Computing Asymptotic Bounds for the Automated Coppersmith Method via Linear Programming

Zhaopeng Ding[1], Zhaopeng Dai[1(✉)], Baofeng Wu[2,3], Yanshuo Zhang[4], and Kejun Zhang[4]

[1] School of Mathematics and Statistics, Qingdao University, Qingdao, China
16678784491@163.com, dzpeng@amss.ac.cn
[2] State Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
wubaofeng@iie.ac.cn
[3] School of Cybersecurity, University of Chinese Academy of Sciences, Beijing, China
[4] Department of Cryptographic Science and Technology, Beijing Electronic Science and Technology Institute, Beijing, China
zhang_yanshuo@163.com, zkj@besti.edu.cn

Abstract. Coppersmith's method is a foundational technique for finding small roots of modular equations, and determining asymptotic bounds for the recoverable roots is a central and challenging part of its analysis. In this paper, we transform the computation of asymptotic bounds for the Automated Coppersmith method, proposed by Meers and Nowakowski (ASIACRYPT 2023), into a linear programming problem, thereby obtaining a provably correct and explicitly computable formula. As applications of our method, we obtain improved asymptotic bounds for five cryptanalytic settings: the Commutative Isogeny Hidden Number Problem, the Modular Inversion Hidden Number Problem, the Elliptic Curve Hidden Number Problem, the Linear Congruential Generators with unknown multiplier, and the Leveled Isogeny Problem with Hints for POKÉ. We believe that our method could be useful for evaluating the security of a broader range of cryptographic settings.

Keywords: Coppersmith's method · automated cryptanalysis · asymptotic bounds · linear programming

1 Introduction

Lattice-based methods for finding small roots can be traced back to Coppersmith's work on small roots of univariate modular equations [12]. The method was subsequently extended to more general polynomial small-root problems [3, 7, 11, 13, 14, 23, 24], and has found wide applications in the cryptanalysis of RSA and its variants [9, 17, 21, 23, 35, 42], various hidden number problems [10, 16, 37–40], and post-quantum cryptographic problems [1, 5, 15, 19, 29, 32, 33]. Coppersmith's method has since become one of the central technical tools in public-key cryptanalysis.

The core idea of Coppersmith's method is to construct a collection of shifted polynomials that share the desired small root with the original polynomial, and to form a lattice L from their coefficient vectors. A lattice basis reduction algorithm is then applied to L, which reduces the problem of finding small roots of modular equations to solving polynomial equations over the integers. The effectiveness of Coppersmith's method is reflected by the small-root bounds $X_1, \ldots, X_k$, which characterize the largest size of solutions that can be recovered. These bounds are typically derived from the following inequality

$$\det(L) = X_1^{p_1(m)} \cdots X_k^{p_k(m)} M^{p_M(m)} < M^{m \dim(L) - \varepsilon},$$

where M is a given modulus and m is a parameter of the algorithm. Here, $p_1(m), \ldots, p_k(m), p_M(m)$ are functions of m determined by the lattice L. To assess the solvability of the corresponding cryptographic problem and its cryptographic security, it is necessary to further analyze the asymptotic behavior of $X_1, \ldots, X_k$, as $m \to \infty$. For specific cryptographic problems, however, the choice of shift polynomials and the derivation of the corresponding asymptotic bounds are usually carried out by hand, making the process time-consuming and laborious. This has motivated the development of automated algorithms for Coppersmith's method, with the aim of automating both the selection of shift polynomials and the derivation of the corresponding asymptotic bounds.

In 2023, Meers and Nowakowski [29] proposed a method for selecting optimal shift polynomials for an arbitrary set of monomials, which is called the Automated Coppersmith method. They pointed out that, for a special class of monomial sets, the corresponding asymptotic bounds can be heuristically determined using Lagrange interpolation. Subsequently, Feng et al. [19] generalized the strategy originally introduced by Jochemsz and May in [23], expanding its application from a single polynomial to multiple polynomials. They introduced sumset theory as a new analytical tool and developed the first provable algorithm for deriving asymptotic small-root bounds by means of computing the associated Newton polytope. Recently, in EUROCRYPT 2025, Ryan [32] proposed that, for an arbitrary set of monomials, shift polynomials can be selected from a suitable ideal via its Gröbner basis. Ryan also suggested that, for monomial sets induced by the integral points of an arbitrary integral polytope, the corresponding asymptotic bounds can be heuristically determined using Lagrange interpolation.

Our Contributions. In this paper, we use linear programming as a rigorous analytical framework for computing asymptotic bounds in the Automated Coppersmith method. We propose a systematic, non-heuristic methodology for deriving asymptotic bounds for this method over general monomial sets. The resulting methodology not only improves the theoretical understanding of Coppersmith's method, but also provides practical tools for evaluating the security of related cryptographic constructions. Our contributions are as follows.

Asymptotic analysis via linear programming. We first generalize Meers and Nowakowski's automated method [29] to multivariable modular equations with

arbitrary powers of an integer: $f_i(x_1, \ldots, x_k) \equiv 0 \pmod{M^{e_i}}$, where $e_i \geq 1$. Then we apply the tools from linear programming to study the asymptotic behavior over monomial sets of the form

$$\mathcal{M}_m = \left\{ x_1^{i_1} \cdots x_k^{i_k} \mid (i_1, \ldots, i_k) \in mP \cap \mathbb{Z}^k \right\},$$

where P is an arbitrary full-dimensional rational polytope. For any sufficiently small $\varepsilon > 0$, we prove that the asymptotic bound of the desired roots (see Theorem 4) satisfies the following formula

$$\prod_{j=1}^{k} X_j^{\int_P x_j \, \mathrm{d}\boldsymbol{x}} < M^{\int_P \varphi_P(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} - \varepsilon},$$

where $\varphi_P(x)$ denotes the optimal value of a parametric linear program determined by the input polynomials and the polytope.

It should be noted that, unlike the heuristic-based approaches in [29] and [32], we prove an explicit formula for computing asymptotic bounds. On the other hand, compared with the integral-polytope framework of [32] and the Newton-polytope framework of [19], our method provides a computational framework based on arbitrary rational polytopes. This greater flexibility enables us to obtain tighter asymptotic bounds.

We also provide an effective algorithm for computing the resulting asymptotic bound[5]; in our experiments, the calculation can be completed within one second. In addition, we propose a heuristic strategy for selecting sufficiently good polytopes. This strategy involves only a small number of tunable parameters and is designed to search for monomial sets that yield strong asymptotic bounds in concrete cryptanalytic applications.

Cryptographic applications. As applications, we instantiate our asymptotic bound formula in five cryptanalytic settings. The resulting computations improve the best-known asymptotic bounds for all considered instances; see Table 1.

Table 1. Comparison of asymptotic bounds between previous works and this work

Setting	Previous Required Leakage	This Work
CIHNP (CSURF)	75.61% [29] 74.03% [33] 74.01% [19] 73.91% [32]	73.38%
MIHNP (3 samples)	55.56% [32]	54.17%
MIHNP (4 samples)	47.92% [32]	46.64%
MIHNP (5 samples)	44.23% [32]	42.06%
ECHNP (3 samples)	69.10% [32]	59.29%
LCG (unknown multiplier a, consecutive values)	66.66% [41, Thm 3]	50.00%
LIPH (unknown degree)	85.29% [15, Thm 4.3]	79.81%

Note. The asymptotic bound computations in [15, 32, 33] are heuristic.

[5] Which is available at https://github.com/ZhaopengDing1/Coppersmith-LP

Concurrent work. For the LCG in Table 1, Feng et al. [20], using the method of [19], independently and concurrently obtained the same asymptotic bound as ours. We also note that, for the LIPH, Feng et al. [18] independently obtained the same asymptotic bound in an ePrint paper released in February 2026. These works follow technical approaches different from ours, and the agreement of the resulting bounds provides independent validation from different perspectives.

Organization. The rest of the paper is organized as follows. Section 2 reviews background on polynomials, lattices, Coppersmith's method, linear programming, and Ehrhart theory. Section 3 derives asymptotic bounds for the Automated Coppersmith method and presents the main results of the paper. Section 4 presents concrete applications and experiments. Section 5 concludes the paper.

2 Preliminaries

Throughout this paper, $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ denote the sets of integers, rational numbers, and real numbers, respectively. Vectors are denoted by bold lowercase letters, e.g., $\boldsymbol{x}$, and are understood to be column vectors unless stated otherwise; matrices are denoted by bold uppercase letters, e.g., $\boldsymbol{A}$. We write $\langle \cdot, \cdot \rangle$ for the standard Euclidean inner product of two vectors. For monomials u and v, $u \mid v$ denotes that u divides v. For $\boldsymbol{x} = (x_1, \ldots, x_k)^\top$ and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k)^\top \in \mathbb{Z}_{\geq 0}^k$, we write $|\boldsymbol{\alpha}| = \alpha_1 + \cdots + \alpha_k$ and

$$\boldsymbol{x}^{\boldsymbol{\alpha}} = x_1^{\alpha_1} \cdots x_k^{\alpha_k}.$$

We also use $f(\boldsymbol{x})$ for $f(x_1, \ldots, x_k)$ and $\mathrm{d}\boldsymbol{x}$ for $\mathrm{d}x_1 \cdots \mathrm{d}x_k$. The notation $f(m) = O(g(m))$ means that there exists a constant $C > 0$, independent of m, such that $|f(m)| \leq C|g(m)|$. For a subset S of a Euclidean space, $\mathrm{int}(S)$ and $\partial(S)$ denote its interior and boundary, respectively.

2.1 Polynomials

Definition 1 (Monomial order). *A* monomial order *on the set of monomials in $\mathbb{Z}[x_1, \ldots, x_k]$ is a total order $\prec$ satisfying the following conditions:*

1. $1 \preceq \boldsymbol{x}^{\boldsymbol{\alpha}}$ *for every* $\boldsymbol{\alpha} \in \mathbb{Z}_{\geq 0}^k$;
2. *whenever* $\boldsymbol{x}^{\boldsymbol{\alpha}} \prec \boldsymbol{x}^{\boldsymbol{\beta}}$, *we have* $\boldsymbol{x}^{\boldsymbol{\alpha}+\boldsymbol{\gamma}} \prec \boldsymbol{x}^{\boldsymbol{\beta}+\boldsymbol{\gamma}}$ *for every* $\boldsymbol{\gamma} \in \mathbb{Z}_{\geq 0}^k$.

In Section 4, we will use the following three monomial orders.

Lexicographic order (lex). Fix a variable order $x_{\pi(1)} \prec x_{\pi(2)} \prec \cdots \prec x_{\pi(k)}$. The lexicographic order associated with this variable order is defined as follows: for two monomials $\boldsymbol{x}^{\boldsymbol{\alpha}}$ and $\boldsymbol{x}^{\boldsymbol{\beta}}$, we write $\boldsymbol{x}^{\boldsymbol{\alpha}} \prec_{\mathrm{lex}} \boldsymbol{x}^{\boldsymbol{\beta}}$ if the largest index s such that $\alpha_{\pi(s)} \neq \beta_{\pi(s)}$ satisfies $\alpha_{\pi(s)} < \beta_{\pi(s)}$.

Degree-lexicographic order (deglex). The degree-lexicographic order associated with the variable order $x_{\pi(1)} \prec x_{\pi(2)} \prec \cdots \prec x_{\pi(k)}$ is defined as follows: $\boldsymbol{x}^{\boldsymbol{\alpha}} \prec_{\mathrm{deglex}} \boldsymbol{x}^{\boldsymbol{\beta}}$ if either $|\boldsymbol{\alpha}| < |\boldsymbol{\beta}|$, or $|\boldsymbol{\alpha}| = |\boldsymbol{\beta}|$ and $\boldsymbol{x}^{\boldsymbol{\alpha}} \prec_{\mathrm{lex}} \boldsymbol{x}^{\boldsymbol{\beta}}$.

Weighted degree-lexicographic order (weighted deglex). Let $\boldsymbol{w} = (w_1, \ldots, w_k) \in \mathbb{R}^k_{>0}$. The weighted degree-lexicographic order associated with $\boldsymbol{w}$ and the variable order $x_{\pi(1)} \prec x_{\pi(2)} \prec \cdots \prec x_{\pi(k)}$ is defined by declaring $\boldsymbol{x}^{\boldsymbol{\alpha}} \prec_{\boldsymbol{w},\mathrm{lex}} \boldsymbol{x}^{\boldsymbol{\beta}}$ if either $\langle \boldsymbol{w}, \boldsymbol{\alpha} \rangle < \langle \boldsymbol{w}, \boldsymbol{\beta} \rangle$, or $\langle \boldsymbol{w}, \boldsymbol{\alpha} \rangle = \langle \boldsymbol{w}, \boldsymbol{\beta} \rangle$ and $\boldsymbol{x}^{\boldsymbol{\alpha}} \prec_{\mathrm{lex}} \boldsymbol{x}^{\boldsymbol{\beta}}$.

For a nonzero polynomial

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{\alpha} \in \mathbb{Z}^k_{\geq 0}} c_{\boldsymbol{\alpha}} \boldsymbol{x}^{\boldsymbol{\alpha}} \in \mathbb{Z}[x_1, \ldots, x_k],$$

we define its total degree as $\deg(f) := \max\{|\boldsymbol{\alpha}| \mid c_{\boldsymbol{\alpha}} \neq 0\}$, and its support as $\mathrm{supp}(f) := \{\boldsymbol{x}^{\boldsymbol{\alpha}} \mid c_{\boldsymbol{\alpha}} \neq 0\}$. We also use the corresponding exponent set

$$A(f) := \{\boldsymbol{\alpha} \in \mathbb{Z}^k_{\geq 0} \mid c_{\boldsymbol{\alpha}} \neq 0\}.$$

We denote by $\|f\| := \left(\sum_{\boldsymbol{\alpha} \in A(f)} |c_{\boldsymbol{\alpha}}|^2\right)^{1/2}$ the Euclidean norm of the coefficient vector of f. Fix a monomial order $\prec$, and let $\boldsymbol{x}^{\boldsymbol{\alpha}_0}$ be the largest monomial in $\mathrm{supp}(f)$ with respect to $\prec$. We define the leading monomial and leading coefficient of f by

$$\mathrm{LM}(f) := \boldsymbol{x}^{\boldsymbol{\alpha}_0}, \qquad \mathrm{LC}(f) := c_{\boldsymbol{\alpha}_0}.$$

If $\mathrm{LC}(f) = 1$, we say that f is monic.

Definition 2 ([29]). *Let $\mathcal{M}$ be a finite set of monomials, and let $\prec$ be a monomial order on $\mathcal{M}$. A set $\mathcal{F}$ of polynomials is said to be $(\mathcal{M}, \prec)$-suitable if the following conditions are satisfied:*

1. *every polynomial $f \in \mathcal{F}$ satisfies $\mathrm{supp}(f) \subseteq \mathcal{M}$;*
2. *for each monomial $\lambda \in \mathcal{M}$, there exists a unique polynomial $f \in \mathcal{F}$ whose leading monomial with respect to $\prec$ is λ.*

Definition 3 (Minkowski sum). *Let $A, B \subseteq \mathbb{R}^k$. The Minkowski sum of A and B is defined by*

$$A + B := \{\boldsymbol{a} + \boldsymbol{b} \mid \boldsymbol{a} \in A,\ \boldsymbol{b} \in B\}.$$

For $m \in \mathbb{Z}_{\geq 1}$, we write

$$mA := \underbrace{A + \cdots + A}_{m \text{ times}}$$

for the m-fold Minkowski sum of A. We also use the convention $0A := \{\mathbf{0}\}$.

We also use the following definition: A function $f(m)$ is called a *quasi-polynomial* if there exists a positive integer n and polynomials $f_0, \ldots, f_{n-1}$ such that $f(m) = f_r(m)$ whenever $m \equiv r \pmod{n}$. The integer n is called a period of f.

2.2 Lattices

A lattice $\mathcal{L} \subseteq \mathbb{R}^n$ is the set of all integer linear combinations of finitely many linearly independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_r \in \mathbb{R}^n$; that is,

$$\mathcal{L} = \mathbb{Z}\boldsymbol{b}_1 + \cdots + \mathbb{Z}\boldsymbol{b}_r.$$

The vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_r$ are called a basis of $\mathcal{L}$. The determinant of $\mathcal{L}$ is defined as

$$\det(\mathcal{L}) = \sqrt{\det(\boldsymbol{B}^\top \boldsymbol{B})},$$

where $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_r)$ denotes the basis matrix of $\mathcal{L}$.

A central problem in algorithmic lattice theory is the *Shortest Vector Problem* (SVP): given a basis of $\mathcal{L}$, find a nonzero vector $\boldsymbol{v} \in \mathcal{L}$ with minimum Euclidean norm. Since solving SVP exactly is computationally difficult, many applications instead rely on sufficiently short approximate solutions.

A fundamental approximation algorithm for SVP is the *LLL algorithm*, introduced by Lenstra, Lenstra, and Lovász [26]. Given a lattice basis, the LLL algorithm computes, in polynomial time, a reduced basis whose vectors are relatively short. In particular, the following standard bound holds.

Lemma 1 (LLL basis reduction [27]). *Let $\mathcal{L} \subseteq \mathbb{Z}^n$ be a full-rank lattice. Given a basis of $\mathcal{L}$ as input, the LLL algorithm outputs a basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ of $\mathcal{L}$ such that*

$$\|\boldsymbol{b}_i\| \le 2^{\frac{n(n-1)}{4(n+1-i)}} \det(\mathcal{L})^{\frac{1}{n+1-i}}, \qquad 1 \le i \le n.$$

2.3 Coppersmith's Method

Let $f_1, \ldots, f_n \in \mathbb{Z}[x_1, \ldots, x_k]$ be given polynomials, and let $X_1, \ldots, X_k$ be positive integer bounds. The goal of Coppersmith's method is to recover a common root $\boldsymbol{x}^{(0)} = (x_1^{(0)}, \ldots, x_k^{(0)})^\top$ modulo M satisfying $|x_i^{(0)}| < X_i$ for $1 \le i \le k$.

The method proceeds as follows. We first construct a family of shift polynomials $\mathcal{G} = \{g_i(\boldsymbol{x})\}_i$ that also vanish at $\boldsymbol{x}^{(0)}$ modulo M^t. We then use the coefficient vectors of the scaled polynomials $g_i(x_1X_1, \ldots, x_kX_k)$ as basis vectors to construct a lattice $\mathcal{L}$. By applying the LLL algorithm (Lemma 1), we obtain relatively short vectors in $\mathcal{L}$. The following lemma shows that sufficiently short vectors yield auxiliary polynomials that vanish at $\boldsymbol{x}^{(0)}$ over $\mathbb{Z}$. These auxiliary polynomials can then be used to recover the target root.

Lemma 2 (Howgrave–Graham [22]). *Let $g \in \mathbb{Z}[x_1, \ldots, x_k]$ have at most ω nonzero monomials. If $g\big(\boldsymbol{x}^{(0)}\big) \equiv 0 \pmod{N}$ and*

$$\|g(x_1X_1, \ldots, x_kX_k)\| < \frac{N}{\sqrt{\omega}},$$

then $g\big(\boldsymbol{x}^{(0)}\big) = 0$ over $\mathbb{Z}$.

Consequently, to obtain k auxiliary polynomials that vanish at $\boldsymbol{x}^{(0)}$ over $\mathbb{Z}$, it is sufficient that the following condition holds.

$$2^{\frac{\dim(\mathcal{L})(\dim(\mathcal{L})-1)}{4(\dim(\mathcal{L})+1-k)}} \det(\mathcal{L})^{\frac{1}{\dim(\mathcal{L})+1-k}} < \frac{M^t}{\sqrt{\dim(\mathcal{L})}}. \tag{1}$$

Finally, to recover the small root $\boldsymbol{x}^{(0)}$ from these auxiliary polynomials, we rely on the following standard heuristic assumption, which holds for most instances arising in practice [19, 25, 27, 28, 31, 32, 35, 37, 38].

Assumption 1. *The ideal in $\mathbb{Q}[x_1, \ldots, x_k]$ generated by the auxiliary polynomials obtained from Coppersmith's method is zero-dimensional.*

Remark 1. In existing analyses of Coppersmith's method, one often assumes that the modulus M is sufficiently large, so that certain lower-order terms can be ignored. Accordingly, (1) can be simplified to $\det(\mathcal{L}) < M^{t(\dim(\mathcal{L})+1-k)-\varepsilon}$ for sufficiently small $\varepsilon > 0$.

We briefly recall the core idea of the Automated Coppersmith method. Details can be found in [29].

Automated Coppersmith method. Meers and Nowakowski [29] proposed an automated method for selecting Coppersmith shift polynomials. The method first chooses a finite monomial set $\mathcal{M}$ and fixes a monomial order $\prec$, together with a positive integer parameter m. For each $\lambda \in \mathcal{M}$, it considers candidate shift polynomials of the form

$$g_{[\lambda,i_1,\ldots,i_n]} = \frac{\lambda}{\mathrm{LM}(f_1)^{i_1} \cdots \mathrm{LM}(f_n)^{i_n}} f_1^{i_1} \cdots f_n^{i_n} M^{m-(i_1+\cdots+i_n)},$$

where $(i_1, \ldots, i_n) \in \mathbb{Z}_{\geq 0}^n$ satisfies $i_1 + \cdots + i_n \leq m$. Such a tuple is called feasible for λ if

$$\prod_{j=1}^{n} \mathrm{LM}(f_j)^{i_j} \mid \lambda \quad \text{and} \quad \mathrm{supp}\big(g_{[\lambda,i_1,\ldots,i_n]}\big) \subseteq \mathcal{M}.$$

The method enumerates all feasible tuples and selects one that maximizes $i_1 + \cdots + i_n$. Repeating this procedure yields one selected shift polynomial for each monomial in $\mathcal{M}$, and the resulting collection is an $(\mathcal{M}, \prec)$-suitable set. Moreover, for the fixed monomial set $\mathcal{M}$, this selection minimizes the determinant of the resulting lattice.

2.4 Linear Programming

In this subsection, we briefly review the concepts and results from linear programming that will be used in this paper. These can be found in standard textbooks; see, for example, [6, 36].

A subset S of $\mathbb{R}^m$ is called convex if, for every $x, y \in S$ and every $0 < t < 1$, $tx + (1-t)y \in S$. For any set S in $\mathbb{R}^m$, we denote by $\mathrm{conv}(S)$ the smallest convex set containing S, called the convex hull of S.

Throughout this paper, vector inequalities are understood componentwise.

Definition 4 (Polyhedron, Polytope). *A polyhedron is a set of the form*

$$P = \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\},$$

for some matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ *and some vector* $\boldsymbol{b} \in \mathbb{R}^m$*. The set* P *is called a rational polyhedron if* $\boldsymbol{A}$ *and* $\boldsymbol{b}$ *are rational. A bounded polyhedron is called a polytope.*

It is straightforward to see that, if $S \subseteq \mathbb{R}^n$ is finite, then $\mathrm{conv}(S)$ is a polytope.

Definition 5 (Vertex). *Let* $P \subseteq \mathbb{R}^n$ *be a polyhedron. A point* $\boldsymbol{v} \in P$ *is called a vertex of* P *if there do not exist two distinct points* $\boldsymbol{x}, \boldsymbol{y} \in P$ *and a scalar* $\lambda \in (0, 1)$ *such that* $\boldsymbol{v} = \lambda\boldsymbol{x} + (1 - \lambda)\boldsymbol{y}$*.*

We use the following standard form of a linear program.

Definition 6 (Linear Program). *A linear program has the following form:*

$$\max \quad \boldsymbol{c}^\top \boldsymbol{x} \quad s.t. \quad \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}, \quad \boldsymbol{x} \geq \boldsymbol{0}, \tag{2}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$*,* $\boldsymbol{b} \in \mathbb{R}^m$*, and* $\boldsymbol{c} \in \mathbb{R}^n$*. The polyhedron*

$$\mathcal{P} := \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0}\}$$

is called the feasible region of the linear program (2)*. Moreover, when the variable* $\boldsymbol{x}$ *in* (2) *is restricted to integer vectors, the program is referred to as an integer linear program.*

Lemma 3 ([36, Thm 3.4]). *The linear program* (2) *admits an optimal solution if and only if its feasible region* $\mathcal{P}$ *is nonempty and the objective function* $\boldsymbol{c}^\top\boldsymbol{x}$ *is bounded above on* $\mathcal{P}$*.*

Definition 7 (Dual Linear Program). *Consider the linear program*

$$\text{(P)} \qquad \max \quad \boldsymbol{c}^\top \boldsymbol{x} \quad s.t. \quad \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}, \quad \boldsymbol{x} \geq \boldsymbol{0}.$$

Its dual linear program is

$$\text{(D)} \qquad \min \quad \boldsymbol{b}^\top \boldsymbol{y} \quad s.t. \quad \boldsymbol{A}^\top \boldsymbol{y} \geq \boldsymbol{c}, \quad \boldsymbol{y} \geq \boldsymbol{0}.$$

The problem (P) *is called the primal problem, and* (D) *is called its dual problem.*

Lemma 4 (Strong Duality [36, Thm 5.2]). *If* (P) *has an optimal solution* $\boldsymbol{x}^*$*, then* (D) *also has an optimal solution* $\boldsymbol{y}^*$ *such that* $\boldsymbol{c}^\top\boldsymbol{x}^* = \boldsymbol{b}^\top\boldsymbol{y}^*$*.*

Lemma 5 ([6, Thm 2.7]). *Consider the linear program* (2)*. Suppose that the feasible region* $\mathcal{P}$ *of* (2) *has at least one vertex and that an optimal solution exists. Then there exists an optimal solution that is a vertex of* $\mathcal{P}$*.*

2.5 Ehrhart Theory

Ehrhart theory studies the problem of counting lattice points in integer dilates of polytopes. Let $P \subseteq \mathbb{R}^k$ be a full-dimensional rational polytope, and let $\phi \in \mathbb{R}[x_1, \ldots, x_k]$ be a polynomial function. The weighted Ehrhart counting function is defined by

$$E_P(m, \phi) = \sum_{\boldsymbol{x} \in mP \cap \mathbb{Z}^k} \phi(\boldsymbol{x}), \qquad m \in \mathbb{Z}_{\geq 0}.$$

When $\phi = 1$, this function reduces to the usual Ehrhart counting function. The case of polynomial weights is described by the theory of weighted Ehrhart quasi-polynomials; see [2]. In particular, we have the following result.

Lemma 6 ([2]). *Let $P \subseteq \mathbb{R}^k$ be a full-dimensional rational polytope. Then $E_P(m, \phi)$ is a quasi-polynomial in m of degree at most $k + \deg(\phi)$. Moreover, if ϕ is homogeneous, then*

$$E_P(m, \phi) = m^{k+\deg(\phi)} \int_P \phi(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O\Big(m^{k+\deg(\phi)-1}\Big).$$

Definition 8 (Affine function). *Let P be a nonempty subset of $\mathbb{R}^k$. A function $L : P \to \mathbb{R}$ is called affine if there exist $\boldsymbol{c} \in \mathbb{R}^k$ and $d \in \mathbb{R}$ such that $L(\boldsymbol{x}) = \boldsymbol{c}^\top \boldsymbol{x} + d$.*

Definition 9 (Piecewise affine function). *Let $P \subseteq \mathbb{R}^k$ be a full-dimensional rational polyhedron. A function $g : P \to \mathbb{R}$ is called piecewise affine if there exist finitely many full-dimensional rational polyhedra $Q_1, \ldots, Q_N \subseteq P$ such that $P = \bigcup_{j=1}^{N} Q_j$, $\operatorname{int}(Q_i) \cap \operatorname{int}(Q_j) = \varnothing$ for $i \neq j$, and the restriction of g to Q_j is affine for each $j \in \{1, \ldots, N\}$.*

We now apply Lemma 6 to affine functions and piecewise affine functions on rational polytopes.

Corollary 1. *Let $Q \subseteq \mathbb{R}^k$ be a full-dimensional rational polytope and let $L : Q \to \mathbb{R}$ be an affine function. Then, for $m \in \mathbb{Z}_{\geq 1}$, we have*

$$\sum_{\boldsymbol{\lambda} \in mQ \cap \mathbb{Z}^k} L\big(\tfrac{\boldsymbol{\lambda}}{m}\big) = m^k \int_Q L(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}).$$

Proof. Write $L(\boldsymbol{x}) = \boldsymbol{c}^\top \boldsymbol{x} + d$. Then, by Lemma 6, we have

$$\begin{aligned}
\sum_{\boldsymbol{\lambda} \in mQ \cap \mathbb{Z}^k} L\big(\tfrac{\boldsymbol{\lambda}}{m}\big) &= \frac{1}{m} \sum_{\boldsymbol{\lambda} \in mQ \cap \mathbb{Z}^k} \boldsymbol{c}^\top \boldsymbol{\lambda} + d \sum_{\boldsymbol{\lambda} \in mQ \cap \mathbb{Z}^k} 1 \\
&= \frac{1}{m} \left(m^{k+1} \int_Q \boldsymbol{c}^\top \boldsymbol{x} \, \mathrm{d}\boldsymbol{x} + O(m^k) \right) + d \left(m^k \int_Q 1 \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}) \right) \\
&= m^k \int_Q (\boldsymbol{c}^\top \boldsymbol{x} + d) \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}) \\
&= m^k \int_Q L(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}).
\end{aligned}$$

This proves the claim. □

Corollary 2. *Let $P \subseteq \mathbb{R}^k$ be a full-dimensional rational polytope, and let $g : P \to \mathbb{R}$ be a piecewise affine function. Then, for $m \in \mathbb{Z}_{\geq 1}$, we have*

$$\sum_{\boldsymbol{\lambda} \in mP \cap \mathbb{Z}^k} g\big(\tfrac{\boldsymbol{\lambda}}{m}\big) = m^k \int_P g(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + O(m^{k-1}).$$

Proof. According to Definition 9, we may take full-dimensional rational polytopes $Q_1, \ldots, Q_N \subseteq P$ with pairwise disjoint interiors such that $P = \bigcup_{j=1}^{N} Q_j$. For each j, denote by L_j the restriction of g to Q_j; then L_j is affine. Write $L_j(\boldsymbol{x}) = \boldsymbol{c}_j^{\top}\boldsymbol{x} + d_j$, where $\boldsymbol{c}_j \in \mathbb{R}^k, d_j \in \mathbb{R}$.

Define

$$S_m := \sum_{\boldsymbol{\lambda} \in mP \cap \mathbb{Z}^k} g\big(\tfrac{\boldsymbol{\lambda}}{m}\big)\,, \qquad T_m := \sum_{j=1}^{N} \sum_{\boldsymbol{\lambda} \in mQ_j \cap \mathbb{Z}^k} L_j\big(\tfrac{\boldsymbol{\lambda}}{m}\big)\,.$$

Since each Q_j is compact and each L_j is continuous, we have

$$C := \max_{1 \leq j \leq N} \sup_{\boldsymbol{x} \in Q_j} |L_j(\boldsymbol{x})| < \infty.$$

Set $B = \bigcup_{j=1}^{N} \partial(Q_j)$. In this proof, $mB = \{m\boldsymbol{b} \mid \boldsymbol{b} \in B\}$. Since every point of $P \setminus B$ lies in exactly one of the polytopes Q_j, we have

$$\begin{aligned} S_m - T_m &= \sum_{\boldsymbol{\lambda} \in mP \cap \mathbb{Z}^k} g\left(\tfrac{\boldsymbol{\lambda}}{m}\right) - \sum_{j=1}^{N} \sum_{\boldsymbol{\lambda} \in mQ_j \cap \mathbb{Z}^k} L_j\left(\tfrac{\boldsymbol{\lambda}}{m}\right) \\ &= \sum_{\boldsymbol{\lambda} \in mP \cap \mathbb{Z}^k} \Big(g\left(\tfrac{\boldsymbol{\lambda}}{m}\right) - \sum_{j\colon \boldsymbol{\lambda}/m \in Q_j} L_j\left(\tfrac{\boldsymbol{\lambda}}{m}\right)\Big) \\ &= \sum_{\boldsymbol{\lambda} \in mB \cap \mathbb{Z}^k} \Big(g\left(\tfrac{\boldsymbol{\lambda}}{m}\right) - \sum_{j\colon \boldsymbol{\lambda}/m \in Q_j} L_j\left(\tfrac{\boldsymbol{\lambda}}{m}\right)\Big). \end{aligned}$$

Therefore,

$$\begin{aligned} |S_m - T_m| &\leq \sum_{\boldsymbol{\lambda} \in mB \cap \mathbb{Z}^k} \Big(\big|g\left(\tfrac{\boldsymbol{\lambda}}{m}\right)\big| + \sum_{j\colon \boldsymbol{\lambda}/m \in Q_j} \big|L_j\left(\tfrac{\boldsymbol{\lambda}}{m}\right)\big|\Big) \\ &\leq (N+1)C\,|mB \cap \mathbb{Z}^k|. \end{aligned}$$

Thus $S_m = T_m + O(|mB \cap \mathbb{Z}^k|)$.

Since B is a finite union of rational polytopes of dimension at most $k-1$, we have $|mB \cap \mathbb{Z}^k| = O(m^{k-1})$. Therefore $S_m = T_m + O(m^{k-1})$.

By Corollary 1, for each $1 \leq j \leq N$, we have

$$\sum_{\boldsymbol{\lambda} \in mQ_j \cap \mathbb{Z}^k} L_j\big(\tfrac{\boldsymbol{\lambda}}{m}\big) = m^k \int_{Q_j} L_j(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + O(m^{k-1}).$$

Summing over $j = 1, \ldots, N$ gives

$$\begin{aligned} T_m &= m^k \sum_{j=1}^{N} \int_{Q_j} L_j(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}) \\ &= m^k \sum_{j=1}^{N} \int_{Q_j} g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}). \end{aligned}$$

Finally, the overlaps $Q_i \cap Q_j$ for $i \neq j$ are contained in B, and hence have zero k-dimensional Lebesgue measure in $\mathbb{R}^k$. Therefore

$$\sum_{j=1}^{N} \int_{Q_j} g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \int_{P} g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}.$$

Combining the preceding estimates yields

$$\sum_{\boldsymbol{\lambda} \in mP \cap \mathbb{Z}^k} g\left(\tfrac{\boldsymbol{\lambda}}{m}\right) = m^k \int_{P} g(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}),$$

as claimed. □

3 Main Results

This section presents a linear-programming-based method for deriving asymptotic bounds in the Automated Coppersmith method. Section 3.1 extends the automated method to systems of modular equations with different powers of the modulus; Section 3.2 reformulates the selection of shift polynomials as a pointwise optimization problem; Section 3.3 derives an asymptotic bound formula via linear programming and presents the corresponding algorithm; and Section 3.4 discusses the choice of the polytope.

3.1 Generalizing the Automated Coppersmith Method

In concrete applications, Coppersmith's method is often not applied directly to the original polynomials. Instead, one first performs a preprocessing step that derives additional congruences from the ideal generated by the original polynomials; see, e.g., [37, 39]. Suppose that the original modular equations are

$$\tilde{f}_i(\boldsymbol{x}) \equiv 0 \pmod{M}, \qquad 1 \leq i \leq s,$$

and let $I = \langle \tilde{f}_1, \ldots, \tilde{f}_s \rangle \subseteq \mathbb{Z}[x_1, \ldots, x_k]$. If $f \in I^q$, then every common root of the original polynomials modulo M satisfies $f(\boldsymbol{x}) \equiv 0 \pmod{M^q}$. In this way, one can exploit the algebraic structure of I^q more fully, which may lead to stronger small-root bounds.

This observation motivates the following general setting. We consider the modular equations

$$f_j(\boldsymbol{x}) \equiv 0 \pmod{M^{e_j}}, \qquad 1 \le j \le n, \tag{3}$$

where each $f_j \in \mathbb{Z}[x_1, \ldots, x_k]$ is nonconstant and monic and satisfies $\mathbf{0} \in A(f_j)$. We assume that the system (3) has a common small root $\boldsymbol{x}^{(0)} = (x_1^{(0)}, \ldots, x_k^{(0)})^\top$ satisfying $|x_i^{(0)}| < X_i$ $(1 \le i \le k)$, where $X_1, \ldots, X_k$ are prescribed bounds. We also write $\boldsymbol{e} = (e_1, \ldots, e_n)^\top \in \mathbb{Z}_{>0}^n$ for the vector of exponents of the modulus powers.

Automated Coppersmith method for (3)**.**

The main idea behind the Automated Coppersmith method is captured in Definition 2. It allows us to bypass all the technical details arising from lattice theory in Coppersmith's method and replace them with purely combinatorial constraints. In this way, we can explicitly construct a triangular lattice basis, whose determinant can be computed simply as the product of its diagonal entries. Therefore, according to Definition 2, the shift polynomials are constructed through the following steps:

Step 1. Choose a finite monomial set $\mathcal{M}_m$ depending on the parameter m.

Step 2. Choose a monomial order $\prec$.

Step 3. Choose a set of shift polynomials $\mathcal{G}_m$ that is $(\mathcal{M}_m, \prec)$-suitable.

To construct a set of shift polynomials satisfying the above condition, for each $\lambda \in \mathcal{M}_m$ and $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_n)^\top \in \mathbb{Z}_{\ge 0}^n$, we define the candidate shifted polynomial

$$F_{[\lambda,\boldsymbol{\ell}]}(\boldsymbol{x}) := \frac{\lambda}{\prod_{j=1}^n \mathrm{LM}(f_j)^{\ell_j}} \prod_{j=1}^n f_j(\boldsymbol{x})^{\ell_j}, \tag{4}$$

satisfying $\prod_{j=1}^n \mathrm{LM}(f_j)^{\ell_j}$ divides λ ($\mathrm{LM}\big(F_{[\lambda,\boldsymbol{\ell}]}\big) = \lambda$) and $\mathrm{supp}\big(F_{[\lambda,\boldsymbol{\ell}]}\big) \subseteq \mathcal{M}_m$. By choosing one such polynomial for each λ, we obtain an $(\mathcal{M}_m, \prec)$-suitable set. We call such $\boldsymbol{\ell}$ a feasible exponent vector for λ. The set of all feasible exponent vectors is denoted by

$$\mathcal{S}_m(\lambda) := \left\{ \boldsymbol{\ell} \in \mathbb{Z}_{\ge 0}^n \;\middle|\; \prod_{j=1}^n \mathrm{LM}(f_j)^{\ell_j} \mid \lambda, \quad \mathrm{supp}\big(F_{[\lambda,\boldsymbol{\ell}]}\big) \subseteq \mathcal{M}_m \right\}.$$

Clearly, $\mathcal{S}_m(\lambda)$ is nonempty and finite.

Note that $F_{[\lambda,\boldsymbol{\ell}]}(\boldsymbol{x}^{(0)}) \equiv 0 \pmod{M^{\langle \boldsymbol{e},\boldsymbol{\ell}\rangle}}$. We further need to lift it to the common modulus M^{t_m}. To ensure that the determinant of the resulting lattice is sufficiently small, we introduce the following definition. For each $\lambda \in \mathcal{M}_m$, define the optimal feasible value

$$r_m(\lambda) := \max\{\langle \boldsymbol{e}, \boldsymbol{\ell}\rangle \mid \boldsymbol{\ell} \in \mathcal{S}_m(\lambda)\}.$$

After constructing $\mathcal{S}_m(\lambda)$ by enumeration, we choose an exponent vector $\boldsymbol{\ell}_\lambda \in \mathcal{S}_m(\lambda)$ satisfying $\langle \boldsymbol{e}, \boldsymbol{\ell}_\lambda \rangle = r_m(\lambda)$ and set $t_m := \max_{\lambda \in \mathcal{M}_m} r_m(\lambda)$. Then the shifted polynomial selected for λ is

$$g_\lambda(\boldsymbol{x}) := M^{\,t_m - r_m(\lambda)} F_{[\lambda, \boldsymbol{\ell}_\lambda]}(\boldsymbol{x}).$$

Thus, our automated method outputs the family $\mathcal{G}_m := \{g_\lambda(\boldsymbol{x}) \mid \lambda \in \mathcal{M}_m\}$, where each g_λ is supported on $\mathcal{M}_m$, its leading monomial is exactly λ and $g_\lambda(\boldsymbol{x}^{(0)}) \equiv 0 \pmod{M^{t_m}}$. Hence $\mathcal{G}_m$ is $(\mathcal{M}_m, \prec)$-suitable. The corresponding algorithm is given in Appendix D.

Remark 2. Whereas [29] takes $t_m = m$, our definition of t_m ensures by construction that $\langle \boldsymbol{e}, \boldsymbol{\ell} \rangle \le t_m$.

We now define the Coppersmith lattice $\mathcal{L}_m$ to be the lattice generated by the coefficient vectors of $g_\lambda(x_1 X_1, \ldots, x_k X_k)$ for $\lambda \in \mathcal{M}_m$, where the columns are indexed by the monomials in $\mathcal{M}_m$ and ordered according to $\prec$. Since $\mathcal{G}_m$ is $(\mathcal{M}_m, \prec)$-suitable, the associated lattice matrix is triangular. Hence, its determinant is

$$\det(\mathcal{L}_m) = \prod_{j=1}^{k} X_j^{p_j(m)} M^{p_M(m)}, \quad \text{where} \begin{cases} p_j(m) := \displaystyle\sum_{x_1^{i_1} \cdots x_k^{i_k} \in \mathcal{M}_m} i_j, \quad 1 \le j \le k, \\ p_M(m) := \displaystyle\sum_{\lambda \in \mathcal{M}_m} \big(t_m - r_m(\lambda)\big). \end{cases}$$

When M is sufficiently large, the determinant condition (1) can be written as

$$\prod_{j=1}^{k} X_j^{p_j(m)} M^{p_M(m)} < M^{t_m |\mathcal{M}_m| - (k-1)t_m - \varepsilon}$$

for sufficiently small $\varepsilon > 0$. Equivalently,

$$\prod_{j=1}^{k} X_j^{p_j(m)} < M^{\,p_0(m) - (k-1)t_m - \varepsilon}, \qquad \text{with} \quad p_0(m) := \sum_{\lambda \in \mathcal{M}_m} r_m(\lambda).$$

For convenience, we assume that $\mathcal{M}_m$ is induced by an exponent set $\mathcal{B}_m \subseteq \mathbb{Z}_{\ge 0}^k$, namely

$$\mathcal{M}_m = \{\boldsymbol{x}^{\boldsymbol{\beta}} \mid \boldsymbol{\beta} \in \mathcal{B}_m\}.$$

In this notation,

$$\begin{cases} p_j(m) = \displaystyle\sum_{(i_1, \ldots, i_k) \in \mathcal{B}_m} i_j, & 1 \le j \le k, \\ p_0(m) = \displaystyle\sum_{(i_1, \ldots, i_k) \in \mathcal{B}_m} r_m(x_1^{i_1} \cdots x_k^{i_k}). \end{cases}$$

Remark 3. The expression above shows that, for a fixed monomial set, the Automated Coppersmith method selects optimal feasible shifts and thereby maximizes the resulting small-root bound. However, the corresponding selection strategy in [19] does not take this point into account.

Asymptotic form. As is standard in previous work, asymptotic bounds are obtained by considering an infinite family of shift polynomials. We therefore focus on the asymptotic regime $m \to \infty$ and assume that $\lim_{m\to\infty} |\mathcal{B}_m| = \infty$. This leads to the following asymptotic notion.

Definition 10 (Asymptotic bound). *If there exist an integer $k_0 > 0$ and real constants $\sigma_0, \sigma_1, \ldots, \sigma_k$ such that*

$$\begin{cases} \lim\limits_{m\to\infty} \dfrac{p_j(m)}{m^{k_0}} = \sigma_j, \quad \textit{for } 1 \le j \le k, \\ \lim\limits_{m\to\infty} \dfrac{p_0(m)}{m^{k_0}} = \sigma_0, \\ \lim\limits_{m\to\infty} \dfrac{t_m}{m^{k_0}} = 0, \end{cases}$$

then we call $\sigma_0, \sigma_1, \ldots, \sigma_k$ the asymptotic-bound coefficients of the Automated Coppersmith method. We express the corresponding asymptotic bound as

$$\prod_{j=1}^{k} X_j^{\sigma_j} < M^{\sigma_0 - \varepsilon},$$

for sufficiently small $\varepsilon > 0$.

3.2 Reformulating Shift Selection

To apply Ehrhart theory and linear programming, we further assume that the exponent sets are of the form

$$\mathcal{B}_m = mP \cap \mathbb{Z}^k,$$

where $P \subseteq \mathbb{R}^k_{\ge 0}$ is a fixed full-dimensional rational polytope. An example is shown in Figure 1.

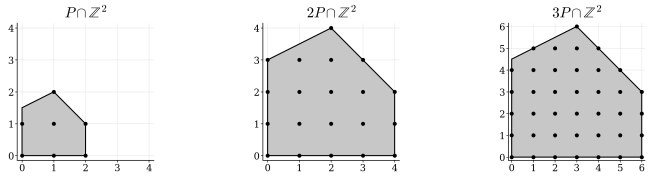

Fig. 1. The lattice-point sets $mP \cap \mathbb{Z}^2$ for $m = 1, 2, 3$, where $P = \mathrm{conv}\{(0,0), (2,0), (2,1), (1,2), (0, \frac{3}{2})\}$.

We now reformulate the construction in Section 3.1 as a pointwise optimization problem, in which feasibility is expressed solely in terms of exponent-set containment. Here, set addition is understood as Minkowski addition.

For each $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_n)^\top \in \mathbb{Z}^n_{\geq 0}$, set $A(\boldsymbol{\ell}) := \sum_{j=1}^n \ell_j A(f_j)$ and $\boldsymbol{\alpha}(\boldsymbol{\ell}) := \sum_{j=1}^n \ell_j \boldsymbol{\alpha}_j$, where $\mathrm{LM}(f_j) = \boldsymbol{x}^{\boldsymbol{\alpha}_j}$ for $1 \leq j \leq n$.

Since $\mathbf{0} \in A(f_j)$ for every j, it follows that $\mathbf{0} \in A(\boldsymbol{\ell})$. For each $m \geq 1$ and $\boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0}$, we further define

$$\mathcal{J}^{(m)}_{\boldsymbol{\ell}} := \left\{ \boldsymbol{x}^{\boldsymbol{\beta}} \;\middle|\; \boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq \mathcal{B}_m \right\}.$$

This yields the following lemma.

Lemma 7. *For any $\boldsymbol{\beta} \in \mathcal{B}_m = mP \cap \mathbb{Z}^k$ and $\boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0}$, we have*

$$\boldsymbol{\ell} \in \mathcal{S}_m(\boldsymbol{x}^{\boldsymbol{\beta}}) \quad \Longleftrightarrow \quad \boldsymbol{x}^{\boldsymbol{\beta}} \in \mathcal{J}^{(m)}_{\boldsymbol{\ell}}.$$

Proof. Fix $\boldsymbol{\beta} \in \mathcal{B}_m$ and $\boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0}$. For $\lambda = \boldsymbol{x}^{\boldsymbol{\beta}}$, whenever the candidate shift in (4) is well defined, it can be written as $F_{[\boldsymbol{x}^{\boldsymbol{\beta}}, \boldsymbol{\ell}]} = \boldsymbol{x}^{\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell})} f_1^{\ell_1} \cdots f_n^{\ell_n}$. Hence, by the Minkowski additivity of Newton polytopes under polynomial multiplication, we have

$$\begin{aligned} \operatorname{conv}\left(A\left(F_{[\boldsymbol{x}^{\boldsymbol{\beta}}, \boldsymbol{\ell}]}\right)\right) &= \boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + \sum_{j=1}^n \ell_j \operatorname{conv}(A(f_j)) \\ &= \boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + \operatorname{conv}(A(\boldsymbol{\ell})). \end{aligned} \tag{5}$$

Suppose first that $\boldsymbol{\ell} \in \mathcal{S}_m(\boldsymbol{x}^{\boldsymbol{\beta}})$. Then, by the definition of $\mathcal{S}_m$, the candidate shift is well defined and $A(F_{[\boldsymbol{x}^{\boldsymbol{\beta}}, \boldsymbol{\ell}]}) \subseteq \mathcal{B}_m \subseteq mP$. Since mP is convex, taking convex hulls and using (5) gives $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + \operatorname{conv}(A(\boldsymbol{\ell})) \subseteq mP$. In particular, $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq mP$, and hence $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq mP \cap \mathbb{Z}^k = \mathcal{B}_m$. By the definition of $\mathcal{J}^{(m)}_{\boldsymbol{\ell}}$, this implies that $\boldsymbol{x}^{\boldsymbol{\beta}} \in \mathcal{J}^{(m)}_{\boldsymbol{\ell}}$.

Conversely, suppose that $\boldsymbol{x}^{\boldsymbol{\beta}} \in \mathcal{J}^{(m)}_{\boldsymbol{\ell}}$. Then $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq \mathcal{B}_m$. Since $\mathbf{0} \in A(f_j)$ for every j, we have $\mathbf{0} \in A(\boldsymbol{\ell})$, and hence $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) \in \mathcal{B}_m \subseteq \mathbb{Z}^k_{\geq 0}$. Thus $\boldsymbol{x}^{\boldsymbol{\alpha}(\boldsymbol{\ell})} \mid \boldsymbol{x}^{\boldsymbol{\beta}}$, and hence the candidate shift $F_{[\boldsymbol{x}^{\boldsymbol{\beta}}, \boldsymbol{\ell}]}$ is well defined.

Moreover, $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq \mathcal{B}_m \subseteq mP$. By the convexity of mP, it follows that $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + \operatorname{conv}(A(\boldsymbol{\ell})) \subseteq mP$. Together with (5), this implies $A(F_{[\boldsymbol{x}^{\boldsymbol{\beta}}, \boldsymbol{\ell}]}) \subseteq mP \cap \mathbb{Z}^k = \mathcal{B}_m$. Equivalently, $\operatorname{supp}(F_{[\boldsymbol{x}^{\boldsymbol{\beta}}, \boldsymbol{\ell}]}) \subseteq \mathcal{M}_m$. Hence, by the definition of $\mathcal{S}_m$, we conclude that $\boldsymbol{\ell} \in \mathcal{S}_m(\boldsymbol{x}^{\boldsymbol{\beta}})$. □

Corollary 3. *For any $\lambda \in \mathcal{M}_m$, we have*

$$r_m(\lambda) = \max\left\{ \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle \;\middle|\; \boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0},\, \lambda \in \mathcal{J}^{(m)}_{\boldsymbol{\ell}} \right\}.$$

Proof. The assertion follows immediately from Lemma 7 and the definition of $r_m(\lambda)$. □

3.3 Computation of Asymptotic Bounds

Computation of $p_j(m)$. We have the following theorem.

Theorem 1. *The function $p_j(m)$ is a quasi-polynomial of degree $k+1$ satisfying*

$$p_j(m) = m^{k+1} \int_P x_j \, \mathrm{d}\boldsymbol{x} + O(m^k).$$

Proof. By Lemma 6, the stated expression follows immediately. Moreover, since $P \subseteq \mathbb{R}^k_{\geq 0}$ is a full-dimensional polytope, we have $\int_P x_j \, \mathrm{d}\boldsymbol{x} > 0$. It follows that the quasi-polynomial has degree $k+1$. □

Computation of $p_0(m)$. We first reduce the computation of $r_m(\boldsymbol{x}^{\boldsymbol{\beta}})$ to an integer linear program. Fix $\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k$. By Corollary 3, we have

$$\begin{aligned} r_m(\boldsymbol{x}^{\boldsymbol{\beta}}) &= \max\Big\{ \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle \;\Big|\; \boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0},\, \boldsymbol{x}^{\boldsymbol{\beta}} \in \mathcal{J}_{\boldsymbol{\ell}}^{(m)} \Big\} \\ &= \max\Big\{ \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle \;\Big|\; \boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0},\, \boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq mP \cap \mathbb{Z}^k \Big\} \\ &= \max\Big\{ \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle \;\Big|\; \boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0},\, \boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq mP \Big\}. \end{aligned}$$

The last equality holds because $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell})$ is already a finite subset of $\mathbb{Z}^k$.

Since P is a rational polytope, we may choose a finite index set T, vectors $\boldsymbol{a}_t \in \mathbb{Q}^k$, and scalars $b_t \in \mathbb{Q}$ such that

$$P = \big\{ \boldsymbol{x} \in \mathbb{R}^k \;\big|\; \langle \boldsymbol{a}_t, \boldsymbol{x} \rangle \leq b_t,\ \forall t \in T \big\}$$

and $mP = \big\{ \boldsymbol{x} \in \mathbb{R}^k \;\big|\; \langle \boldsymbol{a}_t, \boldsymbol{x} \rangle \leq mb_t,\ \forall t \in T \big\}$. Thus, for fixed $\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k$ and $\boldsymbol{\ell} \in \mathbb{Z}^n_{\geq 0}$, the containment $\boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + A(\boldsymbol{\ell}) \subseteq mP$ is equivalent to

$$\max_{\boldsymbol{u} \in A(\boldsymbol{\ell})} \langle \boldsymbol{a}_t,\, \boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + \boldsymbol{u} \rangle \leq mb_t, \qquad \forall t \in T.$$

For each $t \in T$ and $1 \leq j \leq n$, define $\Delta_t(f_j) := \max_{\boldsymbol{u} \in A(f_j)} \langle \boldsymbol{a}_t, \boldsymbol{u} - \boldsymbol{\alpha}_j \rangle$. Since $\boldsymbol{\alpha}_j \in A(f_j)$, we have $\Delta_t(f_j) \geq \langle \boldsymbol{a}_t, \boldsymbol{\alpha}_j - \boldsymbol{\alpha}_j \rangle = 0$. We then obtain

$$\begin{aligned} &\max_{\boldsymbol{u} \in A(\boldsymbol{\ell})} \langle \boldsymbol{a}_t,\, \boldsymbol{\beta} - \boldsymbol{\alpha}(\boldsymbol{\ell}) + \boldsymbol{u} \rangle \\ &= \langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle + \max_{\boldsymbol{u} \in A(\boldsymbol{\ell})} \langle \boldsymbol{a}_t,\, \boldsymbol{u} - \boldsymbol{\alpha}(\boldsymbol{\ell}) \rangle \\ &= \langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle + \sum_{j=1}^n \ell_j \max_{\boldsymbol{u} \in A(f_j)} \langle \boldsymbol{a}_t,\, \boldsymbol{u} - \boldsymbol{\alpha}_j \rangle \\ &= \langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle + \sum_{j=1}^n \ell_j \Delta_t(f_j). \end{aligned}$$

Consequently,

$$\boxed{\begin{aligned} r_m(\boldsymbol{x}^{\boldsymbol{\beta}}) = \max_{\boldsymbol{\ell} \in \mathbb{Z}^n} \quad & \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle \\ \text{s.t.} \quad & \sum_{j=1}^n \ell_j \Delta_t(f_j) \leq mb_t - \langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle, \quad \forall t \in T, \\ & \ell_j \geq 0, \qquad 1 \leq j \leq n. \end{aligned}} \tag{ILP}$$

Thus, for fixed m and $\boldsymbol{\beta}$, the value $r_m(\boldsymbol{x}^{\boldsymbol{\beta}})$ is the optimal value of an integer linear program with n integer variables and $|T|$ linear constraints. Although integer linear programming is NP-hard in general [30], our goal is not to solve this integer programming exactly for each fixed m, but rather to understand the asymptotic behavior of $r_m(\boldsymbol{x}^{\boldsymbol{\beta}})$ as $m \to \infty$. To this end, for each $\boldsymbol{x} \in P$, we introduce the following linear program:

$$\begin{aligned} \varphi_P(\boldsymbol{x}) := \max_{\boldsymbol{\tau} \in \mathbb{R}^n} \quad & \langle \boldsymbol{e}, \boldsymbol{\tau} \rangle \\ \text{s.t.} \quad & \sum_{j=1}^{n} \tau_j \Delta_t(f_j) \le b_t - \langle \boldsymbol{a}_t, \boldsymbol{x} \rangle, \quad \forall t \in T, \\ & \tau_j \ge 0, \qquad 1 \le j \le n. \end{aligned} \tag{LP}$$

We first show that the linear program (LP) is well-defined.

Lemma 8. *For each $\boldsymbol{x} \in P$, the linear program* (LP) *admits an optimal solution.*

Proof. Fix $\boldsymbol{x} \in P$, and let $\mathcal{P}(\boldsymbol{x})$ denote the feasible region of (LP). By Lemma 3, it suffices to show that $\mathcal{P}(\boldsymbol{x})$ is nonempty and that the objective function is bounded above on it.

Nonemptiness is immediate. Since $\boldsymbol{x} \in P$, we have $b_t - \langle \boldsymbol{a}_t, \boldsymbol{x} \rangle \ge 0$ for every $t \in T$. Thus $\boldsymbol{\tau} = \boldsymbol{0}$ satisfies all constraints of (LP), and hence $\mathcal{P}(\boldsymbol{x}) \neq \emptyset$.

We next record a simple consequence of the boundedness of P. For each $1 \le j \le n$, there exists $t_j \in T$ such that $\Delta_{t_j}(f_j) > 0$. Indeed, since f_j is nonconstant and $0 \in A(f_j)$, the set $A(f_j)$ contains an element $\boldsymbol{u}_j \neq \boldsymbol{\alpha}_j$. Set $\boldsymbol{d}_j := \boldsymbol{u}_j - \boldsymbol{\alpha}_j \neq \boldsymbol{0}$. If $\langle \boldsymbol{a}_t, \boldsymbol{d}_j \rangle \le 0$ for all $t \in T$, then, for every $s \ge 0$,

$$\langle \boldsymbol{a}_t, \boldsymbol{x} + s\boldsymbol{d}_j \rangle = \langle \boldsymbol{a}_t, \boldsymbol{x} \rangle + s\langle \boldsymbol{a}_t, \boldsymbol{d}_j \rangle \le b_t \quad \forall t \in T.$$

Hence $\boldsymbol{x} + s\boldsymbol{d}_j \in P$ for all $s \ge 0$, so P contains an unbounded ray, contradicting the boundedness of P. Therefore, for some $t_j \in T$, one has $\langle \boldsymbol{a}_{t_j}, \boldsymbol{d}_j \rangle > 0$. By the definition of $\Delta_t(f_j)$,

$$\Delta_{t_j}(f_j) = \max_{\boldsymbol{u} \in A(f_j)} \langle \boldsymbol{a}_{t_j}, \boldsymbol{u} - \boldsymbol{\alpha}_j \rangle \ge \langle \boldsymbol{a}_{t_j}, \boldsymbol{u}_j - \boldsymbol{\alpha}_j \rangle = \langle \boldsymbol{a}_{t_j}, \boldsymbol{d}_j \rangle > 0.$$

It remains to show that the feasible region is bounded. Let $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_n)^\top \in \mathcal{P}(\boldsymbol{x})$. Since $\Delta_t(f_i) \ge 0$ for all i and t, applying the constraint indexed by t_j gives

$$\tau_j \Delta_{t_j}(f_j) \le \sum_{i=1}^{n} \tau_i \Delta_{t_j}(f_i) \le b_{t_j} - \langle \boldsymbol{a}_{t_j}, \boldsymbol{x} \rangle.$$

As $\Delta_{t_j}(f_j) > 0$, it follows that

$$0 \le \tau_j \le \frac{b_{t_j} - \langle \boldsymbol{a}_{t_j}, \boldsymbol{x} \rangle}{\Delta_{t_j}(f_j)} \qquad (1 \le j \le n).$$

Thus $\mathcal{P}(\boldsymbol{x})$ is contained in a bounded box. In particular, the linear objective $\boldsymbol{\tau} \mapsto \langle \boldsymbol{e}, \boldsymbol{\tau} \rangle$ is bounded above on $\mathcal{P}(\boldsymbol{x})$. Lemma 3 now implies that (LP) admits an optimal solution. □

The following key lemma shows that determining the first-order terms of the asymptotic bound does not require inefficient integer linear programming; ordinary linear programming suffices.

Lemma 9. *For every $\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k$, we have*

$$m\,\varphi_P\Big(\tfrac{\boldsymbol{\beta}}{m}\Big) - \sum_{j=1}^{n} e_j \le r_m(\boldsymbol{x}^{\boldsymbol{\beta}}) \le m\,\varphi_P\Big(\tfrac{\boldsymbol{\beta}}{m}\Big).$$

In particular, $r_m(\boldsymbol{x}^{\boldsymbol{\beta}}) = m\,\varphi_P(\boldsymbol{\beta}/m) + O(1)$.

Proof. Fix $\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k$. We first prove the upper bound. Let $\boldsymbol{\ell} \in \mathbb{Z}^n_{\ge 0}$ be any feasible solution of (ILP). Dividing the constraints in (ILP) by m, we see that $\boldsymbol{\ell}/m$ is feasible for (LP) at $\boldsymbol{\beta}/m$. Hence $\langle \boldsymbol{e}, \boldsymbol{\ell} \rangle = m\,\langle \boldsymbol{e}, \boldsymbol{\ell}/m \rangle \le m\,\varphi_P(\boldsymbol{\beta}/m)$. Taking the maximum over all feasible $\boldsymbol{\ell}$ in (ILP) gives $r_m(\boldsymbol{x}^{\boldsymbol{\beta}}) \le m\,\varphi_P(\boldsymbol{\beta}/m)$.

For the reverse inequality, let $\boldsymbol{\tau}^*$ be an optimal solution of (LP) at $\boldsymbol{\beta}/m$, whose existence follows from Lemma 8. Set $\ell_j = \lfloor m\tau_j^* \rfloor$ for $1 \le j \le n$. Since $\Delta_t(f_j) \ge 0$, for every $t \in T$ we have

$$\sum_{j=1}^{n} \ell_j \Delta_t(f_j) \le \sum_{j=1}^{n} m\tau_j^* \Delta_t(f_j) \le m\,(b_t - \langle \boldsymbol{a}_t, \boldsymbol{\beta}/m \rangle) = mb_t - \langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle.$$

Therefore $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_n)^\top$ is feasible for (ILP). Moreover, using $\lfloor z \rfloor \ge z - 1$ and the optimality of $\boldsymbol{\tau}^*$, we obtain

$$r_m(\boldsymbol{x}^{\boldsymbol{\beta}}) \ge \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle = \sum_{j=1}^{n} e_j \lfloor m\tau_j^* \rfloor \ge m\langle \boldsymbol{e}, \boldsymbol{\tau}^* \rangle - \sum_{j=1}^{n} e_j = m\,\varphi_P\Big(\tfrac{\boldsymbol{\beta}}{m}\Big) - \sum_{j=1}^{n} e_j.$$

This completes the proof. □

The following theorem and its proof establish the computability of φ_P.

Theorem 2. *The function $\varphi_P : P \to \mathbb{R}$ is piecewise affine.*

Proof. Set

$$\boldsymbol{A} := \big(\boldsymbol{a}_t^\top\big)_{t \in T} \in \mathbb{Q}^{|T| \times k}, \qquad \boldsymbol{b} := (b_t)_{t \in T} \in \mathbb{Q}^{|T|},$$
$$\boldsymbol{\Delta} := \big(\Delta_t(f_j)\big)_{t \in T,\ 1 \le j \le n} \in \mathbb{Q}_{\ge 0}^{|T| \times n}.$$

Writing (LP) in vector-inequality form, we see that, for $\boldsymbol{x} \in P$, the value $\varphi_P(\boldsymbol{x})$ is the optimal value of

$$\max \quad \boldsymbol{e}^\top \boldsymbol{\tau} \quad \text{s.t.} \quad \boldsymbol{\Delta}\boldsymbol{\tau} \le \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}, \quad \boldsymbol{\tau} \ge \boldsymbol{0}. \tag{6}$$

The dual linear program is

$$\min \quad (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})^\top \boldsymbol{z} \quad \text{s.t.} \quad \boldsymbol{\Delta}^\top \boldsymbol{z} \geq \boldsymbol{e}, \quad \boldsymbol{z} \geq \boldsymbol{0}. \tag{7}$$

The feasible region of (7) is

$$Y := \{\boldsymbol{z} \in \mathbb{R}^{|T|} \mid \boldsymbol{z} \geq \boldsymbol{0},\ \boldsymbol{\Delta}^\top \boldsymbol{z} \geq \boldsymbol{e}\}.$$

This is a fixed rational polyhedron, independent of $\boldsymbol{x}$. By Lemma 8, the primal problem (6) admits an optimal solution for every $\boldsymbol{x} \in P$. Hence, by Lemma 4,

$$\varphi_P(\boldsymbol{x}) = \min_{\boldsymbol{z} \in Y} (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})^\top \boldsymbol{z} \qquad (\boldsymbol{x} \in P).$$

Moreover, by Lemma 5, for each $\boldsymbol{x} \in P$ there exists an optimal solution of (7) that is a vertex of Y. Since Y is defined by finitely many rational linear inequalities, it has finitely many vertices, all rational. Write

$$\mathrm{Vert}(Y) = \{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(W)}\}.$$

Then

$$\varphi_P(\boldsymbol{x}) = \min_{1 \leq i \leq W} (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})^\top \boldsymbol{z}^{(i)} \qquad (\boldsymbol{x} \in P).$$

For $1 \leq i \leq W$, define

$$L_i(\boldsymbol{x}) := (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})^\top \boldsymbol{z}^{(i)} = \boldsymbol{b}^\top \boldsymbol{z}^{(i)} - \left(\boldsymbol{A}^\top \boldsymbol{z}^{(i)}\right)^\top \boldsymbol{x}.$$

Thus each L_i is affine on P. Relabel the pairwise distinct functions among $L_1|_P, \ldots, L_W|_P$ as $\phi_1, \ldots, \phi_N$. Then

$$\varphi_P(\boldsymbol{x}) = \min_{1 \leq i \leq N} \phi_i(\boldsymbol{x}) \qquad (\boldsymbol{x} \in P).$$

For $1 \leq i \leq N$, define the corresponding cell

$$C_i := \{\boldsymbol{x} \in P \mid \phi_i(\boldsymbol{x}) \leq \phi_j(\boldsymbol{x}) \text{ for all } 1 \leq j \leq N\}.$$

Since each difference $\phi_i - \phi_j$ is affine with rational coefficients, each C_i is a rational polyhedron. The cells $C_1, \ldots, C_N$ cover P, because the minimum of the finite set $\{\phi_1(\boldsymbol{x}), \ldots, \phi_N(\boldsymbol{x})\}$ is attained for every $\boldsymbol{x} \in P$. Moreover, $\varphi_P = \phi_i$ on C_i.

We next restrict to the full-dimensional cells. Let $\mathcal{I} := \{\, i \mid \mathrm{int}(C_i) \neq \varnothing \,\} = \{\, i \mid \dim C_i = k \,\}$. Since the cells C_i form a finite closed polyhedral subdivision of P, the cells with empty interior are contained in the union of the full-dimensional cells. Hence $P = \bigcup_{i \in \mathcal{I}} C_i$. It remains to check that the interiors of the full-dimensional cells are pairwise disjoint. Suppose that, for some distinct $i, j \in \mathcal{I}$, $\mathrm{int}(C_i) \cap \mathrm{int}(C_j) \neq \varnothing$. On this nonempty open set, we have $\varphi_P = \phi_i = \phi_j$. Hence the affine function $\phi_i - \phi_j$ vanishes on a nonempty open set and therefore vanishes identically. This contradicts the choice of $\phi_1, \ldots, \phi_N$ as pairwise distinct affine functions. Thus $\mathrm{int}(C_i) \cap \mathrm{int}(C_j) = \varnothing$ for $i \neq j$, $i, j \in \mathcal{I}$. After reindexing the cells in $\{C_i\}_{i \in \mathcal{I}}$ as $Q_1, \ldots, Q_K$, we have $P = \bigcup_{\nu=1}^{K} Q_\nu$, $\mathrm{int}(Q_\mu) \cap \mathrm{int}(Q_\nu) = \varnothing$ for any $\mu \neq \nu$, and φ_P agrees with an affine function on each Q_ν. Therefore, by Definition 9, the function $\varphi_P : P \to \mathbb{R}$ is piecewise affine. □

Theorem 3. *The function $p_0(m)$ satisfies*

$$p_0(m) = m^{k+1} \int_P \varphi_P(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^k).$$

Proof. By Lemma 9, summing over $\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k$ yields

$$p_0(m) = m \sum_{\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k} \varphi_P\Big(\frac{\boldsymbol{\beta}}{m}\Big) + O\big(|mP \cap \mathbb{Z}^k|\big) \, .$$

Moreover, by Lemma 6, we have $|mP \cap \mathbb{Z}^k| = O(m^k)$. Hence

$$p_0(m) = m \sum_{\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k} \varphi_P\Big(\frac{\boldsymbol{\beta}}{m}\Big) + O(m^k).$$

By Theorem 2, the function φ_P is piecewise affine on P. Therefore, Corollary 2 applies and yields

$$\sum_{\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k} \varphi_P\Big(\frac{\boldsymbol{\beta}}{m}\Big) = m^k \int_P \varphi_P(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^{k-1}).$$

Substituting this estimate into the preceding expression, we obtain

$$p_0(m) = m^{k+1} \int_P \varphi_P(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + O(m^k),$$

as claimed. □

Theorem 4. *For any full-dimensional rational polytope $P \subseteq \mathbb{R}^k_{\geq 0}$, consider the monomial set $\mathcal{M}_m = \{\boldsymbol{x}^{\boldsymbol{\beta}} \mid \boldsymbol{\beta} \in mP \cap \mathbb{Z}^k\}$. Then the Automated Coppersmith method applied with $\mathcal{M}_m$ yields the following asymptotic bound coefficients:*

$$\begin{cases} \sigma_j = \displaystyle\int_P x_j \, \mathrm{d}\boldsymbol{x}, \qquad 1 \leq j \leq k, \\ \sigma_0 = \displaystyle\int_P \varphi_P(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}. \end{cases} \tag{8}$$

For sufficiently small $\varepsilon > 0$, the corresponding asymptotic small-root bound is

$$\prod_{j=1}^{k} X_j^{\sigma_j} < M^{\sigma_0 - \varepsilon}.$$

Proof. By Theorems 1 and 3, the identities in (8) follow immediately. It remains to prove that $\lim_{m \to \infty} t_m / m^{k+1} = 0$, where $t_m = \max_{\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k} r_m(\boldsymbol{x}^{\boldsymbol{\beta}})$.

By Lemma 9, for every $\boldsymbol{\beta} \in mP \cap \mathbb{Z}^k$, we have $r_m(\boldsymbol{x}^{\boldsymbol{\beta}}) \leq m \, \varphi_P(\boldsymbol{\beta}/m)$. Moreover, Theorem 2 shows that φ_P is piecewise affine on P. Since P is bounded, φ_P is bounded above on P. Hence there exists a constant $C > 0$, independent of m, such that $t_m \leq Cm$. Therefore, since $k \geq 1$, we have $0 \leq t_m / m^{k+1} \leq C/m^k \to 0$. This proves the claim. □

Algorithm 1 gives an explicit procedure for computing the asymptotic coefficients. The experiments in the Appendix C show that this procedure is more reliable and substantially faster than heuristic Lagrange interpolation in the tested instances.

Algorithm 1: Computation of the asymptotic-bound coefficients

Input: A rational polytope $P = \{\boldsymbol{x} \in \mathbb{R}^k \mid \langle \boldsymbol{a}_t, \boldsymbol{x} \rangle \leq b_t,\ \forall t \in T\}$ and data $\{(f_j, \boldsymbol{\alpha}_j, e_j)\}_{j=1}^n$.

Output: LP function $\varphi_P(\boldsymbol{x})$ and asymptotic-bound coefficients $\sigma_1, \ldots, \sigma_k, \sigma_0$.

for $t \in T$ *and* $1 \leq j \leq n$ **do**
  $\Delta_t(f_j) \leftarrow \max_{\boldsymbol{u} \in A(f_j)} \langle \boldsymbol{a}_t, \boldsymbol{u} - \boldsymbol{\alpha}_j \rangle$;
Construct $\boldsymbol{A} = (\boldsymbol{a}_t^\top)_{t \in T}$, $\boldsymbol{b} = (b_t)_{t \in T}$, $\boldsymbol{\Delta} = (\Delta_t(f_j))_{t \in T,\, 1 \leq j \leq n}$, and $\boldsymbol{e} = (e_1, \ldots, e_n)^\top$;
Set $Y \leftarrow \{\boldsymbol{z} \in \mathbb{R}_{\geq 0}^{|T|} \mid \boldsymbol{\Delta}^\top \boldsymbol{z} \geq \boldsymbol{e}\}$;
Enumerate $\mathrm{Vert}(Y) = \{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(W)}\}$;
for $i = 1, \ldots, W$ **do**
  $L_i(\boldsymbol{x}) \leftarrow (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})^\top \boldsymbol{z}^{(i)}$;
Relabel the distinct restrictions among $L_1|_P, \ldots, L_W|_P$ as $\phi_1, \ldots, \phi_N$;
$\varphi_P(\boldsymbol{x}) \leftarrow \min\{\phi_1(\boldsymbol{x}), \ldots, \phi_N(\boldsymbol{x})\}$;
$\sigma_0 \leftarrow 0$;
for $i = 1, \ldots, N$ **do**
  $Q_i \leftarrow P \cap \bigcap_{h=1}^N \{\boldsymbol{x} \in \mathbb{R}^k : \phi_i(\boldsymbol{x}) \leq \phi_h(\boldsymbol{x})\}$;
  if $Q_i \neq \emptyset$ *and* $\dim(Q_i) = k$ **then**
    $\sigma_0 \leftarrow \sigma_0 + \int_{Q_i} \phi_i(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}$;
for $j = 1, \ldots, k$ **do**
  $\sigma_j \leftarrow \int_P x_j\, \mathrm{d}\boldsymbol{x}$;
return $\varphi_P(\boldsymbol{x}), (\sigma_1, \ldots, \sigma_k, \sigma_0)$;

Remark 4. In fact, $p_0(m)$ is an eventual quasi-polynomial: there exists an integer $m_0 \geq 1$ such that, for all $m \geq m_0$, $p_0(m)$ is a quasi-polynomial. We defer the full proof to the Appendix A.

Remark 5. Algorithm 1 is stated for a fixed polytope P. In the next subsection, we consider the corresponding computation when P is chosen from a parameterized family of polytopes.

3.4 Choosing a Suitable Polytope

Choosing a suitable monomial set, or equivalently a suitable polytope P, is crucial for obtaining strong asymptotic bounds. This task, however, appears to be challenging: existing work on asymptotic-bound computation, including [19,29,32], does not provide a general-purpose rule for making an optimal choice.

One basic observation is that the asymptotic small-root bound depends primarily on the shape of P, rather than on its size. Therefore, we propose a heuristic for constructing an effective search family of polytopes.

Heuristic strategy for selecting the polytope. Since optimizing over all admissible rational polytopes is infeasible in practice, we restrict the search to a parameterized family whose facet normals are induced by the input polynomials $f_1, \dots, f_n$.

Consider the following two Newton-type polytopes:

$$P_\cup := \operatorname{conv}\Big(\bigcup_{i=1}^{n} A(f_i)\Big), \qquad P_\Sigma := \operatorname{conv}\Big(\sum_{i=1}^{n} A(f_i)\Big),$$

and write their facet descriptions as

$$P_\cup = \big\{\boldsymbol{x} \in \mathbb{R}^k \;\big|\; \langle \boldsymbol{a}_t^\cup, \boldsymbol{x}\rangle \le b_t^\cup,\ \forall t \in T_\cup\big\},$$

and

$$P_\Sigma = \big\{\boldsymbol{x} \in \mathbb{R}^k \;\big|\; \langle \boldsymbol{a}_t^\Sigma, \boldsymbol{x}\rangle \le b_t^\Sigma,\ \forall t \in T_\Sigma\big\}.$$

We use their facet normals as candidate directions; that is,

$$\mathcal{N} = \{\boldsymbol{a}_t^\cup : t \in T_\cup\} \cup \{\boldsymbol{a}_t^\Sigma : t \in T_\Sigma\}.$$

Choose a subcollection $\boldsymbol{v}_1, \dots, \boldsymbol{v}_q \in \mathcal{N}$, and partition $[q]$ into two disjoint subsets J_{fix} and J_{par}. The set J_{fix} indexes fixed constraints, whereas J_{par} indexes tunable facets. This gives the parameterized family

$$P(\boldsymbol{\theta}) = \Big\{\boldsymbol{x} \in \mathbb{R}^k_{\ge 0} \;\Big|\; \langle \boldsymbol{v}_j, \boldsymbol{x}\rangle \le d_j \text{ for } j \in J_{\text{fix}}, \quad \langle \boldsymbol{v}_j, \boldsymbol{x}\rangle \le \theta_j \text{ for } j \in J_{\text{par}}\Big\}.$$

Here, the constants d_j are fixed in advance, while $\boldsymbol{\theta} = (\theta_j)_{j \in J_{\text{par}}}$ is optimized over the prescribed parameter domain.

Remark 6. In principle, one may include all normal vectors in $\mathcal{N}$. In practice, however, after optimization over the resulting family of polytopes, some of the corresponding inequalities are often found to be inactive or implied by the others, and can therefore be discarded.

Remark 7. For the parameterized family $P(\boldsymbol{\theta})$, the function $\varphi_{P(\boldsymbol{\theta})}(\boldsymbol{x}, \boldsymbol{\theta})$ can still be computed by applying Algorithm 1, while keeping the parameters symbolic. The key observation is that the dual feasible polyhedron Y used in the computation is independent of $\boldsymbol{\theta}$. Hence, the algorithm yields an explicit parameterized piecewise-affine expression for $\varphi_{P(\boldsymbol{\theta})}$ as a function of both $\boldsymbol{x}$ and $\boldsymbol{\theta}$.

By contrast, computing the corresponding parametric integral in closed form is substantially more difficult. The resulting expression is typically a complicated piecewise polynomial function of the parameters; for example, in the CIHNP instance considered in the next section, the three-parameter setting already splits into 102 distinct cases. We therefore do not attempt to derive a closed-form formula for the full parametric integral. Instead, in the subsequent computations, we optimize the parameters numerically over a bounded domain to a prescribed finite precision. This approach is considerably more efficient in practice and, in our experiments, suffices to identify strong asymptotic bounds.

Remark 8. Empirically, the above selection strategy, together with suitable parameter tuning, already produces strong asymptotic bounds across our examples. By contrast, enlarging the search space by allowing P to have a more complicated shape typically yields only marginal additional gains.

4 Applications

This section presents several applications of our method for computing asymptotic bounds. We improve previously known bounds for five classes of problems: the Commutative Isogeny Hidden Number Problem for CSURF [32], the Modular Inversion Hidden Number Problem [32], the Elliptic Curve Hidden Number Problem [32], the Linear Congruential Generators with unknown multiplier [41], and the Leveled Isogeny Problem with Hints (LIPH) for unknown-degree POKÉ [15]. For all examples below, the polytopes are chosen according to the heuristic strategy described in Section 3.4. All asymptotic bounds below depend on ε; for convenience, we have omitted ε in the expressions. In Table 1, we compare our results with those of previous work.

4.1 Commutative Isogeny Hidden Number Problem

In [29, Theorem 4], the modular equations associated with CIHNP-CSURF are given by

$$\begin{cases} f_1(x_1, x_2) = x_1^2 + a_1 x_1 x_2^2 + a_2 x_1 x_2 + a_3 x_1 + a_4 x_2^2 + a_5 x_2 + a_6 \equiv 0 \pmod{M}, \\ f_2(x_1, x_3) = x_3^2 + b_1 x_1^2 x_3 + b_2 x_1 x_3 + b_3 x_3 + b_4 x_1^2 + b_5 x_1 + b_6 \equiv 0 \pmod{M}. \end{cases}$$

Improvement of CIHNP-CSURF. We use the lex monomial order with $x_2 \prec x_1 \prec x_3$ and the polytope

$$P(\boldsymbol{\theta}) = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \;\middle|\; \begin{array}{l} x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_2 \leq 1, \quad x_2 + x_3 \leq \theta_1, \\ 2x_1 + x_2 \leq \theta_2, \quad 2x_1 + x_2 + 4x_3 \leq \theta_3 \end{array} \right\}.$$

With this choice, we obtain

$$\varphi_{P(\boldsymbol{\theta})}(\boldsymbol{x}) = \min\Big\{ \frac{x_1 + x_3}{2}, \frac{\theta_1 - x_2}{2}, \frac{\theta_2 - x_2}{4}, \frac{1 - x_2 + x_3}{2}, \frac{\theta_2 + 2 - 2x_1 - 3x_2}{4}, \frac{2\theta_1 + \theta_2 - 2x_1 - 3x_2 - 2x_3}{4} \Big\}.$$

By Theorem 4, we have

$$\prod_{j=1}^{3} X_j^{\int_{P(\boldsymbol{\theta})} x_j \, \mathrm{d}\boldsymbol{x}} < M^{\int_{P(\boldsymbol{\theta})} \varphi_{P(\boldsymbol{\theta})}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}}.$$

When $X_1 = X_2 = X_3 = X$, numerical optimization over $\boldsymbol{\theta}$ gives the near-optimal choice $(\theta_1, \theta_2, \theta_3) = (1.58, 2.51, 4.19)$, which yields the bound $X < M^{0.2662}$.

4.2 Modular Inversion Hidden Number Problem

In [10], MIHNP can be formulated as the following modular equations:

$$f_i(x_i, \alpha) = (x_i + u_i)(\alpha + t_i) - 1 \equiv 0 \pmod{M}, \qquad 1 \le i \le n,$$

where α is the unknown hidden number.

Improvement of MIHNP with 3 samples. Eliminating α via a Gröbner basis computation yields the following modular equations:

$$f_{ij}(x_i, x_j) = x_i x_j + a_{ij} x_i + b_{ij} x_j + c_{ij} \equiv 0 \pmod{M}, \qquad 1 \le i < j \le 3.$$

We use the deglex monomial order with $x_1 \prec x_2 \prec x_3$ and the polytope

$$P(\theta) = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \;\middle|\; 0 \le x_i \le 1 \ (1 \le i \le 3), \quad \sum_{i=1}^{3} x_i \le \theta \right\}.$$

With this choice, we obtain

$$\varphi_{P(\theta)}(\boldsymbol{x}) = \min\Big\{ \frac{x_1 + x_2 + x_3}{2},\, x_1 + x_2,\, x_1 + x_3,\, x_2 + x_3 \Big\}.$$

By Theorem 4, we have

$$\prod_{j=1}^{3} X_j^{\int_{P(\theta)} x_j \, \mathrm{d}\boldsymbol{x}} < M^{\int_{P(\theta)} \varphi_{P(\theta)}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}}.$$

When $X_1 = X_2 = X_3 = X$, numerical optimization over θ gives the near-optimal choice $\theta = 3$ and yields the bound $X < M^{11/24} \approx M^{0.4583}$.

Improvement of MIHNP with 4 samples. Eliminating α via a Gröbner basis computation[6] yields the following modular equations:

$$\begin{cases} f_{ij}(x_i, x_j) = x_i x_j + a_{ij} x_i + b_{ij} x_j + c_{ij} \equiv 0 \pmod{M}, 1 \le i < j \le 4, \\ f_{1234}(x_1, x_2, x_3, x_4) = \displaystyle\sum_{\substack{0 \le i_1 + i_2 + i_3 + i_4 \le 3 \\ 0 \le i_1, i_2, i_3, i_4 \le 1}} d_{i_1 i_2 i_3 i_4} \, x_1^{i_1} x_2^{i_2} x_3^{i_3} x_4^{i_4} \equiv 0 \pmod{M^2}, \end{cases}$$

where $d_{0111} = 1$. We use the deglex monomial order with $x_1 \prec x_2 \prec x_3 \prec x_4$ and the polytope

$$P(\theta) = \left\{ \boldsymbol{x} \in \mathbb{R}^4 \;\middle|\; 0 \le x_i \le 1 \ (1 \le i \le 4), \quad \sum_{i=1}^{4} x_i \le \theta \right\}.$$

[6] The polynomial f_{1234} modulo M^2 is obtained from a Gröbner-basis computation for the ideal generated by all pairwise products of distinct polynomials f_{ij}.

With this choice, we obtain

$$\varphi_{P(\theta)}(\boldsymbol{x}) = \min\Big\{ \frac{x_1 + 2(x_2+x_3+x_4)}{3},\ \frac{1+x_2+x_3+x_4}{2},\\ x_2 + \frac{x_1+x_3+x_4}{2},\ x_3 + \frac{x_1+x_2+x_4}{2},\ x_4 + \frac{x_1+x_2+x_3}{2},\\ x_2+x_3+x_4,\ x_1+x_2+x_3,\ x_1+x_2+x_4,\ x_1+x_3+x_4 \Big\}.$$

By Theorem 4, we have

$$\prod_{j=1}^{4} X_j^{\int_{P(\theta)} x_j \,\mathrm{d}\boldsymbol{x}} < M^{\int_{P(\theta)} \varphi_{P(\theta)}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}}.$$

When $X_1 = X_2 = X_3 = X_4 = X$, numerical optimization over θ gives the near-optimal choice $\theta = 3.15$ and yields the bound $X < M^{0.5336}$.

Improvement of MIHNP with 5 samples. Eliminating α via a Gröbner basis computation yields the following modular equations:

$$\begin{cases} f_{ij}(x_i, x_j) = x_i x_j + a_{ij} x_i + b_{ij} x_j + c_{ij} \equiv 0 \pmod{M}, 1 \le i < j \le 5, \\ f_{ijkl}(x_i, x_j, x_k, x_l) = \displaystyle\sum_{\substack{0 \le i_1+i_2+i_3+i_4 \le 3 \\ 0 \le i_1, i_2, i_3, i_4 \le 1}} d^{(ijkl)}_{i_1 i_2 i_3 i_4} x_i^{i_1} x_j^{i_2} x_k^{i_3} x_l^{i_4} \equiv 0 \pmod{M^2}, \\ 1 \le i < j < k < l \le 5, \qquad (d^{(ijkl)}_{0111} = 1). \end{cases}$$

We use the deglex monomial order with $x_1 \prec x_2 \prec x_3 \prec x_4 \prec x_5$ and the polytope

$$P(\theta) = \left\{ \boldsymbol{x} \in \mathbb{R}^5 \;\middle|\; 0 \le x_i \le 1 \ (1 \le i \le 5), \quad \sum_{i=1}^{5} x_i \le \theta \right\}.$$

With this choice, we obtain

$$\varphi_{P(\theta)}(\boldsymbol{x}) = \min\Big\{ 1 + \frac{x_3+x_4+x_5}{2},\ \frac{x_1+2x_2+2x_3+2x_4+2x_5}{3},\\ x_2+x_3+x_4+x_5,\ \frac{1+x_2+2x_3+x_4+x_5}{2},\ \frac{1+x_2+x_3+2x_4+x_5}{2},\\ \frac{1+x_2+x_3+x_4+2x_5}{2},\ \frac{1+x_1+2x_2+2x_3+3x_4+3x_5}{4},\\ \frac{1+x_1+2x_2+3x_3+2x_4+3x_5}{4},\ \frac{1+x_1+2x_2+3x_3+3x_4+2x_5}{4},\\ \frac{x_1+x_2+x_3+2x_4+2x_5}{2},\ \frac{x_1+x_2+2x_3+x_4+2x_5}{2},\\ \frac{x_1+x_2+2x_3+2x_4+x_5}{2},\ \frac{x_1+2x_2+2x_3+x_4+x_5}{2},\\ \frac{x_1+2x_2+x_3+2x_4+x_5}{2},\ \frac{x_1+2x_2+x_3+x_4+2x_5}{2},\\ x_1+x_3+x_4+x_5,\ x_1+x_2+x_4+x_5,\ x_1+x_2+x_3+x_5,\ x_1+x_2+x_3+x_4 \Big\}.$$

By Theorem 4, we have

$$\prod_{j=1}^{5} X_j^{\int_{P(\theta)} x_j \,\mathrm{d}\boldsymbol{x}} < M^{\int_{P(\theta)} \varphi_{P(\theta)}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}}.$$

When $X_1 = X_2 = X_3 = X_4 = X_5 = X$, numerical optimization over θ gives the near-optimal choice $\theta = 3.40$ and yields the bound $X < M^{0.5794}$.

4.3 Elliptic Curve Hidden Number Problem

In [34], ECHNP can be formulated as the following modular equations:

$$f_i(x_1, x_i) = x_1^2 x_i + a_i x_1 x_i + b_i x_i + c_i x_1^2 + d_i x_1 + e_i \equiv 0 \pmod{M}, \quad 2 \le i \le n.$$

Improvement of ECHNP with 3 samples. A Gröbner basis computation yields the following modular equations:

$$\begin{cases} f_2(x_1, x_2) = x_1^2 x_2 + a_2 x_1 x_2 + b_2 x_2 + c_2 x_1^2 + d_2 x_1 + e_2 \equiv 0 \pmod{M}, \\ f_3(x_1, x_3) = x_1^2 x_3 + a_3 x_1 x_3 + b_3 x_3 + c_3 x_1^2 + d_3 x_1 + e_3 \equiv 0 \pmod{M}, \\ f_4(x_1, x_4) = x_1^2 x_4 + a_4 x_1 x_4 + b_4 x_4 + c_4 x_1^2 + d_4 x_1 + e_4 \equiv 0 \pmod{M}, \\ f_{23}(x_1, x_2, x_3) = x_2 x_3 + a_{23} x_1 x_3 + b_{23} x_1 x_2 + c_{23} x_3 + d_{23} x_2 \\ \qquad\qquad + e_{23} x_1 + h_{23} \equiv 0 \pmod{M}, \\ f_{24}(x_1, x_2, x_4) = x_2 x_4 + a_{24} x_1 x_4 + b_{24} x_1 x_2 + c_{24} x_4 + d_{24} x_2 \\ \qquad\qquad + e_{24} x_1 + h_{24} \equiv 0 \pmod{M}, \\ f_{34}(x_1, x_3, x_4) = x_3 x_4 + a_{34} x_1 x_4 + b_{34} x_1 x_3 + c_{34} x_4 + d_{34} x_3 \\ \qquad\qquad + e_{34} x_1 + h_{34} \equiv 0 \pmod{M}. \end{cases}$$

We use the deglex monomial order with $x_1 \prec x_2 \prec x_3 \prec x_4$ and the polytope

$$P(\boldsymbol{\theta}) = \left\{ \boldsymbol{x} \in \mathbb{R}^4 \;\middle|\; \begin{array}{ll} 0 \le x_2, x_3, x_4 \le 1, & 0 \le x_1 \le \theta_1, \\ x_1 + x_2 + x_3 \le \theta_2, & x_1 + x_2 + x_4 \le \theta_2, \\ x_1 + x_3 + x_4 \le \theta_2, & x_1 + x_2 + x_3 + x_4 \le \theta_3 \end{array} \right\}.$$

With this choice, we obtain

$$\begin{aligned} \varphi_{P(\boldsymbol{\theta})}(\boldsymbol{x}) = \min\Big\{ & \theta_1 - \frac{x_1}{2},\ x_2 + x_3 + x_4,\ \frac{\theta_2}{2}, \\ & \theta_2 - \frac{x_1}{2} - x_2,\ \theta_2 - \frac{x_1}{2} - x_3,\ \theta_2 - \frac{x_1}{2} - x_4,\ \frac{x_1}{4} + \frac{x_2 + x_3 + x_4}{2}, \\ & \theta_2 - \frac{x_1 + x_2 + x_3 + x_4}{2},\ \frac{x_1}{2} + x_2 + x_3,\ \frac{x_1}{2} + x_2 + x_4,\ \frac{x_1}{2} + x_3 + x_4, \\ & \frac{3\theta_2}{2} - x_1 - x_2 - x_3 - x_4,\ 2\theta_2 - \frac{3x_1}{2} - 2x_2 - x_3 - x_4, \\ & 2\theta_2 - \frac{3x_1}{2} - x_2 - 2x_3 - x_4,\ 2\theta_2 - \frac{3x_1}{2} - x_2 - x_3 - 2x_4 \Big\}. \end{aligned}$$

By Theorem 4, we have

$$\prod_{j=1}^{4} X_j^{\int_{P(\boldsymbol{\theta})} x_j \,\mathrm{d}\boldsymbol{x}} < M^{\int_{P(\boldsymbol{\theta})} \varphi_{P(\boldsymbol{\theta})}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}}.$$

When $X_1 = X_2 = X_3 = X_4 = X$, numerical optimization over $\boldsymbol{\theta}$ gives the near-optimal choice $(\theta_1, \theta_2, \theta_3) = (1.40, 2.32, 2.36)$, which yields the bound $X < M^{0.4071}$.

4.4 Linear Congruential Generator

In [41, Theorem 3], the authors consider the modular equations derived from k consecutive outputs of an LCG with unknown multiplier a:

$$f_i(x_i, x_{i+1}, x_{i+2}) =$$
$$x_{i+1}^2 - x_i x_{i+2} + a_i x_i + b_i x_{i+1} + c_i x_{i+2} + d_i \equiv 0 \pmod{M}, \qquad 1 \le i \le k-2.$$

Improvement of LCG in [41, Theorem 3]. We use the weighted deglex order induced by the weight vector $\boldsymbol{w} = (w_1, \ldots, w_k)$, where $w_j = j(k+1-j)$, together with the variable order $x_k \prec x_{k-1} \prec \cdots \prec x_1$, and the polytope

$$P = \left\{ \boldsymbol{x} \in \mathbb{R}^k \;\middle|\; x_i \ge 0 \ (1 \le i \le k), \quad \sum_{i=1}^{k} x_i \le 1 \right\}.$$

With this choice, we obtain

$$\varphi_P(\boldsymbol{x}) = \frac{1}{2} \sum_{j=2}^{k-1} x_j.$$

By Theorem 4, we have

$$\prod_{j=1}^{k} X_j^{\int_P x_j \,\mathrm{d}\boldsymbol{x}} < M^{\int_P \varphi_P(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x}} \quad \Longrightarrow \quad \prod_{j=1}^{k} X_j^{1/(k+1)!} < M^{(k-2)/(2(k+1)!)}.$$

When $X_1 = \cdots = X_k = X$, this gives $X < M^{\frac{1}{2} - \frac{1}{k}} \longrightarrow M^{\frac{1}{2}}$ as $k \to \infty$.

Remark 9. We use the weighted deglex order to ensure that

$$\mathrm{LM}(f_i(x_i, x_{i+1}, x_{i+2})) = x_{i+1}^2.$$

If one uses deglex, then the leading monomial is $x_i x_{i+2}$, which yields the upper bound $M^{1/3}$. Thus, the choice of monomial order can also affect the small-root bound.

Remark 10. In the LCG instance above, the corresponding dual feasible region Y of the linear program has only one vertex. Therefore, the function φ_P is globally affine on the whole polytope P. This example also shows that our method applies equally well to modular equations in which the number of equations varies.

4.5 Leveled Isogeny Problem with Hints for Unknown-Degree POKÉ

In [15, Theorem 4.3], the authors consider the LIPH instance for unknown-degree POKÉ and reduce it to the following modular equation:

$$f(x_1, x_2) = x_1^2 x_2 + a x_1 x_2 + b x_2 + c \equiv 0 \pmod{M},$$

with the bound $X_2 = M^{\frac{6}{17}}$.

Improvement of LIPH for unknown-degree POKÉ. We use the deglex monomial order with $x_2 \prec x_1$ and the polytope

$$P(\theta) = \left\{ (x_1, x_2) \in \mathbb{R}^2 \;\middle|\; x_1 \geq 0, \quad x_2 \geq 0, \quad x_2 \leq 1, \quad x_1 - 2x_2 \leq \theta \right\}.$$

With this choice, we obtain

$$\varphi_{P(\theta)}(\boldsymbol{x}) = \min\left\{ \frac{x_1}{2},\, x_2 \right\}.$$

By Theorem 4, we have

$$\prod_{j=1}^{2} X_j^{\int_{P(\theta)} x_j \, \mathrm{d}\boldsymbol{x}} < M^{\int_{P(\theta)} \varphi_{P(\theta)}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}} \quad \Longrightarrow \quad X_1^{\frac{2}{3}+\theta+\frac{\theta^2}{2}} X_2^{\frac{2}{3}+\frac{\theta}{2}} < M^{\frac{1}{3}+\frac{\theta}{2}}.$$

Choosing θ as a rational approximation to $(2\sqrt{223} - 10)/33$ maximizes the admissible exponent of X_1. This yields $X_1 < M^{\sqrt{223}/17 - 23/34} \approx M^{0.20195}$.

Remark 11. Detailed computations and derivations for the five examples discussed above are deferred to the Appendix E.

Remark 12. Appendix B considers additional cryptographic examples and proves global upper bounds for them within our method.

4.6 Experiments

We conducted several experiments to empirically assess Assumption 1. All experiments were performed on a MacBook Air equipped with an Apple M2 chip and 8 GB of memory. The computations were carried out in SageMath 10.8, using SageMath's built-in LLL implementation for lattice reduction. The results are reported in Table 2.

For each instance, we report the modulus size M, the root bound X, the chosen parameter m, the resulting lattice dimension dim, and the total runtime. In all cases, the correct solution was successfully recovered.

Finally, we used Algorithm 1 to verify the asymptotic bounds obtained for the five applications discussed above with the specified parameter choices. The runtime was less than one second in each case.

Remark 13. We observe that the LCG instances run significantly faster than the other examples, even when the lattice dimension is relatively large. This indicates that the corresponding lattices contain dense sublattices of substantially smaller dimension, consistent with the phenomenon discussed in [32].

Table 2. Experimental results for our method.

Setting	$\log_2 M$	$\log_2 X$	m	dim	Time (s)
CIHNP(CSURF)	256	29	2	16	0.01
	512	70	3	36	0.21
	1024	177	4	76	32.25
MIHNP(3 samples)	512	195	2	27	0.15
	256	102	3	64	4.19
	256	105	4	125	94.73
MIHNP(4 samples)	256	101	1	15	0.01
	512	230	2	76	62.32
MIHNP(5 samples)	512	225	1	26	0.22
ECHNP(3 samples)	512	96	1	11	0.01
	256	69	2	50	1.03
LCG (unknown multiplier a, consecutive values)	256	72	2 ($k = 6$)	28	0.01
	256	76	3 ($k = 8$)	165	3.18
LIPH (unknown degree)	256	27	2	12	0.01
	512	60	3	20	0.04
	1024	149	4	35	3.30

5 Conclusion

This paper brings tools from linear programming to the asymptotic analysis of the Automated Coppersmith method. We develop a provably correct and efficient algorithm for computing the asymptotic bounds achieved by this method. Applying the algorithm to five cryptographic settings, we obtain improved cryptanalytic bounds.

Acknowledgements. We would like to thank the anonymous reviewers of ASIACRYPT 2026 for their insightful suggestions. This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB0690200) and the Fundamental Research Funds for the Central Universities (Grant No. 3282024001).

References

1. Aardal, M.A., Aranha, D.F., Feng, Y., Gao, Y., Pan, Y.: Better bounds for finding fixed-degree isogenies via Coppersmith's method. In: Daemen, J., Thomé, E. (eds.) Advances in Cryptology – EUROCRYPT 2026. LNCS, vol. 16544, pp. 541–572. Springer, Cham (2026). https://doi.org/10.1007/978-3-032-25327-9_19
2. Baldoni, V., Berline, N., De Loera, J.A., Köppe, M., Vergne, M.: Computation of the highest coefficients of weighted Ehrhart quasi-polynomials of rational polyhedra. Foundations of Computational Mathematics **12**, 435–469 (2012). https://doi.org/10.1007/s10208-011-9106-4
3. Bauer, A., Joux, A.: Toward a rigorous variation of Coppersmith's algorithm on three variables. In: Naor, M. (ed.) Advances in Cryptology – EUROCRYPT 2007. LNCS, vol. 4515, pp. 361–378. Springer, Berlin, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72540-4_21

4. Bauer, A., Vergnaud, D., Zapalowicz, J.C.: Inferring sequences produced by non-linear pseudorandom number generators using Coppersmith's methods. In: Public Key Cryptography – PKC 2012. LNCS, vol. 7293, pp. 609–626. Springer (2012). https://doi.org/10.1007/978-3-642-30057-8_36
5. Benčina, B., Kutas, P., Merz, S.P., Petit, C., Stopar, M., Weitkämper, C.: Improved algorithms for finding fixed-degree isogenies between supersingular elliptic curves. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology – CRYPTO 2024, Part V. LNCS, vol. 14924, pp. 183–217. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68388-6_8
6. Bertsimas, D., Tsitsiklis, J.N.: Introduction to Linear Optimization. Athena Scientific Series in Optimization and Neural Computation, Athena Scientific, Belmont, MA (1997)
7. Blömer, J., May, A.: A tool kit for finding small roots of bivariate polynomials over the integers. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. LNCS, vol. 3494, pp. 251–267. Springer, Berlin, Heidelberg (2005). https://doi.org/10.1007/11426639_15
8. Bogart, T., Goodrick, J., Woods, K.: Parametric Presburger arithmetic: Logic, combinatorics, and quasi-polynomial behavior. Discrete Analysis **2017**(4), 1–34 (2017). https://doi.org/10.19086/da.1254
9. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$. In: Stern, J. (ed.) Advances in Cryptology – EUROCRYPT '99. LNCS, vol. 1592, pp. 1–11. Springer, Berlin, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_1
10. Boneh, D., Halevi, S., Howgrave-Graham, N.: The modular inversion hidden number problem. In: Boyd, C. (ed.) Advances in Cryptology – ASIACRYPT 2001. LNCS, vol. 2248, pp. 36–51. Springer, Berlin, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_3
11. Coppersmith, D.: Finding a small root of a bivariate integer equation; factoring with high bits known. In: Maurer, U.M. (ed.) Advances in Cryptology – EUROCRYPT '96. LNCS, vol. 1070, pp. 178–189. Springer, Berlin, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_16
12. Coppersmith, D.: Finding a small root of a univariate modular equation. In: Maurer, U.M. (ed.) Advances in Cryptology – EUROCRYPT '96. LNCS, vol. 1070, pp. 155–165. Springer, Berlin, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_14
13. Coppersmith, D.: Small solutions to polynomial equations, and low-exponent RSA vulnerabilities. Journal of Cryptology **10**(4), 233–260 (1997). https://doi.org/10.1007/s001459900030
14. Coron, J.S.: Finding small roots of bivariate integer polynomial equations revisited. In: Cachin, C., Camenisch, J.L. (eds.) Advances in Cryptology – EUROCRYPT 2004. LNCS, vol. 3027, pp. 492–505. Springer, Berlin, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_29
15. Das, S., Invernizzi, R., Kutas, P., Meers, J.: Leveled isogeny problems with hints. In: Bai, S., Persichetti, E. (eds.) Public-Key Cryptography – PKC 2026, Part III. LNCS, vol. 16553, pp. 31–57. Springer, Cham (2026). https://doi.org/10.1007/978-3-032-26737-5_2
16. Ding, Z., Dai, Z., Wu, B., Wang, R., Zhang, Y.: Coppersmith's method for solving modular inversion hidden number problem via determinant-based elimination. Cryptology ePrint Archive, Paper 2026/423 (2026), https://eprint.iacr.org/2026/423

17. Ernst, M., Jochemsz, E., May, A., de Weger, B.: Partial key exposure attacks on RSA up to full size exponents. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. LNCS, vol. 3494, pp. 371–386. Springer, Berlin, Heidelberg (2005). https://doi.org/10.1007/11426639_22
18. Feng, Y., Gao, Y., Hu, H., Nitaj, A., Pan, Y., Zheng, M.: Revisit unravelled linearization with erhart (quasi-)polynomial. Cryptology ePrint Archive, Paper 2026/244 (2026), https://eprint.iacr.org/2026/244
19. Feng, Y., Luo, H., Chen, Q., Nitaj, A., Pan, Y.: Computing asymptotic bounds for small roots in Coppersmith's method via sumset theory. In: Tauman Kalai, Y., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025, Part I. LNCS, vol. 16000, pp. 3–32. Springer, Cham (2025). https://doi.org/10.1007/978-3-032-01855-7_1
20. Feng, Y., Nitaj, A., Pan, Y., Zheng, M.: New asymptotic results on predicting polynomial congruential generators. Cryptology ePrint Archive, Paper 2025/2044 (2025), https://eprint.iacr.org/2025/2044
21. Herrmann, M., May, A.: Maximizing small root bounds by linearization and applications to small secret exponent RSA. In: Nguyen, P.Q., Pointcheval, D. (eds.) Public Key Cryptography – PKC 2010. LNCS, vol. 6056, pp. 53–69. Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_4
22. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Cryptography and Coding. LNCS, vol. 1355, pp. 131–142. Springer (1997). https://doi.org/10.1007/BFb0024458
23. Jochemsz, E., May, A.: A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. In: Lai, X., Chen, K. (eds.) Advances in Cryptology – ASIACRYPT 2006. LNCS, vol. 4284, pp. 267–282. Springer, Berlin, Heidelberg (2006). https://doi.org/10.1007/11935230_18
24. Jutla, C.S.: On finding small solutions of modular multivariate polynomial equations. In: Nyberg, K. (ed.) Advances in Cryptology – EUROCRYPT '98. LNCS, vol. 1403, pp. 158–170. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0054124
25. Kakvi, S.A., Kiltz, E., May, A.: Certifying RSA. In: Wang, X., Sako, K. (eds.) Advances in Cryptology – ASIACRYPT 2012. LNCS, vol. 7658, pp. 404–414. Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34961-4_25
26. Lenstra, A.K., Lenstra, Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen **261**(4), 515–534 (1982). https://doi.org/10.1007/BF01457454
27. May, A.: New RSA Vulnerabilities Using Lattice Reduction Methods. Ph.D. thesis, University of Paderborn, Paderborn, Germany (2003)
28. May, A.: Using LLL-reduction for solving RSA and factorization problems. In: Nguyen, P.Q., Vallée, B. (eds.) The LLL Algorithm: Survey and Applications, pp. 315–348. Information Security and Cryptography, Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-02295-1_10
29. Meers, J., Nowakowski, J.: Solving the hidden number problem for CSIDH and CSURF via automated Coppersmith. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023, Part IV. LNCS, vol. 14441, pp. 39–71. Springer, Cham (2023). https://doi.org/10.1007/978-981-99-8730-6_2
30. Papadimitriou, C.H.: On the complexity of integer programming. Journal of the ACM **28**(4), 765–768 (1981). https://doi.org/10.1145/322276.322287
31. Peng, L., Hu, L., Lu, Y., Xu, J., Huang, Z.: Cryptanalysis of dual RSA. Designs, Codes and Cryptography **83**(1), 1–21 (2017). https://doi.org/10.1007/s10623-016-0196-5

32. Ryan, K.: Solving multivariate Coppersmith problems with known moduli. In: Fehr, S., Fouque, P.A. (eds.) Advances in Cryptology – EUROCRYPT 2025, Part VI. LNCS, vol. 15606, pp. 355–384. Springer, Cham (2025). https://doi.org/10.1007/978-3-031-91095-1_13
33. Sarkar, S.: Enhanced bound for the commutative isogeny hidden number problem in CSURF. In: Mukhopadhyay, S., Stănică, P. (eds.) Progress in Cryptology – INDOCRYPT 2024, Part II. LNCS, vol. 15496, pp. 201–211. Springer, Cham (2025). https://doi.org/10.1007/978-3-031-80311-6_10
34. Shani, B.: On the bit security of elliptic curve Diffie–Hellman. In: Fehr, S. (ed.) Public-Key Cryptography – PKC 2017, Part I. LNCS, vol. 10174, pp. 361–387. Springer, Cham (2017). https://doi.org/10.1007/978-3-662-54365-8_15
35. Takayasu, A., Lu, Y., Peng, L.: Small CRT-exponent RSA revisited. Journal of Cryptology **32**(4), 1337–1382 (2019). https://doi.org/10.1007/s00145-018-9282-3
36. Vanderbei, R.J.: Linear Programming: Foundations and Extensions. Springer, 4 edn. (2014)
37. Xu, J., Sarkar, S., Hu, L., Wang, H., Pan, Y.: New results on modular inversion hidden number problem and inversive congruential generator. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 297–321. Springer, Berlin, Heidelberg (2019). https://doi.org/10.1007/978-3-030-26948-7_11
38. Xu, J., Sarkar, S., Hu, L., Wang, H., Pan, Y.: Revisiting modular inversion hidden number problem and its applications. IEEE Transactions on Information Theory **69**(8), 5337–5356 (2023). https://doi.org/10.1109/TIT.2023.3263485
39. Xu, J., Sarkar, S., Wang, H., Hu, L.: Improving bounds on elliptic curve hidden number problem for ECDH key exchange. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022, Part III. LNCS, vol. 13793, pp. 771–799. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22969-5_26
40. Xu, J., Sarkar, S., Wang, H., Hu, L.: New results on elliptic curve hidden number problem for ECDH key exchange. Journal of Cryptology **39**(2), 20 (2026). https://doi.org/10.1007/s00145-026-09577-6
41. Zhang, R., Bi, J., Li, L., Peng, H.: Cryptanalysis on two kinds of number theoretic pseudo-random generators using Coppersmith method. IET Information Security **2025**, 1–15 (2025). https://doi.org/10.1049/ise2/5569393
42. Zheng, M., Hu, H., Wang, Z.: Generalized cryptanalysis of RSA with small public exponent. Science China Information Sciences **59**(3), 032108:1–032108:10 (2016). https://doi.org/10.1007/s11432-015-5325-7

A $p_0(m)$ is an eventual quasi-polynomial

To prove that $p_0(m)$ is an eventual quasi-polynomial, we use a counting result for parametric Presburger families due to Bogart et al. [8].

Definition 11 (Parametric Presburger Family [8, Def 1.3]). *Given $d \in \mathbb{N}$, a parametric Presburger family is a collection $\{S_t \mid t \in \mathbb{N}\}$ of subsets of $\mathbb{Z}^d$ that can be defined by a formula using addition, inequalities, multiplication and addition by constants from $\mathbb{Z}$, Boolean operations, multiplication by t, and quantifiers over variables ranging over $\mathbb{Z}$. Equivalently, such families are defined using quantifiers and Boolean combinations of formulas of the form $a(t)\cdot x \le b(t)$, where $a(t) \in \mathbb{Z}[t]^d$ and $b(t) \in \mathbb{Z}[t]$.*

Remark 14. The following simple examples are parametric Presburger families:

1. $S_t = \{x \in \mathbb{Z} \mid 0 \le x \le t\}$.
2. $S_t = \{(x,y) \in \mathbb{Z}^2 \mid 0 \le x \le t,\ 0 \le y \le 2t,\ x+y \le 3t\}$.
3. $S_t = \{(x,y) \in \mathbb{Z}^2 \mid x \ge 0,\ y \ge 0,\ tx + y \le t^2 + 1\}$.
4. $S_t = \{x \in \mathbb{Z} \mid 0 \le x \le t, \text{ or } 2t \le x \le 3t\}$.
5. $S_t = \{x \in \mathbb{Z} \mid 0 \le x \le t,\ \exists q \in \mathbb{Z},\ x = 2q\}$.

The first three examples are defined by linear inequalities whose coefficients and constant terms depend polynomially on t; the fourth uses a Boolean operation, and the fifth uses a quantified integer variable. In contrast, $xy \le t$ is not of the permitted linear form.

Lemma 10 ([8, Prop 2 and Thm 1.15]). *Let $\{S_t \mid t \in \mathbb{N}\}$ be a parametric Presburger family with $S_t \subseteq \mathbb{Z}^d$. Then there exists an eventual quasi-polynomial $g : \mathbb{N} \to \mathbb{N}$ such that, if S_t has finite cardinality, then $g(t) = |S_t|$.*

Corollary 4. *$p_0(m)$ is an eventual quasi-polynomial.*

Proof. We retain the notation from the preceding section. Define

$$\Omega_m := \left\{ (\boldsymbol{\beta}, q) \in \mathbb{Z}^k \times \mathbb{Z}_{\ge 0} \;\middle|\; \begin{array}{l} \boldsymbol{\beta} \in \mathcal{B}_m, \\ \exists\, \boldsymbol{\ell} \in \mathbb{Z}^n_{\ge 0}, \text{ such that } q+1 \le \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle, \\ \langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle + \sum_{j=1}^n \ell_j \Delta_t(f_j) \le m b_t,\ \forall t \in T \end{array} \right\}.$$

We first show that $\{\Omega_m\}_{m \ge 1}$ is a parametric Presburger family. Indeed, $\Omega_m \subseteq \mathbb{Z}^{k+1}$, with free variables $(\boldsymbol{\beta}, q)$ and quantified auxiliary variable $\boldsymbol{\ell}$. The condition $\boldsymbol{\beta} \in \mathcal{B}_m = mP \cap \mathbb{Z}^k$ is equivalent to the finitely many linear inequalities $\langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle \le m b_t$, $t \in T$. The remaining conditions $q \ge 0$, $\ell_j \ge 0$, $q+1 \le \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle$, and

$$\langle \boldsymbol{a}_t, \boldsymbol{\beta} \rangle + \sum_{j=1}^n \ell_j \Delta_t(f_j) \le m b_t, \quad \forall t \in T$$

are also linear inequalities in the integer variables $\boldsymbol{\beta}, q, \boldsymbol{\ell}$. Since there are only finitely many such inequalities and all coefficients are rational, we may clear denominators to obtain integer-coefficient linear inequalities. Hence $\{\Omega_m\}_{m \ge 1}$ is a parametric Presburger family.

Next, we prove that Ω_m is finite. Since P is bounded, $\mathcal{B}_m = mP \cap \mathbb{Z}^k$ is finite. Fix $\boldsymbol{\beta} \in \mathcal{B}_m$. If $(\boldsymbol{\beta}, q) \in \Omega_m$, then there exists $\boldsymbol{\ell} \in \mathbb{Z}^n_{\ge 0}$ satisfying the feasibility inequalities above. Dividing these inequalities by m, we see that $\boldsymbol{\ell}/m$ is feasible for the linear program defining $\varphi_P(\boldsymbol{\beta}/m)$. By Lemma 9, $\varphi_P(\boldsymbol{\beta}/m)$ has a finite optimal value, and hence $\langle \boldsymbol{e}, \boldsymbol{\ell} \rangle \le m \varphi_P(\boldsymbol{\beta}/m)$. Since $q+1 \le \langle \boldsymbol{e}, \boldsymbol{\ell} \rangle$, we have $0 \le q < m\varphi_P(\boldsymbol{\beta}/m)$. Thus, for each fixed $\boldsymbol{\beta}$, there are only finitely many possible values of q. Since $\boldsymbol{\beta}$ ranges over the finite set $\mathcal{B}_m$, the set Ω_m is finite.

It remains to identify its cardinality. For fixed $\boldsymbol{\beta} \in \mathcal{B}_m$ and $q \in \mathbb{Z}_{\ge 0}$, the definition of $r_m(x^{\boldsymbol{\beta}})$ gives

$$(\boldsymbol{\beta}, q) \in \Omega_m \iff q+1 \le r_m(x^{\boldsymbol{\beta}}) \iff 0 \le q < r_m(x^{\boldsymbol{\beta}}).$$

Hence, for each fixed $\boldsymbol{\beta}$, the number of admissible integers q is exactly $r_m(x^{\boldsymbol{\beta}})$. Therefore

$$|\Omega_m| = \sum_{\boldsymbol{\beta} \in \mathcal{B}_m} r_m(x^{\boldsymbol{\beta}}) = p_0(m).$$

Thus $\{\Omega_m\}_{m \geq 1}$ is a parametric Presburger family, and each Ω_m is finite. By Lemma 10, the function $m \mapsto |\Omega_m|$ is an eventual quasi-polynomial. Since $p_0(m) = |\Omega_m|$, the function $p_0(m)$ is an eventual quasi-polynomial. □

B Further Results on Applications

B.1 The FESTA Instance of LIPH

In [15, Proposition 5.13], the authors consider a FESTA instance of the Leveled Isogeny Problem with Hints, which can be formulated as the following modular congruence:

$$f(x, y) = xy + ax + by + c \equiv 0 \pmod{M}.$$

Using the Automated Coppersmith method, the authors show that this problem can be solved when the small-root bounds satisfy $X = Y < M^{1/3-\varepsilon}$ for sufficiently small $\varepsilon > 0$. In fact, we prove that, in the setting considered in this paper, the corresponding bound cannot exceed $M^{1/3-\varepsilon}$. More precisely, we obtain the following theorem.

Theorem 5. *Given a modulus $M \in \mathbb{N}$ and a polynomial*

$$f(x, y) = xy + ax + by + c.$$

Suppose that the asymptotic bound is of the form $X = Y = M^{\delta-\varepsilon}$ for sufficiently small $\varepsilon > 0$. Then, for any full-dimensional rational polytope $P \subseteq \mathbb{R}^2_{\geq 0}$, with the corresponding monomial set $\mathcal{M}_m = \{x^i y^j \mid (i, j) \in mP \cap \mathbb{Z}^2\}$, the Automated Coppersmith method necessarily yields

$$\delta \leq \frac{1}{3}.$$

Moreover, this upper bound is achievable.

Proof. For any monomial order, we have $\mathrm{LM}(f) = xy$, and

$$A(f) = \{(0, 0), (1, 0), (0, 1), (1, 1)\}.$$

Hence the linear programming function in our method can be written as[7]

$$\varphi_P(x, y) = \max \left\{ t \geq 0 \;\middle|\; (x, y) - [0, t]^2 \subseteq P \right\}.$$

[7] It suffices to transform the corresponding vector inequalities into equivalent inclusion relations.

Therefore, by Theorem 4, it suffices to prove that

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\,\mathrm{d}y \le \frac{1}{3}\int_P (x+y)\,\mathrm{d}x\,\mathrm{d}y.$$

For each $t \ge 0$, define $P_t = \{(x,y) \in P \mid \varphi_P(x,y) \ge t\}$. Fix such a t. We decompose P_t along the diagonal as

$$P_t^+ = P_t \cap \{x \ge y\}, \qquad P_t^- = P_t \cap \{x < y\}.$$

First consider P_t^+. If $(x,y) \in P_t^+$, then $\varphi_P(x,y) \ge t$, and hence $(x,y) - [0,t]^2 \subseteq P$. In particular, $(x, y-t) \in P$. Since $x \ge y$, we have $x - (y-t) = x - y + t \ge t$. Thus the translation map $T_+(x,y) = (x, y-t)$ sends P_t^+ into $P \cap \{x - y \ge t\}$. Since this map preserves area, we obtain

$$|P_t^+| \le |P \cap \{x - y \ge t\}|.$$

Similarly, we have $|P_t^-| \le |P \cap \{y - x \ge t\}|$. Adding these two estimates and noting that the two regions are disjoint, we get

$$\begin{aligned} |P_t| &= |P_t^+| + |P_t^-| \\ &\le |P \cap \{x - y \ge t\}| + |P \cap \{y - x \ge t\}| \\ &\le |P \cap \{|x-y| \ge t\}|. \end{aligned}$$

Moreover, since $\varphi_P(x,y) \ge 0$, the layer-cake representation gives

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int_0^\infty |P_t|\,\mathrm{d}t, \qquad \int_P |x-y|\,\mathrm{d}x\,\mathrm{d}y = \int_0^\infty |P \cap \{|x-y| \ge t\}|\,\mathrm{d}t.$$

Therefore,

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\,\mathrm{d}y \le \int_P |x-y|\,\mathrm{d}x\,\mathrm{d}y. \tag{9}$$

On the other hand, since $P \subseteq \mathbb{R}^2_{\ge 0}$, if $(x,y) - [0,t]^2 \subseteq P$, then necessarily $x - t \ge 0$ and $y - t \ge 0$. Hence $\varphi_P(x,y) \le \min(x,y)$. Therefore,

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\,\mathrm{d}y \le \int_P \min(x,y)\,\mathrm{d}x\,\mathrm{d}y. \tag{10}$$

Combining (9) and (10), we obtain

$$\begin{aligned} 3\int_P \varphi_P(x,y)\,\mathrm{d}x\,\mathrm{d}y &\le 2\int_P \min(x,y)\,\mathrm{d}x\,\mathrm{d}y + \int_P |x-y|\,\mathrm{d}x\,\mathrm{d}y \\ &= \int_P \big(2\min(x,y) + |x-y|\big)\,\mathrm{d}x\,\mathrm{d}y \\ &= \int_P (x+y)\,\mathrm{d}x\,\mathrm{d}y. \end{aligned}$$

Finally, when $P = [0,1]^2$, this upper bound is achieved. □

Remark 15. In fact, the class of monomial sets considered in this theorem is already fairly general. Although the theorem does not cover all possible monomial sets, there is currently no compelling reason to believe that a specially structured monomial set could yield an asymptotic bound exceeding 1/3.

B.2 Nonlinear Generators

In [4, Theorem 1], the authors consider the problem of recovering a known nonlinear number-theoretic pseudorandom generator from two consecutive truncated outputs. This problem can be formulated as the following modular congruence:

$$f(x,y) = a_d x^d + a_{d-1}x^{d-1} + \cdots + a_1 x + a_0 - y \equiv 0 \pmod{M}.$$

The authors show that this problem can be solved when the small-root bounds satisfy $X = Y < M^{1/(d+1)-\varepsilon}$ for sufficiently small $\varepsilon > 0$. We obtain the following more general theorem.

Theorem 6. *For any $d \geq 2$, given a modulus $M \in \mathbb{N}$ and a polynomial*

$$f(x,y) = a_d x^d + a_{d-1}x^{d-1} + \cdots + a_1 x + a_0 - y.$$

Suppose that the asymptotic bound is of the form $X = Y = M^{\delta-\varepsilon}$ for sufficiently small $\varepsilon > 0$. Then, for any full-dimensional rational polytope $P \subseteq \mathbb{R}^2_{\geq 0}$, with the corresponding monomial set $\mathcal{M}_m = \{x^i y^j \mid (i,j) \in mP \cap \mathbb{Z}^2\}$, the Automated Coppersmith method necessarily yields

$$\delta \leq \frac{1}{d+1}.$$

Moreover, this upper bound is achievable.

Proof. The exponent set of the polynomial f is

$$A(f) = \{(0,0),(1,0),(2,0),\ldots,(d,0),(0,1)\},$$

and the leading monomial of f has only two possible forms: $\mathrm{LM}(f) = x^d$ or $\mathrm{LM}(f) = y$.

We first consider the case $\mathrm{LM}(f) = x^d$. The linear programming function associated with our method can be written as

$$\varphi_P(x,y) = \max\Big\{ t \geq 0 \;\Big|\; (x,y) + t\big((i,0)-(d,0)\big) \in P \text{ for all } (i,0) \in A(f), \\ (x-dt, y+t) \in P\Big\}.$$

It suffices to prove that

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y \leq \frac{1}{d+1}\int_P (x+y)\,\mathrm{d}x\mathrm{d}y.$$

For each $t \geq 0$, define $P_t = \{(x,y) \in P \mid \varphi_P(x,y) \geq t\}$. If $(x,y) \in P_t$, then, by the definition of $\varphi_P(x,y)$, we have $(x-dt, y+t) \in P \subseteq \mathbb{R}^2_{\geq 0}$. Hence $x - dt \geq 0$, i.e., $x \geq dt$. Therefore, $P_t \subseteq P \cap \{x \geq dt\}$. By the layer-cake representation, we obtain

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y = \int_0^\infty |P_t|\,\mathrm{d}t \leq \int_0^\infty |P \cap \{x \geq dt\}|\,\mathrm{d}t = \frac{1}{d}\int_P x\,\mathrm{d}x\mathrm{d}y.$$

On the other hand, the translation $(x, y) \longmapsto (x-dt, y+t)$ maps P_t into $P \cap \{y \geq t\}$ and preserves area. Thus $|P_t| \leq |P \cap \{y \geq t\}|$. Hence

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y = \int_0^\infty |P_t|\,\mathrm{d}t \leq \int_0^\infty |P \cap \{y \geq t\}|\,\mathrm{d}t = \int_P y\,\mathrm{d}x\mathrm{d}y.$$

Combining these two estimates yields

$$\begin{aligned}\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y &= \frac{d}{d+1}\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y + \frac{1}{d+1}\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y \\ &\leq \frac{d}{d+1}\cdot\frac{1}{d}\int_P x\,\mathrm{d}x\mathrm{d}y + \frac{1}{d+1}\int_P y\,\mathrm{d}x\mathrm{d}y \\ &= \frac{1}{d+1}\int_P (x+y)\,\mathrm{d}x\mathrm{d}y.\end{aligned}$$

Next, we consider the case $\mathrm{LM}(f) = y$. The linear-programming function is then given by

$$\varphi_P(x,y) = \max\left\{t \geq 0 \;\middle|\; (x,y) + t\big((i,0)-(0,1)\big) \in P \text{ for all } (i,0) \in A(f)\right\}.$$

Thus, if $(x, y) \in P_t$, then $(x + dt, y - t) \in P \subseteq \mathbb{R}^2_{\geq 0}$. It follows that $y - t \geq 0$, and hence $P_t \subseteq P \cap \{y \geq t\}$. Therefore,

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y = \int_0^\infty |P_t|\,\mathrm{d}t \leq \int_0^\infty |P \cap \{y \geq t\}|\,\mathrm{d}t = \int_P y\,\mathrm{d}x\mathrm{d}y.$$

On the other hand, the translation $(x, y) \longmapsto (x+dt, y-t)$ maps P_t into $P \cap \{x \geq dt\}$ and preserves area. Thus $|P_t| \leq |P \cap \{x \geq dt\}|$. Hence

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y \leq \int_0^\infty |P \cap \{x \geq dt\}|\,\mathrm{d}t = \frac{1}{d}\int_P x\,\mathrm{d}x\mathrm{d}y.$$

Consequently, in this case we also have

$$\int_P \varphi_P(x,y)\,\mathrm{d}x\mathrm{d}y \leq \frac{1}{d+1}\int_P (x+y)\,\mathrm{d}x\mathrm{d}y.$$

Finally, the upper bound is achievable by $P = \left\{(x,y) \in \mathbb{R}^2_{\geq 0} \mid \frac{x}{d} + y \leq 1\right\}$. □

B.3 Multivariate Polynomials

A recurring phenomenon in applications of Coppersmith's method is that a single modular congruence of total degree d does not appear to yield a symmetric asymptotic small-root bound beyond $M^{1/d}$. We rigorously prove this barrier within our method.

Theorem 7. *For any $d \geq 1$, given a modulus $M \in \mathbb{N}$ and a nonconstant monic polynomial*

$$f(x_1, \ldots, x_k) \in \mathbb{Z}[x_1, \ldots, x_k]$$

of total degree d with $0 \in A(f)$. Suppose that the asymptotic bound is of the form $X_1 = \cdots = X_k = M^{\delta-\varepsilon}$ for sufficiently small $\varepsilon > 0$. Then, for any full-dimensional rational polytope $P \subseteq \mathbb{R}^k_{\geq 0}$, with the corresponding monomial set $\mathcal{M}_m = \{x_1^{i_1} \cdots x_k^{i_k} \mid (i_1, \ldots, i_k) \in mP \cap \mathbb{Z}^k\}$, the Automated Coppersmith method necessarily yields

$$\delta \leq \frac{1}{d}.$$

Proof. Write $\mathrm{LM}(f) = \boldsymbol{x}^{\boldsymbol{\alpha}}$. The linear-programming function is

$$\varphi_P(\boldsymbol{x}) = \max\{t \geq 0 \mid \boldsymbol{x} + t(\boldsymbol{u} - \boldsymbol{\alpha}) \in P \text{ for every } \boldsymbol{u} \in A(f)\}.$$

By Theorem 4, it is enough to prove

$$\int_P \varphi_P(\boldsymbol{x})\, d\boldsymbol{x} \leq \frac{1}{d} \int_P |\boldsymbol{x}|\, d\boldsymbol{x}.$$

Since $\deg(f) = d$, there exists $\boldsymbol{\gamma} \in A(f)$ such that $|\boldsymbol{\gamma}| = d$. For $t \geq 0$, define $P_t = \{\boldsymbol{x} \in P \mid \varphi_P(\boldsymbol{x}) \geq t\}$. If $\boldsymbol{x} \in P_t$, then the choices $\boldsymbol{u} = \boldsymbol{0}$ and $\boldsymbol{u} = \boldsymbol{\gamma}$ give $\boldsymbol{x} - t\boldsymbol{\alpha} \in P \subseteq \mathbb{R}^k_{\geq 0}$ and $\boldsymbol{x} + t(\boldsymbol{\gamma} - \boldsymbol{\alpha}) \in P$. Hence $|\boldsymbol{x}| \geq t|\boldsymbol{\alpha}|$. Consider the translation $T_t(\boldsymbol{x}) = \boldsymbol{x} + t(\boldsymbol{\gamma} - \boldsymbol{\alpha})$. For $\boldsymbol{x} \in P_t$, we have $T_t(\boldsymbol{x}) \in P$, and

$$|T_t(\boldsymbol{x})| = |\boldsymbol{x}| + t(|\boldsymbol{\gamma}| - |\boldsymbol{\alpha}|) \geq t|\boldsymbol{\alpha}| + t(d - |\boldsymbol{\alpha}|) = dt.$$

Thus

$$T_t(P_t) \subseteq P \cap \{\boldsymbol{z} \in \mathbb{R}^k \mid |\boldsymbol{z}| \geq dt\}.$$

Since T_t preserves volume, we have $\mathrm{vol}(P_t) \leq \mathrm{vol}\big(P \cap \{\boldsymbol{x} \in \mathbb{R}^k \mid |\boldsymbol{x}| \geq dt\}\big)$. Therefore, by the layer-cake representation,

$$\begin{aligned}
\int_P \varphi_P(\boldsymbol{x})\, d\boldsymbol{x} &= \int_0^\infty \mathrm{vol}(P_t)\, dt \\
&\leq \int_0^\infty \mathrm{vol}\big(P \cap \{\boldsymbol{x} \in \mathbb{R}^k \mid |\boldsymbol{x}| \geq dt\}\big)\, dt \\
&= \frac{1}{d} \int_0^\infty \mathrm{vol}\big(P \cap \{\boldsymbol{x} \in \mathbb{R}^k \mid |\boldsymbol{x}| \geq t\}\big)\, dt \\
&= \frac{1}{d} \int_P |\boldsymbol{x}|\, d\boldsymbol{x}.
\end{aligned}$$

This proves the claim. □

C More Experiments

We now explain why the heuristic polynomial-interpolation approach used in prior works [29, 32] for determining asymptotic coefficients is no longer feasible in the setting considered in this paper. Consider the case in which p_0 is a quasi-polynomial with minimal period T. For example, consider the polynomial pair

corresponding to Exp. 4 in Table 4. Set $e = (1, 1)$ and $P = [0, 1]^2$. If one directly applies interpolation to compute p_0, then the period of the quasi-polynomial must be known in advance. Otherwise, simply interpolating from consecutive values $m, m + 1, m + 2, m + 3$ may lead to unstable and misleading results.

Even if the minimal period can be determined by some additional method, it is already $T = 52$. Therefore, to interpolate within a fixed residue class, one needs to compute the values at $m = 1, 1+T, 1+2T, 1+3T$, which means that m must be taken as large as 157. For this value of m, the corresponding monomial set has size $|\mathcal{M}_m| = 24964$. Since the Automated Coppersmith method has to enumerate each candidate monomial individually, its running time reaches 35 minutes. In contrast, the asymptotic-bound computation method proposed in this paper avoids such large-scale enumeration and completes the computation for the same example in less than one second.

We report additional results in Table 3. Throughout, we set $e = (1, 1)$ and $P = [0, 1]^2$, and use the deglex monomial order with $y \prec x$. Here, T denotes the period of the corresponding quasi-polynomial p_0.

Table 3. Comparison of running times.

Experiment	T	**Interpolation**	**Ours**
Exp. 1	12	4.5 s	0.06 s
Exp. 2	20	39.1 s	0.02 s
Exp. 3	30	134.3 s	0.02 s
Exp. 4	52	2102.1 s	0.04 s

Table 4. Examples of polynomial supports.

	$\mathrm{supp}(f)$
Exp. 1	$\begin{cases} f_1 \text{ with } \mathrm{supp}(f_1) = \{xy^3,\, y^3,\, xy^2,\, xy,\, y,\, 1\}, \\ f_2 \text{ with } \mathrm{supp}(f_2) = \{x^3y,\, x^3,\, x^2y,\, xy,\, x,\, y,\, 1\}. \end{cases}$
Exp. 2	$\begin{cases} f_1 \text{ with } \mathrm{supp}(f_1) = \{x^4,\, xy,\, y,\, 1\}, \\ f_2 \text{ with } \mathrm{supp}(f_2) = \{y^5,\, xy,\, x,\, 1\}. \end{cases}$
Exp. 3	$\begin{cases} f_1 \text{ with } \mathrm{supp}(f_1) = \{xy^5,\, x,\, y,\, 1\}, \\ f_2 \text{ with } \mathrm{supp}(f_2) = \{x^5y,\, x,\, y,\, 1\}. \end{cases}$
Exp. 4	$\begin{cases} f_1 \text{ with } \mathrm{supp}(f_1) = \{xy^4,\, xy^3,\, y^4,\, xy^2,\, y^2,\, x,\, y,\, 1\}, \\ f_2 \text{ with } \mathrm{supp}(f_2) = \{x^4y^3,\, x^3y^3,\, x^4y^2,\, x^2y^3,\, x^3y,\, xy^2,\, x^2,\, y^3,\, 1\}. \end{cases}$

D Algorithm

Algorithm 2: Automated Coppersmith Method for (3)

Input: A monomial set $\mathcal{M}_m$, a monomial order $\prec$, polynomials $f_1, \ldots, f_n$, exponents $e = (e_1, \ldots, e_n)^\top$, and a modulus M.

Output: An $(\mathcal{M}_m, \prec)$-suitable set of shift polynomials $\mathcal{G}_m$.

$\mathcal{G}_m \leftarrow \varnothing$;
foreach $\lambda \in \mathcal{M}_m$ **do**
  $S_m(\lambda) \leftarrow \varnothing$;
  foreach $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_n) \in \mathbb{Z}_{\geq 0}^n$ *such that* $\prod_{j=1}^n \mathrm{LM}(f_j)^{\ell_j} \mid \lambda$ **do**
    if $\mathrm{supp}(F_{[\lambda,\boldsymbol{\ell}]}) \subseteq \mathcal{M}_m$ **then**
      $S_m(\lambda) \leftarrow S_m(\lambda) \cup \{\boldsymbol{\ell}\}$;
  Choose $\boldsymbol{\ell}_\lambda \in S_m(\lambda)$ such that $\langle e, \boldsymbol{\ell}_\lambda \rangle = \max_{\boldsymbol{\ell} \in S_m(\lambda)} \langle e, \boldsymbol{\ell} \rangle$;
  $r_m(\lambda) \leftarrow \langle e, \boldsymbol{\ell}_\lambda \rangle$;
$t_m \leftarrow \max_{\lambda \in \mathcal{M}_m} r_m(\lambda)$;
foreach $\lambda \in \mathcal{M}_m$ **do**
  $g_\lambda(x) \leftarrow M^{t_m - r_m(\lambda)} F_{[\lambda,\boldsymbol{\ell}_\lambda]}(x)$;
  $\mathcal{G}_m \leftarrow \mathcal{G}_m \cup \{g_\lambda(x)\}$;
return $\mathcal{G}_m$;

Example 1. Let $f(x) = x + 1$, with $\mathrm{LM}(f) = x$. Take $\mathcal{M}_m = \{1, x, x^2\}$ and $e = (1)$. Then we have $t_m = 2$ and the corresponding values are

λ	$S_m(\lambda)$	ℓ_λ	$r_m(\lambda)$	$g_\lambda(x)$
1	$\{0\}$	0	0	M^2
x	$\{0,1\}$	1	1	$Mf(x)$
x^2	$\{0,1,2\}$	2	2	$f(x)^2$

Hence,

$$\mathcal{G}_m = \{M^2,\ Mf(x),\ f(x)^2\}.$$

Then the associated lattice basis matrix is

$$\boldsymbol{B} = \begin{pmatrix} M^2 & 0 & 0 \\ M & MX & 0 \\ 1 & 2X & X^2 \end{pmatrix}.$$

E More Details for Section 4

E.1 Polytope Construction Details

For each example in Section 4, we provide the inequality descriptions of the Newton-type polytopes $P_\cup$ and P_Σ used in the heuristic polytope-selection step.

CIHNP-CSURF.

$$P_\cup = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \;\middle|\; \begin{array}{ll} x_1, x_2, x_3 \geq 0, & x_2 + x_3 \leq 2, \\ 2x_1 + x_2 \leq 4, & 2x_1 + 3x_2 + 4x_3 \leq 8 \end{array} \right\},$$

$$P_\Sigma = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \;\middle|\; \begin{array}{lll} x_1, x_2, x_3 \geq 0, & x_2 \leq 2, & x_3 \leq 2, \\ 2x_1 + x_2 \leq 8, & 2x_1 + x_2 + 4x_3 \leq 12 \end{array} \right\}.$$

MIHNP with 3 samples.

$$P_\cup = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \;\middle|\; \begin{array}{ll} x_1, x_2, x_3 \geq 0, & x_1, x_2, x_3 \leq 1, \\ x_1 + x_2 + x_3 \leq 2 & \end{array} \right\},$$

$$P_\Sigma = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \;\middle|\; x_1, x_2, x_3 \geq 0, \quad x_1, x_2, x_3 \leq 2 \right\}.$$

MIHNP with 4 samples.

$$P_\cup = \left\{ \boldsymbol{x} \in \mathbb{R}^4 \;\middle|\; \begin{array}{ll} x_1, x_2, x_3, x_4 \geq 0, & x_1, x_2, x_3, x_4 \leq 1, \\ x_1 + x_2 + x_3 + x_4 \leq 3 & \end{array} \right\},$$

$$P_\Sigma = \left\{ \boldsymbol{x} \in \mathbb{R}^4 \;\middle|\; \begin{array}{ll} x_1, x_2, x_3, x_4 \geq 0, & x_1, x_2, x_3, x_4 \leq 4, \\ x_1 + x_2 + x_3 + x_4 \leq 15 & \end{array} \right\}.$$

MIHNP with 5 samples.

$$P_\cup = \left\{ \boldsymbol{x} \in \mathbb{R}^5 \;\middle|\; \begin{array}{ll} x_1, x_2, x_3, x_4, x_5 \geq 0, & x_1, x_2, x_3, x_4, x_5 \leq 1, \\ x_1 + x_2 + x_3 + x_4 + x_5 \leq 3 & \end{array} \right\},$$

$$P_\Sigma = \left\{ \boldsymbol{x} \in \mathbb{R}^5 \;\middle|\; \begin{array}{ll} x_i \geq 0, \quad x_i \leq 8 \quad (1 \leq i \leq 5), & \\ \sum_{i=1}^5 x_i \leq 35, & \sum_{j \neq i} x_j \leq 31 \quad (1 \leq i \leq 5) \end{array} \right\}.$$

ECHNP with 3 samples.

$$P_\cup = \left\{ \boldsymbol{x} \in \mathbb{R}^4 \;\middle|\; \begin{array}{ll} x_1, x_2, x_3, x_4 \geq 0, & x_1 \leq 2, \quad x_2, x_3, x_4 \leq 1, \\ x_1 + 2x_2 + 2x_3 + 2x_4 \leq 4 & \end{array} \right\},$$

$$P_\Sigma = \left\{ \boldsymbol{x} \in \mathbb{R}^4 \;\middle|\; \begin{array}{lll} x_1, x_2, x_3, x_4 \geq 0, & x_1 \leq 9, & x_2, x_3, x_4 \leq 3, \\ x_1 + x_2 + x_3 \leq 14, & x_1 + x_2 + x_4 \leq 14, & x_1 + x_3 + x_4 \leq 14, \\ x_1 + x_2 + x_3 + x_4 \leq 15 & & \end{array} \right\}.$$

LCG with unknown multiplier.

$$P_\cup = \left\{ \boldsymbol{x} \in \mathbb{R}^k \;\middle|\; x_1, \ldots, x_k \geq 0, \quad x_1 + \cdots + x_k \leq 2, \quad \cdots \right\},$$

$$P_\Sigma = \left\{ \boldsymbol{x} \in \mathbb{R}^k \;\middle|\; x_1, \ldots, x_k \geq 0, \quad x_1 + \cdots + x_k \leq 2(k-2), \quad \cdots \right\}.$$

where the ellipsis denotes additional inequalities omitted for brevity.

LIPH for unknown-degree POKE.

$$P_{\cup} = P_{\Sigma} = \mathrm{conv}(A(f)) = \left\{ \boldsymbol{x} \in \mathbb{R}^2 \;\middle|\; x_1, x_2 \geq 0, \quad x_2 \leq 1, \quad x_1 - 2x_2 \leq 0 \right\}.$$

E.2 Algebraic Derivations

In Section 4, the Gröbner basis computations are performed on numerical instances to identify structural patterns in the resulting polynomials. To make these computational observations rigorous, we provide explicit algebraic representations of the polynomials in Section 4 that are derived from the initial polynomials. Throughout, we assume that every algebraic quantity appearing in a denominator below is invertible modulo M.

Algebraic Derivation for MIHNP.

$$\begin{aligned} f_{ij}(x_i, x_j) &= x_i x_j + a_{ij} x_i + b_{ij} x_j + c_{ij} \\ &= x_i x_j + \left(u_j - \frac{1}{t_j - t_i} \right) x_i + \left(u_i + \frac{1}{t_j - t_i} \right) x_j + u_i u_j + \frac{u_j - u_i}{t_j - t_i} \\ &= -\frac{x_j + u_j}{t_j - t_i}\, f_i(x_i, \alpha) + \frac{x_i + u_i}{t_j - t_i}\, f_j(x_j, \alpha). \end{aligned}$$

$$\begin{aligned} & f_{1234}(x_1, x_2, x_3, x_4) \\ &= x_2 x_3 x_4 + \frac{a_{34} - a_{24}}{b_{12} - b_{13}} x_1 x_2 x_3 + \frac{b_{34} - a_{13}}{b_{12} - b_{13}} x_1 x_2 x_4 + \frac{a_{12} - b_{24}}{b_{12} - b_{13}} x_1 x_3 x_4 \\ &\quad + \frac{c_{34} - a_{13} a_{24}}{b_{12} - b_{13}} x_1 x_2 + \frac{a_{12} a_{34} - c_{24}}{b_{12} - b_{13}} x_1 x_3 + \frac{a_{12} b_{34} - a_{13} b_{24}}{b_{12} - b_{13}} x_1 x_4 \\ &\quad + \frac{b_{12} a_{34} - b_{13} a_{24}}{b_{12} - b_{13}} x_2 x_3 + \frac{b_{12} b_{34} - c_{13}}{b_{12} - b_{13}} x_2 x_4 + \frac{c_{12} - b_{13} b_{24}}{b_{12} - b_{13}} x_3 x_4 \\ &\quad + \frac{a_{12} c_{34} - a_{13} c_{24}}{b_{12} - b_{13}} x_1 + \frac{b_{12} c_{34} - c_{13} a_{24}}{b_{12} - b_{13}} x_2 + \frac{c_{12} a_{34} - b_{13} c_{24}}{b_{12} - b_{13}} x_3 \\ &\quad + \frac{c_{12} b_{34} - c_{13} b_{24}}{b_{12} - b_{13}} x_4 + \frac{c_{12} c_{34} - c_{13} c_{24}}{b_{12} - b_{13}} \\ &= \frac{1}{b_{12} - b_{13}} \left(f_{12}(x_1, x_2) f_{34}(x_3, x_4) - f_{13}(x_1, x_3) f_{24}(x_2, x_4) \right). \end{aligned}$$

The remaining polynomials admit the same structural representation as the polynomials displayed above.

Algebraic Derivation for ECHNP. Let

$$D_{23} = a_2^2 b_3 - a_2 a_3 b_2 - a_2 a_3 b_3 + a_3^2 b_2 + b_2^2 - 2 b_2 b_3 + b_3^2.$$

$$
\begin{aligned}
&f_{23}(x_1, x_2, x_3)\\
&= x_2 x_3\\
&\quad + \frac{a_2 b_3 c_2 - a_2 e_2 - a_3 b_2 c_2 + a_3 e_2 + b_2 d_2 - b_3 d_2}{D_{23}} x_1 x_3\\
&\quad + \frac{-a_2 b_3 c_3 + a_2 e_3 + a_3 b_2 c_3 - a_3 e_3 - b_2 d_3 + b_3 d_3}{D_{23}} x_1 x_2\\
&\quad + \frac{-a_2 a_3 e_2 + a_2 b_3 d_2 + a_3^2 e_2 - a_3 b_3 d_2 - b_2 b_3 c_2 + b_2 e_2 + b_3^2 c_2 - b_3 e_2}{D_{23}} x_3\\
&\quad + \frac{a_2^2 e_3 - a_2 a_3 e_3 - a_2 b_2 d_3 + a_3 b_2 d_3 + b_2^2 c_3 - b_2 b_3 c_3 - b_2 e_3 + b_3 e_3}{D_{23}} x_2\\
&\quad + \frac{a_2 c_2 e_3 - a_2 c_3 e_2 - a_3 c_2 e_3 + a_3 c_3 e_2 - b_2 c_2 d_3 + b_2 c_3 d_2 + b_3 c_2 d_3 - b_3 c_3 d_2}{D_{23}} x_1\\
&\quad + \frac{a_2 d_2 e_3 - a_2 d_3 e_2 - a_3 d_2 e_3 + a_3 d_3 e_2 - b_2 c_2 e_3 + b_2 c_3 e_2 + b_3 c_2 e_3 - b_3 c_3 e_2}{D_{23}}\\
&= \frac{1}{D_{23}} \Big(-(a_2 d_3 - a_3 d_3 - b_2 c_3 + b_3 c_3)\\
&\qquad - (a_2 a_3 - a_3^2 - b_2 + b_3) x_3 - c_3 (a_2 - a_3) x_1 - (a_2 - a_3) x_1 x_3 \Big) f_2(x_1, x_2)\\
&\quad + \frac{1}{D_{23}} \Big(a_2 d_2 - a_3 d_2 - b_2 c_2 + b_3 c_2\\
&\qquad + (a_2^2 - a_2 a_3 - b_2 + b_3) x_2 + c_2 (a_2 - a_3) x_1 + (a_2 - a_3) x_1 x_2 \Big) f_3(x_1, x_3).
\end{aligned}
$$

The remaining polynomials admit the same structural representation as the polynomials displayed above.