- Replacing $\mathcal{I}$ or $\mathcal{J}$ with $*$ selects all rows or columns, respectively;
- When unambiguous, we abbreviate $A_{\mathcal{I}\,*}$ as $A_{\mathcal{I}}$ and $A_{*\,\mathcal{J}}$ as $A_{\mathcal{J}}$.

2.1 Hermite Normal Form

Let $A \in \mathbb{Q}^{n \times n}$ be a rational matrix of rank n. It is well known (see, for example, [15, 17, 16]) that there exists a unimodular matrix $U \in \mathbb{Z}^{n \times n}$, such that $A = UH$, where $H \in \mathbb{Q}^{n \times n}$ is upper triangular and satisfies $0 \le H_{ij} < H_{ii}$ for every $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, i-1\}$. The matrix H is called the *Hermite Normal Form* (or, shortly, HNF) of the matrix A. Near-optimal polynomial-time algorithms for constructing the HNF of A can be found in [17, 16].

2.2 Preliminaries from Discrepancy Theory

As noted in the Introduction, we employ tools from discrepancy theory. Below we recall the results and definitions we need.

▶ **Definition 10.** *For a matrix $A \in \mathbb{R}^{k \times n}$,* its discrepancy *and its* hereditary discrepancy *are defined by*

$$\begin{aligned} \operatorname{disc}(A) &= \min_{z \in \{-1,1\}^n} \|Az\|_\infty\,, \\ \operatorname{herdisc}(A) &= \max_{\mathcal{I} \subset \{1,\dots,n\}} \operatorname{disc}(A_{*\,\mathcal{J}}). \end{aligned}$$

One of the most important settings in discrepancy theory is the Komlós setting, which considers matrices $A \in \mathbb{R}^{k \times n}$ whose columns have the Euclidean norm at most one. Recall the corresponding definitions from Introduction:

$$\begin{aligned} \mathcal{K}_n &= \left\{ A \in \mathbb{R}^{k \times n} \colon k \ge 1, \|A_{*i}\|_2 \le 1, \forall i \in \{1, \dots, n\} \right\}, \\ \kappa_n &= \max_{A \in \mathcal{K}_n} \operatorname{disc}(A). \end{aligned}$$

The long-standing Komlós conjecture states that $\kappa_n = O(1)$. Considerable work has gone into proving or disproving this conjecture, and the current best bound is due to Bansal & Jiang [3]:

$$\kappa_n = \sqrt[4]{\log n} \cdot \left(\log\log n\right)^{O(1)}. \tag{11}$$

We must also recall an important property concerning the discrepancy of matrices $A \in \mathbb{R}^{k \times n}$ with $k \le n$.

▶ **Lemma 11** (Alon & Spencer [1, Corollary 13.3.3])**.** *Suppose that* $\operatorname{disc}(A_{*\,\mathcal{J}}) \le H$*, for every subset* $\mathcal{J} \subset \{1, \dots, n\}$ *with* $|\mathcal{J}| \le k$*. Then,* $\operatorname{disc}(A) \le 2H$*.*

Originally, this statement was proved only for hypergraph discrepancy. However, it is straightforward to see from the original proof that it extends to matrices as well. Combining Lemma 11 with (11), we obtain

$$\operatorname{herdisc}(A) \le 2\kappa_k = \sqrt[4]{\log k} \cdot \bigl(\log \log k\bigr)^{O(1)}. \tag{12}$$

The following key lemma, due to K. Jansen & L. Rohwedder [12], connects results from discrepancy theory with ILP theory.

▶ **Lemma 12** (K. Jansen & L. Rohwedder [12])**.** *Let $A \in \mathbb{R}^{k \times n}$ with $\operatorname{rank}(A) = k$ and $x \in \mathbb{Z}^n_{\ge 0}$. Then, there exists a vector $z \in \mathbb{Z}^n_{\ge 0}$ with*
1. *$z \le x$;*
2. *$\bigl\|A(z - x/2)\bigr\|_\infty \le \operatorname{herdisc}(A)$.*

Assuming additionally that $\|x\|_1 > 1$, there exists a vector $z \in \mathbb{Z}^n_{\ge 0}$ with
1. *$z \le x$;*
2. *$\frac{1}{6} \cdot \|x\|_1 \le \|z\|_1 \le \frac{5}{6} \cdot \|x\|_1$;*
3. *$\bigl\|A(z - x/2)\bigr\|_\infty \le 2 \cdot \operatorname{herdisc}(A)$.*

2.3 Enumeration of Integer Points in Rational Parallelepipeds

The dynamic programming algorithm we will use requires enumerating integer points of a parallelepiped as a subroutine.

▶ **Lemma 13.** *For a given upper triangular matrix $H \in \mathbb{Q}^{n \times n}$ with strictly positive diagonal elements and $r \in \mathbb{Q}^n$, denote $\mathcal{P} = r + H \cdot [0,1)^n$. The following propositions hold:*
1. $$\prod_{i=1}^n \lfloor H_{ii} \rfloor \le |\mathcal{P} \cap \mathbb{Z}^n| \le \prod_{i=1}^n \lceil H_{ii} \rceil;$$
2. *The elements of $\mathcal{P} \cap \mathbb{Z}^n$ can be enumerated using $O(n)^2 \cdot \prod_{i=1}^n \lceil H_{ii} \rceil$ operations.*

Due to its length, the proof has been moved to Appendix A.1.

▶ **Corollary 14.** *For a given upper triangular matrix $H \in \mathbb{Q}^{n \times n}$ and $r \in \mathbb{Q}^n$, denote $\mathcal{P} = r + H \cdot [0,1)^n$ and $\Delta = |\det H|$. Assume additionally that $H_{ii} = \Omega(1)$ for each i. Then, the following propositions hold:*
1. $$|\mathcal{P} \cap \mathbb{Z}^n| = 2^{O(n)} \cdot \Delta;$$
2. *The elements of $\mathcal{P} \cap \mathbb{Z}^n$ can be enumerated using $2^{O(n)} \cdot \Delta$ operations.*

2.4 Sparse Boolean Convolution via Sparse Polynomial Multiplication

In this subsection, we study the sparse generalized Boolean convolution problem, which will be used to speed up our dynamic programming algorithm for solving feasibility problems.

▶ **Definition 15.** *Let α and β be given functions from $\mathbb{Z}^k$ to $\{0,1\}$ with finite support. We assume that the functions are represented by lists of elements of their supports. The* Boolean convolution *of the functions α and β is the function $\gamma \colon \mathbb{Z}^k \to \{0,1\}$, defined by*

$$\gamma_x = \bigvee_{\substack{y \in \operatorname{supp}(\alpha) \\ x - y \in \operatorname{supp}(\beta)}} \alpha_y \wedge \beta_{x-y}.$$

Note that $\operatorname{supp}(\gamma) = \Bigl\{x \in \mathbb{Z}^k \colon \exists y \in \operatorname{supp}(\alpha)$ *such that* $(x - y) \in \operatorname{supp}(\beta)\Bigr\}$.

▶ **Problem 1** (Sparse generalized Boolean convolution). Given two finite-support functions $\alpha, \beta \colon \mathbb{Z}^k \to \{0,1\}$, each represented by the list of points where it is equal to 1, compute their Boolean convolution γ from Definition 15 in the same sparse representation.

To construct a fast algorithm for Problem 1, we will use the recent result on sparse polynomial multiplication by Giorgi et al. [10]. For a multivariate polynomial $f \in \mathbb{Z}[x_1, \dots, x_n]$, let $\|f\|_\infty$ be the maximum coefficient of f, $\#f$ be the number of non-zero coefficients of f, and $\deg_{x_i} f$ be the maximum degree of the variable x_i in monomials of f.

▶ **Theorem 16** (Giorgi et al. [10, Corollary 4.8]). *There exists an algorithm that takes as inputs $f, g \in \mathbb{Z}[x_1, \dots, x_n]$ and $0 < \varepsilon < 1$, and computes fg with probability at least $1 - \varepsilon$ using*

$$O\big(T \cdot (n \log d + \log C) \cdot \operatorname{polylog}(1/\varepsilon)\big) \quad \textit{bit operations,}$$

where $T = \max\big\{\#(fg), \#f, \#g\big\}$, $d = \max_i \deg_{x_i}(fg)$, and $C = \max\{\|f\|_\infty, \|g\|_\infty\}$.

The following lemma provides a simple reduction of Problem 1 to sparse multivariate polynomial multiplication. Due to its technical nature and space limitations, we have moved the proof to Appendix A.3.

▶ **Lemma 17.** *Problem 15 can be solved in*

$$O\big(T \cdot (k \log L)\big)$$

bit operations with high probability, where $T = \max\Big\{\big|\operatorname{supp}(\alpha)\big|, \big|\operatorname{supp}(\beta)\big|, \big|\operatorname{supp}(\gamma)\big|\Big\}$ and L is the maximum ℓ_∞-norm of elements in $\operatorname{supp}(\alpha)$ and $\operatorname{supp}(\beta)$.

2.5 The Löwner ellipsoid for a set of points

Let $\mathcal{A} \subseteq \mathbb{R}^k$ be a set of n points in general position. For simplicity, we assume that $\mathcal{A}$ is $\mathbf{0}$-symmetric. It is a known fact that there exists a unique minimum-volume ellipsoid $\mathcal{E}$, called the Löwner-John or Löwner ellipsoid, that contains $\mathcal{A}$. Since $\mathcal{A}$ is in general position and $\mathbf{0}$-symmetric, $\mathcal{E}$ is k-dimensional and $\mathbf{0}$-symmetric.

The Löwner ellipsoid $\mathcal{E}$ can be represented by the following pair of primal and dual convex optimization problems (see, for example, Nikolov [13, Section 2.3]). The primal program is

$$\begin{gathered} -\ln\det W \to \min \\ \begin{cases} a^\top W a \leq 1, \quad a \in \mathcal{A}, \\ W \succ \mathbf{0}\,. \end{cases} \end{gathered} \tag{Primal-LJ-Ell}$$

The dual program is

$$\begin{gathered} \ln\det\left(\sum_{i=1}^{n} c_i \cdot a_i a_i^\top\right) \to \max \\ \begin{cases} \mathbf{1}^\top c = k \\ c \in \mathbb{R}^n_{\geq 0}\,. \end{cases} \end{gathered} \tag{Dual-LJ-Ell}$$

The primal problem (Primal-LJ-Ell) is a convex minimization problem over the open domain $\{W \colon W \succ \mathbf{0}\}$ with affine constraints. Hence, due to Slater's condition [5, Section 5.2.3], there is no duality gap between (Primal-LJ-Ell) and (Dual-LJ-Ell).

The following lemma shows that the volume of $\mathcal{E}$ can be bounded in terms of the maximum volume of a parallelepiped spanned by vectors in $\mathcal{A}$. This fact explains the main reason for the existence of *LD-preconditioners* described in Section 3.

▶ **Lemma 18.** *Let A be the matrix whose columns are formed by elements of $\mathcal{A}$. Let $\mathcal{E} = \left\{x \in \mathbb{R}^k \colon x^\top W x \leq 1\right\}$ be the Löwner ellipsoid, i.e., the minimum-volume ellipsoid containing $\mathcal{A}$. Then, $\frac{1}{\sqrt{\det W}} \leq e^{k/2} \cdot \Delta(A)$.*

For the proof of Lemma 18, we will use the following technical lemma, whose proof is deferred to Appendix A.2.

▶ **Lemma 19.** *For $\alpha > 0$ and integers $n \geq k > 0$, consider the following optimization problem:*

$$\sum_{\mathcal{J} \in \binom{\{1,\,\dots,\,n\}}{k}} \prod_{j \in \mathcal{J}} c_j \to \max$$

$$\begin{cases} c_1 + \dots + c_n = \alpha \\ c \in \mathbb{R}^n_{\geq 0} \,. \end{cases}$$

Then, the point $c = \frac{\alpha}{n} \cdot \mathbf{1}$ is an optimal solution of this problem.

Proof Of Lemma 18. Consider the dual program (Dual-LJ-Ell). Denoting $C = \operatorname{diag}(c)$, we have

$$1/\det W = \det\left(\sum_{i=1}^{n} c_i \cdot a_i a_i^\top\right) = \det\bigl(A C A^\top\bigr) =$$

$$\sum_{\mathcal{J} \in \binom{\{1,\,\dots,\,n\}}{k}} \det\bigl((A\sqrt{C})_{*\mathcal{J}}\bigr)^2 = \sum_{\mathcal{J} \in \binom{\{1,\,\dots,\,n\}}{k}} \det\bigl(A_{*\mathcal{J}}\bigr)^2 \cdot \det C_{\mathcal{J}} \leq$$

$$\Delta(A)^2 \cdot \sum_{\mathcal{J} \in \binom{\{1,\,\dots,\,n\}}{k}} \prod_{j \in \mathcal{J}} c_j \overset{\text{by Lemma 19}}{\leq}$$

$$\Delta(A)^2 \cdot \binom{n}{k} \cdot \left(\frac{k}{n}\right)^k \leq e^k \cdot \Delta(A)^2.$$

◀

The following technical lemma provides an algorithm for finding an approximate Löwner ellipsoid in a format convenient for our purposes. We have omitted the proof here owing to the page limit; the full proof is available in the arXiv version of the paper.

▶ **Lemma 20.** *Let $\mathcal{A} \subseteq \mathbb{Q}^k$ be a given finite $\mathbf{0}$-symmetric set of points in general position, and $\varepsilon \in \mathbb{Q}_{>0}$ be a given rational number. Let $W \in \mathbb{R}^{k \times k}$ be the symmetric positive-definite matrix such that*

$$\mathcal{E} = \{x \in \mathbb{R}^k : x^\top W x \leq 1\}$$

is the Löwner ellipsoid of $\mathcal{A}$. Then, there exists a polynomial-time algorithm that computes a matrix $C \in \mathbb{Q}^{k \times k}$ such that

1. *C is upper triangular with positive diagonal,*
2. *for each $a \in \mathcal{A}$, $a^\top C^\top C a \leq 1$,*
3. $\det C \geq e^{-2\varepsilon} \cdot \det W^{1/2}$.

3 Computation of the Preconditioning Matrix with a Small Determinant

The main idea of our approach, which is similar in spirit to the techniques used in the discrepancy minimization paper by Dadush et al. [8], is to precondition a given matrix $A \in \mathbb{R}^{k \times n}$ by choosing a matrix $B \in \mathbb{R}^{k \times k}$ with small determinant such that every column of $B^{-1}A$ has Euclidean norm bounded by an absolute constant. After rescaling, the matrix $B^{-1}A$ belongs to Komlós' setting, which yields $\operatorname{herdisc}\bigl(B^{-1}A\bigr) = O(\kappa_k)$. A natural way to construct such a matrix B is via the Löwner ellipsoid. We refer to such matrices as *LD-preconditioners*, where the letters LD abbreviate Löwner and discrepancy.

▶ **Definition 21** (LD-preconditioner). *Let $A = (a_1, \ldots, a_n) \in \mathbb{R}^{k \times n}$ be a matrix of rank k. A nondegenerate matrix $B \in \mathbb{R}^{k \times k}$ is called an* LD-preconditioner of A *if the following conditions are satisfied:*

1. $|\det B| \leq 2^{O(k)} \cdot \Delta(A)$;
2. $\bigl\|B^{-1} a_i\bigr\|_2 = O(1)$ *for every* $i \in \{1, \ldots, n\}$.

▶ **Proposition 22.** *Let $A \in \mathbb{R}^{k \times n}$ be a matrix of rank k. If B is an LD-preconditioner of A, then* $\operatorname{herdisc}\bigl(B^{-1}A\bigr) = O(\kappa_k)$.

Proof. By the definition, there exists an absolute constant $c > 0$ such that every column of $B^{-1}A$ has Euclidean norm at most c. Hence, the matrix $c^{-1}(B^{-1}A)$ belongs to the Komlós setting. Applying Equation (12), we obtain $\operatorname{herdisc}\bigl(B^{-1}A\bigr) \leq 2c \cdot \kappa_k = O(\kappa_k)$. ◀

The next proposition proves the existence of LD-preconditioners.

▶ **Proposition 23.** *Let $A \in \mathbb{R}^{k \times n}$ be a matrix of rank k. Then there exists an LD-preconditioner B of A such that B is upper triangular with positive diagonal. Moreover, for a given rational matrix A, such a matrix B can be computed in polynomial time.*

Proof. Denote the columns of A by $(a_1, a_2, \ldots a_n)$. Let $W \in \mathbb{R}^{k \times k}$ be the symmetric positive-definite matrix such that

$$\mathcal{E} = \{x \in \mathbb{R}^k : x^\top W x \leq 1\}$$

is the Löwner ellipsoid of the set $\{\pm a_i\}$. Let B be the Cholesky factor of W^{-1}, so that $B^\top B = W^{-1}$. Note that B is upper triangular with positive diagonal. By Lemma 18,

$$|\det B| = \det W^{-1/2} \leq e^{k/2} \cdot \Delta(A). \tag{13}$$

Additionally, if y is the i-th column of $B^{-1}A$, then

$$\|y\|_2^2 = a_i^\top B^{-\top} B^{-1} a_i \leq 1. \tag{14}$$

Hence, B is an LD-preconditioner of A.

Note that B can be chosen as any upper triangular matrix with positive diagonal that satisfies (13) and (14). Thus, for rational A, rather than taking B to be the exact Cholesky factor, we choose C using Lemma 20 with $\varepsilon = 1$, and set $B = C^{-1}$; the same verification applies. ◀

For our purposes, we need a stronger version of LD-preconditioner.

▶ **Definition 24** (Normalized LD-preconditioner). *An LD-preconditioner B of A is called* normalized *if there exists a unimodular matrix $U \in \mathbb{Z}^{k \times k}$ such that UB is upper triangular with diagonal elements bounded below by $\Omega(1)$.*

Or equivalently[4,5]*, an LD-preconditioner B of A is called* normalized *if each diagonal element of the HNF of B is bounded by $\Omega(1)$ from below.*

▶ **Proposition 25.** *Let $A \in \mathbb{R}^{k \times n}$ be a matrix of rank k, and let $B \in \mathbb{Q}^{k \times k}$ be a normalized LD-preconditioner of A. Then, for every $r \in \mathbb{Q}^k$, the points of*

$$\big(r + B \cdot [0,1)^k\big) \cap \mathbb{Z}^k$$

can be enumerated using $2^{O(k)} \cdot \Delta(A)$ operations.

Proof. By definition of a normalized LD-preconditioner, there exists a unimodular matrix $U \in \mathbb{Z}^{k \times k}$ such that $H = UB$ is upper triangular with diagonal elements bounded by $\Omega(1)$ from below. Denoting $r' = Ur$, the map $x \to Ux$ is a bijective map from $\big(r + B \cdot [0,1)^k\big) \cap \mathbb{Z}^k$ to $\big(r' + H \cdot [0,1)^k\big) \cap \mathbb{Z}^k$. By Corollary 14, the points of $\big(r' + H \cdot [0,1)^k\big) \cap \mathbb{Z}^k$ can be enumerated using $2^{O(k)} \cdot |\det H|$ operations. Since U is unimodular and B is an LD-preconditioner of A, we have $|\det H| \le 2^{O(k)} \cdot \Delta(A)$, which finishes the proof. ◀

▶ Remark 26. We will use Proposition 25 later to enumerate the state space of the dynamic program from Section 4.1.

The next lemma shows that every matrix containing the columns of the $k \times k$ identity matrix admits a normalized LD-preconditioner.

▶ **Lemma 27.** *Let $A \in \mathbb{R}^{k \times n}$ be a matrix of rank k which contains a $k \times k$ identity submatrix. Then there exists a normalized LD-preconditioner B of A. Moreover, for rational A, such a matrix B can be computed in polynomial time.*

Proof. Let B be an LD-preconditioner of A constructed using Proposition 23. It is upper triangular with positive diagonal. Thus B is normalized by taking $U = I_k$ in Definition 24. Since A contains the identity matrix, we have

$$0 < \frac{1}{B_{ii}} = \left(B^{-1}\right)_{ii} \le \left\|B^{-1} e_i\right\|_2 = O(1),$$

which completes the proof. ◀

The next theorem constructs, in polynomial time, a rational normalized LD-preconditioner for an arbitrary integer matrix of rank k.

▶ **Theorem 28.** *Let $A \in \mathbb{Z}^{k \times n}$ be a matrix of rank k. Then there exists a normalized LD-preconditioner B of A. Moreover, such a matrix B can be computed in polynomial time.*

Proof. Denote the columns of A by $(a_1, \dots, a_n)$ and let $\mathcal{B}$ be an arbitrary columns base of A. Denote $\delta = |\det A_{\mathcal{B}}|$ and $\Delta = \Delta(A)$. Then

$$\Delta(A_{\mathcal{B}}^{-1} A) = \frac{\Delta}{\delta},$$

and $A_{\mathcal{B}}^{-1} A$ contains an identity submatrix.

---

[4] The second version of this definition is equivalent to the first one for rational matrices. However, since the HNF is not defined for reals, the first generalizes the second.

[5] The equivalence follows from the fact that every upper triangular matrix with strictly positive diagonal can be reduced to an HNF with the same diagonal using only unimodular row operations.

Applying Lemma 27 to $A_{\mathcal{B}}^{-1}A$, we obtain in polynomial time a rational normalized LD-preconditioner B of $A_{\mathcal{B}}^{-1}A$. Hence

$$|\det B| \leq 2^{O(k)} \cdot \Delta(A_{\mathcal{B}}^{-1}A) = 2^{O(k)}\frac{\Delta}{\delta}, \quad \text{and}$$
$$\forall i \in \{1, \dots, n\}, \quad \left\|B^{-1}(A_{\mathcal{B}}^{-1}a_i)\right\|_2 \leq O(1).$$

Let $B' = A_{\mathcal{B}}B$. Then, for every $i \in \{1, \dots, n\}$,

$$\left\|(B')^{-1}a_i\right\|_2 = \left\|B^{-1}(A_{\mathcal{B}}^{-1}a_i)\right\|_2 \leq O(1), \quad \text{and}$$
$$\left|\det B'\right| = |\det A_{\mathcal{B}}| \cdot |\det B| \leq 2^{O(k)}\Delta.$$

Thus B' is an LD-preconditioner of A. Since B is normalized, we can put

$$B = UH,$$

where $U \in \mathbb{Z}^{k \times k}$ is unimodular, $H \in \mathbb{Q}^{k \times k}$ is upper triangular, and the diagonal elements of H are bounded by $\Omega(1)$ from below. Then

$$B' = A_{\mathcal{B}}B = A_{\mathcal{B}}UH \underset{\substack{A_{\mathcal{B}}U = U'H_1 \\ \text{HNF decomposition}}}{=} U'H_1H.$$

The matrix H_1H is upper triangular. Since the diagonal of H_1 consists of positive integers and the diagonal of H is bounded by $\Omega(1)$ from below, the diagonal of H_1H is also bounded by $\Omega(1)$ from below. Therefore, B' is the required normalized LD-preconditioner of A. ◀

4 Proof of Theorem 4

Recall the statement of Theorem 4:

▶ **Theorem 4.** *The* (ILP-SF) *problem can be solved in*

$$O(\kappa_k)^{2k} \cdot \Delta^2 \cdot \log\Delta + \operatorname{poly}(\text{input size})$$

arithmetic operations. Its feasibility variant can be solved in

$$O(\kappa_k)^{k} \cdot \Delta \cdot \operatorname{poly}(\text{input size})$$

bit operations[6].

To prove Theorem 4, we first construct the dynamic programming algorithm described in Section 4.1, and then in Section 4.2 we instantiate its parameters using the construction of rational normalized LD-preconditioners from Section 3.

4.1 Dynamic program

In the seminal work [12], K. Jansen & L. Rohwedder introduced a new class of dynamic programming algorithms for ILP problems, which utilizes results from discrepancy theory and fast algorithms for tropical and Boolean convolutions on sequences. Our dynamic programming algorithm follows the same pattern but exhibits significant differences, as it employs

[6] Note that for the feasibility problem we estimate the bit complexity, because it is easier to do so for several reasons. The arithmetic complexity can be analyzed in further work.

matrix preconditioning (see Section 3), enumeration of integer points in parallelepipeds (see Section 2.3), and handles a more general Boolean convolution problem (see Section 2.4). Unfortunately, we do not know any accelerated algorithm for solving the tropical variant of the generalized convolution arising in the optimization problem.

▶ **Theorem 29.** *Consider the* (ILP-SF) *problem. Let $\rho \in \mathbb{Z}_{>0}$ be such that $\|z^*\|_1 \leq (6/5)^\rho$, for some optimal integer solution z^* of the problem.*

Assume in addition that the following hold for a given nondegenerate matrix $B \in \mathbb{Q}^{k \times k}$:

1. *denoting $\delta = |\det B|$, assume that for every $\gamma \in \mathbb{Z}_{\geq 1}$ and $r \in \mathbb{Q}^k$, all integer points inside $\big(r + \gamma B \cdot [0,1)^k\big)$ can be enumerated with $O(\gamma)^k \cdot \delta$ operations* [7]*;*
2. *denoting $M = B^{-1} \cdot A$, we assume that $\eta \in \mathbb{Z}_{\geq 1}$ is a given upper bound on* $\operatorname{herdisc}(M)$*.*

Under these assumptions on B, the problem can be solved in

$$\rho \cdot O(\eta)^{2k} \cdot \delta^2 + \operatorname{poly}(\text{input size})$$

arithmetic operations. The feasibility variant of the problem can be solved in

$$\rho \cdot O(\eta)^k \cdot \delta \cdot \operatorname{poly}(\text{input size}, \log \eta) \quad \textit{bit operations.}$$

Proof. Set

$$M = B^{-1}A \qquad \text{and} \qquad v = B^{-1}b.$$

Then $Ax = b$ is equivalent to

$$Mx = v, \qquad x \in \mathbb{Z}^n_{\geq 0}\,.$$

Moreover, any feasible right-hand side u of a system $Mx = u$ belongs to $B^{-1} \cdot \mathbb{Z}^k$, because

$$u = Mx = B^{-1}Ax \in B^{-1} \cdot \mathbb{Z}^k\,.$$

Subproblems. For $i \in \{0, \dots, \rho\}$ and $u \in B^{-1} \cdot \mathbb{Z}^k$, let $\mathcal{P}(i, u)$ be the problem

$$c^\top x \to \max$$
$$\begin{cases} Mx = u, \\ \|x\|_1 \leq (6/5)^i, \\ x \in \mathbb{Z}^n_{\geq 0}\,. \end{cases}$$

Denote its optimum value by $\mathrm{OPT}(i, u)$, and set $\mathrm{OPT}(i, u) = -\infty$ if $\mathcal{P}(i, u)$ is infeasible. Since some optimal solution of $Mx = v$ has ℓ_1-norm at most $(6/5)^\rho$, the optimum of $\mathcal{P}(\rho, v)$ is equal to the optimum of the original problem.

The idea is a divide-and-conquer dynamic program: a solution of $\mathcal{P}(i, u)$ will be split into two smaller solutions corresponding to level $i - 1$ with $u', u'' \approx u/2$.

[7] To be absolutely precise formally, we will also note that the bit complexity of the enumeration must also depend polynomially on the lengths of γ and B.

Single splitting step. Fix $i \geq 1$ and let x be a feasible solution of $\mathcal{P}(i, u)$. If $\|x\|_1 > 1$, then by Lemma 12 there exists $z \in \mathbb{Z}^n_{\geq 0}$ such that

$$\begin{aligned}&0 \leq z \leq x, \qquad \left\|M(z - x/2)\right\|_\infty \leq 2\eta, \quad \text{and}\\ &\frac{1}{6}\|x\|_1 \leq \|z\|_1 \leq \frac{5}{6}\|x\|_1 .\end{aligned}$$

Hence

$$\begin{aligned}&\|z\|_1 \leq \frac{5}{6}\|x\|_1 \leq \frac{5}{6}(6/5)^i = (6/5)^{i-1}, \quad \text{and also}\\ &\|x - z\|_1 = \|x\|_1 - \|z\|_1 \leq \frac{5}{6}\|x\|_1 \leq (6/5)^{i-1}.\end{aligned}$$

If $\|x\|_1 \leq 1$, then the first part of Lemma 12 gives a vector $z \in \mathbb{Z}^n_{\geq 0}$ with

$$0 \leq z \leq x, \qquad \left\|M(z - x/2)\right\|_\infty \leq \eta \leq 2\eta.$$

In this case,

$$\|z\|_1 \leq \|x\|_1 \leq 1 \leq (6/5)^{i-1}, \qquad \|x - z\|_1 \leq \|x\|_1 \leq 1 \leq (6/5)^{i-1}.$$

Thus, in both cases, we obtain z such that

$$0 \leq z \leq x, \qquad \left\|M(z - x/2)\right\|_\infty \leq 2\eta, \qquad \|z\|_1 , \|x - z\|_1 \leq (6/5)^{i-1}.$$

Now, we set

$$u' = Mz, \qquad u'' = M(x - z) = u - u'.$$

Then $u', u'' \in B^{-1} \cdot \mathbb{Z}^k$, and

$$\begin{aligned}&\left\|u' - u/2\right\|_\infty = \left\|Mz - u/2\right\|_\infty = \left\|M(z - x/2)\right\|_\infty \leq 2\eta,\\ &\left\|u'' - u/2\right\|_\infty = \left\|M(x - z) - u/2\right\|_\infty = \left\|M(x/2 - z)\right\|_\infty = \left\|Mz - u/2\right\|_\infty \leq 2\eta;\\ &\text{Thus} \quad u', u'' \in u/2 + 2\eta[-1, 1]^k.\end{aligned}$$

Since z and $x - z$ satisfy the required norm bound, they are feasible for $\mathcal{P}(i - 1, u')$ and $\mathcal{P}(i - 1, u'')$, respectively. Therefore, if x is optimal for $\mathcal{P}(i, u)$, then

$$\mathrm{OPT}(i, u) = c^\top x = c^\top z + c^\top (x - z) \leq \mathrm{OPT}(i - 1, u') + \mathrm{OPT}(i - 1, u'').$$

Conversely, if x' and x'' satisfy

$$\begin{aligned}&Mx' = u', \qquad Mx'' = u'',\\ &\text{then} \quad M(x' + x'') = u;\end{aligned}$$

Moreover, if

$$\begin{aligned}&c^\top x' \geq \mathrm{OPT}(i - 1, u') \qquad \text{and} \qquad c^\top x'' \geq \mathrm{OPT}(i - 1, u''),\\ &\text{then} \quad c^\top (x' + x'') \geq \mathrm{OPT}(i, u).\end{aligned}$$

Hence, to obtain a value at least $\mathrm{OPT}(i, u)$, it is enough to know solutions whose objective values are at least $\mathrm{OPT}(i - 1, u')$ and $\mathrm{OPT}(i - 1, u'')$; the corresponding witnesses do not need to satisfy the norm bound.

Relevant state space. For $j \in \{0, \dots, \rho\}$, define

$$\mathcal{U}_j = \left(\frac{v}{2^j} + 4\eta[-1,1]^k\right) \cap B^{-1} \cdot \mathbb{Z}^k .$$

If $u \in \mathcal{U}_j$, then every child state u' produced by the splitting step belongs to $\mathcal{U}_{j+1}$, because

$$\begin{aligned} u' &\in u/2 + 2\eta[-1,1]^k \subseteq \frac{1}{2}\left(\frac{v}{2^j} + 4\eta[-1,1]^k\right) + 2\eta[-1,1]^k \\ &= \frac{v}{2^{j+1}} + 2\eta[-1,1]^k + 2\eta[-1,1]^k \\ &= \frac{v}{2^{j+1}} + 4\eta[-1,1]^k. \end{aligned}$$

Thus, if $u \in \mathcal{U}_j$, then $u' \in \mathcal{U}_{j+1}$.

We will store a state $u \in \mathcal{U}_j$ by its integer representative $b' \in \mathbb{Z}^k$, given by $B^{-1}b' = u$. Accordingly, define

$$\mathcal{B}_j = \left\{ b' \in \mathbb{Z}^k : B^{-1}b' \in \mathcal{U}_j \right\} = \mathbb{Z}^k \cap \left(\frac{b}{2^j} + 4\eta B[-1,1]^k\right).$$

By the assumption on B, each set $\mathcal{B}_j$ can be enumerated in $O(\eta)^k \cdot \delta$ operations. In particular,

$$|\mathcal{B}_j| = O(\eta)^k \cdot \delta.$$

Dynamic programming tables. For $i \in \{0, \dots, \rho\}$ and $b' \in \mathcal{B}_{\rho-i}$, let

$$u = B^{-1}b'.$$

In the feasibility and optimization variant we store

$$\mathrm{DP}_{\mathrm{feas}}[i, b'] \in \{0,1\} \quad \text{and} \quad \mathrm{DP}[i, b'] \in \mathbb{Z} \cup \{-\infty\}, \quad \text{respectively.}$$

In both the optimization and feasibility variants, the DP entry corresponds to a solution of $Mx = u$. This solution need not be feasible for $\mathcal{P}(i, u)$; it may violate the norm bound. However, in the optimization variant we maintain the invariant that its objective value is at least $\mathrm{OPT}(i, u)$. Precisely, for every state (i, b'), we maintain the following invariants:

- If $\mathcal{P}(i, u)$ is feasible, then

$$\begin{aligned} &\mathrm{DP}_{\mathrm{feas}}[i, b'] = 1, \\ &\quad \mathrm{DP}[i, b'] \geq \mathrm{OPT}(i, u). \end{aligned} \tag{INVARIANT-1}$$

- If a table entry certifies feasibility, then a corresponding solution exists:

$$\begin{aligned} \mathrm{DP}_{\mathrm{feas}}[i, b'] = 1 &\implies \exists x \in \mathbb{Z}^n_{\geq 0} \text{ s.t. } Mx = u, \\ \mathrm{DP}[i, b'] > -\infty &\implies \exists x \in \mathbb{Z}^n_{\geq 0} \text{ s.t. } Mx = u \text{ and } c^\top x = \mathrm{DP}[i, b']. \end{aligned} \tag{INVARIANT-2}$$

Initialization. For $i = 0$, the bound $\|x\|_1 \leq 1$ leaves only the possibilities $x \in \{0, e_1, \dots, e_n\}$. We inspect all these vectors and initialize exactly the non-default entries of $\mathrm{DP}[0, \cdot]$ and $\mathrm{DP}_{\mathrm{feas}}[0, \cdot]$; every remaining state keeps the default value $-\infty$ or 0, respectively. This takes only poly(input size) operations.

Transitions. For $i \geq 1$ and $b' \in \mathcal{B}_{\rho-i}$, define

$$\mathrm{DP}[i, b'] = \max_{\substack{b'' \in \mathcal{B}_{\rho-i+1} \\ b'-b'' \in \mathcal{B}_{\rho-i+1}}} \Big(\mathrm{DP}[i-1, b''] + \mathrm{DP}[i-1, b'-b'']\Big), \tag{Opt-Trans}$$

with the convention that the maximum over the empty set is $-\infty$, and

$$\mathrm{DP}_{\mathrm{feas}}[i, b'] = \bigvee_{\substack{b'' \in \mathcal{B}_{\rho-i+1} \\ b'-b'' \in \mathcal{B}_{\rho-i+1}}} \Big(\mathrm{DP}_{\mathrm{feas}}[i-1, b''] \wedge \mathrm{DP}_{\mathrm{feas}}[i-1, b'-b'']\Big). \tag{Feasibility-Trans}$$

Correctness and Running Time. The algorithm's correctness is equivalent to validity of (INVARIANT-1) and (INVARIANT-2), and it follows from the definitions of subproblems and $\mathrm{DP}[i, b']$. The running time depends on how we evaluate formulas (Opt-Trans) and (Feasibility-Trans). We evaluate formula (Opt-Trans) using a trivial algorithm. For the formula (Feasibility-Trans), we use the fast sparse Boolean convolution algorithm from Lemma 17. Due to space constraints, a detailed proof of correctness and a running-time analysis are deferred to Appendix A.5. ◀

4.2 Adjusting the Parameters

To prove Theorem 4, we will use Theorem 29 with appropriately chosen parameters ρ, η, B, and δ. The parameter ρ can be chosen using a standard argument based on the proximity of the LP and ILP solutions. The details are encapsulated in the following lemma, whose proof is deferred to Appendix A.4.

▶ **Lemma 30.** *Any instance of the* (ILP-SF) *problem can be transformed to an equivalent instance with the following property: if the problem is feasible and bounded, then there exists an optimal solution z^* such that $\|z^*\|_1 = (k \cdot \Delta)^{O(1)}$.*

Due to this lemma, we can assume that $\rho = O\big(\log(k \cdot \Delta)\big)$. Applying Theorem 28 to A, we obtain a rational normalized LD-preconditioner B of A. By Proposition 25, the matrix B satisfies the first condition of Theorem 29. By the definition of an LD-preconditioner,

$$\delta = |\det B| \leq 2^{O(k)} \cdot \Delta.$$

Now the proof follows from Theorem 29.

References

1. Noga Alon and Joel H Spencer. *The probabilistic method.* John Wiley & Sons, 2016. URL: `https://dl.acm.org/doi/abs/10.5555/3002498`.
2. Wojciech Banaszczyk. Balancing vectors and gaussian measures of n-dimensional convex bodies. *Random Structures & Algorithms*, 12(4):351–360, 1998. `doi:10.1002/(SICI)1098-2418(199807)12:4<351::AID-RSA3>3.0.CO;2-S`.
3. Nikhil Bansal and Haotian Jiang. Decoupling via affine spectral-independence: Beck-fiala and komlós bounds beyond banaszczyk. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing*, pages 432–442, 2026. `doi:10.1145/3798129.3800762`.
4. Adrian Bock, Yuri Faenza, Carsten Moldenhauer, and Andres Jacinto Ruiz-Vargas. Solving the Stable Set Problem in Terms of the Odd Cycle Packing Number. In Venkatesh Raman and S. P. Suresh, editors, *34th International Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS 2014)*, volume 29 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 187–198, Dagstuhl, Germany, 2014. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.FSTTCS.2014.187`.

5 Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. URL: `https://web.stanford.edu/~boyd/cvxbook/`.

6 Karl Bringmann, Nick Fischer, and Vasileios Nakos. Deterministic and las vegas algorithms for sparse nonnegative convolution. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms (SODA 2022)*, pages 3069–3090. SIAM, 2022. `doi:10.1137/1.9781611977073.119`.

7 M Cherniavskii, D Gribanov, D Malyshev, and Panos M Pardalos. Delta-modular ILP problems of bounded co-dimension, discrepancy, and convolution. *arXiv preprint*, 2024. `arXiv:2405.17001`.

8 Daniel Dadush, Aleksandar Nikolov, Kunal Talwar, and Nicole Tomczak-Jaegermann. Balancing vectors in any norm. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1–10. IEEE, 2018. `doi:10.1109/FOCS.2018.00010`.

9 Friedrich Eisenbrand and Robert Weismantel. Proximity Results and Faster Algorithms for Integer Programming Using the Steinitz Lemma. *ACM Transactions on Algorithms*, 16(1), November 2019. `doi:10.1145/3340322`.

10 Pascal Giorgi, Bruno Grenet, and Armelle Perret du Cray. Essentially optimal sparse polynomial multiplication. In *Proceedings of the 45th International Symposium on Symbolic and Algebraic Computation*, pages 202–209, 2020. `doi:10.1145/3373207.3404026`.

11 V. Gribanov, D., A. Shumilov, I., S. Malyshev, D., and M. Pardalos, P. On δ-modular integer linear problems in the canonical form and equivalent problems. *J Glob Optim*, 2022. `doi:10.1007/s10898-022-01165-9`.

12 K. Jansen and L. Rohwedder. On integer programming, discrepancy, and convolution. *Mathematics of Operations Research*, 2022. `doi:10.1287/moor.2022.1308`.

13 Aleksandar Nikolov. Randomized rounding for the largest simplex problem. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 861–870, 2015. `doi:10.1145/2746539.2746628`.

14 Christos H. Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4):765–768, October 1981. `doi:10.1145/322276.322287`.

15 A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Chichester, 1998. URL: `https://www.wiley-vch.de/de/fachgebiete/mathematik-und-statistik/theory-of-linear-and-integer-programming-978-0-471-98232-6`.

16 Arne Storjohann. *Algorithms for matrix canonical forms*. PhD thesis, ETH Zurich, 2000. URL: `https://cs.uwaterloo.ca/~astorjoh/diss2up.pdf`.

17 Arne Storjohann and George Labahn. Asymptotically fast computation of Hermite normal forms of integer matrices. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation*, ISSAC '96, pages 259–266, New York, NY, USA, 1996. Association for Computing Machinery. `doi:10.1145/236869.237083`.

A Omitted Proofs

A.1 Proof of Lemma 13

Recall the statement of Lemma 13.

▶ **Lemma 13.** *For a given upper triangular matrix $H \in \mathbb{Q}^{n \times n}$ with strictly positive diagonal elements and $r \in \mathbb{Q}^n$, denote $\mathcal{P} = r + H \cdot [0,1)^n$. The following propositions hold:*

1.

$$\prod_{i=1}^{n} \lfloor H_{ii} \rfloor \leq |\mathcal{P} \cap \mathbb{Z}^n| \leq \prod_{i=1}^{n} \lceil H_{ii} \rceil;$$

2. *The elements of $\mathcal{P} \cap \mathbb{Z}^n$ can be enumerated using $O(n)^2 \cdot \prod_{i=1}^{n} \lceil H_{ii} \rceil$ operations.*

Proof. We enumerate the points of $\mathcal{P} \cap \mathbb{Z}^n$ from the last coordinate to the first. Take $y \in \mathcal{P}$, so

$$y = r + Ht, \qquad t \in [0,1)^n.$$

For $k = n, n-1, \ldots, 1$, assume that $y_{k+1}, \ldots, y_n$ have already been fixed (for $k = n$ this means that nothing is fixed yet). The corresponding values $t_{k+1}, \ldots, t_n$ (such that equalities $y_i = H_{i*}t$ for $y_{k+1}, \ldots, y_n$ are satisfied) are known as well. We know that

$$y_k = r_k + \sum_{j=k}^{n} H_{kj} t_j = r_k + \sum_{j=k+1}^{n} H_{kj} t_j + H_{kk} t_k, \qquad t_k \in [0,1)$$

Set

$$\tau_k = r_k + \sum_{j=k+1}^{n} H_{kj} t_j$$

Hence, the admissible values of y_k are exactly the integers in the interval

$$\tau_k + H_{kk} \cdot [0,1) = [\tau_k, \tau_k + H_{kk}).$$

This interval has length H_{kk}, hence it contains at least $\lfloor H_{kk} \rfloor$ and at most $\lceil H_{kk} \rceil$ integers. These integers can be listed simply by scanning this interval, so this step takes $O\big(\lceil H_{kk} \rceil\big)$ time. After choosing y_k, we recover

$$t_k = \frac{y_k - \tau_k}{H_{kk}}.$$

Thus, this procedure defines a search tree: at step k, every node has between $\lfloor H_{kk} \rfloor$ and $\lceil H_{kk} \rceil$ children, and the leaves are exactly the points of $\mathcal{P} \cap \mathbb{Z}^n$. Therefore,

$$\prod_{i=1}^{n} \lfloor H_{ii} \rfloor \leq |\mathcal{P} \cap \mathbb{Z}^n| \leq \prod_{i=1}^{n} \lceil H_{ii} \rceil.$$

For each such node on level k, we spend $O(n) \cdot \lceil H_{kk} \rceil$ operations. Hence the total work at level k is $O(n) \cdot \prod_{i=1}^{n} \lceil H_{ii} \rceil$. Since there are n levels, the overall number of operations coincides with the stated complexity bound. ◀

A.2 Proof of Lemma 19

Recall the statement of Lemma 19.

▶ **Lemma 19.** *For $\alpha > 0$ and integers $n \geq k > 0$, consider the following optimization problem:*

$$\sum_{\mathcal{J} \in \binom{\{1, \ldots, n\}}{k}} \prod_{j \in \mathcal{J}} c_j \to \max$$
$$\begin{cases} c_1 + \cdots + c_n = \alpha \\ c \in \mathbb{R}^n_{\geq 0}. \end{cases}$$

Then, the point $c = \frac{\alpha}{n} \cdot \mathbf{1}$ is an optimal solution of this problem.

Proof. Let c be an optimal solution, chosen so that $\sum_{i=1}^n c_i^2$ is minimal among all optimal solutions. Suppose $c_l \neq c_s$ for some l and s. Define c' to be the vector obtained from c by replacing both c_l and c_s with $\frac{c_l+c_s}{2}$. Then,

$$\sum_{\mathcal{J} \in \binom{\{1,\dots,n\}}{k}} \prod_{j \in \mathcal{J}} c'_j - \sum_{\mathcal{J} \in \binom{\{1,\dots,n\}}{k}} \prod_{j \in \mathcal{J}} c_j =$$
$$\sum_{\mathcal{J} \in \binom{\{1,\dots,n\} \setminus \{l,s\}}{k-2}} \left(c'_l c'_s \prod_{j \in \mathcal{J}} c'_j - c_l c_s \prod_{j \in \mathcal{J}} c_j \right) =$$
$$\left(\left(\frac{c_l + c_s}{2} \right)^2 - c_l c_s \right) \cdot \sum_{\mathcal{J} \in \binom{\{1,\dots,n\} \setminus \{l,s\}}{k-2}} \prod_{j \in \mathcal{J}} c_j \geq 0.$$

Hence the objective value at c' is no smaller than at c. Since c is optimal, it follows that c' is optimal as well. On the other hand,

$$\sum_{i=1}^n (c'_i)^2 < \sum_{i=1}^n c_i^2,$$

which contradicts the choice of c. ◀

A.3 Proof of Lemma 17

Recall the statement of Lemma 17.

▶ **Lemma 17.** *Problem 15 can be solved in*

$$O\left(T \cdot (k \log L)\right)$$

bit operations with high probability, where $T = \max\left\{\left|\operatorname{supp}(\alpha)\right|, \left|\operatorname{supp}(\beta)\right|, \left|\operatorname{supp}(\gamma)\right|\right\}$ *and* L *is the maximum* ℓ_∞*-norm of elements in* $\operatorname{supp}(\alpha)$ *and* $\operatorname{supp}(\beta)$.

Proof. Construct the polynomials $f_\alpha, f_\beta \in \mathbb{Z}[x_1, \dots, x_k]$ in the following way. For each $a \in \operatorname{supp}(\alpha)$, we include the monomial $\mathbf{x}^a$ in f_α. We do the same for each $b \in \operatorname{supp}(\beta)$ to construct f_β. To make the exponents in both polynomials non-negative, we additionally multiply them by $\mathbf{x}^{L \cdot \mathbf{1}}$. Denoting $g = f_\alpha \cdot f_\beta$, we have

$$g[\mathbf{x}^{a + 2L \cdot \mathbf{1}}] = \sum_{\substack{b \in \operatorname{supp}(\alpha) \\ (a-b) \in \operatorname{supp}(\beta)}} \alpha_b \cdot \beta_{a-b}.$$

Since $\gamma_a = 1$ iff $g[\mathbf{x}^{a+2L\cdot\mathbf{1}}] \neq 0$, the computation of γ reduces to multiplying f_α and f_β.

By construction, $\max\left\{\left\|f_\alpha\right\|_\infty, \left\|f_\beta\right\|_\infty\right\} = 1$ and $\max_i \deg_{x_i}(g) \leq 4L$. Therefore, by Theorem 16, the function γ can be computed in

$$O\left(T \cdot (k \log L) \cdot \operatorname{polylog}(1/\varepsilon)\right)$$

bit operations with probability $1 - \varepsilon$. Taking $\varepsilon = 1/k^{O(1)}$, we can guarantee the correctness with high probability. ◀

▶ Remark 31. There is also a deterministic analogue of Lemma 17. Let $q = 4L + 1$ and encode each vector $a = (a_1, \ldots, a_k) \in [-L, L]^k \cap \mathbb{Z}^k$ by

$$\phi(a) = \sum_{i=1}^{k} (a_i + L) q^{i-1}.$$

Then for any $a, b \in [-L, L]^k \cap \mathbb{Z}^k$, the base-$q$ expansion of $\phi(a) + \phi(b)$ is exactly $a + b + 2L \cdot \mathbf{1}$, because every digit belongs to $\{0, \ldots, 4L\}$ and hence no carries occur. Therefore, Problem 1 reduces to sparse nonnegative convolution of two one-dimensional 0/1 vectors of length q^k.

Applying the algorithm of Bringmann, Fischer, and Nakos [6, Theorem 1], we obtain a running time

$$O\big(T \cdot \operatorname{polylog}(q^k \cdot L)\big) = O\big(T \cdot \operatorname{poly}(k, \log L)\big)$$

in the Word-RAM model from Section 1.3. Hence Lemma 17 admits a deterministic version with complexity $O\big(T \cdot \operatorname{poly}(k, \log L)\big)$.
A similar construction for Boolean convolution is used by Jansen and Rohwedder [12].

A.4 Proof of Lemma 30

Recall the statement of Lemma 30.

▶ **Lemma 30.** *Any instance of the* (ILP-SF) *problem can be transformed to an equivalent instance with the following property: if the problem is feasible and bounded, then there exists an optimal solution z^* such that $\|z^*\|_1 = (k \cdot \Delta)^{O(1)}$.*

Proof. Let v be an optimal vertex solution of the relaxation; such a solution can be computed in polynomial time. By [11, Corollary 2], there exists an optimal solution $\hat{z}$ to the original problem such that

$$\|v - \hat{z}\|_1 \le \chi, \qquad \chi = O\Big(k^2 \cdot \Delta \cdot \sqrt[k]{\Delta}\Big).$$

Following [12], we apply the change of variables $x' = x - y$, where

$$y_i = \max\{0, \lceil v_i \rceil - \chi\}.$$

Then $\hat{z} \ge y$. Moreover, since v has at most k nonzero components, it follows that

$$\|\hat{z} - y\|_1 = O(k\chi).$$

Under this substitution, the original problem becomes an equivalent instance of (ILP-SF) with optimal solution $z^* = \hat{z} - y$. Therefore,

$$\|z^*\|_1 = O(k\chi) = (k\Delta)^{O(1)}.$$ ◀

A.5 Proof of the DP-algorithm's Correctness and Running Time Analysis (Theorem 29)

Correctness. We prove the invariants by induction on i. For $i = 0$, they hold by the exact initialization. Assume that they hold on level $i - 1$, and consider a state $b' \in \mathcal{B}_{\rho - i}$ with $u = B^{-1} b'$.

Suppose first that $\mathcal{P}(i, u)$ is feasible, and let x be an optimal solution. By the splitting step, there exist $u', u'' \in u/2 + 2\eta[-1, 1]^k$ with $u = u' + u''$, such that $\mathcal{P}(i-1, u')$ and $\mathcal{P}(i-1, u'')$ are feasible and

$$\mathrm{OPT}(i, u) \le \mathrm{OPT}(i-1, u') + \mathrm{OPT}(i-1, u'').$$

Since $u \in \mathcal{U}_{\rho-i}$, as shown above $u', u'' \in \mathcal{U}_{\rho-i+1}$. Let

$$b'' = Bu' = Az.$$

Then b'', $b' - b'' \in \mathcal{B}_{\rho-i+1}$, and by the induction hypothesis,

$$\mathrm{DP}[i-1, b''] \ge \mathrm{OPT}(i-1, u'), \qquad \mathrm{DP}[i-1, b'-b''] \ge \mathrm{OPT}(i-1, u''),$$

as well as

$$\mathrm{DP}_{\mathrm{feas}}[i-1, b''] = \mathrm{DP}_{\mathrm{feas}}[i-1, b'-b''] = 1.$$

Therefore, (Opt-Trans) and (Feasibility-Trans) imply

$$\mathrm{DP}[i, b'] \ge \mathrm{OPT}(i, u), \qquad \mathrm{DP}_{\mathrm{feas}}[i, b'] = 1.$$

So (INVARIANT-1) is preserved.
To show that (INVARIANT-2) is also preserved, assume first that

$$\mathrm{DP}_{\mathrm{feas}}[i, b'] = 1.$$

By (Feasibility-Trans), there exists $b'' \in \mathcal{B}_{\rho-i+1}$ with $b' - b'' \in \mathcal{B}_{\rho-i+1}$ such that

$$\mathrm{DP}_{\mathrm{feas}}[i-1, b''] = \mathrm{DP}_{\mathrm{feas}}[i-1, b'-b''] = 1.$$

By the induction hypothesis, there exist vectors $x', x'' \in \mathbb{Z}^n_{\ge 0}$ satisfying

$$Mx' = B^{-1}b'', \qquad Mx'' = B^{-1}(b' - b'').$$

Hence

$$M(x' + x'') = B^{-1}b'' + B^{-1}(b' - b'') = B^{-1}b' = u.$$

Thus $\mathrm{DP}_{\mathrm{feas}}[i, b'] = 1$ certifies the existence of a feasible solution of $Mx = u$. The same argument applies to $\mathrm{DP}[i, b'] > -\infty$. So (INVARIANT-2) is preserved.

Conclusion of correctness. Now consider the root state b, corresponding to $u = v$.
For the optimization problem, (INVARIANT-1) gives

$$\mathrm{DP}[\rho, b] \ge \mathrm{OPT}(\rho, v).$$

On the other hand, if $\mathrm{DP}[\rho, b] > -\infty$, then by (INVARIANT-2) there exists $x \in \mathbb{Z}^n_{\ge 0}$ such that

$$Mx = v \qquad \text{and} \qquad c^\top x = \mathrm{DP}[\rho, b].$$

Hence $\mathrm{DP}[\rho, b]$ cannot exceed the true optimum of $Mx = v$. Since some optimal solution has ℓ_1-norm at most $(6/5)^\rho$, the true optimum is exactly $\mathrm{OPT}(\rho, v)$. Therefore,

$$\mathrm{DP}[\rho, b] = \mathrm{OPT}(\rho, v),$$

and the optimization algorithm is correct.

For the feasibility problem, if the instance is feasible, then the assumed optimal solution is feasible for $\mathcal{P}(\rho, v)$, and (INVARIANT-1) yields

$$\mathrm{DP}_{\mathrm{feas}}[\rho, b] = 1.$$

Conversely, if

$$\mathrm{DP}_{\mathrm{feas}}[\rho, b] = 1,$$

then (INVARIANT-2) gives a vector $x \in \mathbb{Z}^n_{\geq 0}$ with $Mx = v$. Thus

$$\mathrm{DP}_{\mathrm{feas}}[\rho, b] = 1 \quad \Longleftrightarrow \quad Mx = v \text{ is feasible,}$$

and the feasibility algorithm is correct as well.

Running time. As noted above, every set $\mathcal{B}_j$ has size

$$|\mathcal{B}_j| = O(\eta)^k \cdot \delta.$$

For the optimization table, a single level is computed by the trivial quadratic scan in (Opt-Trans), so it costs

$$O(\eta)^{2k} \cdot \delta^2$$

arithmetic operations. Summed over all ρ levels, and including the initialization, this gives

$$\rho \cdot O(\eta)^{2k} \cdot \delta^2 + \mathrm{poly}(\text{input size})$$

arithmetic operations.