On the Optimality of Lattices for the Coppersmith Technique

Yoshinori Aono* Manindra Agrawal† Takakazu Sato‡ Osamu Watanabe§

Abstract

We investigate the Coppersmith technique [6] for finding solutions of a univariate modular equation within a range given by range parameter U. This technique converts a given equation to an algebraic equation via a lattice reduction algorithm, and the choice of the lattice is crucial for the performance of the technique. This paper provides a way to analyze a general type of limitation of this lattice construction. Our analysis bounds the possible range of U from above that is asymptotically equal to the bound given by the original result of Coppersmith. It means that Coppersmith has already given the best lattice construction. To show our result, we establish a framework for the technique by following the reformulation of Howgrave-Graham [13], and derive a condition, which we call the *lattice condition*, for the technique to work. We then provide a way to analyze a bound of U for achieving the lattice condition. Technically, we show that (i) the original result of Coppersmith achieves an optimal bound for U when constructing a lattice in a standard way. We then show evidence supporting that (ii) a non-standard lattice construction is generally difficult. We also report on computer experiments demonstrating the tightness of our analysis.

1 Introduction

Coppersmith [6] introduced a polynomial-time algorithm, which we refer to as the *Coppersmith technique*, for finding solutions of a modular equation

$$F(x) = x^D + a_{D-1}x^{D-1} + \cdots + a_0 \equiv 0 \pmod{N} \tag{1}$$

within the range of $|x| < N^{1/D-\varepsilon}$, and showed that it can be used to design an attacking algorithm for RSA cryptography. (Here, for any A, $0 < A < N$, the notation $|x| < A$ under modulo N means that x is an integer satisfying $0 \leq x < A$ or $N - A < x < N$.) Since his work and the reformulation by Howgrave-Graham [13], it has gained attention in relation to attack several cryptographies; (e.g., [2, 3]). The technique has also been generalized for the multivariate case.

The outline of the Coppersmith technique is (i) converting a given modular equation to a certain algebraic equation keeping the same small solutions by using a lattice reduction algorithm and (ii) solving the algebraic equation by a numerical method. One key point of this technique is constructing a good lattice for the lattice reduction algorithm. For instance, considering a bivariate modular equation for RSA cryptanalysis the original result of [3] has been improved essentially by defining better lattices [9, 1]. There should clearly be some limit in such improvements. Here, we focus on the univariate case and investigate the optimality of the lattice construction for the Coppersmith technique. We demonstrate that for solving (1) the range of $|x| < N^{1/D-\varepsilon}$ Coppersmith achieved based on his lattice construction is "in some sense" optimal.

Note that some investigations have shown the limit of polynomial-time algorithms. Konyagin and Steger [16] gave an upper bound of the number of roots of (1) within the range of $|x| < U$,

*National Institute of Information and Communications Technology aono@nict.go.jp
†Dept. of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India.
‡Dept. of Mathematics, Tokyo Institute of Technology, Tokyo, Japan.
§Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan.

which becomes exponential in $\log N$ when $U = N^{1/D+\varepsilon}$. It is also shown [20] that the bound is attained by an equation of the form $x^r \equiv 0 \pmod{p^r}$ for a prime number p and integer r. This clearly provides the limit of any algorithm that runs in polynomial-time in $\log N$. That is, there are *some* equations such that no polynomial-time algorithm works to find *all* solutions within the range of $|x| < N^{1/D+\varepsilon}$. However, their example is somewhat extreme, and insufficient for showing the hardness of solving the particular equations defined for attacking cryptographies. In fact, for most such equations, the number of solutions is easily shown to be quite small. Our objective is to provide a technique to analyze the limit of the Coppersmith technique that is applicable for equations for attacking cryptographies.

Our Results: In this paper, we show a general type limitation of lattices used in the Coppersmith technique for solving any univariate equation (1), by which we can show that the Coppersmith's bound $U = N^{1/D-\varepsilon}$ is best (except for the choice of ε). Our result consists of two technical results. We investigate a certain condition — the *lattice condition* — which is sufficient for the Coppersmith technique to work. This condition is essentially determined by a set of polynomials used for constructing a lattice. We first prove that this bound is not satisfied for $U \geq N^{1/D}$ if the lattice is constructed based on "standard" integer coefficient polynomials. This paper establishes the notion of standard polynomials, in short, it is a natural generalization of a way to select polynomials that have been used in the previous work. We then consider a non-standard lattice construction, that is, constructing a lattice based on non-standard integer coefficient polynomials. We show that any non-standard construction indeed leads either (i) a reduction of the original equation (1) to a strictly simpler one, or (ii) derives a non-trivial factor of the modulo N that we assume difficult to compute. Moreover, neither reduction requires the Coppersmith technique. That is, we show that such a non-standard construction will lead a better way to solve the original problem than the Coppersmith technique. Thus, from these results, we can claim that the range larger than $N^{1/D}$ cannot be achieved by the Coppersmith technique using any lattice construction. Note that the lattice condition is sufficient, and the Coppersmith technique sometimes works even if the lattice condition is not satisfied. We thus discuss the tightness of the lattice condition for showing the limits of the Coppersmith technique. The lattice condition is derived from two inequalities — one for a key algebraic property and one for the length of a short lattice vector. While it seems difficult to show that they are tight, we justify our analysis from the following two points: (i) we show that a limit similar to $U < N^{1/D}$ still holds unless significantly better inequalities can be used, and (ii) we show computer experiments that indicate that these inequalities cannot be significantly improved.

Another contribution of this paper is description of a method for improving lattices. To show the limitations of standard lattice construction, we give a method for decreasing the determinant of a given lattice, which can in fact be given as an explicit procedure. Thus, this method may be used to improve lattices in the Coppersmith technique, providing a better solvable range $U < N^{1/D-\varepsilon}$ with a smaller ε. It should also be noted that this method can be extended for multivariate situations, though we state the procedure for the univariate situation.

Related Work: We first briefly survey applications of the univariate case of the Coppersmith technique in cryptography. Before Coppersmith's work, there were similar ideas for attacking cryptographic schemes. Vallée, Girault and Toffin [25] proposed a lattice based attack for the Okamoto-Shiraishi signature scheme [21] that uses a quadratic inequality. Håstad [12] also proposed a procedure for solving simultaneous modular equations by converting them to one modular equation. Unfortunately, its solvable range is slightly weaker since lattice construction is extremely simple. Based on such previous works, Coppersmith [6] proposed his method for solving general univariate equations and its application for recovering an RSA message with its $(1 - 1/e)$-fraction of MSBs. After his work, Howgrave-Graham [13] reformulated the technique, and many applications have been proposed in the cryptographic area. Shoup [24], for instance, gave an interesting application for proving the security of the RSA-OAEP encryption scheme with $e = 3$. Our result

would show the limitations of these approaches. For example, the RSA message cannot be recovered from its MSBs that have length of less than $(1-1/e)\log_2 N$ by using the direct usage of the Coppersmith technique.

While in this paper, we investigate the limit of the direct usage of the Coppersmith technique, it should be noted here that several extensions of the techniques have been proposed for extending the range $|x| < N^{1/d-\varepsilon}$; see, for example, the survey papers by Coppersmith [7] and by May [20, Chap. 10]. Unfortunately, though, the improvements of these extensions are relatively mild.

The rest of this paper is as follows. Section 2 provides some necessary technical background. Section 3 follows [13] and precisely defines the Coppersmith technique and the lattice condition. Section 4 derives a necessary condition for the solution range to achieve the lattice condition under the standard lattice construction. Section 5 discusses what we are able to compute from a non-standard construction. Section 6 discusses the tightness of our analysis and reports on computer experiments. Finally, Section 7 concludes the paper with remarks. The Appendix gives most of our technical arguments as well as some related topics.

2 Preliminaries

Here we introduce definitions and technical lemmas. For any positive integer n, let $[n]$ to denote the set $\{1,\ldots,n\}$. A vector consisting of $s \geq 2$ coordinates $a_1,\ldots,a_s$ is denoted as $[a_1,\ldots,a_s]$. On the other hand, for polynomials $f_1(x),\ldots,f_s(x)$, we use $(f_1,\ldots,f_s)$ to denote their sequence.

Let $\mathbb{Z}[x]$ denote the ring of integer coefficient univariate polynomials. Denote by $\mathbb{Z}_N$ the ring $\mathbb{Z}/N\mathbb{Z}$, and let $\mathbb{Z}_N[x]$ denote the ring of polynomials whose coefficients are in $\mathbb{Z}_N$. Use $\mathbb{Z}_N^\times$ to denote the set of units; i.e., elements that have inverses, in $\mathbb{Z}_N$. Based on this, we denote the set of units in $\mathbb{Z}_N[x]$ by $\mathbb{Z}_N[x]^\times$. We also use $\mathbb{M}_N[x]$ to denote the set of *monic polynomials* in $\mathbb{Z}_N[x]$; that is, the polynomials whose leading coefficients are one.

By $\equiv_N$ we denote the equivalence between two polynomials under modulo N; that is, for two polynomials $f(x)=\sum_{i=0}^d a_i x^i$ and $g(x)=\sum_{i=0}^e b_i x^i$, we write $f(x) \equiv_N g(x)$ if $a_i \equiv b_i \pmod N$ for any i, $0 \leq i \leq \max(d,e)$. Here we understand that $a_i = 0$ (resp. $b_i = 0$) for $i > d$ (resp. $i > e$.)

For any polynomial $f(x)$, we use $\deg(f)$ and $\mathrm{lc}(f)$ to denote its degree and the leading coefficient, respectively. For any positive integer c and any polynomial $f(x)$, we define $\mathrm{ord}_c(f)$ by the largest integer r such that $f(x) \equiv_{c^r} 0$ holds.

Howgrave-Graham [13] reformulated the Coppersmith technique and gave a lemma that plays a key role in analysis of the technique. We first introduce the notion of U-norm for simplifying notation. Define the *U-norm* of a polynomial $f(x)=\sum_{i=0}^d a_i x^i$ by $||f||_U = \sqrt{\sum_{i=0}^d (a_i U^i)^2}$. Thus, this is just the length of the coefficient vector of $f(Ux)$.

Lemma 1. *(**Howgrave-Graham** [13]) Consider any polynomial $f(x) \in \mathbb{Z}[x]$ consisting of w monomials. Let W be a non-negative integer satisfying*

$$||f||_U < W/\sqrt{w}. \tag{2}$$

Then we have

$$\forall v,\ |v| < U\ [\ f(v) \equiv 0 \pmod W \Leftrightarrow f(v) = 0\ ]. \tag{3}$$

In the Coppersmith technique, we need to find a polynomial having a small U-norm via a lattice reduction algorithm. First, we introduce a way to relate polynomials with vectors (and a lattice). Consider any polynomial $f(x)=\sum_{i=0}^d a_i x^i$. Its *vectorization* is a vector $\mathcal{V}(f,U)$ defined by $[a_0, a_1U, \ldots, a_dU^d]$. On the other hand, as its inverse transformation, for a given vector $\mathbf{v}$, we define the *functionalization* of $\mathbf{v}$ as a unique polynomial $f(x)$ such that $\mathbf{v} = \mathcal{V}(f,U)$ holds, and denote it by $\mathcal{F}(\mathbf{v},U)$. Note that this is undefined if no such $f(x)$ exists. $\mathcal{V}(f,U)$ and $\mathcal{F}(\mathbf{v},U)$ are clearly linear mapping w.r.t. polynomials and vectors, respectively. The Euclidean norm of $\mathbf{v}$

$(= \mathcal{V}(f, U))$ is equal to the U-norm of $f(x)$ $(= \mathcal{F}(\mathbf{v}, U))$. This relation is the motivation for our transformation.

For any $k \geq 1$ and any $w \geq k$, let $\mathbf{b}_1, \ldots, \mathbf{b}_k \in \mathbb{R}^w$ be linearly independent vectors. Then, the *lattice* spanned by these vectors is defined by the set $\{a_1\mathbf{b}_1 + \cdots + a_k\mathbf{b}_k \mid a_1, \ldots, a_k \in \mathbb{Z}\}$. We use the notation $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$ to denote it. The set of vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_k\}$ is called a *basis* of this lattice. We sometimes omit the basis if it is clear from the context. An element of a lattice is called a *lattice vector*. The *determinant* of the lattice is defined by $\det(L) = \prod_{i=1}^{k} |\mathbf{b}_i^*|$, where $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_k^*\}$ is the Gram-Schmidt basis. Here, the notation $|\cdot|$ denotes the standard Euclidean norm.

We need to compute a non-zero short vector in a lattice, which can be computed by a lattice basis reduction algorithm such as the LLL algorithm [17]. For a lattice L, let $\mathbf{v}_1$ denote the first vector in the basis computed by the LLL algorithm. Then it has been known [17] that this length is bounded by

$$|\mathbf{v}_1| \leq A(k) \det(L)^{1/k}, \tag{4}$$

where $A(k)$ is a constant that only depends on k. It has been shown that (4) holds with $A(k) = 2^{(k-1)/4}$. Note that this upper bound may not be tight, and that better algorithms have improved this. On the other hand, it has been observed [10] that for any polynomial-time lattice reduction algorithm, we have $\delta > 1$ such that $|\mathbf{v}_1| \approx \delta^k \det(L)^{1/k}$ holds. (Here by "polynomial-time" we mean polynomial-time w.r.t. k and $\log B \stackrel{\text{def}}{=} \log \max_i |\mathbf{b}_i|$.)

Remarks on the Problem Setting: We consider the problem of finding all solutions for (1) within the range of $|x| < U$ for a given parameter U. We call (1) the *target equation* and the range $|x| < U$ the *target range*. Throughout this paper, we fix the usage of symbols F, D, N, and U. We use the standard unit cost time complexity, and we evaluate complexity measures in terms of $\log N$, because we can assume that $D \leq \text{poly}(\log N)$ and $U < N$. Hence, by "polynomial-time" we mean a time polynomial in $\log N$ unless otherwise stated.

We assume that N is a large composite number whose non-trivial factor cannot be found during the computation that we investigate. This is because the factor of N would give the complete solution to the original problem in almost all applications of the Coppersmith technique. Thus, we can assume that all numbers that appear during the computation are coprime to N. This is used in the argument of Section 5. Because of this, we can assume that the coefficient a_D of x^D of $F(x)$ is one as stated in (1) since otherwise we can "divide" it by multiplying a_D^{-1} modulo N because a_D must be coprime to N.

3 Framework for the Coppersmith Technique

This section introduces our framework for discussing the Coppersmith technique for a univariate equation. As mentioned in the above section, for a given target equation (1) and a target range specified by U, our task is to find all solutions within the target range. For this task, we formulate the Coppersmith technique as an algorithm stated as Figure 1 by following Howgrave-Graham's reformulation [13].

Remarks may be necessary for some steps of the algorithm. First note that the algorithm is given two parameters $k \geq 1$ and $m \geq 2$, which are chosen (often heuristically) for the target equation. They are usually chosen as small because the time complexity of the original LLL algorithm [17] is $O(k^5 u \log^3 B)$ (e.g., [20, Chapter 5]), where u and B are the dimension of each vector, and $\log \max_i ||\mathbf{b}_i||$, respectively. We can at least assume that these parameters relating to time complexity are $\text{poly}(\log N)$, and this assumption is sufficient for our analysis. Thus, throughout the following discussion, we will consider any $k, m, u, \log B \leq (\log N)^c$ for $c > 0$ and let them be fixed.

Input	$F(x)$, N, U; **Parameters** $k \geq 1$, $m \geq 2$;
Output	All solutions of $F(x) \equiv 0 \pmod{N}$ satisfying $\|x\| < U$;
Step 1:	Based on the input, define a sequence of linearly independent polynomials $g_1(x), \ldots, g_k(x)$ that satisfy (5);
Step 2:	Define vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k$ by $\mathbf{b}_i = \mathcal{V}(g_i, U)$ for $i \in [k]$, and carry out the LLL algorithm on $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$; Denote the obtained reduced basis by $\mathbf{v}_1, \ldots, \mathbf{v}_k$;
Step 3:	Define a polynomial $h(x) = \mathcal{F}(\mathbf{v}_1, U)$, and solve the equation $h(x) = 0$ numerically; Output all integer roots within the target range satisfying (1);

Figure 1: Outline of Coppersmith technique

At Step 1, we define linearly independent polynomials $g_1(x), \ldots, g_k(x) \in \mathbb{Z}[x]$, which we call *initial polynomials*, that satisfy

$$\forall v \ [\ F(v) \equiv 0 \pmod{N} \Rightarrow g_i(v) \equiv 0 \pmod{N^m}\ ]. \tag{5}$$

As we will see, the choice of these polynomials determines the lattice used in the algorithm, and this is crucial for the performance of the algorithm. Again, they are defined somewhat heuristically in each application of the technique. We can at least assume that their degrees are bounded by $\mathrm{poly}(\log N)$. From the role of the parameter m in the above, we refer to m as an *initial exponent*.

At Step 3, we enumerate all roots of $h(x)$. Here, we simply assume that a numerical algorithm achieves this task efficiently (which is the case in the reported applications). Note that the degree of $h(x)$ is $\mathrm{poly}(\log N)$; hence, the number of roots is polynomially bounded. Finally, among all obtained roots, output integers within the target range satisfying (1).

For designing an algorithm following the outline of Figure 1, the key point is the choice of initial polynomials that determines the lattice $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$ used to compute a final $h(x)$.

Here, we follow [13] and derive a condition for initial polynomials that is sufficient for guaranteeing the correctness of the algorithm. Clearly, the algorithm works correctly when $\forall v,\ |v| < U\ [\ F(v) \equiv 0 \pmod{N} \Rightarrow h(v) = 0\ ]$. On the other hand, noting that $\mathbf{v}_1$ is an integer linear combination of $\mathbf{b}_1, \ldots, \mathbf{b}_k$, we can show that $h(x)$ $(= \mathcal{F}(\mathbf{v}_1, U))$ is an integer linear combination of initial polynomials. Then, from the requirement (5) for initial polynomials it follows that

$$\forall v \ [\ F(v) \equiv 0 \pmod{N} \Rightarrow h(v) \equiv 0 \pmod{N^m}\ ].$$

Thus, our above goal is satisfied if we have

$$\forall v,\ |v| < U\ [\ h(v) \equiv 0 \pmod{N^m} \Leftrightarrow h(v) = 0\ ]. \tag{6}$$

By Lemma 1, we see that $\|h\|_U < N^m/\sqrt{d_{\max} + 1}$ $(< N^m/\sqrt{\deg(h) + 1})$ is sufficient for (6), where $d_{\max}$ is the largest degree of initial polynomials. Then, our sufficient condition for the algorithm to work is derived by evaluating $\|h\|_U$. First, by definition of $h(x)$ and $\mathcal{F}(\cdot, U)$, and the bound (4), we have $|\mathbf{v}_1| = \|h\|_U \leq A(k) \det(L)^{1/k}$, where $L = L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$. Therefore, (6) is implied by

$$\det(L)^{1/k}/N^m < \left(A(k)\sqrt{d_{\max} + 1}\right)^{-1}. \tag{7}$$

We call this the *lattice condition* (as a sufficient condition for the Coppersmith technique to work). Note that this is a condition for initial polynomials because the basis of L is $\{\mathcal{V}(g_i, U)\}_{i \in [k]}$. In fact, by using the initial polynomials derived from the original work by Coppersmith [6], we can show that this condition is satisfied if $U \leq N^{1/D - \varepsilon}$ (for any $\varepsilon > 0$ if N and m are large enough), thereby confirming in our framework that the original method [6] works for this range of U.

In this paper, we discuss when the condition (7) cannot be achieved by *any* lattice; i.e., any set of initial polynomials, thereby showing the technique's limit. Thus, for our following discussion, we will use a somewhat stronger and much simpler condition

$$\det(L)^{1/k}/N^m < 1 \tag{8}$$

for our target condition, which we refer as a *simplified lattice condition* (for discussing when the lattice condition cannot be achieved). Recall that $A(k) = 2^{(k-1)/4}$ for the LLL algorithm and it has been believed that $A(k)$ cannot be smaller than δ^k for $\delta > 1$ for any polynomial-time lattice reduction algorithm. Then, since $(\sqrt{d_{\max}+1}A(k))^{-1} < 1$ holds for any k, this simplified condition is necessary for (7). See section 6 for analysis with a more relaxed condition.

4 Analysis for Canonical Initial Polynomials

We consider standard lattice construction and investigate its properties. Based on this investigation, we derive a lower bound for U such that the simplified lattice condition (8) fails to hold, which we may regard as the limit of U that the Coppersmith technique works under the standard lattice construction.

Note that initial polynomials need to satisfy the condition (5). One trivial way to define such polynomials $g(x)$ is by

$$g(x) = \sum_{i=0}^{m} q_i(x) N^{m-i} (F(x))^i, \quad \text{where } q_i(x) \in \mathbb{Z}[x]. \tag{9}$$

This is an integer linear combination of polynomials that were usually called "shift polynomials" in previous work. Formally, we introduce the following notion.

Definition 1. *Consider the ideal $\mathfrak{a} = \langle F(x), N\rangle_{\mathbb{Z}[x]}$ in the polynomial ring $\mathbb{Z}[x]$. For any non-zero polynomial $f(x) \in \mathbb{Z}[x]$, let $\nu(f)$ be the $\mathfrak{a}$-adic order of $f(x)$, that is, an integer s that satisfies $f(x) \in \mathfrak{a}^s$ and $f(x) \notin \mathfrak{a}^{s+1}$. For the zero polynomial, define $\nu(0) = \infty$. We say $f(x)$ is an s-canonical polynomial if $\nu(f) \geq s$.*

We simply say that $f(x)$ is *canonical* if $\nu(f) \geq m$ for the initial exponent m. Initial polynomials (or similar ones) used in the previous work are all canonical and we can consider that using canonical polynomials is a standard way to define initial polynomials. This section discusses the case in which initial polynomials are all canonical.

Consider any initial polynomials $g_1(x), \ldots, g_k(x)$. Assume that they are all canonical and linearly independent as requested in the algorithm. Consider a sequence $(g_1, \ldots, g_k)$ and denote it by $\mathbf{G}$. For any sequence $\mathbf{F} = (f_1, \ldots, f_k)$ of linearly independent polynomials, we use $L(\mathbf{F})$ to denote a lattice $L(\mathcal{V}(f_1, U), \ldots, \mathcal{V}(f_k, U))$. Note that $L(\mathbf{G})$ is $L(\mathbf{b}_1, \ldots, \mathbf{b}_k)$ used in the algorithm in Figure 1 for the initial polynomials $g_1(x), \ldots, g_k(x)$.

Our task is to provide a good lower bound of $\det(L(\mathbf{G}))$. For this, we transform $\mathbf{G}$ to a polynomial sequence with some good properties for our analysis. Here, we explain the outline of our transformations and give the proof outline of the first main theorem. Appendix A provides technical discussions with necessary definitions and lemmas.

We say a polynomial sequence $\widetilde{\mathbf{G}} = (\widetilde{g}_1, \ldots, \widetilde{g}_k)$ has a *strictly increasing degree sequence* if it holds that $\deg(\widetilde{g}_1) < \cdots < \deg(\widetilde{g}_k)$. We first transform $\mathbf{G}$ to a sequence $\widetilde{\mathbf{G}} = (\widetilde{g}_1, \ldots, \widetilde{g}_k)$ with this property while maintaining that $\det(L(\mathbf{G})) = \det(L(\widetilde{\mathbf{G}}))$ and that all $\widetilde{g}_i(x)$'s are canonical. This transformation is possible by the procedure described in Appendix A.1 (A.1 Lemma 2). Next, we define a sequence $\mathbf{H} = (h_1, \ldots, h_k)$ by defining each $h_i(x) = \widetilde{g}_i(x)/N^{r_i}$ where we let

$r_i = \mathrm{ord}_N(\widetilde{g}_i)$; that is, r_i is the largest integer r such that every coefficient of $\widetilde{g}_i(x)$ is divisible by N^r. Let $s_i = \nu(h_i)$. Then, since $\widetilde{g}_i(x)$ is canonical, we have $s_i + r_i = \nu(\widetilde{g}_i) \geq m$. Hence we have

$$\det(L(\mathbf{G})) = \det(L(\widetilde{\mathbf{G}})) = \left(\prod_{i=1}^{k} N^{r_i}\right) \times \det(L(\mathbf{H})) \geq \left(\prod_{i=1}^{k} N^{m-s_i}\right) \times \det(L(\mathbf{H})). \tag{10}$$

Finally, we transform $\mathbf{H}$ again to $\widehat{\mathbf{H}}$ with "everywhere linearly independent reduction" (see Appendix A.2 for the definition). This transformation is also possible (A.2 Lemma 3) in such a way that (i) $\det(L(\mathbf{H})) \geq \det(L(\widehat{\mathbf{H}}))$ and (ii) $\widehat{\mathbf{H}}$ also has a strictly increasing degree sequence. Then, by using A.3 Lemma 4, the determinant term of (10) can be bounded by

$$\det(L(\mathbf{H})) \geq \det(L(\widehat{\mathbf{H}})) \geq \prod_{i=1}^{k} |\widehat{a}_i U^{\widehat{d}_i}| \geq \prod_{i=1}^{k} U^{\widehat{d}_i}, \tag{11}$$

where $\widehat{d}_i$ and $\widehat{a}_i$ are used for $\deg(\widehat{h}_i)$ and $\mathrm{lc}(\widehat{h}_i)$, respectively. On the other hand, by defining $\hat{s}_1, \ldots, \hat{s}_k$ as (20) in Appendix A.3, we can show (A.3 Lemma 5) that $\hat{s}_i \geq s_i$ and $\widehat{d}_i \geq \hat{s}_i D$. Thus, from (10) and (11), we have

$$\begin{aligned} \det(L(\mathbf{G})) &\geq \left(\prod_{i=1}^{k} N^{m-\hat{s}_i}\right) \times \left(\prod_{i=1}^{k} U^{\widehat{d}_i}\right) \\ &\geq \left(\prod_{i=1}^{k} N^{m-\hat{d}_i/D}\right) \times \left(\prod_{i=1}^{k} U^{\hat{d}_i}\right) \;=\; N^{mk} \cdot \left(\frac{U}{N^{1/D}}\right)^{\sum \hat{d}_i}. \end{aligned} \tag{12}$$

It is then easy to see that if $U \geq N^{1/D}$, we have $\det(L(\mathbf{G})) \geq N^{mk}$. Thus, $\det(L(\mathbf{G}))^{1/k} \geq N^m$ and the simplified lattice condition (8) fails. This proves our first main theorem.

Theorem 1. *If $U \geq N^{1/D}$, then the simplified lattice condition (8) fails to hold for any lattice constructed from canonical initial polynomials.*

5 Computation from Non-canonical Polynomials

This section considers the possibility of using non-canonical initial polynomials. Recall that in the lattice construction for a given polynomial $F(x)$ and an initial exponent m, an initial polynomial $g(x)$ is said to be canonical if $\nu(g) \geq m$ holds. A *non-canonical initial polynomial* is defined as a polynomial $g(x)$ satisfying (5) and $\nu(g) < m$ w.r.t. $F(x)$ and N. We discuss what we are able to compute if we can indeed construct a non-canonical polynomial. We show technical evidence supporting that there is no polynomial-time algorithm computing such a non-canonical polynomial for any $F(x)$, N, and m. Technically, we show that if $F(x)$ and its derivative has no common factor (a property we call "separability" following the polynomial theory over a field; e.g., [11, Def. 2]), then by using such a non-canonical initial polynomial, it is possible to compute either a non-trivial factor of N or a polynomial $G(x)$ with $\deg(G) \leq \deg(F) - 2$ that keeps the same set of solutions. This computation can also be done in polynomial-time. One can also show that a simpler polynomial $G(x)$ (if it is obtained) also has separability; thus, if there were a general polynomial-time algorithm for computing a non-canonical polynomial, we would be able to continue to create simpler polynomials (unless a factor of N is obtained). Then, however, we would eventually create a linear equation, which derives either a contradiction if $F(x)$ has more than one solution, or a way to compute $F(x)$'s unique solution in polynomial-time. Thus, if we had such a general algorithm, we would be able to use this it to compute either the unique solution of $F(x)$ or a factor of N. Note that this does not rule out the possibility of having *specific* $F(x)$ and m for which non-canonical initial polynomials are easy to compute; but we think that the above one-step reduction itself yields some remarkable consequences for many concrete cases.

5.1 Technical Preliminaries

Our investigation is based on arithmetic computations under modulo N. Since it was assumed that N is not prime; there may be points at which we need to be careful. On the other hand, as explained in the introduction, we can assume that no factor of N appears during these computations; that is, we can treat N as a prime number in the following analysis. Below, we clarify the points where careful arguments are necessary.

We use standard arithmetics in $\mathbb{Z}_N[x]$. There is no problem with addition, subtraction, and multiplication, which can be defined the same as in $\mathbb{Z}[x]$. On the other hand, the division is defined as follows. For any $f(x), g(x) \in \mathbb{Z}_N[x]$, $g(x) \not\equiv_N 0$, consider polynomials $q(x) \in \mathbb{Z}_N[x]$ and $r(x) \in \mathbb{Z}_N[x]$ such that satisfy $f(x) \equiv_N q(x)g(x) + r(x)$ and $\deg(r) < \deg(g)$ (recall that $\equiv_N$ means the polynomial equivalence under modulo N). Note that $q(x)$ and $r(x)$ are unique when the leading coefficient of $g(x)$ is coprime to N. Thus, under our assumption, we can consider $q(x)$ and $r(x)$ the *quotient* and the *remainder* of $f(x)$ divided by $g(x)$, and denote them by $\mathrm{quo}(f, g)$ and $\mathrm{rem}(f, g)$, respectively. We say that $g(x)$ *divides* $f(x)$ under modulo N or $g(x)$ an *N-divisor* of $f(x)$ (and write it as $g(x)|_N f(x)$) if $r(x) \equiv_N 0$ in the above.

For any two polynomials $f(x)$ and $g(x)$, we say they are N-coprime to each other if $h(x)|_N f(x)$ and $h(x)|_N g(x)$ implies that $h(x) \in \mathbb{Z}_N[x]^\times$. On the other hand, for a monic polynomial $h(x) \in \mathbb{M}_N[x]$ satisfying $h(x)|_N f(x)$ and $h(x)|_N g(x)$, we say it is a *greatest monic common N-divisor* if the quotients $\mathrm{quo}(f, h)$ and $\mathrm{quo}(g, h)$ are N-coprime to each other. We can use the standard Euclidean algorithm to compute this divisor of given $f(x)$ and $g(x)$. This computation yields the *unique* greatest monic common N-divisor unless a non-trivial factor of N is computed during the computation.

5.2 From a Non-Canonical Polynomial

We discuss what we are able to compute from a non-canonical initial polynomial for given $F(x)$, N, and m. We need to assume that $F(x)$ is *separable*, that is, $F(x)$ and its derivative are N-coprime to each other. Our result is given by the following theorem. The proof is given in Appendix B.3.

Theorem 2. *Assume that our target polynomial $F(x)$ is separable. For $F(x)$, N, and m, suppose that we have a non-canonical initial polynomial $g(x)$; that is, it satisfies both $\nu(g) \leq m - 1$ and the condition (5). Then we can compute in polynomial-time w.r.t. $\log N$ and $\deg(g)$, either a non-trivial factor of N or a polynomial $G(x)$ with $\deg(G) \leq \deg(F) - 2$ satisfying*

$$\forall v \ [\ F(v) \equiv 0 \pmod{N} \Rightarrow G(v) \equiv 0 \pmod{N}\ ]. \tag{13}$$

Moreover, $G(x)$ is an N-divisor of $F(x)$, and hence, the separability of $G(x)$ is immediate from that of $F(x)$.

Note that this theorem gives a polynomial-time algorithm that reduces a given target equation to a simpler one based on any non-canonical initial polynomial for the target polynomial. We expect that this reduction itself is impossible for various cases. Below, we show one example from the RSA cryptography.

Application for Coppersmith's Attack for RSA: We apply the above theorem for a Coppersmith's attack [6] for RSA. Consider an RSA ciphertext c that is encrypted from a plaintext p using a public key pair (e, N). Here, we use N for both the modulo of an RSA instance and that of the target equation. The situation considered in [6] is that the attacker has a public key pair, valid ciphertext, and a quantity of MSBs of the corresponding plaintext. Let C and P' denote integers respectively corresponding to the ciphertext c and the revealed part of p, and let k denote the length of the unrevealed part of p. Then the unrevealed part Q is the unique solution of $f_{\mathrm{RSA}}(x) \stackrel{\mathrm{def}}{=} (x + 2^k P')^e - C \equiv 0 \pmod{N}$, which we call the *RSA equation*. The

attacker's task is to compute Q. Clearly, Q satisfies $0 \leq Q < 2^k$, and it can be easily shown that x is the unique solution of the RSA equation. Coppersmith showed that the equation is solved in polynomial-time w.r.t. $\log N$ when $0 < Q < N^{1/e}$, which corresponds to the situation in which the bit-length of the unknown part is smaller than $1/e$-times that of N.

From our result of Section 4, the length of the revealed part should be longer than $(1 - 1/e)\log_2 N$ bits in order to use the Coppersmith technique with the canonical initial polynomials. Now suppose that we have a non-canonical initial polynomial for the target equation $f_{\mathrm{RSA}}(x)$ (and an initial exponent m). From the separability of $f_{\mathrm{RSA}}(x)$ and by Theorem 2, we can compute either (i) a non-trivial factor of N, or (ii) a polynomial $G(x)$ with $\deg(G) \leq e-2$ that has the same solutions with $f_{\mathrm{RSA}}(x)$. Clearly the attack succeeds in the former case. Consider the latter case. If $e = 3$, noting that G is a linear function, it is easy to see that the plaintext is computable from $G(x)$. In general, for any $e = \mathrm{poly}(\log N)$, if we can repeat this argument, (either a non-trivial factor of N is obtained, or) a trivial linear equation is derived to compute the plaintext. Note also that the length of the revealed part does not matter in this case. From this example, we can conclude that there is no general and efficient way to construct non-canonical initial polynomials.

6 Tightness of Our Analysis

In Section 3 we derive the (simplified) lattice condition (7) (and (8))

$$\det(L)^{1/k}/N^m < \left(A(k)\sqrt{d_{\max}+1}\right)^{-1} < 1$$

as a sufficient condition that the Coppersmith technique works, and then in the following sections, we show that this condition fails to hold for any initial polynomials when $U \geq N^{1/D}$, thereby claiming that $N^{1/D}$ is the limit of U for the Coppersmith technique. Apparently this argument is not mathematically correct, and it may be possible that the Coppersmith technique works using initial polynomials that do not satisfy the lattice condition, from which we may need to revise the limit $N^{1/D}$. Here, we provide evidence supporting that this situation is quite unlikely to happen.

Recall that the lattice condition is derived by using two inequalities. We recall these inequalities and state them by using the symbols from Section 3 used to derive the lattice condition. First is the inequality (2) stated below. This is a sufficient condition to guarantee that (3) of Lemma 1 holds and that the algorithm works with the designed initial polynomials.

$$(2)\ \ ||h||_U < N^m/\sqrt{d_{\max}+1} \qquad (4)\ \ |\mathbf{v}_1| \leq A(k)\det(L)^{1/k}$$

The second is the upper bound (4) for the length of a short lattice vector obtained by a lattice algorithm (such as LLL) in a lattice L constructed from the initial polynomials. From these inequalities, the lattice condition $A(k)\det(L)^{1/k} < N^m/\sqrt{d_{\max}+1}$ is immediately sufficient for $||h||_U$ to satisfy (2), which guarantees that the algorithm yields a correct answer. We also assume that $(A(k)\sqrt{d_{\max}+1})^{-1} < 1$ for the simplified lattice condition.

We believe that the inequalities (2) and (4) are more or less tight for at least $h(x)$ and L appearing in the execution of the Coppersmith technique. But it is possible that they are not as sharp as we expect. We first show that a similar limit $N^{1/D+\varepsilon}$ can be derived even if much stronger inequalities could be used. For this, consider the following situation for a large β and γ: (i) the property (3) holds if $||h||_U < \beta N^m$, and (ii) a lattice algorithm obtains a short vector satisfying $||\mathbf{v}_1||_U \leq \det(L)^{1/k}/\gamma$. That is, the algorithm works so long as $\det(L)^{1/k}/\gamma < \beta N^m$ ($\Leftrightarrow \det(L)^{1/k}/N^m < \beta\gamma$) holds. Note that β and γ are big numbers that may grow depending on parameters k, $d_{\max}$, etc. Thus, the lattice condition is relaxed considerably. by this condition Even for this relaxed condition, we can show that $U < N^{1/D+\varepsilon}$ is necessary to satisfy it. To see this, we use the lower bound (12) for $\det(L(\mathbf{G}))$, where $L(\mathbf{G})$ is the lattice constructed from any

given canonical initial polynomials. From this bound, it is easy to see that if $U \geq N^{1/D}$,

$$\det(L(\mathbf{G}))^{1/k}/N^m \;\geq\; N^{mk} \cdot \left(\frac{U}{N^{1/D}}\right)^{\sum \hat{d}_i} \;\geq\; N^{mk} \cdot \left(\frac{U}{N^{1/D}}\right)^{k(k-1)/2}$$

holds, where the last inequality follows from the fact that $\widehat{d}_i \geq i-1$ for all $i \in [k]$ (because $\widehat{\mathbf{H}}$ has a strictly increasing degree sequence), and hence, $\sum \widehat{d}_i \geq k(k-1)/2$. Thus, even the relaxed lattice condition fails to hold if the right-hand side $\geq \beta\gamma$, which is equivalent to

$$U \;\geq\; N^{1/D + \frac{2\log(\beta\gamma)}{(k-1)\log N}}. \tag{14}$$

Therefore, even if much stronger inequalities could be used, so long as $\beta\gamma << N$, we can claim that the relaxed lattice condition fails if $U \geq N^{1/D+\varepsilon}$ for any constant $\varepsilon > 0$ and any sufficiently large N.

6.1 Justifications from Computer Experiments

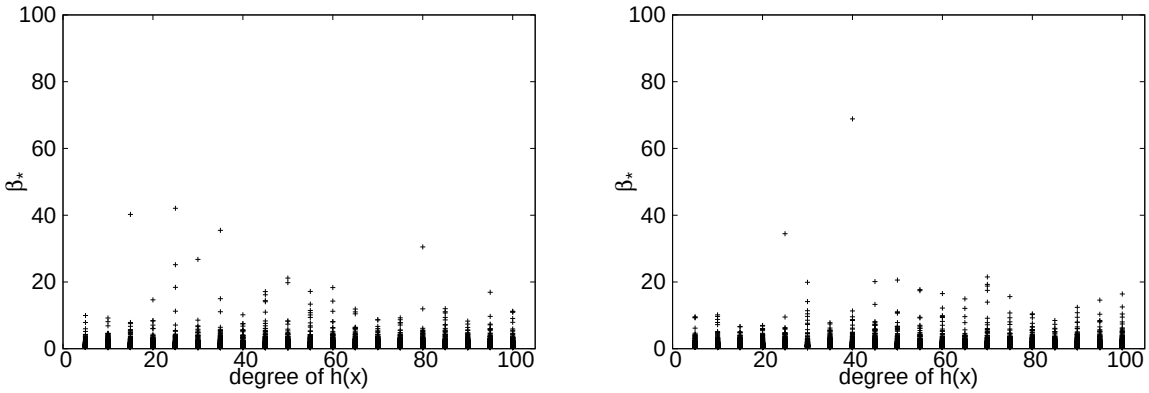
We can provide evidence from computer experiments that supports tightness of (2) and (4). First, consider (4), an upper bound for the length of a short vector given by a lattice reduction algorithm. Due to the importance of obtaining a short lattice vector, the improvement of lattice reduction algorithms and analysis of their performance have been studied extensively. Various computer experiments have also been conducted (e.g., [19, 10]). From such investigation, it has been believed that for any lattice algorithm P, there is $\delta_P > 1$ such that $||\mathbf{v}_1||_U \approx \delta_P^k \det(L)^{1/k}$ holds; that is, the coefficient $A(k)$ of (4) is δ_P^k. In particular, from a large number of computer experiments, we can assume that the bound (4) holds with, at least, $A(k) = 1$.

Next consider the inequality $||h||_U < \beta_0 W$ that is a sufficient condition for

$$(3) \qquad \forall v,\ |v| < U\ [\ h(v) \equiv 0 \pmod{W} \Rightarrow h(v) = 0\ ],$$

where $\beta_0 = (\sqrt{d_{\max}+1})^{-1}$ and $W = N^m$ in our analysis. Since this is only a sufficient condition, it may be possible that (3) holds even if $||h||_U$ is much larger. While it seems quite difficult to show that this is unlikely, it is possible to relate $||h||_U$ and a condition closely related to (3). Note that if $W_* \stackrel{\text{def}}{=} \max_{|x|<U, x\in\mathbb{Z}} |h(x)| < W$, then (3) holds. On the other hand, (3) fails when $W = \max_{|x|<U, x\in\mathbb{Z}} |h(x)|$. This indicates that the condition $\max_{|x|<U, x\in\mathbb{Z}} |h(x)| < W$ and (3) are closely related. On the other hand, letting $\beta_* \stackrel{\text{def}}{=} \|h\|_U / W_*$, we have that $\max_{|x|<U, x\in\mathbb{Z}} |h(x)| < W \iff ||h||_U < \beta_* W$. Thus, we investigate how large β_* could become by computing it for randomly generated polynomials $h(x)$.

Here is the outline of our experiment. For a sufficiently large U (i.e., $U = 10^{10}$ and $= 10^{100}$) and a degree parameter d (i.e., $d = 5, 10, ..., 100$), we generate a polynomial $h(x) = \sum_{i=0}^{d} a_i x^i$ by choosing coefficients a_i randomly so that $|a_i U^i|$ is located in $[0.1BU^d, 10BU^d]$ for a large B (i.e., B is an integer sampled from $[U/2, U]$). This random generation is motivated by our observation that actual polynomials $h(x)$ derived in the Coppersmith technique have coordinates that usually satisfy $|a_i U^i|/U^{d+1} = \Theta(1)$. We then compute $W_* := \max_{|x|<U, x\in\mathbb{Z}} |h(x)|$. Since $h(x)$ is a polynomial of a relatively small degree, this computation can be performed easily by examining integer points near all $\xi \in \mathbb{R}$ satisfying $h'(\xi) = 0$. We then compute $\beta_* = \|h\|_U / W_*$. For each choice of parameters, such β_*'s are computed for 500 randomly generated polynomials. From this experiment (see Figure 2), we can claim that $\beta_* \leq 99$ and $||h||_U \geq 99W \implies \max_{|x|<U} |h(x)| > W$; in other words, it is unlikely that inequality (2) can be improved to $\|h\|_U < 100N^m$.



$U = 10^{10}$ $U = 10^{100}$

For each $U = 10^{10}$ and $= 10^{100}$, and $d = 5, 10, ..., 100$, the graph shows β_*'s computed for 500 randomly generated polynomials. The horizontal axis is the degree d of generated polynomials, and the vertical axis is β_*.

Figure 2: Experimental values of β_*

7 Concluding Remarks

We investigated the optimality of lattice constructions used in the Coppersmith technique for finding small roots of a univariate modular equation (1). For this purpose, we provide a framework of the technique and a sufficient condition, i.e., the lattice condition (7) (and its simplified version (8)) in which the technique works. Then, for any lattice constructed from canonical initial polynomials, we derive a way to estimate a lower bound for the lattice determinant and prove that the condition fails to satisfy if $U \geq N^{1/d}$ (Theorem 1). A similar limit can be shown unless the Coppersmith technique works under a significantly relaxed condition, and we show a computer experiments indicating that it is unlikely that the lattice condition is significantly relaxed. Thus, these results are reasonable evidence of the limit of the standard lattice construction for the Coppersmith technique.

We also discuss the possibility of constructing a better lattice by using polynomials constructed in a non-standard way. We show that such a construction itself would lead to quite a strong method for solving the original problem or factorizing a large number. For example, any non-standard way of constructing a legitimate lattice for the Coppersmith's attack of RSA can be used to decipher messages encrypted by RSA with a small public key e.

From these results, we can claim that our bound $U < N^{1/d}$ is sharp for the direct usage of the Coppersmith technique solving (1) for various target polynomials considered in, for instance, cryptographic applications.

Coppersmith [7] employed a family of integer valued polynomials and Chebychev polynomial representations to extend the range $U < N^{1/d}$. Their approaches indeed increases the range by a poly$(\log N)$ factor. Although our framework considers only polynomials with integer coefficients, it can be adopted for the above framework and provide a similar limitation result. A detailed argument is given in the full-version of this paper.

For a Multivariate Situation: The Coppersmith technique has been generalized to the multivariate case [4, 14]. This multivariate version has been used more actively in cryptographic applications; for example, (i) the Boneh-Durfee [2] attack for an RSA short secret exponent, (ii) the fault-based attack for the EMV signature scheme by Coron et al. [8], and (iii) the method for factoring integers of the form pq^2 by Castagnos et al. [5]. The framework and arguments in Section 3 and 4 can be extended to the multivariate situation. In particular, the multivariate version of our procedure in Section 4 and Appendix A can be used to convert a lattice to one having a smaller determinant. Unfortunately, this extended argument is not strong enough to

prove some useful results in multivariate situations. The primary cause is that it fails to give a sharp lower bound for the lattice determinant. We explain this point with the following example. Consider the equation for recovering small RSA secret exponent by Boneh-Durfee [2]: $f_{\mathrm{BD}}(x, y) = -1 + x(N + y) \equiv 0 \pmod{e}$. We remark that the sequence $(f_{\mathrm{BD}}(x, y))$ consisting of one polynomial has a strictly increasing degree sequence and everywhere linearly independent reduction. Note that both properties are naturally extended for the multivariate situation by selecting an appropriate monomial order. For range parameters X and Y, which are corresponding to x and y, respectively, the converted vector is $\mathcal{V}(f_{\mathrm{BD}}, X, Y) = [-1, eX, XY]$ and the determinant of the constructed lattice is about eX. On the other hand, a multivariate version of Lemma 4 provides a bound $\det(L(\mathbf{G})) \geq XY$ since it ignores the polynomial coefficient. Hence, the gap $e/Y \approx N^{0.5}$ blocks to provide a good limitation result in a multivariate situation. We expect that our argument can be improved to overcome this problem in order to provide interesting analysis for the limitation of several cryptographic applications that use the multivariate version of the Coppersmith technique.

References

[1] Y. Aono, A new lattice construction for partial key exposure attack for RSA, in *Proc. of PKC 2009*, LNCS, vol. 5443, pp. 34-53, 2009.

[2] D. Boneh and G. Durfee, Cryptanalysis of RSA with private Key d Less Than $N^{0.292}$, in *Proc. of Eurocrypt 1999*, LNCS, vol. 1592, pp. 389-401, 1999.

[3] J. Blömer and A. May, New partial key exposure attacks on RSA, in *Proc. of CRTPTO 2003*, LNCS, vol. 2729, pp. 27-43, 2003.

[4] J. Blömer and A. May, A tool kit for finding small roots of bivariate polynomials over the integers, in *Proc. of Eurocrypt 2005*, LNCS, vol. 3494, pp. 351-367, 2005.

[5] G. Castagnos, A. Joux, F. Laguillaumie, and P. Q. Nguyen, Factoring pq^2 with quadratic forms: Nice cryptanalyses, *Proc. of Asiacrypt 2009*, LNCS, vol. 5912, pp. 469-486, 2009.

[6] D. Coppersmith, Finding a small root of a univariate modular equation, *Proc. of Eurocrypt 1996*, LNCS, vol. 1070, pp. 155-165, 1996.

[7] D. Coppersmith, Finding small solutions to small degree polynomials, *Proc. of CaLC 2001*, LNCS, vol. 2146, pp. 20-31, 2001.

[8] J.-S. Coron, A. Joux, I. Kizhvatov, D. Naccache, and P. Paillier, Fault attacks on RSA signatures with partially unknown messages, *Proc. of CHES 2009*, LNCS, vol. 5747, pp. 444-456, 2009.

[9] M. Ernst, E. Jochemsz, A. May, and B. Weger, Partial key exposure attacks on RSA up to full size exponents, in *Proc. of Eurocrypt 2005*, LNCS, vol. 3494, pp. 371-386, 2005.

[10] N. Gama and P. Q. Nguyen, Predicting lattice reduction, in *Proceedings of Eurocrypt 2008*, Lecture Notes in Computer Science, vol. 4965, pp. 31-51, 2008.

[11] P. Gianni and B. Trager, Square-free algorithms in positive characteristic, in *Applicable Algebra in Engineering, Communication and Computing*, Vol. 7, No. 1, pp.1-14, 1996.

[12] J. Håstad, Solving simultaneous modular equations of low degree, SIAM Journal on Computing, Vol. 17, No 2, pp. 336-341, 1988.

[13] N. Howgrave-Graham, Finding small roots of univariate modular equations revisited, *Proc. of Cryptography and Coding*, LNCS, vol. 1355, pp. 131-142, 1997.

[14] E. Jochemsz and A. May, A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants, in *Proc. of Asiacrypt 2006*, LNCS, vol. 4284, pp. 267-282, 2006.

[15] N. Kunihiro, Solving generalized small inverse problems, in *Proc. of ACISP 2010*, LNCS, vol. 6168, pp. 248-263, 2010.

[16] S. V. Konyagin and T. Steger, On polynomial congruences, *Mathematical Notes*, Vol. 55, No. 6, pp. 596-600, 1994.

[17] A. K. Lenstra, H. W. Lenstra, Jr. and L. Lovász Factoring polynomials with rational coefficients, *Mathematische Annalen*, vol. 261, pp. 515-534, 1982.

[18] J. S. Milne, Étale cohomology, Princeton Math. Series 33, Princeton Univ. Press, 1980.

[19] P. Q. Nguyen and D. Stehlé, LLL on the average, in *Proc. of ANTS 2006*, LNCS, vol. 4076, pp. 238-256, 2006.

[20] P. Q. Nguyen and B. Vallée, The LLL Algorithm: Survey and Applications, Springer-Verlag, Berlin Heidelberg, 2009.

[21] T. Okamoto and A. Shiraishi, A fast signature scheme based on quadratic inequalities, in *Proc. of the Symposium on Security and Privacy*, IEEE, pp. 123-132, 1985.

[22] G. Pólya and G. Szegő, Problems and Theorems in Analysis, Vol. II, Springer, 1976.

[23] R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Communications of the ACM*, vol. 21, No. 2, pp. 120-128, 1978.

[24] V. Shoup, OAEP Reconsidered, in *Journal of Cryptology*, vol. 15, No. 4, pp. 223-249, 2002. online version is available at `http://shoup.net/papers/oaep.pdf`.

[25] B. Vallee, M.Girault, and P. Toffin, How to Break Okamoto's Cryptosystems by Reducing Lattices Bases, in *Proc. of Eurocrypt 1988*, LNCS, vol. 330, pp. 281-291, 1988.

A Appendix: Technical Tools for Determinant Analysis

This section explains technical tools for our determinant analysis in Section 4. Below, by "polynomial sequence," we mean a sequence consisting of k linearly independent polynomials in $\mathbb{Z}[x]$, and we use $\mathbf{F}$, $\overline{\mathbf{F}}$, ... to denote such sequences. We consider a transformation from $\mathbf{F} = (f_1, \ldots, f_k)$ to $\overline{\mathbf{F}} = (\overline{f}_1, \ldots, \overline{f}_k)$ that can be specified by using a $k \times k$ integer matrix T as

$$\begin{bmatrix} \overline{f}_1 \\ \vdots \\ \overline{f}_k \end{bmatrix} = T \begin{bmatrix} f_1 \\ \vdots \\ f_k \end{bmatrix}.$$

For simplifying the expression, we state this as $\overline{\mathbf{F}} = T\mathbf{F}$. The following fact is clear from the definition.

Fact 1. *Consider any polynomial sequences* $\mathbf{F} = (f_1, \ldots, f_k)$ *and* $\overline{\mathbf{F}} = (\overline{f}_1, \ldots, \overline{f}_k)$ *satisfying* $\mathbf{F} = T\overline{\mathbf{F}}$ *with a regular* $k \times k$ *matrix* T *consisting of integer elements. Then we have*

$$\det(L(\mathbf{F})) = |\det(T)| \cdot \det(L(\overline{\mathbf{F}})) \geq \det(L(\overline{\mathbf{F}})). \tag{15}$$

In particular, if T *is a unimodular matrix, we have* $\det(L(\mathbf{F})) = \det(L(\overline{\mathbf{F}}))$.

A.1 Analysis for Strictly Increasing Degree Sequence

Recall that the notation $\deg(f)$ is the degree of a polynomial $f(x)$, and it said that a polynomial sequence $\mathbf{F} = (f_1, \ldots, f_k)$ has a strictly increasing degree sequence if $\deg(f_1) < \cdots < \deg(f_k)$.

The following lemma shows that any polynomial sequence can be transformed to a polynomial sequence with a strictly increasing degree sequence while keeping the corresponding lattice determinant.

Lemma 2. *For any polynomial sequence* $\mathbf{F}$*, we can transform it to* $\widetilde{\mathbf{F}}$ *having a strictly increasing degree sequence by a unimodular matrix* $\widetilde{T} \in GL_k(\mathbb{Z})$*; that is,* $\widetilde{\mathbf{F}} = \widetilde{T}\mathbf{F}$*. Hence, we have* $\det(L(\mathbf{F})) = \det(L(\widetilde{\mathbf{F}}))$.

Proof. We first define two series of transformations $C_{i,j}$ and $D_{i,j}(x^e)$ represented by $k \times k$ unimodular matrices, respectively. Then we show that our $\widetilde{T}$ can be written as a finite product of them; hence $\widetilde{T}$ is also a unimodular matrix because the product of unimodular matrices is also unimodular. By this, $\det(L(\widetilde{\mathbf{F}})) = \det(L(\mathbf{F}))$ is clear by Fact 1.

For $i, j \in [k]$ such that $i \neq j$, $C_{i,j}$ is defined as a transformation that swaps $f_i(x)$ and $f_j(x)$; its matrix representation is given by $[c_{i',j'}]$ where $c_{i,i} = c_{j,j} = 0$, $c_{i,j} = c_{j,i} = 1$, and $c_{i',j'} = \delta_{i',j'}$ for other $(i', j') \in [k]^2$. Here, $\delta_{i',j'}$ is the Kronecker's delta that was defined as one if $i' = j'$ and zero otherwise. This is clearly a unimodular matrix since its determinant is -1. For the case $i = j \in [k]$, $C_{i,i}$ is defined as the identity matrix of degree k.

Next we define $D_{i,j}(x^e)$ for $i, j \in [k]$ such that $i \neq j$ and a monomial x^e. It is roughly defined as a transformation that deletes the monomial x^e in $f_i(x)$ by a linear mapping. Let the monomial expressions of $f_i(x)$ and $f_j(x)$ be

$$f_i(x) = \cdots + b_i \cdot x^e + \cdots, \text{ and } f_j(x) = \cdots + b_j \cdot x^e + \cdots,$$

respectively. Our objective here is to define a transformation by which the transformed polynomials $\overline{f}_i(x)$ and $\overline{f}_j(x)$ respectively have the expressions

$$\overline{f}_i(x) = \cdots + 0 \cdot x^e + \cdots, \text{ and } \overline{f}_j(x) = \cdots + b'_j \cdot x^e + \cdots.$$

When $b_i = 0$, no transformation is necessary, and we simply use the identity transformation for $D_{i,j}(x^e)$. Next, for the case $b_i \neq 0$ and $b_j = 0$, we can use $C_{i,j}$ for $D_{i,j}(x^e)$. Finally consider the situation that b_i and b_j are both non-zero. For integers $d_1, \ldots, d_4$, $i, j \in [k]$ and $i \neq j$, consider the integer matrix $D_{i,j}(x^e) \stackrel{\text{def}}{=} [d_{i',j'}]$ where $(d_{i,i}, d_{i,j}, d_{j,i}, d_{j,j}) = (d_1, d_2, d_3, d_4)$, and $d_{i',j'} = \delta_{i',j'}$ for other $(i', j') \in [k]^2$. The transformation creates the following functions at the corresponding positions:

$$\begin{aligned}\overline{f}_i(x) &= d_1 f_i(x) + d_2 f_j(x) = \cdots + (d_1 b_i + d_2 b_j)x^e + \cdots, \text{ and} \\ \overline{f}_j(x) &= d_3 f_i(x) + d_4 f_j(x) = \cdots + (d_3 b_i + d_4 b_j)x^e + \cdots.\end{aligned}$$

Hence, for our purpose, it suffices to choose integers $d_1, \ldots, d_4$ so that

$$d_1 b_i + d_2 b_j = 0 \text{ and } d_1 d_4 - d_2 d_3 = 1 \tag{16}$$

holds. First let $g = \gcd(b_i, b_j)$. Note that this is not zero since $b_i \neq 0$ and $b_j \neq 0$. Then we let $d_1 = b_j/g$ and $d_2 = -b_i/g$. By using them, we set that d_3 and d_4 are defined so that

$d_1d_4 - d_2d_3 = 1$ holds. This is possible since d_1 and d_2 are coprime to each other. These $d_1, \ldots, d_4$ define our desired transformation $D_{i,j}(x^e)$.

We now define a sequence of unimodular matrices such that $\widetilde{T}$ is defined as their product. As explained above, we specify them in terms of corresponding transformations. Let x^L be the maximum degree monomial (without coefficient) appearing in the sequence $(f_1, \ldots, f_k)$, and let i be the index such that $f_i(x)$ contains x^L. Here, we take any one if two or more such polynomials exist. First use $C_{i,k}$ to swap $f_i(x)$ and $f_k(x)$; denote the transformed sequence $(h_1, \ldots, h_k)$. Then, delete the monomial x^L from $h_1(x), \ldots, h_{k-1}(x)$ by using $D_{k-1,k}(x^L), \ldots, D_{1,k}(x^L)$; that is, we apply a transformation by the matrix $D_{k-1,k}(x^L) \cdots D_{1,k}(x^L) \cdot C_{i,k}$. Let $(\overline{f}_1, \ldots, \overline{f}_k)$ be the transformed sequence. Here, $L = \deg(\overline{f}_k)$ is the maximum degree in the new sequence and $\deg(\overline{f}_j) < L$ holds for any $j \in [k-1]$. Then apply similar transformations on the subsequence $\overline{f}_1, \ldots, \overline{f}_{k-1}$. A polynomial sequence $(\widetilde{f}_1, \ldots, \widetilde{f}_k)$ with a strictly increasing degree sequence is obtained by repeating this process. $\square$

Note that if $\mathbf{F}$ is a canonical polynomial sequence, the converted sequence $\widetilde{\mathbf{F}}$ is also canonical since the relation $\widetilde{\mathbf{F}} = \widetilde{T}\mathbf{F}$.

A.2 Analysis for Everywhere Linearly Independent Reduction

We introduce a stronger notion of linearity. For any polynomial sequence $\widehat{\mathbf{F}} = (\widehat{f}_1, \ldots, \widehat{f}_k)$, we say it has *everywhere linearly independent reduction* if we have for any integers $a_1, \ldots, a_k$ and any integer $B \geq 2$,

$$\sum_{i=1}^{k} a_i \widehat{f}_i(x) \equiv_B 0 \;\Leftrightarrow\; \forall i,\ 1 \leq i \leq k\ [\ a_i \equiv 0 \pmod{B}\ ].$$

Lemma 3. *For any polynomial sequence $\mathbf{F}$, there exists a sequence $\widehat{\mathbf{F}}$ having everywhere linearly independent reduction and a regular lower triangular integer matrix $\widehat{T}$ such that $\mathbf{F} = \widehat{T}\widehat{\mathbf{F}}$ holds. Note that from this relation, we have (i) $\det(L(\mathbf{F})) \geq \det(L(\widehat{\mathbf{F}}))$ and (ii) in particular, if $\mathbf{F}$ has a strictly increasing degree sequence, then $\widehat{\mathbf{F}}$ also has a strictly increasing degree sequence.*

Proof. Suppose that a given sequence $\mathbf{F}$ does not have everywhere linearly independent reduction. It can take the minimum i such that there exist $a_1, \ldots, a_i$ and $B \geq 2$ satisfying $\sum_{j=1}^{i} a_j f_j(x) \equiv_B 0$ and $B \not| a_i$. Define $b = \gcd(a_1, \ldots, a_i, B)$. We can assume $b = 1$ without loss of generality; because if not, we can argue with a new polynomial $\sum_{j=1}^{i}(a_j/b)f_j(x)$ and the integer B/b. Letting $B' = \gcd(a_i, B)$, it can also be assumed that $B' = 1$ without loss of generality. Suppose otherwise, there exists $i' < i$ such that $a_{i'}$ is not divisible by B' since $\gcd(a_1, \ldots, a_i, B) = 1$. Let i' be the largest index with this property. Then $a_{i'+1}, \ldots, a_i$ are divisible by B', which yields $\sum_{j=1}^{i'} a_j f_j(x) \equiv_{B'} \sum_{j=1}^{i} a_j f_j(x) \equiv_{B'} 0$, contradicting the choice of i.

Since $\gcd(a_i, B) = 1$, there exist integers c and c' such that $ca_i + c'B = 1$. Consider the sum

$$c\sum_{j=1}^{i} a_j f_j(x) + c'Bf_i(x) \;=\; f_i(x) + \sum_{j=1}^{i-1} c \cdot a_j f_j(x),$$

which is divisible by B by construction. Then, define the new sequence $\overline{\mathbf{F}} = (\overline{f}_1, \ldots, \overline{f}_k)$ by

$$\overline{f}_i(x) = \left(f_i(x) + \sum_{j=1}^{i-1} c \cdot a_j f_j(x) \right) / B$$

and $\overline{f}_j(x) = f_j(x)$ for all $j \in [k] \setminus \{i\}$. Note that the relation between $\mathbf{F}$ and $\overline{\mathbf{F}}$ can be written by

$$\begin{bmatrix} f_1 \\ \vdots \\ f_i \\ \vdots \\ f_k \end{bmatrix} = \begin{bmatrix} 1 & & & & \mathcal{O} & \\ & & \ddots & & & \\ -ca_1 & \cdots & -ca_{i-1} & B & & \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} \overline{f}_1 \\ \vdots \\ \overline{f}_i \\ \vdots \\ \overline{f}_k \end{bmatrix}.$$

Therefore, we have $\mathbf{F} = \overline{T}\overline{\mathbf{F}}$ with a regular lower triangular integer matrix.

Repeating this process defines a new polynomial sequence while it does not have everywhere linearly independent reduction. Note that $\det(\overline{T}) = B \geq 2$ by construction. We also have

$$\det(L(\overline{\mathbf{F}})) = \det(\overline{T})^{-1} \cdot \det(L(\mathbf{F})) \leq \det(L(\mathbf{F}))/2 \tag{17}$$

by Fact 1. That is, the determinant of the matrix corresponding to a new polynomial sequence is reduced by at least 2. On the other hand, the determinants of the matrices corresponding to such polynomial sequences are integers. Thus, this process terminates in a finite number of steps, and the final polynomial sequence must have everywhere linearly independent reduction. Therefore, we let our $\widehat{T}$ by the product of each $\overline{T}$ in all repeats. It is easy to see that $\widehat{T}$ is regular and lower triangular since a finite product of matrices having such properties also has the properties.

Here, the claim (i) is clear from (17). We next prove (ii). From the relation $\mathbf{F} = \widehat{T}\widehat{\mathbf{F}}$, it holds that $\widehat{\mathbf{F}} = (\widehat{T})^{-1}\mathbf{F}$ and it is easy to see that $(\widehat{T})^{-1}$ is also a regular lower triangular matrix (whose entries may not be integers). Thus, it can be written $\widehat{f}_i(x) = \sum_{j=1}^{i} a_{i,j} f_j(x)$ where $a_{i,j}$ are the entries of $(\widehat{T})^{-1}$, and $\deg(\widehat{f}_i) = \deg(f_i)$ since $\mathbf{F}$ has a strictly increasing degree sequence. Hence $\widehat{\mathbf{F}}$ also has a strictly increasing degree sequence. □

A.3 Bounding Determinant

In order to show (11), we bound below the determinant of a lattice constructed from a polynomial sequence with a strictly increasing degree sequence.

Lemma 4. *Let $\mathbf{F} = (f_1, \ldots, f_k)$ be a polynomial sequence with a strictly increasing degree sequence. Then we have*

$$\det(L(\mathbf{F})) \geq \prod_{i=1}^{k} |a_i U^{d_i}| \geq \prod_{i=1}^{k} U^{d_i}. \tag{18}$$

Here, d_i is the degree of $f_i(x)$ and a_i is the coefficient of x^{d_i} in $f_i(x)$.

Proof. Here we show the first inequality. The second one is clear from the relation $|a_i U^{d_i}| \geq U^{d_i}$ for $\forall i \in [k]$.

For each $i \in [k]$, let $\mathbf{y}_i = \mathcal{V}(f_i, U)$. Recall that $L(\mathbf{F})$ is the lattice $L(\mathbf{y}_1, \ldots, \mathbf{y}_k)$. Let $\{\mathbf{y}_1^*, \ldots, \mathbf{y}_k^*\}$ and $\mu_{i,j}$ be the Gram-Schmidt basis of $\{\mathbf{y}_1, \ldots, \mathbf{y}_k\}$ and the corresponding Gram-Schmidt coefficients, respectively. Hence, by definition we have $\det(L(\mathbf{F})) = \prod_{i=1}^{k} |\mathbf{y}_i^*|$. Then for the claim, it suffices to show that $|\mathbf{y}_i^*| \geq |a_i U^{d_i}|$ for all $i \in [k]$. Fix any $i \in [k]$. Note that we have

$$\mathbf{y}_i^* \stackrel{\text{def}}{=} \mathbf{y}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{y}_j^* = \mathbf{y}_i - \sum_{j=1}^{i-1} \alpha_{i,j} \mathbf{y}_j \tag{19}$$

for rational numbers $\alpha_{i,j}$. Since $\mathbf{F}$ has a strictly increasing degree sequence, $\deg(f_j) < \deg(f_i)$ holds for any $j \in [i-1]$. Thus, in the vector $\mathbf{y}_j$ for any $j \in [i-1]$, the coordinate corresponding

to x^{d_i} is zero. Then, the same coordinate of the vector $\sum_{j=1}^{i-1} \alpha_{i,j}\mathbf{y}_j$ in (19) is also zero. Hence $\mathbf{y}_i$ and $\mathbf{y}_i^*$ both have the same value $a_i U^{d_i}$ at this coordinate. Hence we have

$$|a_i U^{d_i}| = |(0,\ldots,0,a_i U^{d_i},0,\ldots,0)| \le |(*,\ldots,*,a_i U^{d_i},0,\ldots,0)| = |\mathbf{y}_i^*|. \quad \square$$

Next, we bound below the degree of canonical polynomial. This helps to show the last inequalities (12) in Section 4. Let $\mathbf{F}$ and $\widehat{\mathbf{F}}$ be a pair of polynomial sequences given by Lemma 3. Assume that both have strictly increasing degree sequences. W.r.t. $\widehat{\mathbf{F}}$, we introduce a sequence of integers $\hat{s}_1,\ldots,\hat{s}_k$, where each $\hat{s}_i$ is defined by

$$\hat{s}_i = \max_{(*)} \nu\left(\sum_{j=1}^{i} a_j \widehat{f}_j(x)\right). \tag{20}$$

Here the condition $(*)$ is that $a_1,\ldots,a_i \in \mathbb{Z}$, $a_i \neq 0$, and at least one a_j is not divisible by N. Then, we can show the following relations (recall that D is the degree of the target equation).

Lemma 5. *Let $\mathbf{F}$, $\widehat{\mathbf{F}}$, and $\hat{s}_1,\ldots,\hat{s}_k$ be those considered above. Assume that both $\mathbf{F}$ and $\widehat{\mathbf{F}}$ have strictly increasing degree sequences, and $f_i(x) \not\equiv_N 0$ for all $i \in [k]$. Let $s_1,\ldots,s_k$ be the sequence $\nu(f_1),\ldots,\nu(f_k)$, and let $\widehat{d}_1,\ldots,\widehat{d}_k$ be the degree sequence of $\widehat{\mathbf{F}}$. Then for every $i \in [k]$, we have (i) $\hat{s}_i \ge s_i$, and (ii) $\widehat{d}_i \ge \hat{s}_i D$.*

Proof. Consider any $i \in [k]$. From the relation $\mathbf{F} = \widehat{T}\widehat{\mathbf{F}}$, we have $f_i(x) = \sum_{j=1}^{i} b_{i,j}\widehat{f}_j(x)$ where $b_{i,j}$ are the entries of $\widehat{T}$; note that $b_{i,i} \neq 0$. Moreover, there exists $j \in [i]$ such that $N \not| b_{i,j}$ since $f_i(x) \not\equiv_N 0$ and that $\widehat{\mathbf{F}}$ has everywhere linearly independent reduction. Thus, the coefficients $b_{1,i},\ldots,b_{i,i}$ satisfy $(*)$ of (20). Therefore, we have $\hat{s}_i \ge s_i = \nu\left(\sum_{j=1}^{i} b_{i,j}\widehat{f}_j\right)$.

For showing (ii), let $h(x)$ be a polynomial that attains $\nu(h) = \hat{s}_i$; we take any one if two or more such polynomials exist. Fix $a_1,\ldots,a_i \in \mathbb{Z}$ so that $h(x) = \sum_{j=1}^{i} a_j \widehat{f}_j(x)$ holds. Since $a_i \neq 0$, we have $\deg(h) = \widehat{d}_i$ $(= \deg(\widehat{f}_i))$. On the other hand, the polynomial can be expressed as $h(x) = \sum_{j=0}^{\hat{s}_i} q_j(x) N^{\hat{s}_i - j} \cdot (F(x))^j$ by $q_j(x) \in \mathbb{Z}[x]$. Note that $q_{\hat{s}_i}(x) \not\equiv_N 0$ holds from $h(x) \not\equiv_N 0$. Therefore, we have $\widehat{d}_i = \deg(h) = \deg(q_{\hat{s}_i}) + \deg\left((F(x))^{\hat{s}_i}\right) \ge \hat{s}_i D$. $\square$

B Appendix: Proof of Theorem 2

For proving Theorem 2, we first prepare several technical tools. Recall that $f(x)$ and $g(x) \in \mathbb{Z}_N[x]$ are called N-coprime if $h(x)|_N f(x)$ and $h(x)|_N g(x)$ imply $h(x) \in \mathbb{Z}_N[x]^\times$. On the other hand, we say that $f(x)$ and $g(x)$ are *strictly coprime* if $f(x)$ and $g(x)$ generate the unit ideal of $\mathbb{Z}_N[x]$. The following example described in [18, p.32] would illustrate this: $f(x)$ and $x-a$ are N-coprime if and only if $f(a) \neq 0$ while they are strictly coprime if and only if $f(a) \in \mathbb{Z}_N^\times$. Recall also that a polynomial $f(x)$ is said to be separable if $f(x)$ and its derivative are N-coprime to each other.

B.1 Extended Euclidean Algorithm for Polynomials Under Modulo N

We first give the algorithm shown in Figure 3, which is a modification of the extended Euclidean algorithm for polynomials. For given N-coprime polynomials $f(x)$ and $g(x) \in \mathbb{M}_N[x]$, it computes a pair of polynomials stated in the following lemma if the polynomial division under modulo N is always defined throughout the execution of the algorithm. If otherwise, i.e., the division is not defined in iteration, it returns a non-trivial factor of the modulo.

Input:	$f(x)$, $g(x) \in \mathbb{M}_N[x]$ which are N-coprime;
Output:	Either a non-trivial factor of N or $u(x), v(x) \in \mathbb{Z}_N[x]$ satisfying $u(x)f(x) + v(x)g(x) \equiv_N 1$;
Step 1:	Initialize as $r_0(x) \leftarrow f(x)$; $u_0(x) \leftarrow 1$; $v_0(x) \leftarrow 0$;
	$r_1(x) \leftarrow g(x)$; $u_1(x) \leftarrow 0$; $v_1(x) \leftarrow 1$;
	while $(\mathrm{rem}(r_0, r_1) \not\equiv_N 0)$ {
Step 2:	$q(x) \leftarrow \mathrm{quo}(r_0, r_1)$;
	$r_2(x) \leftarrow r_0(x) - q(x)r_1(x)$; $u_2(x) \leftarrow u_0(x) - q(x)u_1(x)$;
	$v_2(x) \leftarrow v_0(x) - q(x)v_1(x)$;
Step 3:	**if** $(\mathrm{lc}(r_2) \notin \mathbb{Z}_N^\times$ **then return** $\gcd(N, \mathrm{lc}(r_2))$; // a non-trivial factor of N
Step 4:	$r_0(x) \leftarrow r_1(x)$; $u_0(x) \leftarrow u_1(x)$; $v_0(x) \leftarrow v_1(x)$;
	$c \leftarrow \mathrm{lc}(r_2)^{-1} \pmod N$; $r_1(x) \leftarrow cr_2(x)$; $u_1(x) \leftarrow cu_2(x)$; $v_1(x) \leftarrow cv_2(x)$;
	}
Step 5:	**return** $(u_1(x), v_1(x))$;

Figure 3: Extended Euclidean algorithm for two polynomials in $\mathbb{M}_N[x]$

Lemma 6. *Let $f(x)$ and $g(x) \in \mathbb{M}_N[x]$ be N-coprime. Then, the algorithm in Figure 3 computes either a non-trivial factor of N, or a pair of polynomials $(u(x), v(x))$ satisfying $u(x)f(x) + v(x)g(x) \equiv_N 1$ in polynomial-time w.r.t. $\deg(f)$, $\deg(g)$, and $\log N$. In particular, in the latter case, $f(x)$ and $g(x)$ are strictly coprime.*

Proof. As in the case of the standard Euclidean algorithm, the ideal $\langle r_0(x), r_1(x)\rangle_{\mathbb{Z}_N[x]}$ is equal to $\langle f(x), g(x)\rangle_{\mathbb{Z}_N[x]}$ at the beginning of the while loop. Suppose the algorithm terminated with returning a pair $(u_1(x), v_1(x))$ of polynomials. Then, $\langle f(x), g(x)\rangle_{\mathbb{Z}_N[x]} = \langle r_1(x)\rangle_{\mathbb{Z}_N[x]}$; hence, both $r_1(x)|_N f(x)$ and $r_1(x)|_N g(x)$ hold. Since $f(x)$ and $g(x)$ are N-coprime, $r_1(x) \in \mathbb{Z}_N[x]^\times$. Moreover, it is monic by the construction. Hence $r_1(x) \equiv_N 1$ and $(u_1(x), v_1(x))$ has the required property. The assertion on time complexity follows from the same argument as the one for the standard Euclidean algorithm. □

In the proof of Theorem 2, we give a procedure to compute either $G(x)$ stated in the theorem, or a non-trivial factor of N. If the procedure fails to compute the desired $G(x)$, it gives a way to convert a given non-canonical polynomial to a pair of polynomials having special properties. The following proposition shows that the algorithm in Figure 3 always outputs a non-trivial factor of N from such a polynomial pair.

Proposition 1. *Let $f(x)$ and $g(x) \in \mathbb{M}_N[x]$ be N-coprime. Fix an integer $s \geq 1$. Then, by the algorithm in Figure 3, we can obtain either a non-trivial factor of N, or a pair of polynomials $(u(x), v(x))$ satisfying $u(x)f(x) + v(x)(g(x))^s \equiv_N 1$ in polynomial-time w.r.t. $\deg(f)$, $\deg(g)$, s, and $\log N$. Particularly, if $f(x)$ and $(g(x))^s$ are not N-coprime, a factor of N is always obtained.*

Proof. Consider the execution of the algorithm on $f(x)$ and $g(x)$. In the case we obtain a non-trivial factor of N, we have finished. Otherwise, we obtain $u(x), v(x) \in \mathbb{Z}_N[x]$ satisfying $u(x)f(x) + v(x)g(x) \equiv_N 1$. Hence, we have

$$1 \equiv_N \big(u(x)f(x) + v(x)g(x)\big)^s \equiv_N \left(\sum_{t=1}^{s} \binom{s}{t} (u(x))^t (f(x))^{t-1} (v(x)g(x))^{s-t}\right) f(x) + (v(x))^s (g(x))^s$$

which implies $f(x)$ and $(g(x))^s$ are strictly coprime. A fortiori, $f(x)$ and $(g(x))^s$ are N-coprime. Thus, if $f(x)$ and $(g(x))^s$ are not N-coprime, it must return a non-trivial factor of N. □

B.2 Derivative of Polynomials

We use the derivative of polynomials that are defined in the standard way in $\mathbb{Z}[x]$ (even if they are sometimes treated as polynomials in $\mathbb{Z}_N[x]$). For $s \in \mathbb{N}$, use $f^{(s)}(x)$ to denote the s-th derivative of a polynomial $f(x)$.

Proposition 2. *For $f(x), g(x) \in \mathbb{Z}[x]$ and $s \in \mathbb{N}$ satisfying $\gcd(s!, N) = 1$, if it holds that*

$$\forall v \left[\; f(v) \equiv 0 \pmod{N} \Rightarrow g(v) \equiv 0 \pmod{N^{s+1}} \; \right], \tag{21}$$

we then have

$$\forall v \left[\; f(v) \equiv 0 \pmod{N} \Rightarrow g^{(s)}(v) \equiv 0 \pmod{N} \; \right].$$

Proof. Considering the Taylor expansion of $g(x)$, we have for each $i \in [s+1]$,

$$g(x + iN) = g(x) + (iN) \cdot g^{(1)}(x) + (iN)^2 \cdot \frac{1}{2!} \cdot g^{(2)}(x) + \cdots .$$

Thus, under modulo N^{s+1}, the following polynomial relation holds.

$$\begin{bmatrix} g(x+N) \\ g(x+2N) \\ g(x+3N) \\ \vdots \\ g(x+(s+1)N) \end{bmatrix} \equiv_{N^{s+1}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 3 & \cdots & s+1 \\ 1 & 2^2 & 3^2 & \cdots & (s+1)^2 \\ & \vdots & & \ddots & \vdots \\ 1 & 2^s & 3^s & \cdots & (s+1)^s \end{bmatrix} \begin{bmatrix} g(x) \\ N \cdot g^{(1)}(x) \\ (N^2/2!) \cdot g^{(2)}(x) \\ \vdots \\ (N^s/s!) \cdot g^{(s)}(x) \end{bmatrix} \tag{22}$$

We can easily see the matrix is invertible under modulo N^{s+1} since it is the transpose of a van der Monde matrix and $\gcd(s!, N) = 1$. Then, for constants $c_i \in \mathbb{Z}/N^{s+1}\mathbb{Z}$, we have

$$\Big(N^s/s!\Big) \cdot g^{(s)}(x) \equiv_{N^{s+1}} \sum_{i=1}^{s+1} c_i g(x + iN).$$

On the other hand, we have for any integer i,

$$\forall v \Big[f(v) \equiv 0 \pmod{N} \Rightarrow f(v + iN) \equiv 0 \pmod{N} \Rightarrow g(v + iN) \equiv 0 \pmod{N^{s+1}} \Big].$$

Thus, combining them we have

$$\forall v \Big[f(v) \equiv 0 \pmod{N} \Rightarrow N^s g^{(s)}(v) \equiv 0 \pmod{N^{s+1}} \Big],$$

and the claim holds. □

Consider any s-canonical polynomial $g(x)$ that is defined by $g(x) = \sum_{i=0}^{s} q_i(x) N^{s-i} (F(x))^i$ where $q_i(x) \in \mathbb{Z}[x]$. Then, it is easy to see that $g^{(s)}(x) \equiv_N s! \cdot q_s(x) \cdot (F^{(1)}(x))^s + r(x) \cdot F(x)$ holds for $r(x) \in \mathbb{Z}_N[x]$. Thus, by using the above proposition, if (21) holds for $F(x)$ and $g(x)$, and we then have

$$\forall v \left[\; F(v) \equiv 0 \pmod{N} \Rightarrow q_s(v)(F^{(1)}(v))^s \equiv 0 \pmod{N} \; \right]. \tag{23}$$

B.3 Proof of Theorem 2

We are now ready to prove Theorem 2 of Section 5, stated again as follows.

Theorem 2. *Assume that our target polynomial $F(x)$ is separable. For $F(x)$, N, and m, suppose that we have a non-canonical polynomial $g(x)$, that is, it satisfies $\nu(g) \leq m-1$ and the condition for an initial polynomial (5). We can then compute in polynomial-time w.r.t. $\log N$ and $\deg(g)$, either a non-trivial factor of N or a polynomial $G(x)$ with $\deg(G) \leq \deg(F) - 2$ satisfying*

$$\forall v \left[F(v) \equiv 0 \pmod{N} \Rightarrow G(v) \equiv 0 \pmod{N} \right]. \tag{13}$$

Moreover, $G(x)$ is an N-divisor of $F(x)$, and hence, the separability of $G(x)$ is immediate from that of $F(x)$.

Proof. Put $h(x) = N^{-r}g(x)$ where $r = \mathrm{ord}_N(g)$ and $s = \nu(h)$. (Recall that $\mathrm{ord}_N(g)$ is the largest integer r such that $g(x) \equiv_{N^r} 0$, and $\nu(h)$ is the $\langle F(x), N\rangle_{\mathbb{Z}[x]}$-adic order of $h(x)$.) Then, $s \leq m-r-1$ and it holds that

$$\forall v \left[F(v) \equiv 0 \pmod{N} \Rightarrow h(v) \equiv 0 \pmod{N^{s+1}} \right]. \tag{24}$$

Express $h(x)$ as $h(x) = q_s(x)(F(x))^s + \cdots + N^s q_0(x)$. by using $q_s(x), \ldots, q_0(x) \in \mathbb{Z}_N[x]$. It can be assumed that $q_s(x) \not\equiv_N 0$ and each $q_i(x)$ satisfies $\deg(q_i) < \deg(F)$ without loss of generality. If the latter case fails, we reconstruct $h(x)$ by replacing current $q_i(x)$ by $\mathrm{rem}(q_i, F)$. Next suppose the former case fails to hold, i.e., $q_s(x) \equiv_N 0$. Removing such zero terms, we would have $h(x) = N^a q_{s-a}(x)(F(x))^{s-a} + \cdots$ and $q_{s-a}(x) \not\equiv_N 0$ for $a > 0$. Then, dividing $h(x)$ by N^a, we obtain t $(= s-a)$-canonical polynomial $\overline{h}(x)$ satisfying

$$\forall v \left[F(v) \equiv 0 \pmod{N} \Rightarrow \overline{h}(v) \equiv 0 \pmod{N^{t+1}} \right].$$

Thus, by renaming $\overline{h}(x)$ and t to $h(x)$ and s, we again have (24) with $q_s(x) \not\equiv_N 0$.

Now consider the s-th derivative of $h(x)$. Note that we have (23) for $F(x)$ and this $h(x) = q_s(x)(F(x))^s + \cdots$. Define monic polynomials $q(x)$ and $\bar{F}(x)$ from $q_s(x)$ and $F^{(1)}(x)$ by dividing their leading coefficients, and define $Q(x) = q(x) \cdot (\bar{F}(x))^s$. It clearly satisfies

$$\forall v \left[F(v) \equiv 0 \pmod{N} \Rightarrow Q(v) \equiv 0 \pmod{N} \right]. \tag{25}$$

Noting that for $s = 0$, it can directly take $Q(x)$ by $q(x)$ $(= q_0(x)/\mathrm{lc}(q_0).)$

Consider two cases: $F(x) \not|_N Q(x)$ and $F(x)|_N Q(x)$. For these cases, we show our claim.

The case $F(x)|_N Q(x)$: In this case, a non-trivial factor of N is always obtained efficiently from $F(x)$ and $\bar{F}(x)$.

Note first, that $s \geq 1$. Because otherwise, i.e., $s = 0$, we have $F(x)|_N q_0(x)$ which derives $q_0(x) \equiv_N 0$ by $\deg(q_0) < \deg(F)$, this contradicts the assumption that $q_s(x) \not\equiv_N 0$.

Then, write $Q(x) \equiv_N p(x)F(x)$ by using $p(x) \in \mathbb{M}_N[x]$. Apply the first half of Proposition 1 to $F(x)$ and $\bar{F}(x)$; if we obtain the desired factor, we have finished. Hence, suppose that we obtain a pair of polynomials $u(x), v(x) \in \mathbb{Z}_N[x]$ satisfying $1 \equiv_N u(x)F(x) + v(x)(\bar{F}(x))^s$. Multiply $q(x)$ to both sides, we have

$$\begin{aligned} q(x) &\equiv_N q(x)u(x)F(x) + v(x)Q(x) \equiv_N q(x)u(x)F(x) + v(x)p(x)F(x) \\ &\equiv_N (q(x)u(x) + v(x)p(x))F(x), \end{aligned}$$

which implies that $\deg(q) \geq \deg(F)$ since $q(x)$ and $F(x)$ are monic polynomial. This contradicts the assumption that $\deg(q) < \deg(F)$.

The case $F(x) \not|_N Q(x)$: Let $G(x)$ denote the greatest monic common N-divisor of $F(x)$ and $Q(x)$. Note that the divisor is computed by the standard Euclidean algorithm for polynomials under modulo N; from the problem-setting in the preliminary section, the polynomial division throughout the execution of the algorithm is always defined.

We show that $G(x)$ satisfies the sentence of this theorem. Since it can be written $G(x) \equiv_N u(x)F(x) + v(x)Q(x)$ by $u(x), v(x) \in \mathbb{Z}_N[x]$, (25) implies (13). Hence, it suffices to show that $\deg(G) \leq \deg(F) - 2$. It is clear that $G(x)|_N F(x)$ by definition, and write $F(x) \equiv_N w(x)G(x)$ for $w(x) \in \mathbb{M}_N[x]$. We show that $\deg(w) \geq 2$. Clearly $\deg(w) \geq 1$; thus, assume that $\deg(w) = 1$, that is, $w(x) = x - v_0$ for $v_0 \in \mathbb{Z}_N$. By (13), $F(v_0) \equiv G(v_0) \equiv 0 \pmod{N}$; thus, $(x - v_0)|_N G(x)$ holds by the factor theorem. Hence, we have $(x - v_0)^2|_N F(x)$, which implies that $(x - v_0)|_N F^{(1)}(x)$. This contradicts the separability of $F(x)$. Therefore, it needs that $\deg(w) \geq 2$ and $\deg(G) = \deg(F) - \deg(w) \leq \deg(F) - 2$. $\square$

C Extending Our Framework

We provide a detailed explanation to extend our framework based on his survey paper [7]. By the following two frameworks, it increases the bound of U by a factor in polynomial in $\log N$, but can not achieve an exponential factor.

C.1 Using Integer Valued Polynomials

Lenstra and Howgrave-Graham proposed using an integer valued polynomial, which is defined as a rational coefficient polynomial taking on integer values for all integer points. For example, $\frac{x(x+1)}{2} \in \mathbb{Q}[x]$ is such a polynomial. Use **Int** to denote the set of integer-valued polynomials. Pólya and Szegő [22] proved that $b_i(x) = x(x-1)\cdots(x-i+1)/i!$ for $i = 0, 1, \ldots$ are the integer basis of **Int**, i.e., any $h(x) \in \mathbf{Int}$ can be expressed as

$$h(x) = \sum_{j=0}^{\deg h} a_j b_j(x), \quad a_j \in \mathbb{Z}.$$

We introduce an idea for improvement. For fixed $F(x)$, N, and m, consider the Coppersmith's standard polynomial selection: $g_j(x) = x^{j'} N^{m-i'} (F(x))^{i'}$ for $j = 0, \ldots, Dm+(D-1)$, where we let $i' = \lfloor j/D \rfloor$ and $j' = j \bmod D$. For each j, define a new integer valued polynomial satisfying (5) by $g'_j(x) = A_j g_j(x) + B_j N^m b_j(x)$ where A_j and B_j are integers such that satisfy $A_j \cdot j! + B_j N^m = 1$. This is possible since N was assumed to be hard to factor. Noting that the leading monomial of $g'_j(x)$ is $x^j/j!$, which is decreased by the factor $j!$ comparing to the original polynomial. Hence, the determinant of the constructed lattice is decreased by the factor $\prod_{j=0}^{DM+(D-1)} j!$, and [7] claimed that this extends the bound by a factor around $Dm/4.5$, which is proportional to the degree of polynomials used in the technique.

Following this idea, we extend our framework to allow use of the integer-valued polynomials as the initial polynomials. Thus, w.r.t. $F(x)$, N and m, we say a polynomial $g(x) \in \mathbf{Int}$ is *rational canonical* if it satisfies

$$g(x) = \sum_{i=0}^{m} r_i(x) N^{m-i} (F(x))^i \tag{26}$$

where $r_i(x) \in \mathbf{Int}$. It is easy to see that the polynomial satisfies (5), the condition for an initial polynomial in our standard framework. Then, we also extend the notion of everywhere linearly independent reduction. For a sequence of integer valued sequence $\mathbf{F} = (f_1, \ldots, f_k)$, we say it has

rational everywhere linearly independent reduction if for any integers $a_1, \ldots, a_k$ and any integer $B \geq 2$,

$$\left[v! \sum_{i=1}^{k} a_i \widehat{f_i}(x) \equiv_B 0 \text{ for } v = \max\{\deg f_i | a_i \neq 0\} \right] \Leftrightarrow \forall i,\ 1 \leq i \leq k\ [\ a_i \equiv 0 \pmod{B}\ ].$$

Following the proof of Lemma 3, it can be shown that any sequence in $\mathbf{Int}^k$ can be converted to a sequence having this property and a strictly increasing degree sequence. Let the converted sequence be $\widehat{\mathbf{H}} = (\widehat{h_1}, \ldots, \widehat{h_k})$. The leading coefficient of $\widehat{h_i}(x)$ is a multiple of $1/(\deg(\widehat{h_i}))!$ by the result of Pólya and Szegő.

Hence, by letting $w = \max \deg g_i$, we adopt the inequalities (10) and (11) to the sequence of integer-valued polynomials. For $U \geq w \cdot N^{1/d}$, we have

$$\det(L(\mathbf{G})) \geq \left(\prod_{i=1}^{k} N^{m-\widehat{s_i}} \right) \times \left(\prod_{i=1}^{k} \frac{U^{\widehat{d_i}}}{\widehat{d_i}!} \right) \geq \left(\prod_{i=1}^{k} N^{m-\widehat{s_i}+\widehat{d_i}/D} \right) \times \left(\prod_{i=1}^{k} \frac{w^{\widehat{d_i}}}{\widehat{d_i}!} \right) \geq N^{mk}.$$

Since we assumed the degrees of considered polynomials are bounded in polynomial in $\log N$, the new framework could increase the range by a polynomial factor, but not in exponential.

Next, suppose we have a "rational" non-canonical polynomial, i.e., $g(x) \in \mathbf{Int}$ such that satisfies (5) and does not have the form (26) w.r.t. $F(x)$ and m. Then, the polynomial $(\deg(g))! \cdot g(x)$ is an integer coefficient non-canonical polynomial and derives the same result as in Section 5.

C.2 Using Chebychev Polynomials

Another improvement was achieved by changing representation of polynomials by Boneh. Again we assume that the initial polynomials are selected from $\mathbb{Z}[x]$. Consider the Chebychev polynomials $T_i(x) = \cos(i \cos^{-1} x) \in \mathbb{Z}[x]$; noting that the leading monomial is $2^{i-1}x^i$, and for $|x| \leq 1$, $|T_i(x)| \leq 1$. It is well known that $T_i(x)$ is the degree i polynomial that maximizes the leading coefficient under the condition that $|T_i(x)| \leq 1$ for $|x| \leq 1$.

Using these polynomials, we rewrite the framework of the Coppersmith technique. For fixed bound U and a polynomial $f(x)$, write $f(x)$ as $\sum_{i=0}^{d} c_i T_i(x/U) U^i$. Then, define the *$\mathcal{C}$-vectorization* and *$\mathcal{C}$-$U$-norm* of the polynomial by

$$\mathcal{V}_{\mathcal{C}}(f; U) = [c_0, c_1 U, \ldots, c_d U^d] \text{ and } ||f||_{C-U}^2 = \sum_{i=0}^{d} c_i^2 U^{2i},$$

respectively. Adopting the Howgrave-Graham lemma to this norm as follows, we can provide another framework for the Coppersmith technique.

Lemma 7. *(***Adopted Howgrave-Graham***) Consider any polynomial $f(x) \in \mathbb{Z}[x]$ that is a sum of w Chebychev polynomials. Let W be a non-negative integer satisfying*

$$||f||_{C-U} < W/\sqrt{w}. \tag{27}$$

Then, we have

$$\forall v,\ |v| < U\ \left[\ f(v) \equiv 0 \pmod{W} \Leftrightarrow f(v) = 0\ \right]. \tag{28}$$

The proof is provided by the fact that $|T_i(x/U)| \leq 1$ holds for $|x| < U$. For a sequence of initial polynomials $\mathbf{G} = (g_1, \ldots, g_k)$, use $L_{\mathcal{C}}(\mathbf{G})$ to denote the lattice $L(\mathcal{V}_{\mathcal{C}}(g_1; U), \ldots, \mathcal{V}_{\mathcal{C}}(g_k; U))$. We can then prove that if $\det(L_{\mathcal{C}}(\mathbf{G}))^{1/k} < N^m / A(k)\sqrt{d_{\max} + 1}$, the technique works as the framework in Section 3. Here we recall that $A(k)$ and $d_{\max}$ are the constant that satisfies (4), and the maximum degree of initial polynomials.

Using the Coppersmith's standard polynomial selection, let $\mathbf{G} = (g_1, \ldots, g_k)$, it can be shown that $\det(L_{\mathcal{C}}(\mathbf{G})) = \det(L(\mathbf{G})) \times 2^{-(k-1)(k-2)/2}$. Then it also shown that the range U increases by a factor of $(2^{(k-1)(k-2)/2})^{2/k(k-1)} \approx 2$.

Following this idea, we define our new framework based on that in Section 3. Only expression of polynomials differs from the original one. Fix a series of rational coefficient polynomials $\mathcal{T} = (t_0, t_1, \ldots)$ such that $\deg(t_i) = i$ and $|t_i(x)| \leq 1$ for $|x| \leq 1$ and $i = 0, 1, \ldots$. Using these polynomials, express a polynomial $f(x)$ as $\sum_{i=0}^{d} c_i t_i(x/U) U^i$ and define the $\mathcal{T}$-vectorization and $\mathcal{T}$-U-norm by $\mathcal{V}_{\mathcal{T}}(f; U) = [c_0, c_1 U, \ldots, c_d U^d]$ and $||f||^2_{\mathcal{T}-U} = \sum_{i=0}^{d} c_i^2 U^{2i}$ respectively. As per the above argument, w.r.t. the series $\mathcal{T}$, the adopted Howgrage-Graham lemma holds. Then, for a sequence of initial polynomials $\mathbf{G} = (g_1, \ldots, g_k)$, define $L_{\mathcal{T}}(\mathbf{G})$ to denote the lattice $L(\mathcal{V}_{\mathcal{T}}(g_1; U), \ldots, \mathcal{V}_{\mathcal{T}}(g_k; U))$. We can then prove that the adopted Coppersmith technique works if $\det(L_{\mathcal{T}}(\mathbf{G}))^{1/k} < N^m / A(k) \sqrt{d_{\max} + 1}$.

We prove the impossibility result in this framework. Define ℓ_i as the leading coefficient of $t_i(x)$. Under the condition that $|t_i(x)| \leq 1$ holds for $|x| \leq 1$, we have $|\ell_i| \leq 2^{i-1}$ for $i \geq 1$ and $|\ell_0| \leq 1$; recall that for each i, the equality holds iff $t_i(x) = T_i(x)$. The relation between $\mathcal{T}$-vectorization and the standard vectorization is

$$\mathcal{V}(f; U) = \mathcal{V}_{\mathcal{T}}(f; U) \times \begin{bmatrix} \ell_0 & & & \\ * & \ell_1 & & \\ \vdots & & \ddots & \\ * & * & \cdots & \ell_d \end{bmatrix}, \text{ where } d = \deg(f).$$

Particularly, we have $b_d = \ell_d^{-1} a_d$ for $f(x) = \sum_{i=0}^{d} a_i x^i = \sum_{i=0}^{d} b_i t_i(x/U) U^i$. Then, for a sequence $\widehat{\mathbf{H}} = (\widehat{h_1}, \ldots, \widehat{h_k})$ having everywhere linearly independent reduction and strictly increasing degree sequence (w.r.t. $\mathcal{T}$) that was converted from $\mathbf{G}$, we have by a similar argument to Section 3,

$$\det(L_{\mathcal{T}}(\widehat{\mathbf{H}})) \geq \prod_{i=1}^{k} |\widehat{b_i}| U^{\widehat{d_i}} = \prod_{i=1}^{k} |\widehat{a_i} \ell_{\widehat{d_i}}^{-1}| U^{\widehat{d_i}} \geq \prod_{i=1}^{k} 2^{1-\widehat{d_i}} U^{\widehat{d_i}}$$

where $\widehat{d_i} = \deg(\widehat{h_i})$, $\widehat{a_i}$ and $\widehat{b_i}$ are the leading coefficient of $\widehat{h_i}(x)$ and the coefficient of $T_{\widehat{d_i}}(x/U) U^{\widehat{d_i}}$, respectively. Hence, it can be shown that for $U \geq 2N^{1/d}$,

$$\det(L_{\mathcal{T}}(\widehat{\mathbf{G}})) \geq \left(\prod_{i=1}^{k} N^{m - \hat{s}_i} \right) \times \prod_{i=1}^{k} 2^{1-\widehat{d_i}} U^{\widehat{d_i}} \geq \left(\prod_{i=1}^{k} N^{m - \hat{s}_i + \widehat{d_i}/D} \right) \prod_{i=1}^{k} \left(2^{1-\widehat{d_i}} \cdot 2^{\widehat{d_i}} \right) \geq N^{mk}.$$

Thus, we can increase the range by the factor two compared to the original bound. This is the adopted result for canonical initial polynomials.

We remark that in this framework, the result on the non-canonical polynomials holds in the same way same as Section 5.

As mentioned in [7], two ideas can apply simultaneously. Thus, for the combined framework, the impossibility result is also given for $U \geq 2w \cdot N^{1/d}$.