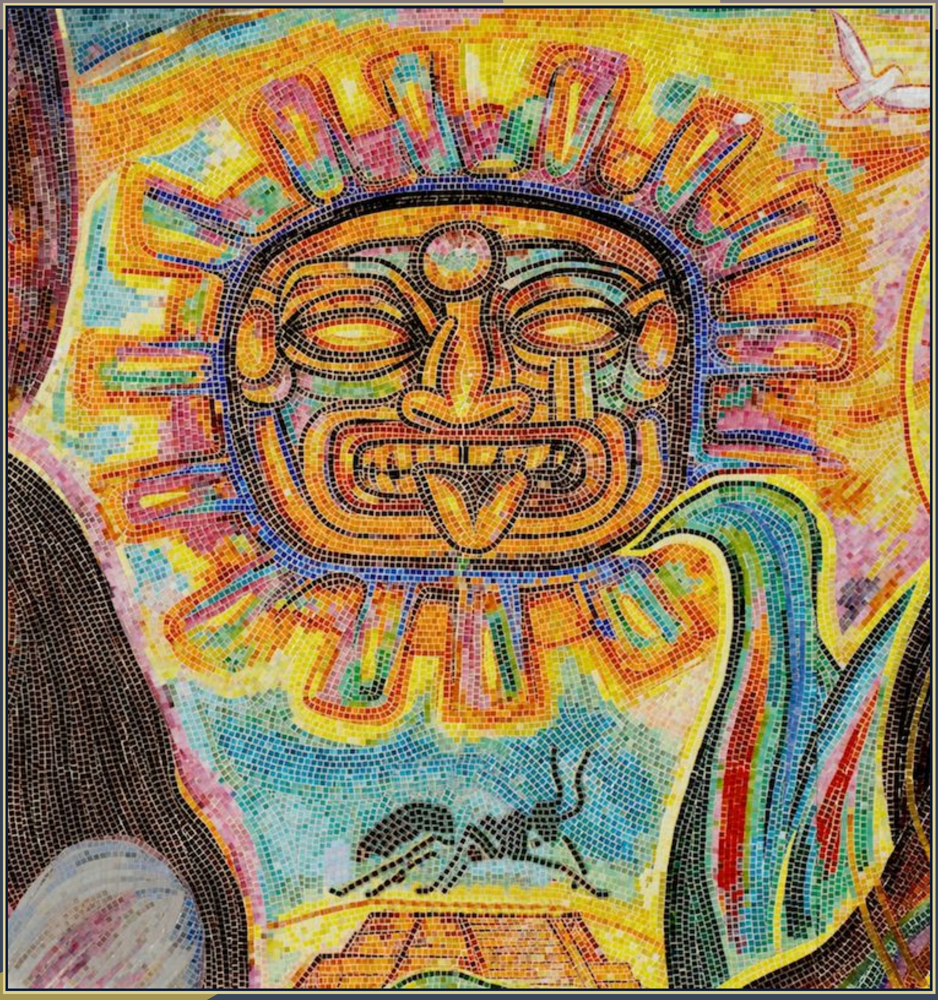
THE OPEN BOOK SERIES 1

ANTS X
Proceedings of the Tenth
Algorithmic Number Theory Symposium

Approximate common divisors via lattices

Henry Cohn and Nadia Heninger



msp

THE OPEN BOOK SERIES 1 (2013)
Tenth Algorithmic Number Theory Symposium
dx.doi.org/10.2140/obs.2013.1.271

Approximate common divisors via lattices

Henry Cohn and Nadia Heninger

We analyze the multivariate generalization of Howgrave-Graham's algorithm for the approximate common divisor problem. In the m-variable case with modulus N and approximate common divisor of size N^{β}, this improves the size of the error tolerated from N^{β^2} to $N^{\beta^{(m+1)/m}}$, under a commonly used heuristic assumption. This gives a more detailed analysis of the hardness assumption underlying the recent fully homomorphic cryptosystem of van Dijk, Gentry, Halevi, and Vaikuntanathan. While these results do not challenge the suggested parameters, a $2^{n^{\varepsilon}}$ approximation algorithm with $\varepsilon < 2/3$ for lattice basis reduction in n dimensions could be used to break these parameters. We have implemented the algorithm, and it performs better in practice than the theoretical analysis suggests.

Our results fit into a broader context of analogies between cryptanalysis and coding theory. The multivariate approximate common divisor problem is the number-theoretic analogue of multivariate polynomial reconstruction, and we develop a corresponding lattice-based algorithm for the latter problem. In particular, it specializes to a lattice-based list decoding algorithm for Parvaresh-Vardy and Guruswami-Rudra codes, which are multivariate extensions of Reed-Solomon codes. This yields a new proof of the list decoding radii for these codes.

1. Introduction

Given two integers, we can compute their greatest common divisor efficiently using Euclid's algorithm. Howgrave-Graham [28] formulated and gave an algorithm to solve an approximate version of this problem, asking the question "What if instead

MSC2010: primary 11Y16; secondary 94A60, 94B35.
Keywords: Coppersmith's algorithm, lattice basis reduction, fully homomorphic encryption, approximate common divisors, list decoding, Parvaresh-Vardy codes, noisy polynomial reconstruction.

of exact multiples of some common divisor, we only know approximations?" In the simplest case, we are given one exact multiple $N = pq_0$ and one near multiple $a_1 = pq_1 + r_1$, and the goal is to learn p, or at least $p \gcd(q_0, q_1)$.

In this paper, we generalize Howgrave-Graham's approach to the case when one is given many near multiples of p. The hardness of solving this problem for small p (relative to the size of the near multiples) was recently proposed as the foundation for a fully homomorphic cryptosystem [21]. Specifically, we can show that improving the approximation of lattice basis reduction for the particular lattices L we are looking at from $2^{\dim L}$ to $2^{(\dim L)^{\varepsilon}}$ with $\varepsilon < 2/3$ would break the suggested parameters in the system. See Section 3 for the details. The approximate common divisor problem is also closely related to the problem of finding small solutions to multivariate polynomials, a problem first posed by Coppersmith [15], and whose various extensions have many applications in cryptanalysis [9].

The multivariate version of the problem allows us to improve the bounds for when the approximate common divisor problem is solvable. Given $N = pq_0$ and m randomly chosen approximate multiples $a_i = pq_i + r_i$ of $p = N^{\beta}$, as well as upper bounds X_i for each $|r_i|$, we can find the perturbations r_i when

$$\sqrt[m]{X_1 \cdots X_m} < N^{(1+o(1))\beta^{(m+1)/m}}.$$

In other words, we can compute approximate common divisors when r_i is as large as $N^{\beta^{(m+1)/m}}$. For $m = 1$, we recover Howgrave-Graham's theorem [28], which handles errors as large as N^{β^2}. As the number m of samples grows large, our bound approaches N^{β}, i.e., the size of the approximate common divisor p. The algorithm runs in polynomial time for fixed m. We cannot rigorously prove that it always works, but it is supported by a heuristic argument and works in practice.

There is an analogy between the ring of integers and the ring of polynomials over a field. Under this analogy, finding a large approximate common divisor of two integers is analogous to reconstructing a polynomial from noisy interpolation information, as we explain in Section 1.2.2. One of the most important applications of polynomial reconstruction is decoding of Reed-Solomon codes. Guruswami and Sudan [25] increased the feasible decoding radius of these codes by giving a list-decoding algorithm that outputs a list of polynomially many solutions to a polynomial reconstruction problem. The analogy between the integers and polynomials was used in [14] to give a proof of the Guruswami-Sudan algorithm inspired by Howgrave-Graham's approach, as well as a faster algorithm.

Parvaresh and Vardy [40] developed a related family of codes with a larger list-decoding radius than Reed-Solomon codes. The decoding algorithm corresponds to simultaneous reconstruction of several polynomials.

In this paper, we observe that the problem of simultaneous reconstruction of multiple polynomials is the exact analogue of the approximate common divisor problem with many inputs, and the improved list-decoding radius of Parvaresh-Vardy codes corresponds to the improved error tolerance in the integer case. We adapt the algorithm for the integers to give a corresponding algorithm to solve the multiple polynomial reconstruction problem.

This algorithm has recently been applied to construct an optimally Byzantine-robust private information retrieval protocol [20]. The polynomial lattice methods we describe are extremely fast in practice, and they speed up the client-side calculations by a factor of several thousand compared with a related scheme that uses the Guruswami-Sudan algorithm. See [20] for more information and timings.

1.1. ***Related work.*** Howgrave-Graham first posed the problem of approximate integer common divisors in [28], and used it to address the problem of factoring when information is known about one of the factors. His algorithm gave a different viewpoint on Coppersmith's proof [15] that one can factor an RSA modulus $N = pq$ where $p \approx q \approx \sqrt{N}$ given the most significant half of the bits of one of the factors. This technique was applied by Boneh, Durfee, and Howgrave-Graham [10] to factor numbers of the form $p^r q$ with r large. Jochemsz and May [29] and Jutla [30] considered the problem of finding small solutions to multivariate polynomial equations, and showed how to do so by obtaining several equations satisfied by the desired roots using lattice basis reduction. Herrmann and May [26] gave a similar algorithm in the case of finding solutions to multivariate linear equations modulo divisors of a given integer. They applied their results to the case of factoring with bits known when those bits might be spread across $\log \log N$ chunks of p. Notably, their results display similar behavior to ours as the number of variables grows large. Sarkar and Maitra [45] studied the multivariate extension of Howgrave-Graham's method and applied it to the problem of implicit factorization.

Most relevantly, van Dijk, Gentry, Halevi, and Vaikuntanathan [21] discussed extensions of Howgrave-Graham's method to larger m and provided a rough heuristic analysis in Appendix B.2 of the longer version of their paper available on the Cryptology ePrint Archive. In particular, they carried out the calculation using the parameter settings $t = k = 2$ from Section 2 below and estimating the determinant by the product of row lengths. They briefly sketched how to extend it to $t = k = d$ for larger values of d. However, they did not optimize the choice of parameters or provide a detailed analysis. They concluded that including products of pairs of equations does worse than the original Howgrave-Graham attack and does not threaten their parameter choices.

Chen and Nguyen [13] gave an algorithm to find approximate common divisors which is not related to the Coppersmith/Howgrave-Graham lattice techniques and which provides an exponential speedup compared with exhaustive search over the possible perturbations.

In addition to the extensive work on polynomial reconstruction and noisy polynomial interpolation in the coding theory literature, the problem in both the single and multiple polynomial cases has been used as a cryptographic primitive, for example in [33], [32], and [3] (broken in [17]). Coppersmith and Sudan [16] gave an algorithm for simultaneous reconstruction of multiple polynomials, assuming random (rather than adversarially chosen) errors. Bleichenbacher, Kiayias, and Yung [7] gave a different algorithm for simultaneous reconstruction of multiple polynomials under a similar probabilistic model. Parvaresh and Vardy [40] were the first to beat the list-decoding performance of Reed-Solomon codes for adversarial errors, by combining multiple polynomial reconstruction with carefully chosen constraints on the polynomial solutions; this allowed them to prove that their algorithm ran in polynomial time, without requiring any heuristic assumptions. Finally, Guruswami and Rudra [24] combined the idea of multipolynomial reconstruction with an optimal choice of polynomials to construct codes that can be list-decoded up to the information-theoretic bound (for large alphabets).

1.2. *Problems and results.*

1.2.1. *Approximate common divisors.* Following Howgrave-Graham, we define the "partial" approximate common divisor problem to be the case when one has $N = pq_0$ and m approximate multiples $a_i = pq_i + r_i$ of p. We want to recover an approximate common divisor. To do so, we will compute $r_1, \dots, r_m$, after which we can simply compute the exact greatest common divisor of $N, a_1 - r_1, \dots, a_m - r_m$.

If the perturbations r_i are allowed to be as large as p, then it is clearly impossible to reconstruct p from this data. If they are sufficiently small, then one can easily find them by a brute force search. The following theorem interpolates between these extremes. As m grows, the bound on the size of r_i approaches the trivial upper bound of p.

Theorem 1 (Partial approximate common divisors). *Given positive integers N, $a_1, \dots, a_m$ and bounds $\beta \gg 1/\sqrt{\log N}$ and $X_1, \dots, X_m$, we can find all $r_1, \dots, r_m$ such that*

$$\gcd(N, a_1 - r_1, \dots, a_m - r_m) \geq N^{\beta}$$

and $|r_i| \leq X_i$, provided that

$$\sqrt[m]{X_1 \cdots X_m} < N^{(1+o(1))\beta^{(m+1)/m}}$$

and that the algebraic independence hypothesis discussed in Section 2 holds. The algorithm runs in polynomial time for fixed m, and the $\gg$ and $o(1)$ are as $N \to \infty$.

For $m = 1$, this theorem requires no algebraic independence hypothesis and is due to Howgrave-Graham [28]. For $m > 1$, not all inputs $N, a_1, \dots, a_m$ will satisfy the hypothesis. Specifically, we must rule out attempting to improve on the $m = 1$ case by deriving $a_2, \dots, a_m$ from a_1, for example by taking a_i to be a small multiple of a_1 plus an additional perturbation (or, worse yet, $a_1 = \dots = a_m$). However, we believe that generic integers will work, for example integers chosen at random from a large range, or at least integers giving independent information in some sense.

We describe the algorithm to solve this problem in Section 2. We follow the general technique of Howgrave-Graham; that is, we use LLL lattice basis reduction to construct m polynomials for which $r_1, \dots, r_m$ are roots, and then we solve the system of equations. The lattice basis reduction is for a lattice of dimension at most $\beta \log N$, regardless of what m is, but the root finding becomes difficult when m is large.

This algorithm is heuristic, because we assume we can obtain m short lattice vectors representing algebraically independent polynomials from the lattice that we will construct. This assumption is commonly made when applying multivariate versions of Coppersmith's method, and has generally been observed to hold in practice. See Section 2 for more details. This is where the restriction to generic inputs becomes necessary; if $a_1, \dots, a_m$ are related in trivial ways, then the algorithm will simply recover the corresponding relations between $r_1, \dots, r_m$, without providing enough information to solve for them.

Note that we are always able to find one nontrivial algebraic relation between $r_1, \dots, r_m$, because LLL will always produce at least one short vector. If we were provided in advance with $m - 1$ additional relations, carefully chosen to ensure that they would be algebraically independent of the new one, then we would have no need for heuristic assumptions. We will see later in this section that this situation arises naturally in coding theory, namely in Parvaresh-Vardy codes [40].

The condition $\beta \gg 1/\sqrt{\log N}$ arises from the exponential approximation factor in LLL. It amounts to $N^{\beta^2} \gg 1$. An equivalent formulation is $\log p \gg \sqrt{\log N}$; i.e., the number of digits in the approximate common factor p must be more than the square root of the number of digits in N. When $m = 1$, this is not a restriction at all, because when p is small enough that N^{β^2} is bounded, there are only a bounded number of possibilities for r_1 and we can simply try all of them. When $m > 1$, the multivariate algorithm can handle much larger values of r_i for a given p, but the $\log p \gg \sqrt{\log N}$ condition dictates that p cannot be any smaller than when $m = 1$. Given a lattice basis reduction algorithm with approximation factor $2^{(\dim L)^\varepsilon}$, one could replace this condition with $\beta^{1+\varepsilon} \log N \gg 1$. If $\varepsilon = 1/m$, then the constraint could be removed entirely in the m-variable algorithm. See Section 2 for the details.

The $\log p \gg \sqrt{\log N}$ condition is the only thing keeping us from breaking the fully homomorphic encryption scheme from [21]. Specifically, improving the approximation of lattice basis reduction for the particular lattices L we are looking at to $2^{(\dim L)^\varepsilon}$ with $\varepsilon < 2/3$ would break the suggested parameters in the system. See Section 3 for the details.

We get nearly the same bounds for the "general" approximate common divisor problem, in which we are not given the exact multiple N.

Theorem 2 (General approximate common divisors). *Given positive integers $a_1, \dots, a_m$ (with $a_i \approx N$ for all i) and bounds $\beta \gg 1/\sqrt{\log N}$ and X, we can find all $r_1, \dots, r_m$ such that*

$$\gcd(a_1 - r_1, \dots, a_m - r_m) \geq N^\beta$$

and $|r_i| \leq X$, provided that

$$X < N^{(C_m + o(1))\beta^{m/(m-1)}},$$

where

$$C_m = \frac{1 - 1/m^2}{m^{1/(m-1)}} \approx 1 - \frac{\log m}{m},$$

and that the algebraic independence hypothesis holds. The algorithm runs in polynomial time for fixed m, and the $\gg$ and $o(1)$ are as $N \to \infty$.

Again, for $m = 2$, this result is due to Howgrave-Graham [28], and no algebraic independence hypothesis is needed.

The proof is very similar to the case when N is known, but the calculations are more tedious because the determinant of the lattice is more difficult to bound. See Section 2.2 for the details.

In [28], Howgrave-Graham gave a more detailed analysis of the behavior for $m = 2$. Instead of our exponent $C_2\beta^2 = \frac{3}{8}\beta^2$, he obtained $1 - \beta/2 - \sqrt{1 - \beta - \beta^2/2}$, which is asymptotic to $\frac{3}{8}\beta^2$ for small β but is slightly better when β is large. We are interested primarily in the case when β is small, so we have opted for simplicity, but one could carry out a similar analysis for all m.

1.2.2. *Noisy multipolynomial reconstruction.* Let F be a field. Given m single-variable polynomials $g_1(z), \dots, g_m(z)$ over F and n distinct points $z_1, \dots, z_n$ in F, evaluating the polynomials at these points yields mn elements $y_{ij} = g_i(z_j)$ of F.

The noisy multipolynomial reconstruction problem asks for the recovery of $g_1, \dots, g_m$ given the evaluation points $z_1, \dots, z_n$, degree bounds ℓ_i on g_i, and possibly incorrect values y_{ij}. Stated more precisely, we wish to find all m-tuples of polynomials $(g_1, \dots, g_m)$ satisfying $\deg g_i \leq \ell_i$, for which there are at least βn values of j such that $g_i(z_j) = y_{ij}$ for all i. In other words, some of the data may have been corrupted, but we are guaranteed that there are at least βn points at which all the values are correct.

Bleichenbacher and Nguyen [8] distinguish the problem of "polynomial reconstruction" from the "noisy polynomial interpolation" problem. Their definition of "noisy polynomial interpolation" involves reconstructing a single polynomial when there are several possibilities for each value. The multivariate version of this problem can be solved using Theorem 5.

This problem is an important stepping stone between single-variable interpolation problems and full multivariate interpolation, in which we reconstruct polynomials of many variables. The multipolynomial reconstruction problem allows us to take advantage of multivariate techniques to prove much stronger bounds, without having to worry about issues such as whether our evaluation points are in general position.

We can restate the multipolynomial reconstruction problem slightly to make the analogy with the integer case clear. Given evaluation points z_j and values y_{ij}, define $N(z) = \prod_j (z - z_j)$, and use ordinary interpolation to find polynomials $f_i(z)$ such that $f_i(z_j) = y_{ij}$. Then we will see shortly that $g_1, \dots, g_m$ solve the noisy multipolynomial reconstruction problem if and only if

$$\deg \gcd \bigl(f_1(z) - g_1(z), \dots, f_m(z) - g_m(z), N(z)\bigr) \geq \beta n.$$

This is completely analogous to the approximate common divisor problem, with $N(z)$ as the exact multiple and $f_1(z), \dots, f_m(z)$ as the approximate multiples.

To see why this works, observe that $g_i(z_j) = y_{ij}$ if and only if $g_i(z) - y_{ij}$ is divisible by $z - z_j$. Thus, $g_i(z_j) = f_i(z_j) = y_{ij}$ if and only if $f_i(z) - g_i(z)$ is divisible by $z - z_j$, and $\deg \gcd\bigl(f_i(z) - g_i(z), N(z)\bigr)$ counts how many j satisfy $g_i(z_j) = y_{ij}$. Finally, to count the j such that $g_i(z_j) = y_{ij}$ for all i, we use

$$\deg \gcd\bigl(f_1(z) - g_1(z), \dots, f_m(z) - g_m(z), N(z)\bigr).$$

This leads us to our result in the polynomial case.

Theorem 3. *Given polynomials $N(z), f_1(z), \dots, f_m(z)$, degree bounds $\ell_1, \dots, \ell_m$, and $\beta \in [0, 1]$, we can find all $g_1(z), \dots, g_m(z)$ such that*

$$\deg \gcd\bigl(f_1(z) - g_1(z), \dots, f_m(z) - g_m(z), N(z)\bigr) \geq \beta \deg N(z)$$

and $\deg g_i \leq \ell_i$, *provided that*

$$\frac{\ell_1 + \dots + \ell_m}{m} < \beta^{(m+1)/m} \deg N(z)$$

and that the algebraic independence hypothesis holds. The algorithm runs in polynomial time for fixed m.

As in the integer case, our analysis depends on an algebraic independence hypothesis, but it may be easier to resolve this issue in the polynomial case, because lattice basis reduction is far more effective and easier to analyze over polynomial rings than it is over the integers.

Parvaresh-Vardy codes [40] are based on noisy multipolynomial reconstruction. A codeword is constructed by evaluating polynomials $f_1, \dots, f_m$ at points $z_1, \dots, z_n$ to obtain mn elements $f_i(z_j)$. In their construction, $f_1, \dots, f_m$ are chosen to satisfy $m-1$ polynomial relations, so that they only need to find one more algebraically independent relation to solve the decoding problem. Furthermore, the $m-1$ relations are constructed so that they must be algebraically independent from the relation constructed by the decoding algorithm. This avoids the need for the heuristic assumption discussed above in the integer case. Furthermore, the Guruswami-Rudra codes [24] achieve improved rates by constructing a system of polynomials so that only n symbols need to be transmitted, rather than mn.

Parvaresh and Vardy gave a list-decoding algorithm using the method of Guruswami and Sudan, which constructs a polynomial by solving a system of equations to determine the coefficients. In our terms, they proved the following theorem:

Theorem 4. *Given polynomials $N(z), f_1(z), \dots, f_m(z)$, degree bounds $\ell_1, \dots, \ell_m$, and $\beta \in [0, 1]$ satisfying*

$$\frac{\ell_1 + \cdots + \ell_m}{m} < \beta^{(m+1)/m} \deg N(z),$$

we can find a nontrivial polynomial $Q(x_1, \dots, x_m)$ such that

$$Q(g_1(z), \dots, g_m(z)) = 0$$

for all $g_1(z), \dots, g_m(z)$ satisfying $\deg g_i \le \ell_i$ and

$$\deg \gcd\bigl(f_1(z) - g_1(z), \dots, f_m(z) - g_m(z), N(z)\bigr) \ge \beta \deg N(z).$$

The algorithm runs in polynomial time.

In Section 4, we give an alternative proof of this theorem using the analogue of lattice basis reduction over polynomial rings. This algorithm requires neither heuristic assumptions nor conditions on β.

2. Computing approximate common divisors

In this section, we describe the algorithm to solve the approximate common divisor problem over the integers.

To derive Theorem 1, we will use the following approach:

(1) Construct polynomials $Q_1, \dots, Q_m$ of m variables such that

$$Q_i(r_1, \dots, r_m) = 0$$

for all $r_1, \dots, r_m$ satisfying the conditions of the theorem.

(2) Solve this system of equations to learn candidates for the roots $r_1, \dots, r_m$.

(3) Test each of the polynomially many candidates to see if it is a solution to the original problem.

In the first step, we will construct polynomials Q satisfying

$$Q(r_1, \dots, r_m) \equiv 0 \pmod{p^k}$$

(for a k to be chosen later) whenever $a_i \equiv r_i \pmod{p}$ for all i. We will furthermore arrange that

$$|Q(r_1, \dots, r_m)| < N^{\beta k}.$$

These two facts together imply that $Q(r_1, \dots, r_m) = 0$ whenever $p \geq N^{\beta}$.

To ensure that $Q(r_1, \dots, r_m) \equiv 0 \pmod{p^k}$, we will construct Q as an integer linear combination of products

$$(x_1 - a_1)^{i_1} \cdots (x_m - a_m)^{i_m} N^{\ell}$$

with $i_1 + \cdots + i_m + \ell \geq k$. Alternatively, we can think of Q as being in the integer lattice generated by the coefficient vectors of these polynomials. To ensure that $|Q(r_1, \dots, r_m)| < N^{\beta k}$, we will construct Q to have small coefficients; i.e., it will be a short vector in the lattice.

More precisely, we will use the lattice L generated by the coefficient vectors of the polynomials

$$(X_1 x_1 - a_1)^{i_1} \cdots (X_m x_m - a_m)^{i_m} N^{\ell}$$

with $i_1 + \cdots + i_m \leq t$ and $\ell = \max\left(k - \sum_j i_j, 0\right)$. Here t and k are parameters to be chosen later. Note that we have incorporated the bounds $X_1, \dots, X_m$ on the desired roots $r_1, \dots, r_m$ into the lattice. We define Q to be the corresponding integer linear combination of $(x_1 - a_1)^{i_1} \cdots (x_m - a_m)^{i_m} N^{\ell}$, without $X_1, \dots, X_m$.

Given a polynomial $Q(x_1, \dots, x_m)$ corresponding to a vector $v \in L$, we can bound $|Q(r_1, \dots, r_m)|$ by the ℓ_1 norm $|v|_1$. Specifically, if

$$Q(x_1, \dots, x_m) = \sum_{j_1, \dots, j_m} q_{j_1 \dots j_m} x_1^{j_1} \cdots x_m^{j_m},$$

then v has entries $q_{j_1 \dots j_m} X_1^{j_1} \cdots X_m^{j_m}$, and

$$\begin{aligned} |Q(r_1, \dots, r_m)| &\leq \sum_{j_1, \dots, j_m} |q_{j_1 \dots j_m}| |r_1|^{j_1} \cdots |r_m|^{j_m} \\ &\leq \sum_{j_1, \dots, j_m} |q_{j_1 \dots j_m}| X_1^{j_1} \cdots X_m^{j_m} \\ &= |v|_1. \end{aligned}$$

Thus, every vector $v \in L$ satisfying $|v|_1 < N^{\beta k}$ gives a polynomial relation between $r_1, \dots, r_m$.

It is straightforward to compute the dimension and determinant of the lattice:

$$\dim L = \binom{t+m}{m},$$

and

$$\det L = (X_1 \cdots X_m)^{\binom{t+m}{m}\frac{t}{m+1}} N^{\binom{k+m}{m}\frac{k}{m+1}}.$$

To compute the determinant, we can choose a monomial ordering so that the basis matrix for this lattice is upper triangular; then the determinant is simply the product of the terms on the diagonal.

Now we apply LLL lattice basis reduction to L. Because all the vectors in L are integral, the m shortest vectors $v_1, \dots, v_m$ in the LLL-reduced basis satisfy

$$|v_1| \le \cdots \le |v_m| \le 2^{(\dim L)/4} (\det L)^{1/(\dim L + 1 - m)}$$

(see Theorem 2 in [26]), and $|v|_1 \le \sqrt{\dim L}\,|v|$ by Cauchy-Schwarz, so we know that the corresponding polynomials Q satisfy

$$|Q(r_1, \dots, r_m)| \le \sqrt{\dim L}\, 2^{(\dim L)/4} (\det L)^{1/(\dim L + 1 - m)}.$$

If

$$\sqrt{\dim L}\, 2^{(\dim L)/4} \det L^{1/(\dim L + 1 - m)} < N^{\beta k}, \tag{1}$$

then we can conclude that $Q(r_1, \dots, r_m) = 0$.

If t and k are large, then we can approximate $\binom{t+m}{m}$ with $t^m/m!$ and $\binom{k+m}{m}$ with $k^m/m!$. The $\sqrt{\dim L}$ factor plays no significant role asymptotically, so we simply omit it (the omission is not difficult to justify). After taking a logarithm and simplifying slightly, our desired equation (1) becomes

$$\frac{t^m}{4km!} + \frac{1}{1 - \frac{(m-1)m!}{t^m}} \left(\frac{m \log_2 X}{m+1} \frac{t}{k} + \frac{\log_2 N}{m+1} \frac{k^m}{t^m} \right) < \beta \log_2 N,$$

where X denotes the geometric mean of $X_1, \dots, X_m$.

The $t^m/(4km!)$ and $(m-1)m!/t^m$ terms are nuisance factors, and once we optimize the parameters they will tend to zero asymptotically. We will take $t \approx \beta^{-1/m} k$ and $\log X \approx \beta^{(m+1)/m} \log N$. Then

$$\frac{m \log X}{m+1} \frac{t}{k} + \frac{\log N}{m+1} \frac{k^m}{t^m} \approx \frac{m}{m+1} \beta \log N + \frac{1}{m+1} \beta \log N = \beta \log N.$$

By setting $\log X$ slightly less than this bound (by a $1 + o(1)$ factor), we can achieve the desired inequality, assuming that the $1 - (m-1)!/t^m$ and $t^m/(4km!)$ terms do not interfere. To ensure that they do not, we take $t \gg m$ and $t^m \ll \beta \log N$ as $N \to \infty$. Note that then $\dim L \le \beta \log N$, which is bounded independently of m.

Specifically, when N is large we can take

$$t = \left\lfloor \frac{(\beta \log N)^{1/m}}{(\beta^2 \log N)^{1/(2m)}} \right\rfloor$$

and

$$k = \lfloor \beta^{1/m} t \rfloor \approx (\beta^2 \log N)^{1/(2m)}.$$

With these parameter settings, t and k both tend to infinity as $N \to \infty$, because $\beta^2 \log N \to \infty$, and they satisfy the necessary constraints. We do not recommend using these parameter settings in practice; instead, one should choose t and k more carefully. However, these choices work asymptotically. Notice that with this approach, $\beta^2 \log N$ must be large enough to allow t/k to approximate $\beta^{-1/m}$. This is a fundamental issue, and we discuss it in more detail in the next subsection.

The final step of the proof is to solve the system of equations defined by the m shortest vectors in the reduced basis to learn $r_1, \dots, r_m$. One way to do this is to repeatedly use resultants to eliminate variables; alternatively, we can use Gröbner bases. See, for example, Chapter 3 of [19].

One obstacle is that the equations may be not algebraically independent, in which case we will not have enough information to complete the solution. In the experiments summarized in Section 6, we sometimes encountered cases when the m shortest vectors were algebraically dependent. However, in every case the vectors represented either (1) irreducible, algebraically independent polynomials, or (2) algebraically dependent polynomials that factored easily into polynomials which all had the desired properties. Thus when the assumption of algebraic dependence failed, it failed because there were fewer than m independent factors among the m shortest relations. In these cases, there were always more than m vectors of ℓ_1 norm less than $N^{\beta k}$, and we were able to complete the solution by using all these vectors. This behavior appears to depend sensitively on the optimization of the parameters t and k.

2.1. ***The*** $\boldsymbol{\beta^2 \log N \gg 1}$ ***requirement.*** The condition that $\beta^2 \log N \gg 1$ is not merely a convenient assumption for the analysis. Instead, it is a necessary hypothesis for our approach to work at all when using a lattice basis reduction algorithm with an exponential approximation factor. In previous papers on these lattice-based techniques, such as [15] or [28], this issue seemingly does not arise, but that is because it is hidden in a degenerate case. When $m = 1$, we are merely ruling out the cases when the bound N^{β^2} on the perturbations is itself bounded, and in those cases the problem can be solved by brute force.

To see why a lower bound on $\beta^2 \log N$ is necessary, we can start with (1). For that equation to hold, we must at least have $2^{(\dim L)/4} < N^{\beta k}$ and $(\det L)^{1/(\dim L)} < N^{\beta k}$, and these inequalities imply that

$$\frac{1}{4}\binom{t+m}{m} < \beta k \log_2 N$$

and

$$\frac{\binom{k+m}{m}\log_2 N}{\binom{t+m}{m}(m+1)} < \beta \log_2 N.$$

Combining them with $\binom{k+m}{m} > k$ yields

$$\frac{1}{4(m+1)} < \beta^2 \log_2 N,$$

so we have an absolute lower bound for $\beta^2 \log N$. Furthermore, one can check that in order for the $2^{(\dim L)/4}$ factor to become negligible compared with $N^{\beta k}$, we must have $\beta^2 \log N \gg 1$.

Given a lattice basis reduction algorithm with approximation factor $2^{(\dim L)^{\varepsilon}}$, we could replace t^m with $t^{\varepsilon m}$ in the nuisance term coming from the approximation factor. Then the condition $t^m \ll \beta \log N$ would become $t^{\varepsilon m} \ll \beta \log N$, and if we combine this with $k \approx \beta^{1/m} t$, we find that

$$k^{\varepsilon m} \approx \beta^{\varepsilon} t^{\varepsilon m} \ll \beta^{1+\varepsilon} \log N.$$

Because $k \geq 1$, the condition $\beta^{1+\varepsilon} \log N \gg 1$ is needed, and then we can take

$$t = \left\lfloor \frac{(\beta \log N)^{1/(\varepsilon m)}}{(\beta^{1+\varepsilon} \log N)^{1/(2\varepsilon m)}} \right\rfloor$$

and

$$k = \lfloor \beta^{1/m} t \rfloor \approx (\beta^{1+\varepsilon} \log N)^{1/(2\varepsilon m)}.$$

2.2. ***Theorem 2.*** The algorithm for Theorem 2 is identical to the above, except that we do not have an exact N, so we omit all vectors involving N from the construction of the lattice L.

The matrix of coefficients is no longer square, so we have to do more work to bound the determinant of the lattice. Howgrave-Graham [28] observed in the two-variable case that the determinant is preserved even under nonintegral row operations, and he used a nonintegral transformation to hand-reduce the matrix before bounding the determinant as the product of the ℓ_2 norms of the basis vectors; furthermore, the ℓ_2 norms are bounded by $\sqrt{\dim L}$ times the ℓ_∞ norms.

The nonintegral transformation that he uses is based on the relation

$$(x_i - a_i) - \frac{a_i}{a_1}(x_1 - a_1) = x_i - \frac{a_i}{a_1}x_1.$$

Adding a multiple of $f(x)(x_1 - a_1)$ reduces $f(x)(x_i - a_i)$ to $f(x)\big(x_i - \frac{a_i}{a_1}x_1\big)$. The advantage of this is that if $x_1 \approx x_i$ and $a_1 \approx a_i$, then $x_i - (a_i/a_1)x_1$ may be much smaller than $x_i - a_i$ was. The calculations are somewhat cumbersome, and we will omit the details (see [28] for more information).

When $a_1, \dots, a_m$ are all roughly N (as in Theorem 2), we get the following values for the determinant and dimension in the m-variable case:

$$\det L \le (N/X)^{\binom{k+m-1}{m}(t-k+1)} X^{m\left(\binom{t+m}{m}\frac{t}{m+1} - \binom{k-1+m}{m}\frac{k-1}{m+1}\right)}$$

and

$$\dim L = \binom{t+m}{m} - \binom{k-1+m}{m}.$$

To optimize the resulting bound, we take $t \approx (m/\beta)^{1/(m-1)}k$.

3. Applications to fully homomorphic encryption

In [21], the authors build a fully homomorphic encryption system whose security relies on several assumptions, among them the hardness of computing an approximate common divisor of many integers. This assumption is used to build a simple "somewhat homomorphic" scheme, which is then transformed into a fully homomorphic system under additional hardness assumptions. In this section, we use Theorem 1 to provide a more precise theoretical understanding of the security assumption underlying this somewhat homomorphic scheme, as well as the related cryptosystem of [18].

For ease of comparison, we will use the notation from the above two papers (see Section 3 of [21]). Let γ be the bit length of N, η be the bit length of p, and ρ be the bit length of each r_i. Using Theorem 1, we can find $r_1, \dots, r_m$ and the secret key p when

$$\rho \le \gamma\beta^{(m+1)/m}.$$

Substituting in $\beta = \eta/\gamma$, we obtain

$$\rho^m \gamma \le \eta^{m+1}.$$

The authors of [21] suggest as a "convenient parameter set to keep in mind" to set $\rho = \lambda$, $\eta = \lambda^2$, and $\gamma = \lambda^5$. Using $m > 3$ we would be able to solve this parameter set, if we did not have the barrier that η^2 must be much greater than γ.

As pointed out in Section 1.2.1, this barrier would no longer apply if we could improve the approximation factor for lattice basis reduction. If we could improve the approximation factor to $2^{(\dim L)^\varepsilon}$, then the barrier would amount to $\beta^{1+\varepsilon}\lambda^5 \gg 1$, where $\beta = \eta/\gamma = \lambda^{-3}$. If $\varepsilon < 2/3$, this would no longer be an obstacle. Given a $2^{(\dim L)^{2/3}/\log \dim L}$ approximation factor, we could take $m = 4$, $k = 1$, and $t = \lfloor 3\lambda^{3/4} \rfloor$ in the notation of Section 2. Then (1) holds, and thus the algorithm works, for all $\lambda \geq 300$.

One might try to achieve these subexponential approximation factors by using blockwise lattice reduction techniques [22]. For an n-dimensional lattice, one can obtain an approximation factor of roughly $\kappa^{n/\kappa}$ in time exponential in κ. For the above parameter settings, the lattice will have dimension on the order of λ^3, and even a $2^{n^{2/3}}$ approximation will require $\kappa > n^{1/3} = \lambda$, for a running time that remains exponential in λ. (Note that for these parameters, using a subexponential-time factoring algorithm to factor the modulus in the "partial" approximate common divisor problem is super-exponential in the security parameter.)

In general, if we could achieve an approximation factor of $2^{(\dim L)^\varepsilon}$ for arbitrarily small ε, then we could solve the approximate common divisor problem for parameters given by any polynomials in λ. Furthermore, as we will see in Section 6, the LLL algorithm performs better in practice on these problems than the theoretical analysis suggests.

In [18], Coron, Mandal, Naccache, and Tibouchi suggest explicit parameter sizes for a modified version of the scheme from [21]. The parameters are well within the range for which the algorithm works, assuming typical LLL performance. However, although our attacks run in time polynomial in the input size, the running time is dependent on the largest input (the total key size) and for these parameters the performance of the lattice-based approach is not competitive with attacks such as Chen and Nguyen [13], which run in time subexponential in the size of the error.

4. Multipolynomial reconstruction

4.1. ***Polynomial lattices.*** For Theorem 3 and Theorem 4, we can use almost exactly the same technique, but with lattices over the polynomial ring $F[z]$ instead of the integers.

By a d-dimensional lattice L over $F[z]$, we mean the $F[z]$-span of d linearly independent vectors in $F[z]^d$. The degree $\deg v$ of a vector v in L is the maximum degree of any of its components, and the determinant $\det L$ is the determinant of a basis matrix (which is well-defined, up to scalar multiplication).

The polynomial analogue of lattice basis reduction produces a basis $b_1, \dots, b_d$ for L such that

$$\deg b_1 + \cdots + \deg b_d = \deg \det L.$$

Such a basis is called a reduced basis (sometimes column or row-reduced, depending on how the vectors are written), and it can be found in polynomial time; see, for example, Section 6.3 in [31]. If we order the basis so that $\deg b_1 \le \cdots \le \deg b_d$, then clearly

$$\deg b_1 \le \frac{\deg \det L}{d},$$

and more generally

$$\deg b_i \le \frac{\deg \det L}{d-(i-1)},$$

because

$$\deg \det L - (d-(i-1)) \deg b_i = \sum_{j=1}^{d} \deg b_j - \sum_{j=i}^{d} \deg b_i \ge 0.$$

These inequalities are the polynomial analogues of the vector length bounds in LLL-reduced lattices, but notice that the exponential approximation factor does not occur. See [14] for more information about this analogy, and [20] for applications that demonstrate the superior performance of these methods in practice.

4.2. ***Theorems 3 and 4.*** In the polynomial setting, we will choose $Q(x_1, \ldots, x_m)$ to be a linear combination (with coefficients from $F[z]$) of the polynomials

$$(x_1 - f_1(z))^{i_1} \cdots (x_m - f_m(z))^{i_m} N(z)^{\ell}$$

with $i_1 + \cdots + i_m \le t$ and $\ell = \max(k - \sum_j i_j, 0)$. We define the lattice L to be spanned by the coefficient vectors of these polynomials, but with x_i replaced with $z^{\ell_i} x_i$ to incorporate the bound on $\deg g_i$, much as we replaced x_i with $X_i x_i$ in Section 2.

As before, we can easily compute the dimension and determinant of L:

$$\dim L = \binom{t+m}{m}$$

and

$$\deg \det L = (\ell_1 + \cdots + \ell_m) \binom{t+m}{m} \frac{t}{m+1} + n \binom{k+m}{m} \frac{k}{m+1},$$

where $n = \deg N(z)$.

Given a polynomial $Q(x_1, \ldots, x_m)$ corresponding to a vector $v \in L$, we can bound $\deg Q(g_1(z), \ldots, g_m(z))$ by $\deg v$. Specifically, suppose

$$Q(x_1, \ldots, x_m) = \sum_{j_1, \ldots, j_m} q_{j_1 \ldots j_m}(z) x_1^{j_1} \cdots x_m^{j_m};$$

then v is the vector whose entries are $q_{j_1 \dots j_m}(z) z^{j_1 \ell_1 + \dots + j_m \ell_m}$, and

$$\begin{aligned}
\deg Q(g_1(z), \dots, g_m(z)) \\
&\le \max_{j_1,\dots,j_m} (\deg q_{j_1 \dots j_m}(z) + j_1 \deg g_1(z) + \dots + j_m \deg g_m(z)) \\
&\le \max_{j_1,\dots,j_m} (\deg q_{j_1 \dots j_m}(z) + j_1 \ell_1 + \dots + j_m \ell_m) \\
&= \deg v.
\end{aligned}$$

Let $v_1, \dots, v_{\dim L}$ be a reduced basis of L, arranged in increasing order by degree. If

$$\frac{\deg \det L}{\dim L - (m-1)} < \beta k n, \tag{2}$$

then each of $v_1, \dots, v_m$ yields a polynomial relation Q_i such that

$$Q_i(g_1(z), \dots, g_m(z)) = 0,$$

because by the construction of the lattice, $Q_i(g_1(z), \dots, g_m(z))$ is divisible by the k-th power of an approximate common divisor of degree βn, while

$$\deg Q_i(g_1(z), \dots, g_m(z)) \le \deg v_i < \beta k n.$$

Thus we must determine how large $\ell_1 + \dots + \ell_m$ can be, subject to the inequality (2).

If we set $t \approx k \beta^{-1/m}$ and

$$\frac{\ell_1 + \dots + \ell_m}{m} < n \beta^{(m+1)/m},$$

then inequality (2) is satisfied when t and k are sufficiently large. Because there is no analogue of the LLL approximation factor in this setting, we do not have to worry about t and k becoming too large (except for the obvious restriction that $\dim L$ must remain polynomially bounded), and there is no lower bound on β. Furthermore, we require no $1 + o(1)$ factors, because all degrees are integers and all the quantities we care about are rational numbers with bounded numerators and denominators; thus, any sufficiently close approximation might as well be exact, and we can achieve this when t and k are polynomially large. More precisely, without loss of generality we can take βn to be an integer. Then the inequality

$$\frac{\ell_1 + \dots + \ell_m}{m} < n \beta^{(m+1)/m}$$

is equivalent to $n(\ell_1 + \dots + \ell_m)^m < (n\beta)^{m+1} m^m$ and hence

$$n(\ell_1 + \dots + \ell_m)^m \le (n\beta)^{m+1} m^m - 1$$

by integrality. Thus, $(\ell_1 + \cdots + \ell_m)/m$ is smaller than $n\beta^{(m+1)/m}$ by at least a factor of $(1 - n^{-(m+1)} m^{-m})^{1/m}$, and this factor is enough to ensure that inequality (2) holds when t and k are only polynomially large.

5. Higher-degree polynomials

It is possible to generalize the results in the previous sections to find solutions of a system of higher-degree polynomials modulo divisors of N.

Theorem 5. *Given a positive integer N and m monic polynomials $h_1(x), \ldots, h_m(x)$ over the integers, of degrees $d_1, \ldots, d_m$, and given any $\beta \gg 1/\sqrt{\log N}$ and bounds $X_1, \ldots, X_m$, we can find all $r_1, \ldots, r_m$ such that*

$$\gcd(N, h_1(r_1), \ldots, h_m(r_m)) \geq N^{\beta}$$

and $|r_i| \leq X_i$, provided that

$$\sqrt[m]{X_1^{d_1} \cdots X_m^{d_m}} < N^{(1+o(1))\beta^{(m+1)/m}}$$

and that the algebraic independence hypothesis holds. The algorithm runs in polynomial time for fixed m.

The $m = 1$ case does not require the algebraic independence hypothesis, and it encompasses both Howgrave-Graham and Coppersmith's theorems [28; 15]; it first appeared in [36].

In the case where $X_1 = \cdots = X_m$, the bound becomes $N^{\beta^{(m+1)/m}/\bar{d}}$, where $\bar{d} = (d_1 + \cdots + d_m)/m$ is the average degree.

Theorem 6. *Given a polynomial $N(z)$ and m monic polynomials $h_1(x), \ldots, h_m(x)$ over $F[z]$, of degrees $d_1, \ldots, d_m$ in x, and given degree bounds $\ell_1, \ldots, \ell_m$ and $\beta \in [0, 1]$, we can find all $g_1(z), \ldots, g_m(z)$ in $F[z]$ such that*

$$\deg \gcd\big(N(z), h_1(g_1(z)), \ldots, h_m(g_m(z))\big) \geq \beta \deg N(z)$$

and $\deg g_i(z) \leq \ell_i$, provided that

$$\frac{\ell_1 d_1 + \cdots + \ell_m d_m}{m} < \beta^{(m+1)/m} \deg N(z)$$

and that the algebraic independence hypothesis holds. The algorithm runs in polynomial time for fixed m.

The algorithms are exactly analogous to those for the degree-1 cases, except that $x_i - a_i$ (or $x_i - f_i(z)$) is replaced with $h_i(x_i)$.

6. Implementation

We implemented the number-theoretic version of the partial approximate common divisor algorithm using Sage [47]. We used Magma [11] to do the LLL and Gröbner basis calculations.

We solved the systems of equations by computing a Gröbner basis with respect to the lexicographic monomial ordering, to eliminate variables. Computing a Gröbner basis can be extremely slow, both in theory and in practice. We found that it was more efficient to solve the equations modulo a large prime, to limit the bit length of the coefficients in the intermediate and final results. Because $r_1, \dots, r_m$ are bounded in size, we can simply choose a prime larger than $2 \max_i |r_i|$.

We ran our experiments on a computer with a 3.30 GHz quad-core Intel Core i5 processor and 8 GB of RAM. Table 1 shows a selection of sample running times (in seconds) for various parameter settings. For comparison, the table includes the $m = 1$ case, which is Howgrave-Graham's algorithm. The rows for which no timing information is listed give example lattice dimensions for larger inputs, in order to illustrate the limiting behavior of the algorithm.

The performance of the algorithm depends on the ratio of t to k, which should be approximately $\beta^{-1/m}$. Incorrectly optimized parameters often perform much worse than correctly optimized parameters. For example, when $m=3$, $\log_2 N = 1000$, and $\log_2 p = 200$, taking $(t, k) = (4, 2)$ can handle 84-bit perturbations r_i, as one can see in Table 1, but taking $(t, k) = (4, 3)$ cannot even handle 60 bits.

For large m, we experimented with using the nonoptimized parameters $(t, k) = (1, 1)$, as reported in Table 1. For the shortest vector only, the bounds would replace the exponent $\beta^{(m+1)/m}$ with $(m + 1)\beta/m - 1/m$, which is its tangent line at $\beta = 1$. This bound is always worse, and it is trivial when $\beta \le 1/(m + 1)$, but it still approaches the optimal exponent β for large m. Our analysis does not yield a strong enough bound for the m-th largest vector, but in our experiments the vectors found by LLL are much shorter than predicted by the worst-case bounds, as described below. Furthermore, the algorithm runs extremely quickly with these parameters, because the lattices have lower dimensions and the simultaneous equations are all linear.

The last column of the table gives the value of the "LLL factor" λ, which describes the approximation ratio obtained by LLL in the experiment. Specifically, the value of λ satisfies

$$|v_m| \approx \lambda^{\dim L} (\det L)^{1/(\dim L)},$$

where v_m is the m-th smallest vector in the LLL-reduced basis for L. Empirically, we find that all of the vectors in the reduced basis are generally quite close in size, so this estimate is more appropriate than using $1/(\dim L - (m - 1))$ in the exponent

m	$\log_2 N$	$\log_2 p$	$\log_2 r$	t	k	$\dim L$	LLL	Gröbner	λ
1	1000	200	36	41	8	42	12.10	—	1.037
1	1000	200	39	190	38	191			
1	1000	400	154	40	16	41	34.60	—	1.023
1	1000	400	156	82	33	83	4554.49	—	1.029
1	1000	400	159	280	112	281			
2	1000	200	72	9	4	55	25.22	0.94	1.030
2	1000	200	85	36	16	703			
2	1000	400	232	10	6	66	126.27	5.95	1.038
2	1000	400	238	15	9	136	15720.95	25.86	1.019
2	1000	400	246	46	29	1128			
3	1000	200	87	5	3	56	18.57	1.20	1.038
3	1000	200	102	14	8	680			
3	1000	400	255	4	3	35	2.86	2.13	1.032
3	1000	400	268	7	5	120	1770.04	25.43	1.040
3	1000	400	281	19	14	1540			
4	1000	200	94	3	2	35	1.35	0.54	1.028
4	1000	200	111	8	5	495			
4	1000	400	279	4	3	70	38.32	9.33	1.035
4	1000	400	293	10	8	1001			
5	1000	200	108	3	2	56	7.35	1.42	1.035
5	1000	200	110	4	3	126	738.57	7.28	1.037
5	1000	400	278	3	2	56	1.86	0.90*	0.743
6	1000	200	115	3	2	84	31.51	3.16	1.038
6	1000	400	297	3	2	84	3.97	1.34*	0.586
7	1000	200	120	3	2	120	203.03	7.73	1.046
7	1000	400	311	3	2	120	12.99	2.23*	0.568
12	1000	400	347	1	1	13	0.01	0.52	1.013
18	1000	400	364	1	1	19	0.03	1.08	1.032
24	1000	400	372	1	1	25	0.04	1.93	1.024
48	1000	400	383	1	1	49	0.28	8.37	1.030
96	1000	400	387	1	1	97	1.71	27.94	1.040

Table 1. Timings, in seconds, of the LLL and Gröbner basis portions of our implementation of the integer partial approximate common divisor algorithm, for various choices of the parameters m, N, p, r, t, and k. Rows for which no timings are listed give sample parameters for more extreme calculations. The meanings of the final column and of the timings marked with an asterisk are explained in the text. We include results for the nonoptimized parameters $t = k = 1$, which perform well for a large number of samples but give a weaker result than Theorem 1.

(which we did in the theoretical analysis, in order to get a rigorous bound). The typical value is about 1.02, which matches the behavior one would expect from LLL on a randomly generated lattice [37], whose successive minima will all be close to $\det L^{1/(\dim L)}$.

Because of this, the reduced lattice bases in practice contain many more than m suitable polynomials, and we were able to speed up some of the Gröbner basis calculations in Table 1 by including all of them in the basis. Even using all the vectors with ℓ_1 norm less than $N^{\beta k}$ is overly conservative in many cases, because vectors that do not satisfy this constraint can still lead to valid relations. Our code initially tries using every vector in the reduced basis except the longest one; if that fails, we fall back on the m shortest vectors. We also experimented with using just those with ℓ_1 norm less than $N^{\beta k}$, but in our experiments this bound was often violated even for polynomials that did vanish. Including more polynomials in the Gröbner basis calculation in many cases leads to substantially better running times than using just m vectors.

A handful of our experimental parameters resulted in lattices whose shortest vectors were much shorter than the expected bounds; this tended to correlate with a small sublattice of algebraically dependent vectors. We marked cases where we encountered algebraically dependent relations with an asterisk in Table 1. In each case, we were still able to solve the system of equations by including more relations from the lattice and solving this larger system.

Acknowledgments

We thank Chris Umans and Alex Vardy for suggesting looking at Parvaresh-Vardy codes, and Martin Albrecht for advice on computing Gröbner bases in practice. N.H. would like to thank MIT, CSAIL, and Microsoft Research New England for their hospitality during the course of this research. This material is based upon work supported by an AT&T Labs Graduate Fellowship and by the National Science Foundation under Award No. DMS-1103803.

References

[1] ACM (ed.), *STOC'03: Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, New York, ACM Press, 2003. MR 2005j:68009

[2] ACM (ed.), *STOC'08: Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, New York, ACM Press, 2008. MR 2010m:68004

[3] Daniel Augot and Matthieu Finiasz, *A public key encryption scheme based on the polynomial reconstruction problem*, in Biham [6], 2003, pp. 229–240. MR 2005d:94075

[4] Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger (eds.), *Automata, languages and programming*, Lecture Notes in Computer Science, no. 2719, Berlin, Springer, 2003. MR 2005b:68008

[5] Feng Bao, Robert Deng, and Jianying Zhou (eds.), *Public Key Cryptography—PKC* 2004, Lecture Notes in Computer Science, no. 2947, Berlin, Springer, 2004.

[6] Eli Biham (ed.), *Advances in cryptology—EUROCRYPT* 2003, Lecture Notes in Computer Science, no. 2656, Berlin, Springer, 2003. MR 2005c:94003

[7] Daniel Bleichenbacher, Aggelos Kiayias, and Moti Yung, *Decoding of interleaved Reed Solomon codes over noisy data*, in Baeten et al. [4], 2003, pp. 97–108. MR 2005b:94062

[8] Daniel Bleichenbacher and Phong Q. Nguyen, *Noisy polynomial interpolation and noisy Chinese remaindering*, in Preneel [43], 2000, pp. 53–69. MR 2001b:94030

[9] Dan Boneh and Glenn Durfee, *Cryptanalysis of RSA with private key* d *less than* $N^{0.292}$, IEEE Trans. Inform. Theory **46** (2000), no. 4, 1339–1349. MR 2002g:94034

[10] Dan Boneh, Glenn Durfee, and Nick Howgrave-Graham, *Factoring* $N = p^r q$ *for large* r, in Wiener [49], 1999, pp. 326–337. MR 2000i:11188

[11] Wieb Bosma, John Cannon, and Catherine Playoust, *The Magma algebra system. I. The user language*, J. Symbolic Comput. **24** (1997), no. 3-4, 235–265. MR 1484478

[12] Bernard Chazelle (ed.), *Innovations in Computer Science—ICS* 2011, Beijing, Tsinghua University Press, 2011.

[13] Yuanmi Chen and Phong Q. Nguyen, *Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers*, in Pointcheval and Johansson [42], 2012, pp. 502–519.

[14] Henry Cohn and Nadia Heninger, *Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding*, in Chazelle [12], 2011, pp. 298–308, full version at arXiv:1008.1284 [math.NT].

[15] Don Coppersmith, *Small solutions to polynomial equations, and low exponent RSA vulnerabilities*, J. Cryptology **10** (1997), no. 4, 233–260. MR 99b:94027

[16] Don Coppersmith and Madhu Sudan, *Reconstructing curves in three (and higher) dimensional space from noisy data*, in ACM [1], 2003, pp. 136–142. MR 2005k:94004

[17] Jean-Sébastien Coron, *Cryptanalysis of a public-key encryption scheme based on the polynomial reconstruction problem*, in Bao et al. [5], 2004, pp. 14–27.

[18] Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi, *Fully homomorphic encryption over the integers with shorter public keys*, in Rogaway [44], 2011, pp. 487–504. MR 2874875

[19] David Cox, John Little, and Donal O'Shea, *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*, Undergraduate Texts in Mathematics, Springer, New York, 1992. MR 93j:13031

[20] Casey Devet, Ian Goldberg, and Nadia Heninger, *Optimally robust private information retrieval*, in Kohno [34], 2012, pp. 269–283, extended version at http://eprint.iacr.org/2012/083.

[21] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan, *Fully homomorphic encryption over the integers*, in Gilbert [23], 2010, pp. 24–43, extended version at http://eprint.iacr.org/2009/616. MR 2660481

[22] Nicolas Gama and Phong Q. Nguyen, *Finding short lattice vectors within Mordell's inequality*, in ACM [2], 2008, pp. 207–216. MR 2011c:68214

[23] Henri Gilbert (ed.), *Advances in cryptology—EUROCRYPT* 2010, Lecture Notes in Computer Science, no. 6110, Berlin, Springer, 2010. MR 2011g:94001

[24] Venkatesan Guruswami and Atri Rudra, *Explicit codes achieving list decoding capacity: error-correction with optimal redundancy*, IEEE Trans. Inform. Theory **54** (2008), no. 1, 135–150. MR 2010b:94096

[25] Venkatesan Guruswami and Madhu Sudan, *Improved decoding of Reed-Solomon and algebraic-geometry codes*, IEEE Trans. Inform. Theory **45** (1999), no. 6, 1757–1767. MR 2000j:94033

[26] Mathias Herrmann and Alexander May, *Solving linear equations modulo divisors*: *on factoring given any bits*, in Pieprzyk [41], 2008, pp. 406–424. MR 2546108

[27] Florian Hess, Sebastian Pauli, and Michael Pohst (eds.), *Algorithmic number theory*, Lecture Notes in Computer Science, no. 4076, Berlin, Springer, 2006. MR 2007h:11001

[28] Nick Howgrave-Graham, *Approximate integer common divisors*, in Silverman [46], 2001, pp. 51–66. MR 2003h:11160

[29] Ellen Jochemsz and Alexander May, *A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants*, in Lai and Chen [35], 2006, pp. 267–282. MR 2009h:94128

[30] Charanjit S. Jutla, *On finding small solutions of modular multivariate polynomial equations*, in Nyberg [38], 1998, pp. 158–170. MR 2000k:94033

[31] Thomas Kailath, *Linear systems*, Prentice-Hall, Englewood Cliffs, NJ, 1980. MR 82a:93001

[32] Aggelos Kiayias and Moti Yung, *Polynomial reconstruction based cryptography* (*a short survey*), in Vaudenay and Youssef [48], 2001, pp. 129–133. MR 2054433

[33] ———, *Secure games with polynomial expressions*, in Orejas et al. [39], 2001, pp. 939–950. MR 2066563

[34] Tadayoshi Kohno (ed.), 21*st USENIX Security Symposium*, Berkeley, The USENIX Association, 2012.

[35] Xuejia Lai and Kefei Chen (eds.), *Advances in cryptology—ASIACRYPT* 2006, Lecture Notes in Computer Science, no. 4284, Berlin, Springer, 2006. MR 2009e:94091

[36] Alexander May, *New RSA vulnerabilities using lattice reduction methods*, Ph.D. thesis, Universität Paderborn, 2003. http://nbn-resolving.de/urn:nbn:de:hbz:466-20030101205

[37] Phong Q. Nguyen and Damien Stehlé, *LLL on the average*, in Hess et al. [27], 2006, pp. 238–256. MR 2008a:11154

[38] Kaisa Nyberg (ed.), *Advances in cryptology—EUROCRYPT* '98, Lecture Notes in Computer Science, no. 1403, Berlin, Springer, 1998. MR 2000h:94004

[39] Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen (eds.), *Automata*, *languages and programming*, Lecture Notes in Computer Science, no. 2076, Berlin, Springer, 2001. MR 2005a:68007

[40] Farzad Parvaresh and Alexander Vardy, *Correcting errors beyond the Guruswami-Sudan radius in polynomial time*, Proceedings of the 46th IEEE Symposium on Foundations of Computer Science held in Pittsburgh, October 23–25, 2005 (Los Alamitos, CA), Institute of Electrical and Electronics Engineers, IEEE Computer Society, 2005, pp. 285–294.

[41] Josef Pieprzyk (ed.), *Advances in cryptology—ASIACRYPT* 2008, Lecture Notes in Computer Science, no. 5350, Berlin, Springer, 2008. MR 2010j:94005

[42] David Pointcheval and Thomas Johansson (eds.), *Advances in cryptology—EUROCRYPT* 2012, Lecture Notes in Computer Science, no. 7237, Berlin, Springer, 2012.

[43] Bart Preneel (ed.), *Advances in cryptology—EUROCRYPT* 2000, Lecture Notes in Computer Science, no. 1807, Berlin, Springer, 2000. MR 2001b:94028

[44] Phillip Rogaway (ed.), *Advances in cryptology—CRYPTO* 2011, Lecture Notes in Computer Science, no. 6841, Heidelberg, Springer, 2011. MR 2012i:94009

[45] Santanu Sarkar and Subhamoy Maitra, *Approximate integer common divisor problem relates to implicit factorization*, IEEE Trans. Inform. Theory **57** (2011), no. 6, 4002–4013. MR 2012f:94121

[46] Joseph H. Silverman (ed.), *Cryptography and lattices*, Lecture Notes in Computer Science, no. 2146, Berlin, Springer, 2001. MR 2002m:11002

[47] W. A. Stein et al., *Sage Mathematics Software* (*version* 4.6.2), 2011. http://www.sagemath.org

[48] Serge Vaudenay and Amr M. Youssef (eds.), *Selected areas in cryptography*: *Revised papers from the* 8*th Annual International Workshop* (*SAC* 2001) *held in Toronto*, *ON*, *August* 16–17, 2001, Lecture Notes in Computer Science, no. 2259, Springer, Berlin, 2001. MR 2004k:94066

[49] Michael Wiener (ed.), *Advances in cryptology—CRYPTO* '99, Lecture Notes in Computer Science, no. 1666, Berlin, Springer, 1999. MR 2000h:94003

HENRY COHN: cohn@microsoft.com
Microsoft Research New England, One Memorial Drive, Cambridge, MA 02142, United States

NADIA HENINGER: nadiah@cis.upenn.edu
Department of Computer Science, Princeton University, Princeton, NJ 08540, United States
Current address: *Department of Computer and Information Science, 3330 Walnut St., Philadelphia, PA 19104, United States*

VOLUME EDITORS

Everett W. Howe
Center for Communications Research
4320 Westerra Court
San Diego, CA 92121-1969
United States

Kiran S. Kedlaya
Department of Mathematics
University of California, San Diego
9500 Gilman Drive #0112
La Jolla, CA 92093-0112

Front cover artwork based on a detail of
Chicano Legacy 40 Años ©2010 Mario Torero.

Electronic copies can be obtained free of charge from http://msp.org/obs/1
and printed copies can be ordered from MSP (contact@msp.org).

The Open Book Series is a trademark of Mathematical Sciences Publishers.

ISSN: 2329-9061 (print), 2329-907X (electronic)

ISBN: 978-1-935107-00-2 (print), 978-1-935107-01-9 (electronic)

First published 2013.

---

msp

MATHEMATICAL SCIENCES PUBLISHERS

798 Evans Hall #3840, c/o University of California, Berkeley CA 94720-3840
contact@msp.org **http://msp.org**

THE OPEN BOOK SERIES 1

Tenth Algorithmic Number Theory Symposium

The Algorithmic Number Theory Symposium (ANTS), held biennially since 1994, is the premier international forum for research in computational number theory. ANTS is devoted to algorithmic aspects of number theory, including elementary, algebraic, and analytic number theory, the geometry of numbers, arithmetic algebraic geometry, the theory of finite fields, and cryptography.

This volume is the proceedings of the tenth ANTS meeting, held July 9–13, 2012, at the University of California, San Diego. It includes revised and edited versions of the 25 refereed papers presented at the conference, together with extended abstracts of two of the five invited talks.

TABLE OF CONTENTS