New Results for Coppersmith's Method from the Perspective of Sumsets Theory

Yansong Feng[1,2], Abderrahmane Nitaj[3], and Yanbin Pan[1,2]

[1] Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China
[2] School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China
{fengyansong,panyanbin}@amss.ac.cn
[3] Normandie Univ, UNICAEN, CNRS, LMNO, 14000 Caen, France
abderrahmane.nitaj@unicaen.fr

Abstract. Coppersmith's method, combined with the Jochemsz-May strategy, is widely used to find the small roots of multivariate polynomials for cryptanalysis. At Asiacrypt'23, Meers and Nowakowski improved the Jochemsz-May strategy from a single polynomial equation to a system of polynomial equations and proposed a new method, called Automated Coppersmith. Note that it is typically a tedious and non-trivial task to determine asymptotic upper bounds for Coppersmith's method and manual analysis has to be performed anew when a new set of polynomials is considered. By making certain heuristic assumption, Meers and Nowakowski showed that the bound can be obtained using Lagrange interpolation with the computer, but it is still time-consuming. Moreover, we find that sometimes the interpolation method may get stuck in local convergence, which will result in an incorrect bound when a natural termination strategy is employed in the method.

In this paper, we revisit the Jochemsz-May strategy as well as the work of Meers and Nowakowski and point out that the bound can be obtained by calculating the leading coefficient of some Hilbert function, which is exactly the volume of the corresponding Newton polytope. To this end, we introduce the concept of Sumsets theory and propose a series of related results and algorithms. Compared with the Automated Coppersmith, we overcome the issue of getting stuck in local convergence and directly eliminate the time-consuming calculation for f^m in Automated Coppersmith when m is large, which brings a 1000x$\sim$1200x improvement in running time for some polynomials in our experiment.

Additionally, our new method offers a new perspective on understanding Automated Coppersmith, thus providing proof of Meers and Nowakowski's Heuristic 2 for the system of a single polynomial.

Keywords: Coppersmith · Sumsets · Newton polytopes · Additive combinatorics

1 Introduction

In 1996, Coppersmith [6,7] introduced a method to find the small solutions of a univariate polynomial modular equation, and another method to find the small roots of a bivariate polynomial. Since then, these methods have been extended in several ways, such as [14,21], and have found significant applications in the cryptanalysis [4,5,13,20,21,22,24,30].

The main idea behind Coppersmith's methods lies in constructing a set of polynomials sharing common roots with the original polynomial. In general, the coefficients of such polynomials are used to build a lattice $\mathcal{L}$ to be reduced in the sequel. To improve Coppersmith's method, the key is to construct a better family of polynomials G.

In 2006, Jochemsz and May [16] presented a heuristic strategy, known as the Jochemsz-May strategy, for choosing the collection G_m of the polynomials $g_{i,j}(x_1, \ldots, x_k)$ satisfying a congruence of the form $g_{i,j}(x_1, \ldots, x_k) \equiv 0 \pmod{M}^m$ for a specific integer m. The main idea in the Jochemsz-May strategy is to decrease the order of M in $g_{i,j}$. The Jochemsz-May strategy applies to all multivariate polynomials that have either modular or integer roots, which generalizes the work of Blömer and May [2] that finds optimal bound for small integer roots of bivariate polynomials.

At Asiacrypt'23, Meers and Nowakowski's [23] proposed an automated method, called Automated Coppersmith. They improved on Jochemsz-May from a single polynomial equation to a system of polynomial equations. Moreover, it is typically a tedious and non-trivial task to determine asymptotic upper bounds for Coppersmith's method and manual analysis has to be performed anew when a new set of polynomials is considered. It seems convoluted to prove the asymptotic bound. By making certain heuristic assumption, Meers and Nowakowski showed that the bound can be obtained using Lagrange interpolation with the computer, but it is still time-consuming.

More precisely, both strategies encounter estimating the exponents of the following inequality at the end [1], where X_i is the upper bound of the absolute value of x_i for $i = 1, \cdots, k$:

$$X_1^{p_1(m)} \cdot \ldots \cdot X_k^{p_k(m)} M^{p_{\mathcal{F}}(m)} < M^{p_{\mathcal{M}}(m)-\epsilon}.$$

How to quickly calculate these $p_1, \ldots, p_k, p_{\mathcal{F}}$ and $p_{\mathcal{M}}(m)$ is an unavoidable issue.

For some simple polynomials, we can compute them by summation. Taking the modular polynomial equation $f(x_1, x_2) = a_1x_1 + a_2x_2 + C \equiv 0 \pmod{M}$ as an example, $p_{\mathcal{M}}(m)$ is $m|\mathrm{supp}\{f^m\}| = m\sum_{i_1=0}^{m}\sum_{i_2=0}^{m-i_1} 1 \approx \frac{m^3}{2}$. However, for polynomials with more variables, manual analysis becomes tedious and time-consuming significantly. In [23], Meers and Nowakowski claimed that the functions $p_j(m)$'s $(j = 1, \ldots, k)$ and $p_{\mathcal{M}}(m)$ are polynomials in m when m is large enough, but gave no proof. Furthermore, they assumed that the function $p_{\mathcal{F}}(m)$ becomes a polynomial when m is sufficiently large. This allows them to select

[1] This is just a simplified version, details can be found in Section 2

specific values m and then utilize Lagrange interpolation to solve for $p_{\mathcal{M}}(m)$, $p_j(m)$'s and $p_{\mathcal{F}}(m)$.

However, the interpolation method will usually involve the computation of f^m, which needs lots of time in the worst case since the number of monomials of the power of f grows very quickly. Hence, Automated Coppersmith is still very time-consuming for general polynomials, which is also verified by the experiments.

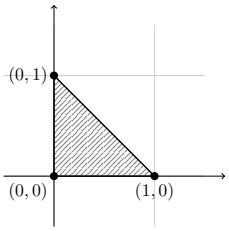
Furthermore, when using the interpolation method, it's necessary to ensure that m is big enough for $p_{\mathcal{M}(m)}$ and $p_j(m)$ to be polynomials. The current bound for m in theory is very huge [11]. A natural idea in practice is to continuously adjust the value of m and add the corresponding interpolation polynomials into a sequence, and output the polynomial when some new added interpolation polynomials remain the same. However, another question of how many times we should adjust m arises. We find in our experiments that sometimes there can be a continuous subsequence of unchanged polynomials before the correct polynomial appears, resembling the phenomenon of local convergence, which means that an incorrect result will be outputted if the times we adjust m is not enough. This significantly affects our confidence in the correctness of the outputted result from the interpolation method.

As a consequence, the following natural question arises:

Can we compute $p_{\mathcal{M}}(m)$, $p_j(m)$'s and $p_{\mathcal{F}}(m)$ more efficiently?

In fact, for the asymptotic upper bound of the roots, only the leading coefficient of these polynomials (if they are) are needed.

Note that for the single polynomial f, $p_{\mathcal{M}}(m) = m|\text{supp}\{f^m\}|$. In 1992, Khovanskii [17] proved that $|\text{supp}\{f^m\}|$ is indeed polynomial in m when m is big enough and the leading coefficient of $|\text{supp}\{f^m\}|$ is exactly the volume of the convex hull related to f. See Fig. 1 for an example. Hence we can compute the leading coefficient of $p_{\mathcal{M}}(m)$ by computing the volume, which is usually very fast in practice.



$f(x_1, x_2) = a_1x_1 + a_2x_2 + C$,
Δ is a triangle of $\{(0,0),(0,1),(1,0)\}$,
$S_\Delta = \frac{1}{2}$.

Fig. 1: Newton polytope corresponding to $\text{supp}\{f\} = \{x_1, x_2, 1\}$

Subsequently, researchers investigated how large m needs to be such that $|\text{supp}\{f^m\}|$ is a polynomial in m. Such explicit results were only previously

known in the special cases when the number of variables $k = 1$ [10,12,26,32], when the convex hull of f is a simplex or when $|\text{supp}\{f\}| = k + 2$ [8] until 2023 Granville et al. [11] give the first effective upper bounds for this threshold for arbitrary f. When $|\text{supp}\{f\}| = n$, it tends to be as least n^n! It might imply that in certain worst-case, if we using Lagrange interpolation, we would need to compute f^m for very large $m = O(n^n)$.

Obviously, Khovanskii's results can help improve the computation of $\dim(\mathcal{L})$ for the Jochemsz-May strategy, while to our best knowledge, no similar results have been found for computing $\dim(\mathcal{L})$ for a system of multi polynomial equations with Automated Coppersmith. Moreover, it remains unsolved to compute the leading coefficients of $p_j(m)$'s and $p_{\mathcal{F}}(m)$ more efficiently for a system of polynomial equations with Automated Coppersmith, even for a single polynomial equation with Jochemsz-May strategy.

Our Contribution. Inspired by Khovanskii's results [17], we try to solve the problem of computing the leading coefficients of $p_{\mathcal{M}}(m), p_j(m)$'s and $p_{\mathcal{F}}(m)$ from the perspective of sumsets theory, and present some more efficient algorithms in this paper. These algorithms are aimed to construct Newton polytopes, whose volume equals the desired leading coefficient of $p_{\mathcal{M}}(m)$, $p_j(m)$'s and $p_{\mathcal{F}}(m)$. Therefore, we avoid encountering the phenomenon of local convergence and the calculation of f^m using Lagrange interpolation.

Compute the leading coefficient of $p_{\mathcal{M}}$. By [17], we know that for a single polynomial f, the number of monomials in f^m becomes a polynomial in m when m is sufficiently large. The leading coefficient of this polynomial is exactly the volume of the convex hull of f. However, no similar results are found for a system of multi polynomials $f_1, \ldots, f_n$ when $n > 1$. We try to generalize Khovanskii's results for such case.

To solve the system of multi polynomials with Coppersmith's method, two types of collections of polynomials have been utilized to construct the lattice. The first one is considered in the Automated Coppersmith, which is $\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} | i_1, \ldots, i_n \leq \frac{m}{n}\}$, and the second one is $\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} | i_1 + \ldots + i_n \leq m\}$ used in [9], which has been found performing better for linear modular equations. Hence, we consider computing $\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} | i_1, \ldots, i_n \leq \frac{m}{n}\}$ and $\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} | i_1 + \ldots + i_n \leq m\}$ to yield $p_{\mathcal{M}}(m)$. By successfully reducing the multi-polynomial case to the single polynomial case, we prove that $p_{\mathcal{M}}(m)$ is also a polynomial in m when m is big enough and present two more efficient algorithms to calculate the leading coefficient of $p_{\mathcal{M}}(m)$ via computing the volume of some Newton polytope.

Compute the leading coefficient of p_j. Meers and Nowakowski [23] claimed that p_j's are polynomials in m when m is large enough, but no proof is presented. We find that it is non-trivial to prove this. Furthermore, we also aim to construct a Newton polytope such that its volume equals $\text{LC}(p_j)$.

By introducing the concept of "High Dimension Duplicate", a method for constructing higher-dimensional convex hull N_j, we prove that p_j's are exactly

polynomials in m when m is large enough and the leading coefficient of p_j is the volume of N_j. The definition we propose is quite intriguing, possessing many desirable properties, such as Lemma 6-8. From these useful Lemma, we can show that its vertex count doesn't need to exceed twice the number of vertices of the original convex hull, making the computation of N_j not too challenging. It is worth mentioning that we provided an algorithm to compute this higher-dimensional convex hull, along with Theorem 5 and Algorithm 5.

Compute the leading coefficient of $p_{\mathcal{F}}$. Meers and Nowakowski assumed that $p_{\mathcal{F}}$ becomes a polynomial in m when m is sufficiently large [23]. The difficulty with $p_{\mathcal{F}}$ lies in characterizing the optimal polynomial for each monomial, making it challenging to prove that $p_{\mathcal{F}}$ becomes a polynomial in m for large m. However, by introducing sumset theory, we realize that for $n = 1$, that is, the system has a single polynomial f, and the selected monomial sets correspond to the saturated Newton polytope of f. Regardless of the order, the leading monomial of f is definitely a vertex of this Newton polytope. Besides, we not only prove that when $n = 1$ $p_{\mathcal{F}}$ becomes a polynomial in m for large m, i.e., Heuristic 2 in [23], but also provide a symbolic solution for the leading coefficient of $p_{\mathcal{F}}$.

As with Meers and Nowakowski in [23], we also do not consider the "Extended Strategy" mentioned in [16]. Just "Basic Strategy" is enough in practice, such as CI-HNP in [23]. Besides, using the "Extended Strategy" leads to an increase in the dimension of the lattice.

However, we have made an interesting discovery. Upon studying the univariate case, we observed that there is no need to introduce additional shift polynomials for the "Extended Strategy". Instead, we simply modify N^{m-i} to $N^{\max(t-i,0)}$. This modification yields the same results as before but with a smaller lattice dimension. Using the same idea, we studied multivariate case. In the case of modular unknown divisors, this result can produce an effect similar to shift polynomials without increasing the lattice dimension.

Roadmap. The paper is organized as follows: In Section 2 we give some basic preliminaries about polynomials (such as the definition of **Hilbert function**), Coppersmith's method (**Jochemsz-May Strategy** and **Automated Coppersmith**) and Sumsets Theory (such as the definition of **Newton polytope**). Our algorithms for computing Automated Coppersmith's method quickly is described in Section 3, where we omit the calculation about the values of $p_{\mathcal{M}}$ and p_j when m_i is relatively large. Then we provided our proof of Heuristic 2 in Automated Coppersmith for $n = 1$ in Section 4. In order to fully demonstrate the superiority of our algorithm, we have conducted sufficient experiments, and the experimental results can be found in Section 5. We also provide an example to illustrate the phenomenon of local convergence encountered with interpolation methods. Finally, We conclude our work in Section 6.

2 Notations and Preliminaries

Let $\mathbb{Z}$ denote the ring of integers and $\mathbb{Q}$ denote the field of rational numbers. We use lowercase bold letters (e.g., $\mathbf{v}$) for vectors and uppercase bold letters (e.g., $\mathbf{A}$) for matrices. The notation $\binom{n}{m}$ represents the number of ways to select m items out of n items, which is defined as $\frac{n!}{m!(n-m)!}$. If $m > n$, we set $\binom{n}{m} = 0$.

2.1 Polynomials

Let $x_1, \ldots, x_k$ be k variables. Suppose f is a polynomial in $\mathbb{Z}[x_1, \ldots, x_k]$, then the polynomial f can be expressed as

$$f(x_1, \ldots, x_k) = \sum_{i_1, \ldots, i_k \in \mathbb{N}} \alpha_{i_1, \ldots, i_k} \cdot x_1^{i_1} \cdot \ldots \cdot x_k^{i_k}.$$

Here, $x_1^{i_1} \cdot \ldots \cdot x_k^{i_k}$ is termed as a monomial of f if its coefficient $\alpha_{i_1, \ldots, i_k} \neq 0$. The set of all monomials of f is denoted as $\mathrm{supp}\{f\}$. The total degree $\deg(f)$ of f is defined as

$$\deg(f) := \max_{\alpha_{i_1, \ldots, i_k} \neq 0} (i_1 + \cdots + i_k).$$

The following definitions serve to simplify the notations related to multivariate polynomials.

Definition 1. *Let M be a positive integer. For a set of polynomials $\mathcal{F} \subset \mathbb{Z}[x_1, \ldots, x_k]$, we define the set $\mathbb{Z}_M(\mathcal{F})$ of its roots as*

$$\mathbb{Z}_M(\mathcal{F}) := \left\{ r = (r_1, \ldots, r_k) \in \mathbb{Z}^k \mid \forall f \in \mathcal{F} : f(r) \equiv 0 \mod M \right\}.$$

Similarly, for parameters $M, X_1, \ldots, X_k \in \mathbb{N}$, we define the corresponding set of its small modular roots as

$$\mathbb{Z}_{M, X_1, \ldots, X_k}(\mathcal{F}) := \left\{ r = (r_1, \ldots, r_k) \in \mathbb{Z}^k \mid \forall f \in \mathcal{F} : f(r) \equiv 0 \mod M, \forall j : |r_j| \leq X_j \right\}.$$

Definition 2 (Monomial Order). *Let $\mathcal{M}$ be a set of monomials. A monomial order on $\mathcal{M}$ is a total order $\prec$ that satisfies the following two properties:*

1. *For every $\lambda \in \mathcal{M}$, it holds that $1 \prec \lambda$.*
2. *If $\lambda_1 \prec \lambda_2$, then $\lambda \cdot \lambda_1 \prec \lambda \cdot \lambda_2$ for every monomial $\lambda \in \mathcal{M}$.*

For example, suppose $x_1 \prec x_2 \prec x_3$, then $x_2^2 \prec x_3$ and $x_1 \prec x_2 \prec x_2^2$ when using the lexicographic monomial order $\prec_{\mathrm{lex}}$. Because lexicographic monomial order ($\prec_{\mathrm{lex}}$) first compares exponents of x_1 in the monomials, and in case of equality compares exponents of x_2, and so forth.

If $\prec$ is a monomial order, the leading monomial of a polynomial f is the unique monomial λ of f that satisfies $\lambda' \prec \lambda$ for every monomial λ' of f. We denote the leading monomial, and the leading coefficient of the leading monomial of f by $\mathrm{LM}(f)$ and $\mathrm{LC}(f)$ respectively. The leading term of f is denoted $\mathrm{LT}(f)$ and satisfies

$$\mathrm{LT}(f) = \mathrm{LC}(f) \times \mathrm{LM}(f).$$

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

Definition 3 (Ideal). *Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be a set of polynomials in $\mathbb{Z}[x_1, \ldots, x_k]$. The ideal I generated by $\mathcal{F}$ is the set of all linear polynomial combinations of $f_1, \ldots, f_n$, that is*

$$I = \{a_1 f_1 + \ldots + a_n f_n : a_i \in \mathbb{Z}[x_1, \ldots, x_k]\}.$$

If I is ideal, then the set of all leading terms of the elements of I is denoted $\mathrm{LT}(I)$ and satisfies $\mathrm{LT}(I) = \{\mathrm{LT}(f) | f \in I\}$.

Definition 4. *Fix a monomial order, and let I be an ideal. A finite subset $G = \{g_1, \ldots, g_r\} \subset I$ is a Gröbner basis for I if*

$$\mathrm{LT}(I) = \{\mathrm{LT}(g_1), \ldots, \mathrm{LT}(g_r)\}.$$

Before we introduce the Hilbert Function of an ideal I, we need the following definition:

Definition 5. *Suppose I is an ideal in $\mathbb{Z}[x_1, x_2, \ldots, x_k]$ and then we define $\mathbb{Z}[x_1, x_2, \ldots, x_k]_{\leqslant s}$ to be the set of polynomials in $\mathbb{Z}[x_1, x_2, \ldots, x_k]$ of total degree $\leqslant s$, and $I_{\leqslant s}$ is the set of polynomials in I of total degree $\leqslant s$. That is,*

$$\mathbb{Z}[x_1, x_2, \ldots, x_k]_{\leqslant s} = \{f \in \mathbb{Z}[x_1, x_2, \ldots, x_k] : \deg(f) \leqslant s\},$$
$$I_{\leqslant s} = I \cap \mathbb{Z}[x_1, x_2, \ldots, x_k]_{\leqslant s} = \{f \in I : \deg(f) \leqslant s\}.$$

Both $\mathbb{Z}[x_1, x_2, \ldots, x_k]_{\leqslant s}$ and $I_{\leqslant s}$ are vector spaces over $\mathbb{Z}$, with $I_{\leqslant s}$ exactly being a subspace of $\mathbb{Z}[x_1, x_2, \ldots, x_k]_{\leqslant s}$. Now, we are prepared to introduce the Hilbert function.

Definition 6 (Hilbert function). *Let I be an ideal in $\mathbb{Z}[x_1, x_2, \ldots, x_k]$, and let $I_{\leqslant s}$ be the space of elements of I of degree at most s. The (affine) Hilbert function $HF(s)$ of I is defined to be the dimension of $\mathbb{Z}[x_1, x_2, \ldots, x_k]_{\leqslant s}/I_{\leqslant s}$ as a vector space over $\mathbb{Z}$. That is,*

$$HF_I(s) = \dim(\mathbb{Z}[x_1, x_2, \ldots, x_k]_{\leqslant s}/I_{\leqslant s}).$$

There is a useful lemma for Hilbert function, which is called Hilbert's theorem (see [[25], Theorem 6.21]):

Lemma 1. *Let $I \subset \mathbb{Z}[x_1, x_2, \ldots, x_k]$ be a proper ideal. Then there exists a polynomial $h(z) \in \mathbb{Q}[z]$ such that $\deg(h) = \dim(I)$, for sufficiently large m,*

$$HF_I(m) = h(m).$$

The polynomial $h(z)$ is often referred to as the Hilbert polynomial of I.

Remark 1. The concepts of Hilbert functions and Hilbert polynomials of graded algebras are crucial in commutative algebra. For more detailed results, please refer to [29].

Finally, there is one more result about the sum of the p-th powers of the first m positive integers.

Lemma 2 (Faulhaber's formula,[15]). *The sum of the p-th powers of the first m positive integers*

$$\sum_{k=1}^{m} k^p = 1^p + 2^p + 3^p + \cdots + m^p, \tag{1}$$

is a polynomial in m with leading term $\frac{m^{p+1}}{p+1}$.

2.2 Lattices, SVP, and LLL

Let $m \geq 2$ be an integer. A lattice is a discrete additive subgroup of $\mathbb{R}^m$. A more explicit definition is presented as follows.

Definition 7 (Lattice). *Let* $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n} \in \mathbb{R}^m$ *be n linearly independent vectors with* $n \leq m$. *The lattice* $\mathcal{L}$ *spanned by* $\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}\}$ *is the set of all integer linear combinations of* $\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}\}$, *i.e.,*

$$\mathcal{L} = \left\{ \mathbf{v} \in \mathbb{R}^m \mid \mathbf{v} = \sum_{i=1}^{n} a_i \mathbf{v_i}, a_i \in \mathbb{Z} \right\}.$$

The integer n denotes the rank of the lattice $\mathcal{L}$, while m represents its dimension. The lattice $\mathcal{L}$ is said to be full rank if $n = m$. We use the matrix $\mathbf{B} \in \mathbb{R}^{n \times m}$, where each vector $\mathbf{v_i}$ contributes a row to $\mathbf{B}$. The determinant of $\mathcal{L}$ is defined as $\det(\mathcal{L}) = \sqrt{\det(\mathbf{B}\mathbf{B}^t)}$, where $\mathbf{B}^t$ is the transpose of $\mathbf{B}$. If $\mathcal{L}$ is full rank, this reduces to $\det(\mathcal{L}) = |\det(\mathbf{B})|$.

Definition 8 (Fundamental domain). *For a lattice basis* $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n} \in \mathbb{R}^m$, *the space generated by all real number combinations in* $[0, 1)^n$ *is called the fundamental domain of the lattice* $\mathcal{L}$. *It is denoted as*

$$\mathcal{P}(\mathcal{L}) = \left\{ \sum_{i=1}^{n} a_i \mathbf{v_i} | 0 \leq a_i < 1 \right\}.$$

The volume of the fundamental domain $\mathcal{P}$ is equal to the determinant of the lattice, that is $\mathrm{vol}(\mathcal{P}) = \det(\mathcal{L})$.

In lattice theory, numerous hard problems are used to secure several cryptosystems. The Shortest Vector Problem (SVP) is one of them.

Definition 9 (Shortest Vector Problem (SVP)). *Given a lattice* $\mathcal{L}$, *the Shortest Vector Problem (SVP) asks to find a non-zero lattice vector* $\mathbf{v} \in \mathcal{L}$ *of minimum Euclidean norm, i.e., find* $\mathbf{v} \in \mathcal{L} \backslash \{\mathbf{0}\}$ *such that* $\|\mathbf{v}\| \leq \|\mathbf{w}\|$ *for all non-zero* $\mathbf{w} \in \mathcal{L}$.

Although SVP is NP-hard under randomized reductions [1], there exist algorithms that can find a relatively short vector, instead of the exactly shortest vector, in polynomial time, such as the famous LLL algorithm proposed by Lenstra, Lenstra, and Lovász [18] in 1982. The following result is useful for our analysis [21].

Theorem 1 (LLL). *Let $\mathcal{L}$ be a lattice spanned by a basis $(\mathbf{u_1}, \ldots, \mathbf{u}_\omega)$. In polynomial time, the LLL algorithm finds a new basis $(\mathbf{v}_1, \ldots, \mathbf{v}_\omega)$ of $\mathcal{L}$ satisfying*

$$\|\mathbf{v}_1\| \leq \ldots \leq \|\mathbf{v}_i\| \leq 2^{\frac{\omega(\omega-1)}{4(\omega+1-i)}} \det(\mathcal{L})^{\frac{1}{\omega+1-i}},$$

for $i = 1, \ldots, \omega$.

The following result [14] is useful to find the small solutions of a multivariate modular polynomial equation, where, for $h(x_1, \ldots, x_k) = \sum a_{i_1,\ldots,i_k} x_1^{i_1} \cdots x_k^{i_k}$, the Euclidean norm of $h(X_1 x_1, \ldots, X_k x_k)$ is

$$\|h(X_1 x_1, \ldots, X_k x_k)\| = \sqrt{\sum a_{i_1,\ldots,i_k}^2 X_1^{2i_1} \cdots X_k^{2i_k}}.$$

Theorem 2 (Howgrave-Graham). *Let $h(x_1, \ldots, x_k)$ be a polynomial with at most ω monomials, and let $m, M, X_1, \ldots, X_k \in \mathbb{N}$. Suppose h has a root $r = (r_1, \ldots, r_k)$ modulo M^m, satisfying $|r_i| \leq X_i$ for every $i = 1, \ldots, k$. If*

$$\|h(X_1 x_1, \ldots, X_k x_k)\| < \frac{M^m}{\sqrt{\omega}},$$

then $h(r_1, \ldots, r_k) = 0$ holds over the integers.

2.3 Growth of sumsets

For any given finite subset A of an abelian group G, suppose $0 \in A$, we consider the sumset $mA := \{a_1 + a_2 + \cdots + a_m : a_i \in A\}$. Khovanskii's 1992 theorem [17] states that if $A \subset \mathbb{Z}^k$ is finite, then there exists $p(x) \in \mathbb{Q}[x]$ of degree k and $N_{\mathrm{Kh}}(A)$ such that if $m \geq N_{\mathrm{Kh}}(A)$, then $|mA| = p(m)$. Moreover, if the difference set $A - A$ generates all of $\mathbb{Z}^k$ additively, then $\deg(p) = k$ and the leading coefficient of p is the volume of the convex hull of A, which we define as $H(A)$.

To make things more straightforward, we introduce the Newton polytope.

Definition 10 (Set of points). *Let $G = \mathbb{Z}^K$. For a polynomial f, consider $A(f)$ as the set of points corresponding to the monomials of f as follows:*

$$A(f) = \{(i_1, \ldots, i_k) | x_1^{i_1} \cdot \ldots \cdot x_k^{i_k} \text{ is a monomial of } f\}.$$

Definition 11 (Newton polytope). *Let f be a polynomial in $\mathbb{Z}[x_1, \ldots, x_k]$. The Newton polytope $N(f)$ of f is defined as the convex hull of $A(f)$*

Obviously, the Newton polytope has the following property:

Property 1. For all polynomials f_1, f_2 in $\mathbb{Z}[x_1, \ldots, x_k]$, it holds that

$$N(f_1 \cdot f_2) = N(f_1) + N(f_2).$$

Definition 12 (Saturated Newton polytope). *We say that a polynomial f has Saturated Newton Polytope if every integer point of the convex hull of its exponent vectors corresponds to a monomial of f.*

For example, when $\text{supp}\{f\} = \{x_1^2, x_1, x_2, 1\}$, $A(f)$ is $\{(0,0),(1,0),(2,0),(0,1)\}$, corresponding to $\{1, x, x^2, y\}$ and the Newton Polytope of f is a triangle with $\{(0,0),(2,0),(0,1)\}$. Then $|mA(f)|$ corresponds to $\text{supp}\{f^m\}$. For simplicity, we write $A(f)$ as A. So Khovanskii's 1992 theorem [17] can be stated as follows:

Lemma 3 (Khovanskii, [17]). *There exists a value N_{kh} such that if $m > N_{kh}$, then there exists a polynomial $p(x) \in \mathbb{Q}[x]$ of degree k such that $|\bigcup_{j=0}^{m} \text{supp}\{f^j\}| = p(m)$.*

Khovanskii proved this by constructing a finitely generated graded module M over the polynomial ring $\mathbb{C}[t_1, \ldots, t_s]$, where the cardinality of set A is denoted by s. This module possesses the characteristic that its homogeneous component M_m forms a vector space over $\mathbb{C}$ with precisely $\text{supp}\{f^m\}$ dimensions for all $m \geq 1$. Therefore, the dimension of M_m over $\mathbb{C}$ is exactly the Hilbert Function. According to Hilbert's theorem, the dimension of M_m over $\mathbb{C}$ is a polynomial in m for sufficiently large m, thereby yielding the desired result.

Suppose $A \subset \mathbb{Z}^k$ is full rank, which means there exist $\mathbf{v}_1, \ldots, \mathbf{v}_k$ that are linearly independent. We denote the linear space spanned by A over $\mathbb{Z}$ as $\text{span}(A)$, that is

$$\text{span}(A) = \{a_1\mathbf{v}_1 + \ldots + a_k\mathbf{v}_k | a_1, \ldots, a_k \in \mathbb{Z}\}.$$

Moreover, if $\text{span}(A) = \mathbb{Z}^k$ additively, then $\deg(p) = k$ and the leading coefficient of p is the volume of $N(f)$.

When $\text{span}(A) \neq \mathbb{Z}^k$, we use the following definition to calculate the leading coefficient of p:

Definition 13. *For a polynomial f, let $A(f)$ be the set of points corresponding to the monomials of f, and* $\text{span}(A)$ *be the corresponding lattice over $\mathbb{Z}$. Then we denote the fundamental domain of* $\text{span}(A)$ *as $\mathcal{P}(f)$.*

See Fig 2 as an example, suppose $f = a_1x_1 + a_2x_2 + C$. We have $\text{supp}\{f\} = \{x_1, x_2, 1\}$. Now $A(f) = \{(1,0),(0,1),(0,0)\}$ and $\text{span}(A) = \{(1,0)z_1 + (0,1)z_2 | z_1, z_2 \in \mathbb{Z}\}$ is a lattice over $\mathbb{Z}$. Then $\mathcal{P}(f)$ is a unit square.

Fig. 2: $A(f)$ and $\mathcal{P}(f)$: we can see that $A(f)$ is a triangle and $\mathcal{P}(f)$ is a unit square.

Therefore, we can rewrite Corollary 2 in [17] as follows:

Lemma 4. *There exists a value N_{kh} such that if $m > N_{kh}$, then there exists a polynomial $p(x) \in \mathbb{Q}[x]$ of degree k such that $|\bigcup_{j=0}^{m} \mathrm{supp}\{f^j\}| = p(m)$ and the leading coefficient of p is the volume of $\frac{V(N(f))}{V(\mathcal{P}(f))}$.*

Proof. The idea of the proof is as follows: since the points in the fundamental domain are unreachable, we consider $\mathbb{Z}^k \mod \mathcal{P}(f)$. In this case, it is equivalent to the situation where $\mathrm{span}(A) = \mathbb{Z}^k$, and thus the proof is completed. □

Remark 2. If f has a constant term and all coefficients are positive, i.e., 0 belongs to $A(f)$, then it holds that: $\bigcup_{j=0}^{m} \mathrm{supp}\{f^j\} = \mathrm{supp}\{f^m\}$.

Regarding the size of N_{kh}, in 2023, Granville et al. [11] provided the first effective upper bounds for this threshold for arbitrary A. For any such A in terms of the width of A, $w(A) = \mathrm{width}(A) := \max_{a_1,a_2 \in A} \|a_1 - a_2\|_\infty$. Then the upper bound proposed by Granville et al. is as follows:

Lemma 5 (Granville et al., [11]). *If $A \subset \mathbb{Z}^k$ is finite, then $\left|\bigcup_{j=0}^{m} \mathrm{supp}\{f^j\}\right| = p(m)$ for all $m \geq (2|A| \cdot \mathrm{width}(A))^{(k+4)|A|}$.*

We note that the former upper bound is too large. When $|A| = n$, it tends to be at least n^n!

2.4 Jochemsz-May Strategy [2] and Automated Coppersmith Method

At Asiacrypt'06, Jochemsz and May [16] described a strategy to find small modular and integer roots of multivariate polynomials. Recently, Meers and Nowakowski [23] generalized the idea of the Jochemsz-May strategy and proposed a new method called Automated Coppersmith. Their idea is based on the notion of $(\mathcal{M}, \prec)$-suitability of a set of polynomials.

Definition 14. *Let $\mathcal{M}$ be a finite set of monomials, and let $\prec$ be a monomial order on $\mathcal{M}$. A set of polynomials $\mathcal{F}$ is called $(\mathcal{M}, \prec)$-suitable if:*

1. *Every polynomial $f \in \mathcal{F}$ is defined over $\mathcal{M}$.*
2. *For every monomial $\lambda \in \mathcal{M}$, there is a unique polynomial $f \in \mathcal{F}$ with a leading monomial λ with respect to $\prec$.*

If $\mathcal{F}$ is $(M, \prec)$-suitable and $\lambda \in \mathcal{M}$, then we denote by $\mathcal{F}[\lambda]$ the unique polynomial $f \in \mathcal{F}$ with the leading monomial λ.

Therefore, it is crucial to understand how to generate an $(\mathcal{M}, \prec)$-suitable set of polynomials $\mathcal{F}$ from $\mathcal{M}$. Unlike in all other Coppersmith-type results, simply construct $\mathcal{F}$ using so-called shift-polynomials, i.e., polynomials of the form

$$f_{j_1,\ldots,j_k,i_1,\ldots,i_n} := x_1^{j_1} \cdot \ldots \cdot x_k^{j_k} \cdot f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \cdot M^{m-(i_1+\ldots+i_n)}, \tag{2}$$

[2] Because the Jochemsz-May strategy can be viewed as Automated Coppersmith's method when $\boldsymbol{n = 1}$, we will only provide a detailed introduction to the latter.

for some appropriately chosen integers $j_1, \ldots, j_k, i_1, \ldots, i_n \in \mathbb{N}$. For a monomial $\lambda \in \mathcal{M}$, Meers and Nowakowski define the polynomials of the form

$$f_{[\lambda,i_1,\ldots,i_n]} := \frac{\lambda}{\mathrm{LM}(f_1)^{i_1} \cdot \ldots \cdot \mathrm{LM}(f_n)^{i_n}} \cdot f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \cdot M^{m-(i_1+\ldots+i_n)}, \quad (3)$$

and provide the following algorithm in [23]:

Algorithm 1: Constructing an $(\mathcal{M}, \prec)$-suitable set $\mathcal{F}$

Input: Set of monomials $\mathcal{M}$, monomial order $\prec$ on $\mathcal{M}$, monic polynomials $f_1, \ldots, f_n$, and integer $m \in \mathbb{N}$

Output: $(\mathcal{M}, \prec)$-suitable set of shift-polynomials $\mathcal{F}$, satisfying $\mathbb{Z}_{M,X_1,\ldots,X_k}(f_1, \ldots, f_n) \subseteq \mathbb{Z}_{M^m,X_1,\ldots,X_k}(\mathcal{F})$, and minimizing $\sum_{\lambda\in\mathcal{M}} |\mathrm{LC}(\mathcal{F}[\lambda])|$

```
F ← ∅;
for λ ∈ M do
    Enumerate all shift-polynomials f_[λ,i_1,...,i_n] as in Equation (3) such that
      LM(f_{i_1}) · ... · LM(f_{i_n}) divides λ and f_[λ,i_1,...,i_n] is defined over M;
    Among all such f_[λ,i_1,...,i_n], pick one that maximizes i_1 + ... + i_n and
      include it in F;
end
return F;
```

Meers and Nowakowski also provide a way to choose monomials set $\mathcal{M}$. Denote $m_i = i \cdot n$. Let $f_1, \ldots, f_n \in \mathbb{Z}[x_1, \ldots, x_k]$. For $i \in \mathbb{N}$, define

$$\mathcal{M}_{m_i} := \bigcup_{0\le j_1,\ldots,j_n\le i} \mathrm{supp}\{f_1^{j_1} \cdot \ldots \cdot f_n^{j_n}\}.$$

This is a breakthrough idea, but it introduces a new heuristic. Suppose we want to find the small modular roots as $\mathbb{Z}_{M,X_1,\ldots,X_k}(f_1, \ldots, f_n)$ with an $(\mathcal{M}, \prec)$-suitable set of polynomials $\mathcal{F} \subseteq \mathbb{Z}_{M^m}[x_1, \ldots, x_k]$. We need the following condition before applying the LLL algorithm:

$$\det(\mathcal{L}) = \prod_{\lambda\in\mathcal{M}} \lambda(X_1, \ldots, X_k) \cdot M^{\sum_{\lambda\in\mathcal{M}} |\mathrm{LC}(\mathcal{F}[\lambda])|} \le M^{(m-k)|\mathcal{M}|}, \quad (4)$$

When i is sufficiently large, the terms $M^{(m-k)|\mathcal{M}_i|}$ and $\prod_{\lambda\in\mathcal{M}_i} \lambda(X_1, \ldots, X_k)$ in Equation (4) grow as $M^{p_{\mathcal{M}}(m_i)}, X_1^{p_1(m_i)} \cdot \ldots \cdot X_k^{p_k(m_i)}$, where $p_{\mathcal{M}}, p_1, \ldots, p_k$ are polynomials of degree $k+1$. Therefore, we need $\prod_{\lambda\in\mathcal{M}_i} |\mathrm{LC}(\mathcal{F}_i[\lambda])|$ to be writable as $M^{p_{\mathcal{F}}(m_i)}$ when i is sufficiently large, where $p_{\mathcal{F}}$ is also a polynomial of degree $k+1$. Although it often holds true in experiments, it is challenging to prove that the set $\mathcal{F}$ obtained from Algorithm 1 satisfies this property for arbitrary $f_1, \ldots, f_n$. Hence, the following heuristic is used in Algorithm 2, and we call it *the Heuristic in Automated Coppersmith* in the sequel.

Heuristic 1. *Let $f_1, \ldots, f_n \in \mathbb{Z}[x_1, \ldots, x_k]$, let $\prec$ be a monomial order on $x_1, \ldots, x_k$. Define an increasing sequence $\mathcal{M}_1 \subset \mathcal{M}_2 \subset \mathcal{M}_3 \subset \ldots$ of sets of monomials and $m_i := i \cdot n$. Then there exists a polynomial $p_{\mathcal{F}}(m)$ of degree $k+1$, such that for any set $\mathcal{F}_i$, that is obtained from Algorithm 1 on input $(\mathcal{M}_i, \prec, (f_1, \ldots, f_n), m_i)$, it holds that*

$$\prod_{\lambda \in \mathcal{M}_i} |\mathrm{LC}(\mathcal{F}_i[\lambda])| = M^{p_{\mathcal{F}}(m_i)}.$$

After constructing the lattice $\mathcal{L}$ and applying the LLL algorithm, Coppersmith's method needs to assume the following assumption.

Assumption 1. *The polynomials obtained from the LLL-reduced basis in Coppersmith's method generate an ideal of a zero-dimensional variety.*

Assumption 1 is often used in connection with Coppersmith's method in the multivariate scenario [3,19,21,28,31], the heuristic holds for most instances arising in practice.

Finally, we give a formal description of the Automated Coppersmith Method by the following algorithm.

Algorithm 2: Coppersmith's Method

Input: Integers M, $m \in \mathbb{N}$, polynomials $f_1, \ldots, f_n \in \mathbb{Z}[x_1, \ldots, x_k]$, bounds $0 \leq X_1, \ldots, X_k$, set of monomials $\mathcal{M}$, monomial order $\prec$ on $\mathcal{M}$, and a $(M, \prec)$-suitable set of polynomials $\mathcal{F} \subseteq \mathbb{Z}_{M^m}[x_1, \ldots, x_k]$ satisfying Equation (4).

Output: All roots $r \in \mathbb{Z}_{M, X_1, \ldots, X_k}(f_1, \ldots, f_n)$.

1 Construct an $|\mathcal{M}| \times |\mathcal{M}|$ basis matrix $\mathbf{B}$, whose columns are the coefficient vectors of the polynomials $\mathcal{F}[\lambda](X_1 x_1, \ldots, X_k x_k)$, where $\lambda \in \mathcal{M}$;

2 LLL-reduce $\mathbf{B}$;

3 Interpret the first k columns of the resulting matrix as the coefficient vectors of polynomials $h_i(X_1 x_1, \ldots, X_k x_k)$;

4 **return** all $r \in \mathbb{Z}^k(h_1, \ldots, h_k) \cap \mathbb{Z}_{M, X_1, \ldots, X_k}(f_1, \ldots, f_n)$.

3 Algorithms related to Newton Polytope

3.1 Algorithm for quickly calculating LC($p_{\mathcal{M}}$)

Next, we will associate the leading coefficient of $p_{\mathcal{M}}$ and p_j with the volume of the convex hull by analyzing the Hilbert Function of some graded algebra. Therefore, we only need to calculate the volume of the convex hull to obtain the desired value, which is a very fast operation. In [23], since m needs to be sufficiently large to ensure that $p_{\mathcal{M}}, p_j, p_{\mathcal{F}}$ are polynomials about m, it is necessary to calculate the values of $p_{\mathcal{M}}, p_j, p_{\mathcal{F}}$ when m is relatively large, and then

calculate the results of $p_{\mathcal{M}}, p_j, p_{\mathcal{F}}$ using Lagrange interpolation based on these values. When m is relatively large, the computation of the values of $p_{\mathcal{M}}$ and p_j is time-consuming, so our method avoids this operation, which is very useful.

When we consider the case $n = 1$, which means considering that m is sufficiently large, $\text{supp}\{f^m\} = p_{\mathcal{M}}(m)$. Recall from Lemma 4 that we know that the leading coefficient of $p_{\mathcal{M}}$ is the volume of the convex hull of A divided by the volume of fundamental domain of $\mathcal{L}(f)$, that is $\text{LC}(p_{\mathcal{M}}) = \frac{V(N(f))}{V(\mathcal{P}(f))} = \frac{V(N(f))}{\det(L(f))}$.

For example, when we consider a modular polynomial equation $f \equiv 0 \pmod{M}$ with $\text{supp}\{f\} = \{x_1, x_2, 1\}$, we use $\text{supp}\{f^m\}$ as the dimension of $\mathcal{L}$. In Coppersmith's method, we usually compute $\sum_{i_1=0}^{m} \sum_{i_2=0}^{m-i_1} 1 \approx \frac{m^2}{2}$. For Automated Coppersmith method, we have to choose a large value m, then calculate $|\text{supp}\{f^m\}|$ and use Lagrange interpolation.

However, with our new method, we have an easier way to compute it. What we need is just $\text{LC}(p_{\mathcal{M}})$, so we do not need the other coefficients of $p_{\mathcal{M}}$.

From Fig 2, we have $N(f)$ is a triangle of $\{(0,0),(0,1),(1,0)\}$ and $\mathcal{P}(f)$ is a unit square. Then we can compute $\text{LC}(p_{\mathcal{M}})$ as follows:

$$\text{LC}(p_{\mathcal{M}}) = \frac{V(N(f))}{V(\mathcal{P}(f))} = \frac{1}{2}.$$

When we select a polynomial f which is slightly more complex, the advantage of our method is more obvious because it does not require finding f^m, which is time-consuming.

Next, when considering $n > 1$ and the corresponding situations for $f_1, \ldots, f_n$, we consider two generalizations. The first one is proposed in Automated Coppersmith, using $\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} | i_1, \ldots, i_n \leq \frac{m}{n}\}$. The second one is introduced in [9] using $\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} | i_1 + \ldots + i_n \leq m\}$. It performs better computations for linear modular equations.

For the first generalization, we have the following result.

Theorem 3. *There exists a polynomial f such that*

$$\left|\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \mid 0 \leq i_1, \ldots, i_n \leq \frac{m}{n}\}\right| = |\text{supp}\{f^{\frac{m}{n}}\}|.$$

Moreover, when m is sufficiently large, there exists a polynomial $p_{\mathcal{M}}$ with degree $k+1$ such that

$$m|\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \mid 0 \leq i_1, \ldots, i_n \leq \frac{m}{n}\}| = m|\text{supp}\{f^m\}| = p_{\mathcal{M}}(m),$$

where the leading coefficient of $p_{\mathcal{M}}$ is $\frac{V(N(f))}{n^k V(\mathcal{P}(f))}$

Proof. Consider $f = f_1 \cdot \ldots \cdot f_n$. Define $\mathcal{M}_m^{(1)} = \text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \mid 0 \leq i_1, \ldots, i_n \leq \frac{m}{n}\}$, we have

$$\mathcal{M}_m^{(1)} = \text{supp}\{f_1^{\frac{m}{n}} \cdot \ldots \cdot f_n^{\frac{m}{n}}\} = \text{supp}\{f^{\frac{m}{n}}\}.$$

This way, we reduce it to the case of $n = 1$. Hence, the leading coefficient of $p_{\mathcal{M}}$ is $\frac{V(N(f))}{n^k V(\mathcal{P}(f))}$. □

Remark 3. This can also be seen as a special case of Theorem 1 in [27] with $B = 0, h_1 = ... = h_k$.

By Lemma 4, we can quickly compute the leading coefficient of the corresponding $p_{\mathcal{M}}$ by computing the volume of the convex hull of f. We provide the following algorithm for the corresponding calculation process.

Algorithm 3: Calculate LC($p_{\mathcal{M}}$)

Input: Set of monic polynomials $\mathcal{F} = \{f_1, \ldots, f_n\}$
Output: LC($p_{\mathcal{M}}$), satisfying Theorem 3

```
Define N(f) = ∅;
for j ∈ 1, ..., n do
    Compute N(f_j);
    N(f) ← N(f) + N(f_j);
end
Compute V(N(f)) and V(P(N(f)));
return V(N(f)) / (n^k V(P(f))).
```

Next, for the second case, we have the following result.

Theorem 4. *There exists a polynomial f such that*

$$\left|\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \mid 0 \leq i_1 + \ldots + i_n \leq m\}\right| = |\text{supp}\{f^m\}|.$$

Moreover, when m is sufficiently large, there exists a polynomial $p_{\mathcal{M}}$ with degree $k+1$ such that

$$m|\text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \mid 0 \leq i_1 + \ldots + i_n \leq m\}| = m|\text{supp}\{f^m\}| = p_{\mathcal{M}}(m).$$

The leading coefficient of $p_{\mathcal{M}}$ is $\frac{V(N(f))}{V(\mathcal{P}(f))}$.

Proof. Consider $f = f_1 + \ldots + f_n$. Define $\mathcal{M}_m^{(2)} = \text{supp}\{f_1^{i_1} \cdot \ldots \cdot f_n^{i_n} \mid 0 \leq i_1 + \ldots + i_n \leq m\}$. We have

$$\mathcal{M}_m^{(2)} = \text{supp}\{(f_1 + \ldots + f_n)^m\} = \text{supp}\{f^m\}.$$

This way, we also reduce it to the case of $n = 1$. Hence, the leading coefficent of $p_{\mathcal{M}}$ is $\frac{V(N(f))}{V(\mathcal{P}(f))}$. □

Similarly, we can provide an upper bound for m, but we can still devise an algorithm to directly compute the leading coefficient of the corresponding $p_{\mathcal{M}}$.

Algorithm 4: Calculate $\mathrm{LC}(p_{\mathcal{M}})$

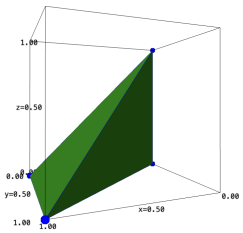
Input: Set of monic polynomials $\mathcal{F} = \{f_1, \ldots, f_n\}$
Output: $\mathrm{LC}(p_{\mathcal{M}})$, satisfying Theorem 3

```
Define f = 0 so N(f) = {0};
for j ∈ 1, ..., n do
    f ← f + f_j;
end
Compute V(N(f)) and V(P(N(f)));
return V(N(f)) / V(P(f)).
```

3.2 An Algorithm to efficiently compute $\mathrm{LC}(p_j)$

Through the analysis above, for $p_1, \ldots, p_k$ when $n > 1$, we can also reduce it to the case when $n = 1$. Before giving an algorithm to compute the leading coefficient of p_j, we introduce a new definition called High dimension duplicate in Definition 15.

The idea is to transform p_j of M_m into a number of elements in a higher Newton polytope. For example, we choose f with $\mathrm{supp}\{f\} = \{x_1, x_2, 1\}$. If we want to directly calculate p_x, we need to calculate $\sum_{i_1=0}^{m} \sum_{i_2=0}^{m-i_1} i_1 \approx \frac{m^3}{6}$. However, we can use the following method to compute $\mathrm{LC}(p_j)$. Now we think about the convex hull of $\{(0,0,0),(0,1,0),(1,0,0),(1,0,1)\}$ instead of $\{(0,0),(0,1),(1,0)\}$. To introduce the above idea more formally, we introduce the following definition.



A tetrahedron of $\{(0,0,0),(0,1,0),(1,0,0),(1,0,1)\}$,
Volume of tetrahedron $\frac{1}{6} \cdot 1 \cdot 1 \cdot 1 = \frac{1}{6}$,
$V(\mathcal{P}(f)) = 1$,
$\mathrm{LC}(p_1) = \frac{1}{6}$.

Fig. 3: $\mathrm{supp}\{f\} = \{x_1, x_2, 1\}$

Definition 15 (High dimension duplicate). *Suppose λ is a monomial in $\mathbb{Z}[x_1, \ldots, x_k]$, we use the following map from $\mathbb{Z}[x_1, \ldots, x_k]$ to $\mathbb{Z}^{k+h}$ to generate a h-dim duplicate over the j-th coordinate:*

$$\begin{array}{rccc} \mathcal{H}_{h,j}: & f & \longrightarrow & \mathbb{Z}^{k+h} \\ & x_1^{i_1} \cdot \ldots \cdot x_k^{i_k} & \longmapsto & (i_1, \ldots, i_k) \oplus \{0, i_j\}^h \end{array}$$

Here $\oplus$ means direct sum, for example, $(1,1) \oplus (0,0) = (1,1,0,0)$.

Definition 16 (Full High dimension duplicate). *Suppose λ is a monomial in $\mathbb{Z}[x_1, \ldots, x_k]$, we use the following map from $\mathbb{Z}[x_1, \ldots, x_k]$ to $\mathbb{Z}^{k+h}$ to generate a h-full dim duplicate over the j-th coordinate:*

$$\begin{array}{rccc} \overline{\mathcal{H}}_{h,j}: & f & \longrightarrow & \mathbb{Z}^{k+h} \\ & x_1^{i_1} \cdot \ldots \cdot x_k^{i_k} & \longmapsto & (i_1, \ldots, i_k) \oplus \{0, 1, \ldots, i_j\}^h \end{array}$$

We choose f with $\text{supp}\{f\} = \{x_1^2, x_2, 1\}$ as an example.

- $A(f) = \{(2,0), (0,1), (0,0)\}$;
- $N(f) = \{(2,0), (1,0), (0,1), (0,0)\}$;
- $\mathcal{H}_{0,j}(f) = \overline{\mathcal{H}}_{0,j}(f) = A(f)$
- $\mathcal{H}_{1,1}(f) = \{(2,0,0), (2,0,2), (0,1,0), (0,0,0)\}$;
- $\overline{\mathcal{H}}_{1,1}(f) = \{(2,0,0), (2,0,1), (2,0,2), (0,1,0), (0,0,0)\}$;
- $\mathcal{H}_{1,2}(f) = \{(2,0,0), (0,1,0), (0,1,1), (0,0,0)\}$;
- $\overline{\mathcal{H}}_{1,1}(f) = \{(2,0,0), (0,1,0), (0,1,1), (0,0,0)\}$.

There are some useful properties related to Definition 15 and Definition 16. The first property gives a relation between Definition 15 and Definition 16.

Lemma 6. *For $f \in \mathbb{Z}[x_1, \ldots, x_k]$, denote the 1-dimension duplicate of f and the full 1-dimension duplicate of f as $\mathcal{H}_{1,j}(f)$ and, $\overline{\mathcal{H}}_{1,j}(f)$ respectively. Then, we have*

$$H(\mathcal{H}_{1,j}(f)) = H(\overline{\mathcal{H}}_{1,j}(f)), \tag{5}$$

where $H(x)$ is the convex hull of x.

The proof of this lemma is provided in Appendix A. The second property is related to $\text{span}(A)$, $\text{span}(\mathcal{H}_{1,j}(f))$, or $\text{span}(\overline{\mathcal{H}}_{1,j}(f))$.

Lemma 7. *For $f \in \mathbb{Z}[x_1, \ldots, x_k]$, we have*

$$[\mathbb{Z}^{k+1} : H(\overline{\mathcal{H}}_{1,j}(f))] = [\mathbb{Z}^k : A(f)].$$

Moreover, if $\text{span}(A) = \mathbb{Z}^k$, then

$$\text{span}(\mathcal{H}_{1,j}(f)) = \text{span}(\overline{\mathcal{H}}_{1,j}(f)) = \mathbb{Z}^{k+1}.$$

The proof of this lemma is provided in Appendix B. Another useful property is as follows.

Lemma 8. *Let $f \in \mathbb{Z}[x_1, \ldots, x_k]$. Then*

$$\overline{\mathcal{H}}_{1,j}(f^m) = m\overline{\mathcal{H}}_{1,j}(f). \tag{6}$$

The proof of this lemma is provided in Appendix C.

We also have the following result.

Theorem 5. *Consider $f \in \mathbb{Z}[x_1, \ldots, x_k]$. Suppose N_j is the convex hull of 1-dim duplicate of f over the j-th coordinate, that is $N_j = H(\mathcal{H}_{1,j}(f))$, then $\text{LC}(p_j) = \frac{V(N_j)}{V(\mathcal{P}(f))}$.*

Proof. Define the j-th coordinate projection mapping as:

$$\begin{array}{rccc} \psi_j: & \mathbb{Z}^k & \longrightarrow & \mathbb{Z} \\ & (i_1,\ldots,i_k) & \longmapsto & i_j \end{array}$$

We want to prove that there exists polynomial p_j, such that $\sum_{\lambda\in mA}\psi_j(\lambda) = p_j(M)$, when m is large enough.

From the definition of $\overline{\mathcal{H}}_{1,j}$, we have

$$\sum_{\lambda\in\overline{\mathcal{H}}_{1,j}(f^m)} 1 = \sum_{\lambda\in mA}(\psi_j(\lambda)+1) = \sum_{\lambda\in mA}\psi_j(\lambda) + \sum_{\lambda\in mA} 1.$$

So we have

$$\sum_{\lambda\in mA}\psi_j(\lambda) = \sum_{\lambda\in\overline{\mathcal{H}}_{1,j}(f^m)} 1 - \sum_{\lambda\in mA} 1. \tag{7}$$

Then we analysis the growth of $\sum_{\lambda\in\overline{\mathcal{H}}_{1,j}(f^m)} 1$. From Lemma 8, we have

$$\sum_{\lambda\in\overline{\mathcal{H}}_{1,j}(f^m)} 1 = \sum_{\lambda\in m\overline{\mathcal{H}}_{1,j}(f)} 1.$$

And we know that there exists a polynomial $p'_j(m)$, when m is large enough, such that $\sum_{\lambda\in m\overline{\mathcal{H}}_{1,j}(f)} 1 = p'_j(m)$, that is

$$\sum_{\lambda\in\overline{\mathcal{H}}_{1,j}(f^m)} 1 = \sum_{\lambda\in m\overline{\mathcal{H}}_{1,j}(f)} 1 = p'_j(m),$$

where the leading coefficient of p'_j is the volume of the convex hull of $\overline{\mathcal{H}}_{1,j}(f)$ divided by the volume of the fundamental domain generated by $\overline{\mathcal{H}}_{1,j}(f)$, that is

$$\mathrm{LC}(p'_j) = \frac{V(H(\overline{\mathcal{H}}_{1,j}(f)))}{V(\mathcal{P}(\overline{\mathcal{H}}_{1,j}(f)))},$$

and the degree of p'_j is $k+1$.

From Lemma 6, we have $H(\mathcal{H}_{1,j}(f)) = H(\overline{\mathcal{H}}_{1,j}(f))$. Then

$$\mathrm{LC}(p'_j) = \frac{V(H(\mathcal{H}_{1,j}(f)))}{V(\mathcal{P}(\overline{\mathcal{H}}_{1,j}(f)))} = \frac{V(N_j)}{V(\mathcal{P}(\overline{\mathcal{H}}_{1,j}(f)))}.$$

Besides, using Lemma 7, we have

$$V(\mathcal{P}(\overline{\mathcal{H}}_{1,j}(f))) = V(\mathcal{P}(f)).$$

Therefore, we have

$$\mathrm{LC}(p'_j) = \frac{V(N_j)}{V(\mathcal{P}(\overline{\mathcal{H}}_{1,j}(f)))} = \frac{V(N_j)}{V(\mathcal{P}(f))}.$$

When m is large enough, there exists a polynomial $p_{\mathcal{M}}$ with degree k, such that

$$\sum_{\lambda \in mA} 1 = p_{\mathcal{M}}(m).$$

Therefore, using Equation (7), there exists a polynomial p_j with degree $k+1$, such that

$$\sum_{\lambda \in mA} \psi_j(\lambda) = p_j(m),$$

and the leading coefficient of p_j is $\frac{V(N_j)}{V(\mathcal{P}(f))}$. □

We provide the following Algorithm 5 for the corresponding calculation.

Algorithm 5: Calculate $\mathrm{LC}(p_j)$

Input: Set of monic polynomials $\mathcal{F} = \{f_1, \ldots, f_n\}$
Output: $\mathrm{LC}(p_j)$, satisfying Theorem 5

$\mathrm{LC}(p_j) = 0$;
if $\mathcal{M}_m$ is chosen as $\mathcal{M}_m^{(1)}$ in Theorem 3 **then**
  Compute $f = \prod_{j=1}^{n} f_j$;
  Compute the convex hull of $\mathcal{H}_{1,j}(f)$ as N_j in Definition 15;
  Compute $V(N_j)$ and $V(\mathcal{P}(f))$;
  $\mathrm{LC}(p_j) \leftarrow \frac{V(N_j)}{n^{k+1}V(\mathcal{P}(f))}$;
else
  Compute $f = \sum_{j=1}^{n} f_j$;
  Choose $\mathcal{M}_m^{(2)}$ in Theorem 4 as $\mathcal{M}_m$;
  Compute the convex hull of $\mathcal{H}_{1,j}(f)$ as N_j in Definition 15;
  Compute $V(N_j)$ and $V(\mathcal{P}(f))$;
  $\mathrm{LC}(p_j) \leftarrow \frac{V(N_j)}{V(\mathcal{P}(f))}$;
end
return $\mathrm{LC}(p_j)$.

Remark 4. Here we require A to be full rank. Otherwise, we can use an isometric projection mapping to achieve this, for example, f with $\mathrm{supp}\{f\} = \{x_1x_2, 1\}$. Although it corresponds to a convex hull of $\mathbb{Z}^2$, it is not actually two-dimensional. We consider another polynomial g with $\mathrm{supp}\{g\} = \{x_1, 1\}$ as a full rank version of f.

Remark 5. Obviously, computing $\mathrm{LC}(p_M), \mathrm{LC}(p_j)$ when m is large enough is as hard as computing $V(N(f)), V(H(\mathcal{H}_{1,j}(f)))$.

Remark 6. There exists a $\mathcal{O}(n^4)$ estimate algorithm to compute $V(N(f))$ if f has n monomials with error ϵ.

Remark 7. Using the same idea, we can prove that there exists a polynomial p with degree $k+h$ such that

$$\sum_{\lambda \in mA} \psi_j(\lambda)^h = p(m), \tag{8}$$

when m is large enough. We can use the same idea to calculate $\mathrm{LC}(p(m))$, which is the volume of $\mathcal{H}_{h,j}(f)$. We think it is an interesting problem whether the threshold of Equation (8) changes when h changes.

Remark 8. Observe that Automated Coppersmith requires more than one equation. Therefore, we can generalize Lemma 5 and provide similar results. However, since Lagrange interpolation method are no longer considered at this time, we will not elaborate on the bound of threshold N_{kh} further.

4 Algorithm for efficient calculation of $\mathrm{LC}(p_{\mathcal{F}})$

When we use Coppersmith's method, we require that each coefficient of f^m is used to construct the lattice. This implies that each monomial of f^m must be in the lattice. However, we know through the theory of sumset that when m is large enough, $A(f^m)$ tends to $N(f^m)$. This means that the monomial corresponding to each point of the convex hull can be obtained.

In [21], regarding f being a univariate polynomial, May selected the polynomials $x^j f^i M^{m-i}$ and $xf^m, x^2f^m, ..., x^t f^m$. If we only look at the first part, we actually need to find the corresponding optimal polynomial for each set of monomials $\{x^j \mid 0 \le j \le \delta m\}$, and this is the monomial set corresponding to $\mathrm{supp}\{f^m\}$. The latter part is looking for whether $x^t \mathrm{LM}(f^m)$ will be smaller than M^m, which is certainly possible. This is called Extended Strategy in [16].

However, for these shift polynomials, we can use another method to replace them, that is, choose $m-(i_1+\ldots+i_n)$ as $\max\{t-(i_1+\ldots+i_n),0\}$ with the optimal parameter t in Equation (3), which can also achieve the bound of the shift polynomial. It is worth to be noticed that the dimension in $\max\{t-(i_1+\ldots+i_n),0\}$ is smaller than the one using the Extended Strategy. The detail can be found in Appendix D.

In the next part, we mainly perform two tasks. The first one is to use the fact that when m is large enough, $A(f^m)$ tends to $N(f^m)$. This enables us to prove Heuristic 2 ($n=1$) in [23]. This also enables us to solve the open problem of this paper. The second part is to introduce a new variable t to achieve the effect of replacing part of the Extended Strategy in [16]. We propose our result in Theorem 7 and Corollart 2, which shows that introducing t is effective when the modular is an unknown modular.

4.1 Proof of Heuristic 2 of Automated Coppersmith

Following the growth of mA, we know that the set of the monomials tends to $mN(f)$. This implies that we can assume that A is a Saturated Newton Polytope, as in the following definition, showing the partition of mA.

Definition 17 (Partition of mA). *Suppose $A(f) \subset \mathbb{Z}^k$ is finite and saturated, $\mathrm{LM}(f)$ is related to $\alpha \in A$, then we define $S_\ell = \ell A + (m-\ell)\alpha$. We define $T_\ell = S_\ell \backslash S_{\ell-1}$ for $\ell = 1, \ldots, m$ and $T_0 = m\alpha$. Therefore, we can write mA as follows:*

$$mA = T_0 \cup T_1 \cup \ldots \cup T_m.$$

We call $\{T_\ell\}_{0\le\ell\le m}$ as a partition of mA.

Using $m = 2$ as an example, we have

$$\begin{aligned} T_0 &= 2\alpha, \\ T_1 &= (A+\alpha)\backslash 2\alpha, \\ T_2 &= 2A\backslash(A+\alpha). \end{aligned}$$

Then the following holds: $2A = T_0 \cup T_1 \cup T_2$.

The following result is directly connected to the former definition, which is useful to understand Algorithm 1.

Corollary 1. *Therefore, we can estimate $p_{\mathcal{F}}$ as follows, when m is large enough:*

$$p_{\mathcal{F}}(m) = \sum_{\ell=0}^{m} (m-\ell)|T_{m-\ell}|$$

The following theorem characterizes $p_{\mathcal{F}}$ very well.

Theorem 6. *If m is sufficiently large, then $p_{\mathcal{F}}$ is a polynomial of m with degree $k+1$, and its leading coefficient is $\frac{kV(A(f))}{(k+1)V(\mathcal{P}(f))}$.*

Proof. Suppose $\{T_\ell\}_{0\le\ell\le m}$ is a partition of mA denoted in Definition 17 and $\mathrm{LM}(f)$ is related to $\alpha \in A$. Then we have $mA = \bigcup_{\ell=0}^{m} T_\ell$.

For finite $A \subset \mathbb{N}^k$, we have $V(mA(f)) = m^k V(A(f))$. Then

$$|T_\ell| = (\ell^k - (\ell-1)^k)\frac{V(A(f))}{V(\mathcal{P}(f))} = (k\ell^{k-1} + \mathcal{O}(\ell^{k-2}))\frac{V(A(f))}{V(\mathcal{P}(f))}. \tag{9}$$

Using Corollary 1 and Equation (9), we can estimate $p_{\mathcal{F}}$ as follows

$$\begin{aligned} p_{\mathcal{F}} &= \sum_{\ell=0}^{m}(m-\ell)|T_{m-\ell}| = \sum_{\ell=0}^{m}\ell|T_\ell| \\ &= \sum_{\ell=0}^{m}\ell(\ell^k-(\ell-1)^k)\frac{V(A(f))}{V(\mathcal{P}(f))} \\ &= \frac{V(A(f))}{V(\mathcal{P}(f))}\sum_{\ell=0}^{m}\ell(\ell^k-(\ell-1)^k) \\ &= \frac{V(A(f))}{V(\mathcal{P}(f))}\sum_{\ell=0}^{m}\ell(\ell^k-\sum_{i=0}^{k}(-1)^i\binom{k}{i}\ell^{k-i}) \\ &= \frac{V(A(f))}{V(\mathcal{P}(f))}\sum_{\ell=0}^{m}\left(\sum_{i=0}^{k-1}(-1)^i\binom{k}{i+1}\ell^{k-i}\right). \end{aligned}$$

From Lemma 2, we know that, for each i_0, the sum

$$\sum_{\ell=0}^{m}(-1)^{i_0}\binom{k}{i_0+1}\ell^{k-i_0}$$

is a polynomial of m with degree $k+1-i_0$. Therefore, when m is large enough, $p_{\mathcal{F}}$ is a polynomial of degree $k+1$ in m and the leading coefficient of $p_{\mathcal{F}}$ is $\frac{kV(A(f))}{(k+1)V(\mathcal{P}(f))}$. □

4.2 Benefit of introducing the parameter t

We introduce a parameter t to partially overcome the Extended Strategy of Jochemsz and May [16] without altering the dimension of the lattice $\mathcal{L}$.

We replace $m-(i_1+\ldots+i_k)$ by $\max\{t-(i_1+\ldots+i_k),0\}$ with the optimal parameter t in Equation (3). Sometimes, we have $f(x_1,\ldots,x_k)\equiv 0 \mod \hat{M}$ instead of $f(x_1,\ldots,x_k)\equiv 0 \mod M$, where $\hat{M}$ is an unknown divisor of M. We need to make full use of this condition. For an unknown divisor, we will show that the extremum point of t is not $\frac{t}{m}=1$, which means that introduction t is beneficial. Suppose the known modular M is a multiple of the unknown divisor $\hat{M}$. In our scenario, we want to solve $\mathbb{Z}_{\hat{M},X_1,\ldots,X_k}(f_1,\ldots,f_n)$. To be more precise, it takes the following form

$$f_{[\lambda,\ell]} := \frac{\lambda}{\mathrm{LM}(f)^{\ell}}\cdot f^{\ell}\cdot M^{\max\{t-\ell,0\}}. \tag{10}$$

Now we have the following results.

Theorem 7. *If m is sufficiently large, then there exists a polynomial $p_{\mathcal{F}}(t,m)$ with total degree $k+1$ such that $\sum_{\lambda\in\mathcal{M}_m}\max\{t-\ell,0\}=p_{\mathcal{F}}(t,m)$, and the leading term of p_F is*

$$\frac{V(A(f))}{V(\mathcal{P}(f))}\left(m^k t-\frac{m^{k+1}-(m-t)^{k+1}}{k+1}\right).$$

Proof. We only need to sum up to t instead of m. We adopt a similar approach to the proof of Theorem 6, thus only considering the leading term. "$\approx$" indicates that only the leading term is considered. For example, $2m^{10}+\mathcal{O}(m^9)\approx 2m^{10}$.

For a finite $A\subset\mathbb{N}^k$, we have $V(mA(f))=m^kV(A(f))$. By Equation (9) it holds that

$$|T_\ell|=\left(\ell^k-(\ell-1)^k\right)V(A(f))\approx\frac{k\ell^{k-1}V(A(f))}{V(\mathcal{P}(f))}.$$

Therefore, we can estimate $p_{\mathcal{F}}$ as follows:

$$\begin{aligned} p_{\mathcal{F}} &= \sum_{\ell=0}^{t}(t-\ell)|T_{m-\ell}| \\ &\approx \sum_{\ell=0}^{t}(t-\ell)\cdot k\cdot(m-\ell)^{k-1}\frac{V(A(f))}{V(\mathcal{P}(f))} \\ &= \frac{V(A(f))}{V(\mathcal{P}(f))}(I_1 - I_2), \end{aligned}$$

where $I_1 = \sum_{\ell=0}^{t}(m-\ell)\cdot k\cdot(m-\ell)^{k-1}$ and $I_2 = \sum_{\ell=0}^{t}(m-t)\cdot k\cdot(m-\ell)^{k-1}$. First, we calculate $I_1 = \sum_{\ell=0}^{t}(m-\ell)\cdot k\cdot(m-\ell)^{k-1}$ using Lemma 2 as follows:

$$\begin{aligned} I_1 &= k\sum_{\ell=0}^{t}(m-\ell)^k \\ &= k\sum_{\ell=0}^{m}\ell^k - k\sum_{\ell=0}^{m-t}\ell^k \\ &\approx k\left(\frac{m^{k+1}}{k+1} - \frac{(m-t)^{k+1}}{k+1}\right). \end{aligned}$$

Second, we calculate $I_2 = \sum_{\ell=0}^{t}(m-t)\cdot k\cdot(m-\ell)^{k-1}$ using Lemma 2 as follows:

$$\begin{aligned} I_2 &= (m-t)k\sum_{\ell=0}^{t}(m-\ell)^{k-1} \\ &= (m-t)k\left(\sum_{\ell=0}^{m}\ell^{k-1} - \sum_{\ell=0}^{m-t}\ell^{k-1}\right) \\ &\approx (m-t)k\left(\frac{m^k}{k} - \frac{(m-t)^k}{k}\right). \end{aligned}$$

Therefore, we continue to calculate $p_{\mathcal{F}}$ as

$$p_{\mathcal{F}} \approx \frac{V(A(f))}{V(\mathcal{P}(f))}\left(m^k t - \frac{m^{k+1}-(m-t)^{k+1}}{k+1}\right).$$

So the leading term of $p_{\mathcal{F}}$ is $\frac{V(A(f))}{V(\mathcal{P}(f))}\left(m^k t - \frac{m^{k+1}-(m-t)^{k+1}}{k+1}\right)$. □

Moreover, we show that the introduction of t will lead to a better result when the modulus is an unknown divisor of a known integer.

Corollary 2. *For unknown disivor, the extremum point of t is not $\frac{t}{m} = 1$, which means the introduction of t is useful.*

Proof. Suppose $p = M^\beta$ is an unknown divisor of M, now we consider the following inequality

$$X_1^{p_1} \cdot \ldots \cdot X_k^{p_k} \cdot M^{p_{\mathcal{F}}} < p^{t \dim(\mathcal{L})}.$$

Only focus on terms that contain t, we have

$$m^k t - \frac{m^{k+1} - (m-t)^{k+1}}{k+1} < \beta t m^k.$$

Suppose $\frac{t}{m} = \delta$, the above inequality can be rewritten as

$$(k+1)\delta - (1 - (1-\delta)^{k+1}) - (k+1)\beta\delta < 0.$$

The optimal value for δ is $\delta_0 = 1 - (1-\beta)^{\frac{1}{k}}$. Hence, when $\beta \neq 1$, we have $\delta_0 \neq 1$. □

Remark 9. The leading term of $p_{\mathcal{F}}$ is the most important part. Considering at $\frac{kV(A(f))}{(k+1)V(\mathcal{P}(f))}$ in the leading coefficient of $p_{\mathcal{F}}$, choosing a polynomial f whose Newton Polytope is smaller yields better results.

If we have more than one equation, we must carefully select the polynomials with different leading monomials. Additionally, a smaller Newton Polytope leads to better results.

Note our Newton Polytope based Coppersmith's method isn't only used to compute the bound for small roots but used to design the lattice for Coppersmith's method.

5 Experiments

Our experiments were performed using Sagemath 10.3 on a MacBook Pro with an M1 chip, boasting a maximum CPU clock rate of 3.2 GHz. We first present an example to illustrate the phenomenon of local convergence in Section 5.1. Then, we provide details of the example mentioned in the abstract in Section 5.2. Finally, we provide more examples to validate the efficiency of our algorithm in Section 5.3.

5.1 Example for Local Convergence

Let's illustrate the phenomenon of local convergence encountered when using the interpolation method proposed in [23]. Consider the following polynomial f with

$$\text{supp}\{f\} = \{x_1^3, x_1x_2, x_1x_3, x_2, x_3^2x_4^2, x_4^5, 1\}.$$

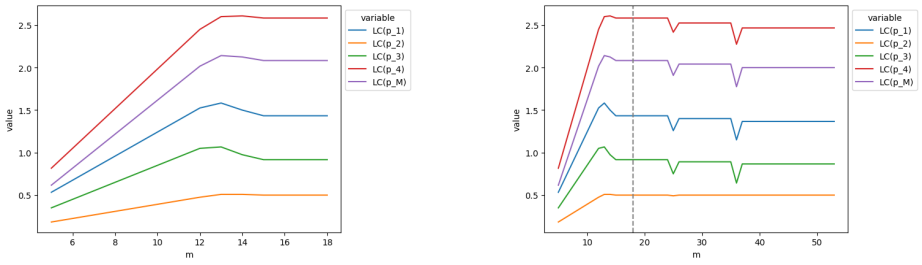
We compute f^m and then track the corresponding $p_{\mathcal{M}}(m)$ and $p_j(m)$. Here, f has four variables, i.e., $k = 4$. According to the results of [17], we know that

$p_{\mathcal{M}}(m)$ and $p_j(m)$ should be polynomials in m with degree 5 when $m > N_{\|\langle}$. However, according to the result in [11], the upper bound,

$$N_{kh} = (2 \times 7 \times 5)^{8\times 7} \approx 2^{343},$$

is extremely large, making it impractical.

A natural idea is to consider the values of $p_{\mathcal{M}}$ and p_j at $m-5, m-4, m-3, m-2, m-1, m$ when $m \geq 5$, and then interpolate to obtain a fifth-degree polynomial, recording the leading coefficient. The relevant numerical values are presented in Figure 4. In Figure 4a, we observe that as m increases, the leading coefficient of $p_{\mathcal{M}}$ stabilizes at 25/12. However, this is incorrect. Continuing to increase m, we eventually find that the leading coefficient of $p_{\mathcal{M}}$ stabilizes at 2. All the information can be found in Figure 4b, where the part to the left of the gray dashed line corresponds to Figure 4a.



(a) No.1 (b) No.2

Fig. 4: An example for Local Convergence

In practical applications of the interpolation method, one might encounter issues with results getting stuck in local convergence. Additionally, the utilization of N_{kh} as proposed in [11] may lead to overly large computations. However, our method effectively tackles these challenges.

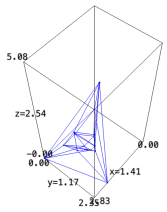
5.2 A Toy Example

We conducted experiments to demonstrate the significant reduction in computational time achieved by our new method.

For a slightly more complex example:

$$\begin{cases} f_1 \text{ with } \operatorname{supp}\{f_1\} = \{x_3^3x_4^2, x_2^2, x_1x_2, 1\} \\ f_2 \text{ with } \operatorname{supp}\{f_2\} = \{x_4^5, x_3^2x_1, x_1^3, x_2, 1\}. \end{cases}$$

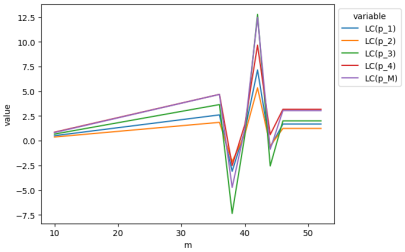
We must choose $m > 40$ if we select $\mathcal{M}_m$ as in Theorem 3, which costs more than 20 minutes! However, if we just compute the volume of Newton Polytope, the time costs is less than 0.5s!



The leading coefficient of p_M is 583/192,
The leading coefficient of p_1 is 431/256,
The leading coefficient of p_2 is 317/256,
The leading coefficient of p_3 is 769/384,
The leading coefficient of p_4 is 2437/768.

Now we see the following figure to see how large m needs to be to satisfy the interpolation and get the value we want, that is, when the leading coefficient remains stable.

The leading coefficient of p_M is 583/192,
The leading coefficient of p_1 is 431/256,
The leading coefficient of p_2 is 317/256,
The leading coefficient of p_3 is 769/384,
The leading coefficient of p_4 is 2437/768.



Seeing the above figure, we must choose $m > 40$ if we select $\mathcal{M}_m$ as in Theorem 3, that is why it costs about 10 minutes!

5.3 More Experiments

We propose a metric for polynomials. This metric is the ratio of the number of monomials in f to the number of points in the corresponding saturated Newton polytope. Intuitively, the smaller this ratio, the larger m needs to be chosen to approach the Hilbert function, and the more apparent our advantage becomes. We call this metric *the saturation of* f, denoted as sat(f). Formally speaking, we define the saturation of f as follows:

$$\mathrm{sat}(f) = \frac{|\mathrm{supp}\{f\}|}{\text{the number of points in } A(f)}. \tag{11}$$

We conducted experiments for different saturations. It is worth noting that this result also depends on the number of variables involved. We restrict our analysis to cases with up to four variables. Naturally, as the number of variables increases, the complexity grows, making our advantage more pronounced.

In Section 3, there are two ways to select the monomial sets: one is to multiply all the polynomials together (Theorem 3), and the other one is to add them (Theorem 4). Therefore, we compared these two approaches. For our previously mentioned toy example in Sec. 5.2, we chose to multiply the polynomials. Now we record the time needed for the method that adds them together, which is shown in (Table 1, No. 2). No. 3 represents the toy example mentioned earlier, facilitating the comparison.

The required values of m can be found in Fig 5 in Appendix E. As m increases, we select new m along with their corresponding values to compute Lagrange

sat(f)	Method for choosing $\mathcal{M}$	Time for [23] (s)	Time for our method (s)
1	Theorem 3	0.027	0.063
<1	Theorem 4	23.507	0.078
<1	Theorem 3	681.299≈10 minutes	0.271

Table 1: Experimental results for Section 3.

interpolation and record the leading coefficients. Hence, when m is sufficiently large, the leading coefficient remains constant. It can be observed that when f is saturated, there is no need to compute f^m for large m. Consequently, sometimes our algorithm may be slower, but it still manages to compute the solution very quickly, less than one second.

We also conducted experimental simulations for the results in Section 4, focusing solely on the case where $n = 1$. Since we characterized the polynomials constructing the lattice, there is no need to search for the optimal polynomial for each monomial, unlike in the Automated Coppersmith method. This saved a significant amount of time.

sat(f)	Time for [23] (s)	Time for our method (s)
1	10.886	0.223
<1	46.157≈ 1 minute	0.037
<1	2238.237≈ 2 hours	0.069
<0.5	828.326≈ 14 minutes	0.073

Table 2: Experimental results for Section 4.

The required values of m can be found in Fig 6 in Appendix E. The detailed information about the polynomials used in our experiments in Table 3 in Appendix E. It is worth noting that a higher proportion of randomly selected polynomials f are not saturated. When f is not saturated, the Automated Coppersmith method requires selecting a large m, leading to increased computational time.

6 Conclusion

In this paper, we introduced a new and powerful mathematical tool called Growth of Sumsets Theory from Additive Combinatorics. We revisited the Jochemsz-May strategy as well as the work of Meers and Nowakowski and pointed out that their bounds can be obtained by calculating the leading coefficient of some Hilbert function, which is exactly the volume of the corresponding Newton polytope. To this end, we introduced the concept of Sumsets theory and proposed

a series of related results and algorithms that improve the former methods for solving modular polynomial equations.

References

1. Ajtai, M.: The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract). In: Symposium on the Theory of Computing (1998)
2. 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, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3494, pp. 251–267. Springer (2005). https://doi.org/10.1007/11426639_15, https://doi.org/10.1007/11426639_15
3. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key less than $n^{0.292}$. In: EUROCRYPT '99. Lecture Notes in Computer Science, vol. 1592, pp. 1–11. Springer (1999). https://doi.org/10.1007/3-540-48910-X_1
4. Boneh, D., Halevi, S., Howgrave-Graham, N.: The modular inversion hidden number problem. In: Advances in Cryptology—ASIACRYPT 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast, Australia, December 9–13, 2001 Proceedings 7. pp. 36–51. Springer (2001)
5. Boneh, D., Venkatesan, R.: Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In: Annual International Cryptology Conference. pp. 129–142. Springer (1996)
6. Coppersmith, D.: Finding a small root of a univariate modular equation. In: EUROCRYPT '96. Lecture Notes in Computer Science, vol. 1070, pp. 155–165. Springer (1996). https://doi.org/10.1007/3-540-68339-9_14
7. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. J. Cryptol. **10**(4), 233–260 (1997). https://doi.org/10.1007/s001459900030
8. Curran, M.J., Goldmakher, L.: Khovanskii's theorem and effective results on sumset structure (Dec 2021). https://doi.org/10.48550/arXiv.2009.02140
9. Feng, Y., Nitaj, A., Pan, Y.: Provable Automated Coppersmith for linear equations and its applications (2024)
10. Granville, A., Shakan, G.: The frobenius postage stamp problem, and beyond (Apr 2020)
11. Granville, A., Shakan, G., Walker, A.: Effective results on the size and structure of sumsets. Combinatorica **43**(6), 1139–1178 (Dec 2023). https://doi.org/10/gtf3bf
12. Granville, A., Walker, A.: A tight structure theorem for sumsets (Mar 2021)
13. Heninger, N., Ryan, K.: The hidden number problem with small unknown multipliers: Cryptanalyzing MEGA in six queries and other applications. In: Boldyreva, A., Kolesnikov, V. (eds.) Public-Key Cryptography - PKC 2023 - 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7-10, 2023, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13940, pp. 147–176. Springer (2023). https://doi.org/10.1007/978-3-031-31368-4_6, https://doi.org/10.1007/978-3-031-31368-4_6
14. Howgrave-Graham, N.: Approximate integer common divisors. In: CaLC 2001. Lecture Notes in Computer Science, vol. 2146, pp. 51–66. Springer (2001). https://doi.org/10.1007/3-540-44670-2_6

15. Jacobi, C.G.J.: De usu legitimo formulae summatoriae maclaurinianae. (1834)
16. Jochemsz, E., May, A.: A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. In: ASIACRYPT 2006. Lecture Notes in Computer Science, vol. 4284, pp. 267–282. Springer (2006). https://doi.org/10.1007/11935230_18
17. Khovanskii, A.G.: Newton polyhedron, Hilbert polynomial, and sums of finite sets. Functional Analysis and Its Applications **26**(4), 276–281 (1992). https://doi.org/10/dsb2rr
18. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen **261**, 515–534 (1982)
19. Lu, Y., Peng, L., Zhang, R., Hu, L., Lin, D.: Towards optimal bounds for implicit factorization problem. In: SAC 2016. pp. 462–476. Springer (2016)
20. May, A.: Cryptanalysis of unbalanced RSA with small crt-exponent. In: Yung, M. (ed.) Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings. Lecture Notes in Computer Science, vol. 2442, pp. 242–256. Springer (2002). https://doi.org/10.1007/3-540-45708-9_16, https://doi.org/10.1007/3-540-45708-9_16
21. May, A.: New RSA Vulnerabilities Using Lattice Reduction Methods. Ph.D. thesis, University of Paderborn (2003)
22. May, A., Nowakowski, J., Sarkar, S.: Approximate divisor multiples - factoring with only a third of the secret crt-exponents. IACR Cryptol. ePrint Arch. p. 271 (2022), https://eprint.iacr.org/2022/271
23. Meers, J., Nowakowski, J.: Solving the hidden number problem for CSIDH and CSURF via Automated Coppersmith (2023)
24. Micheli, G.D., Heninger, N.: Recovering cryptographic keys from partial information, by example. Cryptology ePrint Archive, Paper 2020/1506 (2020), https://eprint.iacr.org/2020/1506, https://eprint.iacr.org/2020/1506
25. Mumford, D.: Algebraic geometry I: complex projective varieties, vol. 221. Springer (1976)
26. Nathanson, M.B.: Sums of finite sets of integers. The American Mathematical Monthly **79**(9), 1010–1012 (1972), http://www.jstor.org/stable/2318072
27. Nathanson, M.B.: Growth of sumsets in abelian semigroups (Feb 2000)
28. Sarkar, S., Maitra, S.: Approximate integer common divisor problem relates to implicit factorization. IEEE Trans. Inf. Theory **57**(6), 4002–4013 (2011). https://doi.org/10.1109/TIT.2011.2137270
29. Stanley, R.P.: Hilbert functions of graded algebras. Advances in Mathematics **28**(1), 57–83 (1978)
30. Takayasu, A., Lu, Y., Peng, L.: Small crt-exponent rsa revisited. Cryptology ePrint Archive, Paper 2017/092 (2017), https://eprint.iacr.org/2017/092, https://eprint.iacr.org/2017/092
31. Wang, S., Qu, L., Li, C., Fu, S.: A better bound for implicit factorization problem with shared middle bits. Sci. China Inf. Sci. **61**(3), 032109:1–032109:10 (2018). https://doi.org/10.1007/s11432-017-9176-5
32. Wu, J.D., Chen, F.J., Chen, Y.G.: On the structure of the sumsets. Discrete Mathematics **311**(6), 408–412 (2011). https://doi.org/https://doi.org/10.1016/j.disc.2010.11.014, https://www.sciencedirect.com/science/article/pii/S0012365X10004449

A Proof of Lemma 6

Proof. We take this proof into two parts. One is

$$H(\mathcal{H}_{1,j}(f)) \subset H(\overline{\mathcal{H}}_{1,j}(f))$$

and the other is

$$H(\overline{\mathcal{H}}_{1,j}(f)) \subset H(\mathcal{H}_{1,j}(f)).$$

For the first part, we know $A(f_1) \subset A(f_2)$ implies $H(f_1) \subset H(f_2)$. It is obvious that $\mathcal{H}_{1,j}(f) \subset \overline{\mathcal{H}}_{1,j}(f)$, so we have

$$H(\mathcal{H}_{1,j}(f)) \subset H(\overline{\mathcal{H}}_{1,j}(f)).$$

For the second part, suppose $(i_1, \ldots, i_k) \in A(f)$, we know

$$\{(i_1, \ldots, i_k, 0), (i_1, \ldots, i_k, i_j)\} \in \mathcal{H}_{1,j}(f),$$

so $\{(i_1, \ldots, i_k, 0), (i_1, \ldots, i_k, 1), \ldots, (i_1, \ldots, i_k, i_j)\}$ exist in $H(\mathcal{H}_{1,j}(f))$.

Therefore, we have $\overline{\mathcal{H}}_{1,j}(f) \subset H(\mathcal{H}_{1,j}(f))$. Then we have

$$H(\overline{\mathcal{H}}_{1,j}(f)) \subset H(H(\mathcal{H}_{1,j}(f))).$$

We know $H(H(\cdot)) = H(\cdot)$, so we have $H(\overline{\mathcal{H}}_{1,j}(f)) \subset H(\mathcal{H}_{1,j}(f))$. Therefore, we know $H(\mathcal{H}_{1,j}(f)) = H(\overline{\mathcal{H}}_{1,j}(f))$ holds. □

B Proof of Lemma 7

Proof. From the definition of $\overline{\mathcal{H}}_{1,j}(f)$, we known it is saturated at the $k+1$-th coordinate. So $[\mathbb{Z}^{k+1} : H(\overline{\mathcal{H}}_{1,j}(f))] = [\mathbb{Z}^k : A(f)]$ holds true.

Obviously, the $k+1$-th coordinate is generated by j-th coordinate, when $(0, 0, ..., 1, ..., 0) \in \mathrm{span}(A)$, we know $(0, 0, ..., 1, ..., 0, 1)$ and $(0, 0, ..., 1, ..., 0, 0)$ are exist in $\mathcal{H}_{1,j}(f)$ and $\overline{\mathcal{H}}_{1,j}(f)$. So do $(0, 0, ..., 0, ..., 0, 1)$. □

C Proof of Lemma 8

Proof. Here we prove a stronger conclusion, where the "+" represents Minkowski sum:

$$\overline{\mathcal{H}}_{1,j}(f_1 \cdot f_2) = \overline{\mathcal{H}}_{1,j}(f_1) + \overline{\mathcal{H}}_{1,j}(f_2).$$

If this holds, then considering the m-fold addition of $\overline{\mathcal{H}}_{1,j}(f)$, we have

$$\overline{\mathcal{H}}_{1,j}(f^m) = m\overline{\mathcal{H}}_{1,j}(f).$$

First, let's consider the scenario where $A(f_1)\backslash\{0\}$ and $A(f_2)\backslash\{0\}$ are both single points. Assuming $A(f_1)\backslash\{0\} = \{(i_1, \ldots, i_k)\}$ and $A(f_2)\backslash\{0\} = \{(i'_1, \ldots, i'_k)\}$, if we only consider the first k components, then $\overline{\mathcal{H}}_{1,j}(f_1 \cdot f_2)$ and $\overline{\mathcal{H}}_{1,j}(f_1) + \overline{\mathcal{H}}_{1,j}(f_2)$ are equal.

For the $(k+1)$-th component, which is generated by a full high dimension duplicate, we know that for $(i_1, \ldots, i_k) \in A(f_1 \cdot f_2)$, we have $\overline{\mathcal{H}}_{1,j}(\{(i_1, \ldots, i_k)\}) \in \overline{\mathcal{H}}_{1,j}(f_1) + \overline{\mathcal{H}}_{1,j}(f_2)$.

Conversely, for $\lambda \in \overline{\mathcal{H}}_{1,j}(f_1) + \overline{\mathcal{H}}_{1,j}(f_2)$, suppose its first components is $(i'_1, \ldots, i'_k)$, then we know $(i'_1, \ldots, i'_k) \in A(f_1 \cdot f_2)$. Therefore

$$\overline{\mathcal{H}}_{1,j}(f_1 \cdot f_2) = \overline{\mathcal{H}}_{1,j}(f_1) + \overline{\mathcal{H}}_{1,j}(f_2).$$

Hence, $\overline{\mathcal{H}}_{1,j}(f^m) = m\overline{\mathcal{H}}_{1,j}(f)$.

□

D More results for Section 4

In Section 4, we said for univariate shift polynomials, we can use another method to replace them, that is, choose $m-(i_1+\ldots+i_n)$ as $\max\{t-(i_1+\ldots+i_n), 0\}$ with the optimal parameter t in Equation (3), which can also achieve the bound of the shift polynomial. Besides, it is worth to be motioned that the dim in this method is smaller than using Extended Strategy. Now we give detailed proof to show its correctness.

Theorem 8. *Let N be an integer of unknown factorization, which has a divisor $p \geq N^\beta$. Let $f(x)$ be a univariate monic polynomial of degree δ. We can choose $m-(i_1+\ldots+i_n)$ as $\max\{t-(i_1+\ldots+i_n), 0\}$ with the optimal parameter t in Equation (3) to achieve the bound of solving $f(x) \equiv 0 \mod p$ in Theorem 6 in [21].*

Proof. For $f(x) \in \mathbb{Z}[x]$ with degree δ, we know $N(f) = [0, \delta]$. And we need to compute $p_{\mathcal{M}}, p_1, p_{\mathcal{F}}$ in the following equation:

$$X^{p_1} N_{p_{\mathcal{F}}} < p^{p_{\mathcal{M}}}. \tag{12}$$

From Section 3, we know that the leading coefficient satisfy

$$\begin{aligned} p_{\mathcal{M}} &\approx \frac{V(N(f))}{V(\mathcal{P}(f))} mt = \frac{\delta}{\gcd(A(f))} mt, \\ p_1 &\approx \frac{V(N_1)}{V(\mathcal{P}(f))} m^2 = \frac{\delta^2}{2\gcd(A(f))} m^2. \end{aligned}$$

From Theorem 7, we know the leading term of p_F is

$$\begin{aligned} \mathrm{LC}(p_F) &= \frac{V(A(f))}{V(\mathcal{P}(f))}(mt - \frac{m^2-(m-t)^2}{2}) \\ &= \frac{\delta}{\gcd(A(f)}(mt - \frac{m^2-(m-t)^2}{2}) \end{aligned}$$

Suppose $X = N^\gamma$ and $\frac{t}{m} = \alpha$, then we can rewrite Equation (12) is as follows:

$$\frac{\gamma \cdot \delta}{2} + \alpha - \frac{1-(1-\alpha)^2}{2} < \beta \cdot \alpha, \tag{13}$$

where the extremum point for α is $\alpha = \beta$.

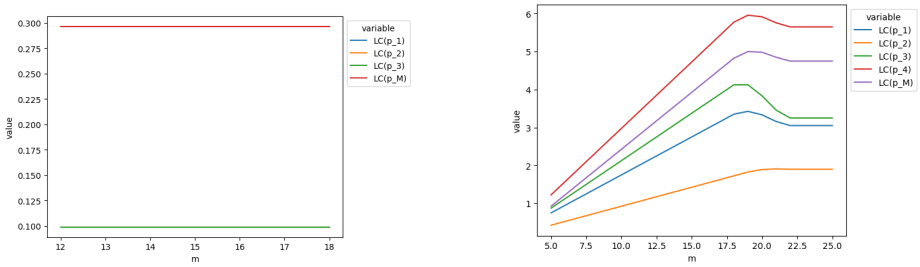
Therefore, Equation (13) yields

$$\gamma < \frac{\beta^2}{\delta},$$

which is the same as the bound in Theorem 6 in [21]. □

E Details for f in Section 5

The required values of m in Table 1.

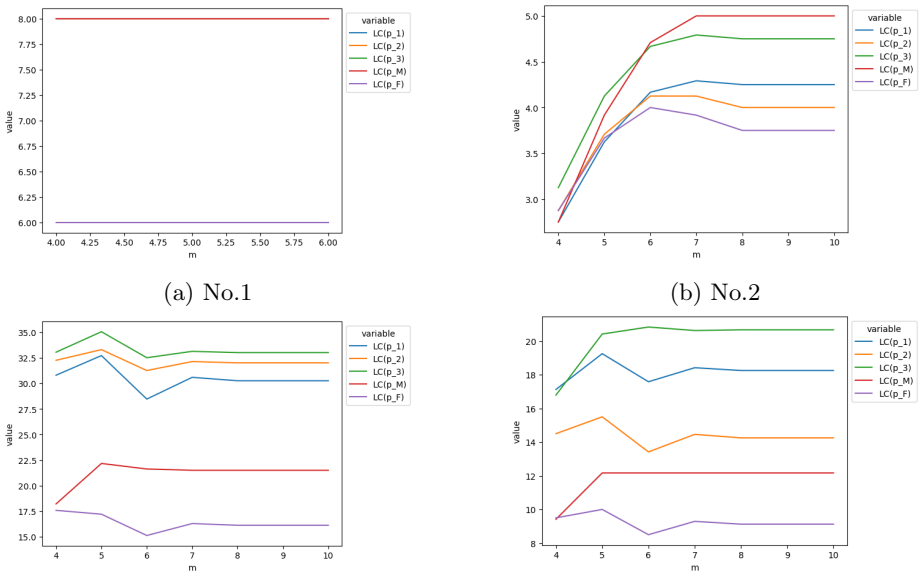


(a) No.1 (b) No.2

Fig. 5: Corresponding values of m in Table 1.

The required values of m in Table 2.

We also provide detailed information about the polynomials used in our experiments in Table 3.



(a) No.1 (b) No.2

(c) No.3 (d) No.4

Fig. 6: Corresponding values of m in Table 2.

No.	$\text{supp}\{f\}$
f in Table 1	
No.1	$\begin{cases} f_1 \text{ with } \text{supp}\{f_1\} = \{x_1 * x_2, x_1, x_2, 1\} \\ f_2 \text{ with } \text{supp}\{f_2\} = \{x_2 * x_3, x_2, x_3, 1\} \\ f_3 \text{ with } \text{supp}\{f_2\} = \{x_1 * x_3, x_1, x_3, 1\} \end{cases}$
No.2	$\begin{cases} f_1 \text{ with } \text{supp}\{f_1\} = \{x_3^3x_4^2, x_2^2, x_1x_2, 1\} \\ f_2 \text{ with } \text{supp}\{f_2\} = \{x_4^5, x_3^2x_1, x_1^3, x_2, 1\} \end{cases}$
No.3	$\begin{cases} f_1 \text{ with } \text{supp}\{f_1\} = \{x_3^3x_4^2, x_2^2, x_1x_2, 1\} \\ f_2 \text{ with } \text{supp}\{f_2\} = \{x_4^5, x_3^2x_1, x_1^3, x_2, 1\} \end{cases}$
f in Table 2	
No.1	$\text{supp}\{(x_1 * x_2 + x_1 + x_2 + 1) * (x_2 * x_3 + x_2 + x_3 + 1) * (x_1 * x_3 + x_1 + x_3 + 1)\}$
No.2	$\text{supp}\{x_1^3 + x_1 * x_2 + x_1 * x_3^2 + x_2^2 * x_3^3 + x_2^2 + x_2 + 2\}$
No.3	$\text{supp}\{(x_3^3 * x_2^2 + x_2^2 + x_1 * x_2 + 1) * (x_3^2 * x_1 + x_1^3 + x_2 + 1)\}$
No.4	$\text{supp}\{(x_2^2 + x_1^3 * x_2 + 1) * (x_1^2 + x_2 * x_3 + x_3^4 + 1)\}$

Table 3: Details of f in Experiments.