Fast Coppersmith method over the polynomials: finding a reduced basis via approximation

Vincent Neiger

ENS de Lyon
Laboratoire LIP, CNRS, Inria, UCBL, U. Lyon

Monthly lattice and crypto meetings
March 3, 2016
http://perso.ens-lyon.fr/damien.stehle/LATTICE_MEETINGS.html

Abstract

In this report, we present a technique to compute small modular roots of a polynomial $F(Y)$. When $F(Y)$ is over the integers $\mathbb{Z}$, this is the well-known *Coppersmith technique* [18] which has been largely studied in the last twenty years, in particular for its applications in cryptography. Here, we focus on the situation of a polynomial $F(Y)$ over the univariate polynomials $\mathbb{K}[X]$, which has mainly received attention through the specific case of the Guruswami-Sudan algorithm for the list-decoding of Reed-Solomon codes [24].

Here, we first present this analogue to the Coppersmith technique over $\mathbb{K}[X]$. Then, we focus on the so-called *interpolation step*, and we sketch the folklore solution via fast polynomial lattice basis reduction. Finally, we show that this interpolation step can be solved more efficiently by rather solving a system of linear modular equations over $\mathbb{K}[X]$.

1 Coppersmith technique over the univariate polynomials

In what follows, $\mathbb{K}$ denotes the base field, $\mathcal{P} = \mathbb{K}[X]$ is the ring of univariate polynomials over $\mathbb{K}$, and $\mathcal{P}[Y]$ is the ring of univariate polynomials in Y over $\mathcal{P}$.

Problem. In the polynomial case, the Coppersmith technique solves Problem 1 below.

We note that one can consider M monic without loss of generality. The size of the input, meaning the number of coefficients from $\mathbb{K}$ used to represent it, is at most dn for F and n for M. The size of the output is not obvious to analyze: the number of solutions depends on the four parameters d, n, k and t, and may be exponential in d or n.

Under the assumption $t^2 > knd$, the $\mathbb{K}[X]$ version of the Coppersmith technique solves Problem 1 in time polynomial in d and n (for a detailed study, see for example [15, 17]). This technique consists of two main steps:

- first, the *interpolation step* which builds a polynomial $Q \in \mathcal{P}[Y]$ that satisfies $Q(p) = 0$ for all solutions p;
- second, the *root-finding step* which computes the roots of $Q(Y)$ in $\mathcal{P}$ and returns those that are actual solutions p.

Problem 1 (Small modular roots)**.**
Input:

- the base field $\mathbb{K}$,
- nonnegative integers d, n, k and t,
- $F \in \mathcal{P}[Y]$ monic of degree d with coefficients of degree $< n$,
- $M \in \mathcal{P}$ of degree n.

Output: all $p \in \mathcal{P}$ such that

(1.i) $\deg(\gcd(F(p), M)) \geqslant t$;

(1.ii) $\deg(p) \leqslant k$.

Guruswami-Sudan decoding. The best-known particular case of Problem 1 is certainly the Guruswami-Sudan algorithm for the list-decoding of Reed-Solomon codes [46, 24]. In this context, $k+1$ is the message length (or dimension of the code) and $n-t$ is an upper bound on the number of errors that can be corrected by the list-decoder; we have as input the code evaluation points $x_1, \ldots, x_n$ which are pairwise distinct in $\mathbb{K}$ as well as a received word $(y_1, \ldots, y_n) \in \mathbb{K}^n$. Then, the Guruswami-Sudan algorithm solves the above problem with the input $M = (X - x_1) \cdots (X - x_n)$ and $F = Y - L$ where L is the Lagrange polynomial such that $L(x_i) = y_i$ for all i. This particular input implies that $\deg(\gcd(F(p), M))$ counts the number of indices i for which $p(x_i) = y_i$: this is the *number of correct locations*, or the *agreement*, with respect to the given $p \in \mathcal{P}$.

There has been a lot of work on this specific case, mainly focusing on reducing the cost of the interpolation step (the root-finding step has been discussed for example in [42, 2, 41, 9]). In the original Guruswami-Sudan paper, the authors find the interpolation polynomial $Q(Y)$ by solving a linear system over $\mathbb{K}$ via Gaussian elimination; the fact that we can build such a system with more unknowns than equations follows from the assumption $t^2 > kn$. There are several branches of faster algorithms, relying on

- *structured linear algebra* [37, 42, 49, 14], which fully linearizes the problem into a linear system over $\mathbb{K}$, and takes advantage of the structure of the matrix of this system (block-Vandermonde, quasi-Toeplitz, ...) to solve it more efficiently;
- $\mathbb{K}[X]$*-lattice basis reduction* [1, 40, 30, 7, 9, 15, 34, 17] which, in a top-down fashion, first builds a known basis of the $\mathbb{K}[X]$-module of interpolation polynomials $Q(Y)$ and then combines the elements of this basis to obtain another basis of this module with smaller degrees;
- *polynomial approximation* [36, 31, 35, 26] which, in a bottom-up fashion, incrementally builds a basis of this $\mathbb{K}[X]$-module with minimal degrees, starting from the identity matrix.

The lattice-based method of the second item, that we sketch in Section 2, is analogous to the Coppersmith technique over the integers and to the CRT list-decoding [10]; yet it is unclear whether the authors of [1, 30, 7] were aware of this analogy. In this approach, unlike in the methods of the first and third items, one does not use the specificity of L and M. As such, it thus works in

general with $F = Y - L$ and M for any L and M of degree $< n$ and n, respectively; as we will see in Section 2, it is easily adapted to work for the general case with $d \geqslant 1$.

Other applications. Multivariate extensions of Problem 1 arise in and are tackled by list-decoding algorithms for Parvaresh-Vardy codes [38] and for folded Reed-Solomon codes [23], as well as an algorithm for optimally robust Private Information Retrieval [19]. In this case, the root-finding step becomes more involved; in [19] the overall process is heuristic since the interpolation step may not give enough independent equations to conclude, while in [38, 23] one has pre-existing equations and only needs to find one more.

Still, concerning the interpolation step we have the same three approaches, with fast algorithms using fast structured systems solvers [14], $\mathbb{K}[X]$-lattice basis reduction [13, 12, 16], and approximation algorithms [26, 27][1].

The interpolation step. In the rest of this report, we focus on the interpolation step, which can be formalized as follows.

Problem 2 (Interpolation step)**.**

Input:

- the base field $\mathbb{K}$,
- nonnegative integers d, n, k and t,
- $F \in \mathcal{P}[Y]$ monic of degree d with coefficients of degree $< n$,
- $M \in \mathcal{P}$ of degree n,
- positive integer μ.

Output: $Q \in \mathcal{P}[Y]$ nonzero such that

(2.i) Q belongs to the ideal

$$\mathcal{I} = \langle M, F \rangle^{\mu} = \langle M^{\mu}, M^{\mu-1}F, \ldots, MF^{\mu-1}, F^{\mu} \rangle$$

of $\mathcal{P}[Y]$,

(2.ii) $\deg_X(Q(X^k Y)) < \mu t$.

Writing $Q = Q_0 + Q_1 Y + Q_2 Y^2 + \cdots$, the second item requires that $\deg(Q_j) < \mu t - jk$ for all $j \leqslant \deg_Y(Q)$. In particular, the degree in Y of Q cannot grow arbitrarily large: it is bounded from above by $\mu t/k$.

The integer μ is called the *multiplicity parameter* and is introduced to give more power to the technique (allowing it to work in more cases). For example, the Sudan list-decoding algorithm [46] is with $\mu = 1$ and is able to correct up to about $n - \sqrt{2kn}$ errors, while the Guruswami-Sudan algorithm [24] improves this to $n - \sqrt{kn}$ by using $\mu > 1$.

[1] Although this is not presented here, it is reasonable to expect that one can derive approximation equations in the multivariate case following the same ideas as those in Section 3 in the univariate case.

Now suppose that we have computed a solution Q to this problem. Then, every solution p to Problem 1 satisfies

- $\deg(\gcd(F(p), M)) \geqslant t$ and thus $\deg(\gcd(Q(p), M^\mu)) \geqslant \mu t$ since Q belongs to $\mathcal{I}$,
- $\deg(p) \leqslant k$ and thus $\deg(Q(p)) \leqslant \deg_X(Q(X^k Y)) < \mu t$,

which together imply $Q(p) = 0$. Thus, having such a Q, it remains to compute its roots over $\mathcal{P}$ and to verify which of them are solutions to Problem 1.

The choice of the parameter $\mu \geqslant 1$ can be done by counting how many linear unknowns and linear equations we have in the $\mathbb{K}$-linear system corresponding to the given instance of Problem 2, and by taking μ sufficiently large so that there are more unknowns than equations. This is only feasible under some assumption on the parameters n, d, k, t.

Namely, assuming that $t^2 > knd$, one can choose

$$\mu \;=\; \left\lfloor \frac{k(nd-t)}{t^2 - knd} \right\rfloor + 1 \quad \text{and} \quad \ell = \left\lfloor \mu \frac{t}{k} \right\rfloor , \tag{1}$$

where ℓ is called the *list-size parameter*: then μ and ℓ are such that there is a solution Q to Problem 2 with $\deg_Y(Q) \leqslant \ell$. Indeed, this choice implies the inequality

$$\sum_{j \leqslant \ell} (\mu t - jk) \;>\; \sum_{i<m} (\mu - i) nd \, , \tag{2}$$

which precisely states that there are more unknowns than equations in the linearized problem. (We will also derive this inequality from the lattice-based approach in Section 2.) Furthermore, this shows that the number of solutions to Problem 1 is at most $\ell < 2tnd$.

In multivariate extensions of Problem 1, which we will not discuss in the rest of this report, we do not have a nice closed-form expression for a "good" choice of the parameters. This choice becomes more involved, even though the idea is the same as above: it is made so that some well-identified inequality is satisfied, so as to ensure the existence of a solution.

Lattice-based and approximation-based approaches. In the next section, we present a lattice-based solution to the interpolation step, which is the direct analogue to the Coppersmith technique over the integers; we also give an overview of the literature of lattice basis reduction over $\mathbb{K}[X]$. This leads to a first solution to Problem 2, which is deterministic and uses $\widetilde{\mathcal{O}}(\ell^\omega \mu n)$ operations in $\mathbb{K}$. Here, the notation $\widetilde{\mathcal{O}}(\cdot)$ means that we omit logarithmic factors in the cost bound, and ω stands for the exponent of matrix multiplication.

In Section 3, we show how the first condition on Q belonging to $\mathcal{I}$ can be efficiently rewritten as a system of linear modular equations over $\mathbb{K}[X]$. This leads to the fastest solutions we are aware of for Problem 2, using one of the following algorithms.

- [14] which is probabilistic and relies on the fast structured system solver in [11] to solve a quasi-Toeplitz system efficiently; it solves Problem 2 using an expected number of $\widetilde{\mathcal{O}}(\ell^{\omega-1}\mu^2 nd)$ operations in $\mathbb{K}$, returning only one solution Q which satisfies the degree constraints.
- [33] which is deterministic and builds upon previous work on Hermite-Padé approximation [4, 20, 45, 50, 26, 27] to solve this kind of systems; it uses $\widetilde{\mathcal{O}}(\ell^{\omega-1}\mu^2 nd)$ operations in $\mathbb{K}$ and returns a whole basis of solutions in the so-called *shifted Popov form*, which contains in particular at least one solution which satisfies the degree constraints.

In some contexts, this approach represents a significant improvement over the lattice-based approach; for example, when $\mu = 1$ and d as in Sudan decoding, the speed-up factor is ℓ. In general, the factor is $\ell/(\mu d)$, with $\ell \geqslant \mu d$ by choice of the parameters.

2 Interpolation step via basis reduction

In this section, we present an approach to solve Problem 2 via reduction of matrices over $\mathbb{K}[X]$.

Overview of the approach. The lattice-based approach to solve Problem 1 uses the parameter ℓ to bound the degree in Y of the sought solution Q. In coding theory, the integer ℓ is called the *list-size parameter* since it bounds the size of the list of solutions to Problem 1. Then, Q is written $Q = Q_0 + Q_1 Y + \cdots + Q_\ell Y^\ell$ for unknown polynomials $Q_j \in \mathcal{P}$, and it is identified with the vector $[Q_0, Q_1, \ldots, Q_\ell]$. This is a linearization of the problem with respect to the variable Y: we will work in $\mathcal{P}$-submodules of $\mathcal{P}^{\ell+1}$, ignoring the bivariate nature of the initial problem. In other words, the initial problem in $\mathcal{P}[Y]$ has been transformed into a problem of linear algebra over $\mathcal{P}$.

In what follows, we call *polynomial lattice* any free $\mathcal{P}$-module of finite rank. As a consequence, a polynomial lattice has a basis which can be represented by a matrix $\mathbf{B}$ with entries in $\mathcal{P}$, each row of $\mathbf{B}$ containing the representation of an element of the basis. Then, a polynomial matrix $\mathbf{P}$ is *reduced* if its rows have some type of minimal degree (precise definitions are given below). What we call *lattice basis reduction* is the problem of computing a reduced form of a given matrix $\mathbf{B}$, that is, computing a reduced matrix $\mathbf{P}$ whose rows generate the same lattice as $\mathbf{B}$.

Then, the strategy is to first build a known basis $\mathbf{B}$ of the lattice of all $[Q_j]_j$ such that Q belongs to the ideal $\mathcal{I}$: the vectors in this lattice correspond to polynomials Q that satisfy (2.i). Then, we compute a reduced form $\mathbf{P}$ of this basis. By definition, at least one of the rows of $\mathbf{P}$ has minimal degree, and thus corresponds to some Q that satisfies the weighted-degree condition (2.ii). We will see how to take the degree weight into account during the basis reduction by using the so-called *degree shifts*.

Building the lattice basis. This construction, detailed in [15], is analogous to a construction encountered in the Coppersmith technique over the integers, such as in [25]. The same construction was used in lattice-based algorithms for the Guruswami-Sudan decoding [1, 40, 30, 13, 7, 9].

The condition (2.i) on Q states that Q belongs to the ideal $\mathcal{I} = \langle F, M \rangle^\mu$: with the degree constraint $\deg_Y(Q) \leqslant \ell$, this is equivalent to having Q in the polynomial lattice generated by

$$\mathcal{B} \;=\; \{M^{\mu-i} Y^r F^i,\; i < \mu, r < d, id + r \leqslant \ell\} \;\cup\; \{Y^r F^m,\; r \leqslant \ell - md\}. \tag{3}$$

This basis is conveniently represented as a matrix $\mathbf{B} \in \mathcal{P}^{\ell+1 \times \ell+1}$ as shown in Figure 2 (page 6).

This matrix $\mathbf{B}$ has a triangular structure, and its diagonal elements are

$$\underbrace{M^\mu, \ldots, M^\mu}_{d}, \underbrace{M^{\mu-1}, \ldots, M^{\mu-1}}_{d}, \ldots, \underbrace{M, \ldots, M}_{d}, \underbrace{1, \ldots, 1}_{\ell+1-\mu d}.$$

To obtain the polynomials in $\mathcal{B}$, it is enough to compute $M^\mu, M^{\mu-1}F, \ldots, MF^{\mu-1}, F^\mu$; these can be computed using a total of $\widetilde{\mathcal{O}}(\mu^3 nd)$ operations in the field $\mathbb{K}$.

The entries of $\mathbf{B}$ can be computed modulo M^μ (except for the top-left entry which is M^μ itself) without loss of generality. Thus, in the dense representation of $\mathbf{B}$, the maximum degree of its entries is μn. This representation uses $\mathcal{O}(\ell \mu^2 nd)$ elements from $\mathbb{K}$, a bound that is reached generically.

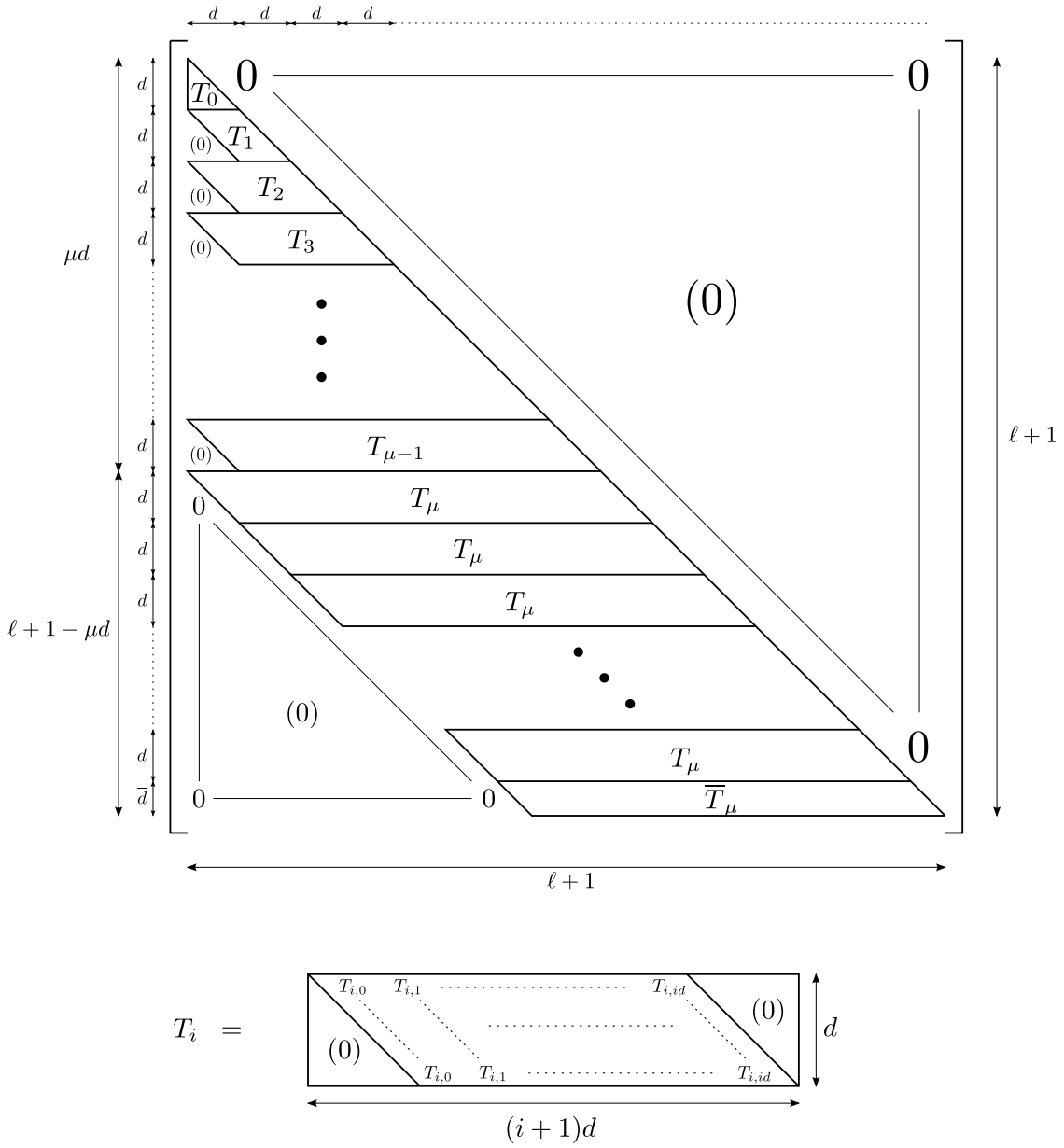

Figure 1: The basis $\mathbf{B}$ of the polynomial lattice, represented as a square $\ell+1 \times \ell+1$ matrix over $\mathbb{K}[X]$. Each T_i is a trapezoidal Toeplitz block with the shape given above, where $[T_{i,0}, \ldots, T_{i,id}]$ is the coefficient vector of the polynomial $F^i M^{\mu-i}$. The last block $\overline{T}_\mu$ is formed by the upper $\overline{d} = (\ell+1) \text{ rem } d$ rows of T_μ.

We remark that $\mathbf{B}$ is lower triangular, with a triangle of zeroes in the bottom left corner, and it has Toeplitz blocks: as of today, this structure is not used by the fastest algorithms. Hence, as we will see below, for most reduction algorithms, $\mathbf{B}$ is essentially considered as an $\ell+1 \times \ell+1$ matrix of degree μn, whose size is $\mathcal{O}(\ell^2 \mu n)$. This bound is beyond our target cost $\widetilde{\mathcal{O}}(\ell^{\omega-1}\mu^2 n d)$.

Reduced bases. This notion is defined for example in [28, Section 6.3.2], and has received a lot of attention [39, 28, 8, 48, 32, 20, 43, 22]. It can be thought of as similar to basis reduction of Euclidean lattices, but one should keep in mind that, in short, *everything is easier over the polynomials*. In particular, we are able to compute shortest vectors and minimal bases in polynomial time (even in quasi-optimal time if it turns out that $\omega = 2$).

In the problem of *reduction*, we have a basis of a lattice represented as a matrix $\mathbf{A}$, and we want to compute another basis which has some type of minimality. Here, the measure is the degree of the rows of the matrix: for a row $\mathbf{p} = [p_1, \ldots, p_m] \in \mathcal{P}^{1\times m}$, its degree is the maximum of the degrees of its entries. Here, we focus on matrices $\mathbf{A}$ that are square and nonsingular.

To take into account the degree constraints in our problem, we will need the notion of *shifted degree*, where some degree weights are put on the columns: for a *shift* $\mathbf{s} = (s_1, \ldots, s_m) \in \mathbb{Z}^m$, the $\mathbf{s}$-*degree* of a row $\mathbf{p} = [p_1, \ldots, p_m] \in \mathcal{P}^{1\times m}$ is $\max_{1\leqslant j\leqslant m}(\deg(p_j) + s_j)$. Then, we also define the $\mathbf{s}$-*row degree* of a matrix $\mathbf{A} \in \mathcal{P}^{m\times m}$ by $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{A}) = (d_1, \ldots, d_m)$ with d_i the $\mathbf{s}$-degree of the i-th row of $\mathbf{A}$.

In this context, any two bases $\mathbf{A}$ and $\mathbf{P}$ of a given lattice are left-unimodularly equivalent, which means that there exists a matrix $\mathbf{U}$ with determinant in $\mathbb{K} - \{0\}$ and such that $\mathbf{A} = \mathbf{UP}$. For a given shift $\mathbf{s}$, a nonsingular matrix $\mathbf{A} \in \mathcal{P}^{m\times m}$ is said to be $\mathbf{s}$-*reduced* [28, 6] if its shifted row degree $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{A})$, sorted in nondecreasing order, is lexicographically minimal among the $\mathbf{s}$-row degrees of all matrices left-unimodularly equivalent to $\mathbf{A}$ (that is, which represent bases of the same lattice).

Then, an $\mathbf{s}$-*reduced form of* $\mathbf{A}$ is an $\mathbf{s}$-reduced matrix $\mathbf{P}$ which generates the same lattice as $\mathbf{A}$, that is, $\mathcal{P}^{1\times m}\mathbf{P} = \mathcal{P}^{1\times m}\mathbf{A}$. In particular, $\mathbf{P}$ is left-unimodularly equivalent to $\mathbf{A}$, and at least one of its rows is a vector in the lattice with minimal $\mathbf{s}$-degree.

Let δ denote the maximum degree in the matrix $\mathbf{A}$. Then, algorithms for computing reduced form of $\mathbf{A}$ include:

- [32] which iteratively cancels the high-degree entries of $\mathbf{A}$ until it is reduced, using $\mathcal{O}(m^3\delta^2)$ operations in $\mathbb{K}$ in the worst case;
- [1, 2] which proposes the same algorithm and a divide-and-conquer improvement, leading to the cost bound $\widetilde{\mathcal{O}}(m^4\delta)$;
- [20] which uses the so-called high-order lifting and Hermite-Padé approximation to obtain $\widetilde{\mathcal{O}}(m^\omega\delta)$ (probabilistic);
- [22] which gives a deterministic variant of the latter with the same cost bound;
- [33] which is discussed below.

All these algorithms compute reduced bases for the uniform shift $\mathbf{s} = (0, \ldots, 0)$. In our context, we want to use them with the shift $(0, k, \ldots, \ell k)$: this can be done by multiplying the columns of the matrix by the corresponding powers of X. Besides, we are only interested in a single sufficiently small vector in the lattice, but as of today it is not known how to compute such a vector more efficiently than by computing a whole reduced basis.

Back to Coppersmith technique. Here, our lattice matrix $\mathbf{B}$ has dimension $\ell + 1$ and the largest degree of its entries is $\mu n = \deg(M^\mu)$. Before applying a reduction algorithm, we multiply the columns of $\mathbf{B}$ by the powers $1, X^k, \ldots, X^{\ell k}$ so as to take the degree constraints into account; since $\ell k \leqslant \mu t \leqslant \mu n$, the degree of the new input matrix $\widetilde{B}$ remains in $\mathcal{O}(\mu n)$.

Then, using one of the basis reduction algorithms in [20, 22], this matrix $\widetilde{B}$ is reduced using $\widetilde{\mathcal{O}}(\ell^\omega \mu n)$ operations in $\mathbb{K}$. Among the rows of the obtained matrix, it is guaranteed that we will find a row $[Q_j X^{jk}]_j$ which has minimal degree: this one corresponds to $Q = Q_0 + Q_1 Y + \cdots + Q_\ell Y^\ell$ that solves Problem 2 (unless no solution exists, in which case either $t^2 \leqslant knd$ or the parameters μ and ℓ have not been chosen properly).

Besides, it is known that such a row with minimal degree satisfies

$$\max_{0 \leqslant j \leqslant \ell} \deg(Q_j X^{jk}) \leqslant \frac{\deg(\det(\widetilde{\mathbf{B}}))}{\dim(\widetilde{\mathbf{B}})},$$

that is,

$$\deg(Q(X^k Y)) \leqslant \frac{\frac{1}{2}\ell(\ell+1)k + \frac{1}{2}\mu(\mu+1)nd}{\ell+1}.$$

Thus, to ensure the existence of a solution, we require that

$$\mu t > \frac{\frac{1}{2}\ell(\ell+1)k + \frac{1}{2}\mu(\mu+1)nd}{\ell+1},$$

which is equivalent to the inequality in (2).

A possible improvement. For an arbitrary shift $\mathbf{s}$, the algorithm in [33] computes the $\mathbf{s}$-*Popov form of* $\mathbf{A}$ (which is a canonical form among the $\mathbf{s}$-reduced forms of $\mathbf{A}$) with an expected number of $\widetilde{\mathcal{O}}(m^\omega \lceil \sigma \rceil)$ operations, where $\sigma > 0$ is a parameter that is bounded from above by both the average of the column degrees and the average of the row degrees of $\mathbf{A}$.

We may be able to take advantage of this result if we rather build the lattice basis matrix in the shape $\begin{bmatrix} \mathbf{T} & \mathbf{0} \\ \mathbf{A} & \mathbf{I} \end{bmatrix}$ with $\mathbf{T}$ the principal $\mu d \times \mu d$ submatrix of $\mathbf{B}$, and $\mathbf{I}$ the identity matrix. Since the first μd columns of this matrix have degree at most μn and its remaining columns are constant, the average of its column degrees is $\mu^2 nd/(\ell+1)$, and the algorithm in [33] finds a shifted reduced form of such a matrix in expected $\widetilde{\mathcal{O}}(\ell^{\omega-1}\mu^2 nd)$ operations.

This is our target cost announced in Section 1.[2] With this method, here would be the sketch of the global algorithm:

1. compute the basis of the lattice of solutions Q with the shape $\begin{bmatrix} \mathbf{T} & \mathbf{0} \\ \mathbf{A} & \mathbf{I} \end{bmatrix}$ as above,
2. compute the Smith form of this matrix (probabilistic, [44]),
3. compute partial information on a right-multiplier corresponding to this Smith form (probabilistic, [21]),
4. use these to set up a system of linear modular equations
5. solve this system using the algorithm in [33] (see Section 3.3 for some more details).

In the next section, we propose to exploit the particularities of our problem here to avoid steps 2 and 3, which are probabilistic and non-trivial. Instead of building a basis of the lattice and trying to reduce it, we directly follow the dual approach: we build a system of modular equations that describe the lattice. It remains to find a small degree solution to this system, which is done deterministically in [33]. This yields the cost bound $\widetilde{\mathcal{O}}(\ell^{\omega-1}\mu^2 nd)$ for Problem 2.

[2] Still, one has to verify that the construction of this other basis can also be done within a reasonable cost bound, such as $\widetilde{\mathcal{O}}(\ell\mu^2 nd)$; I have not checked this yet.

3 Interpolation step via approximation

In this section, we present a reduction of Problem 2 to a problem of polynomial approximation, or more precisely to a system of linear modular equations over $\mathcal{P}$. Then, we give an overview of fast algorithms for solving such systems, in Section 3.3.

3.1 Warm-up: the specific case $d = 1$

Here, we first present this reduction to a problem of approximation in a simple case, but nonetheless important: when $d = \deg(F) = 1$, which includes the Guruswami-Sudan algorithm. Although we follow a different presentation to ease the transition to the general case in the next subsection, the material here is very close to results proved and used in [42, 49, 14].

In what follows, we write $d = Y - L$ for some $L \in \mathcal{P}$ of degree less than n. Assuming that $\mu = 1$, the reduction is as follows: we have $Q \in \mathcal{I} = \langle Y - L, M\rangle$ if and only if $Q(L) = 0 \bmod M$, which is rewritten as the single modular linear equation $Q_0 + Q_1 L + \cdots + Q_\ell L^\ell = 0 \bmod M$ (this was used in [42] to speed-up the Sudan algorithm). Now, when $\mu \geqslant 1$, this was generalized in [49, Proposition 3] using the notion of *Hasse derivative*: $Q \in \mathcal{I} = \langle Y - L, M\rangle^\mu$ if and only if the derivatives of $Q(Y)$ at order $0, \ldots, \mu - 1$ vanish at L modulo $M^\mu, \ldots, M$, respectively.

Here, we show this property, reformulated as a change of basis which can be computed via some Taylor expansions. We are looking for $Q \in \mathcal{P}[Y]$ which belongs to the ideal $\mathcal{I}$ of $\mathcal{P}[Y]$ generated by

$$M^\mu, M^{\mu-1}(Y - L), \ldots, M(Y - L)^{\mu-1}, (Y - L)^\mu.$$

This is equivalent to

$$Q \;=\; \widehat{Q}_0 M^\mu + \widehat{Q}_1 M^{\mu-1}(Y - L) + \cdots + \widehat{Q}_{\mu-1} M(Y - L)^{\mu-1} + \widehat{Q}(Y)(Y - L)^\mu$$

for some $\widehat{Q}_0, \ldots, \widehat{Q}_{\mu-1} \in \mathcal{P}$ and some $\widehat{Q} \in \mathcal{P}[Y]$. Adding the constraint $\deg(Q) \leqslant \ell$, this is equivalent to Q belonging to the lattice $\mathcal{M} = \mathcal{I} \cap \mathcal{P}[Y]_{\leqslant \ell}$ of dimension $\ell + 1$ with basis

$$\mathcal{B} \;=\; \{M^\mu, M^{\mu-1}(Y - L), \ldots, M(Y - L)^{\mu-1}, (Y - L)^\mu, Y(Y - L)^\mu, \ldots, Y^{\ell-\mu}(Y - L)^\mu\}. \tag{4}$$

In order to transform this property of belonging to the lattice $\mathcal{M}$ into a property of satisfying modular equations, one first rewrites $Q = Q_0 + \cdots + Q_\ell Y^\ell$ in the basis

$$\mathcal{B}' \;=\; \{1, Y - L, \ldots, (Y - L)^{\mu-1}, (Y - L)^\mu, Y(Y - L)^\mu, \ldots, Y^{\ell-\mu}(Y - L)^\mu\} \tag{5}$$

of $\mathcal{P}[Y]_{\leqslant \ell}$, that is,

$$Q = \widehat{Q}_0 + \widehat{Q}_1(Y - L) + \cdots + \widehat{Q}_{\mu-1}(Y - L)^{\mu-1} + \widehat{Q}(Y)(Y - L)^\mu$$

for some $\widehat{Q}_0, \ldots, \widehat{Q}_{\mu-1} \in \mathcal{P}$ and some $\widehat{Q} \in \mathcal{P}[Y]$ of degree $\leqslant \ell - \mu$. We remark that the coefficients $\widehat{Q}_i$ of Q written in $\mathcal{B}$ are unique: ensuring that Q is in the lattice $\mathcal{M}$ is thus equivalent to ensuring the modular equations

$$M^{\mu-i} \text{ divides } \widehat{Q}_i \quad \text{for } i < \mu. \tag{6}$$

This is the linear system of modular equations over $\mathcal{P}$ that we are going to focus on. It remains to show that we can compute the $\widehat{Q}_i$'s efficiently as $\mathcal{P}$-linear combinations of the unknown Q_i's.

In this special context where $F = Y - L$, the computation of the $\widehat{Q}_i$'s is particularly straightforward since explicit formula are known for the change of basis from $\{1, Y, \ldots, Y^\ell\}$ to $\mathcal{B}'$, namely thanks to the Taylor formula

$$Q \;=\; \sum_{j \leqslant \ell} Q_j (Y - L + L)^j \;=\; \sum_{i \leqslant \ell} \left(\sum_{j \geqslant i} \binom{j}{i} Q_j L^{j-i} \right) (Y - L)^i.$$

This gives an explicit formula for the coefficients $[\widehat{Q}_i]_i$ of Q in the basis $\mathcal{B}'$, as linear combinations with explicit coefficients in $\mathcal{P}$ of the coefficients $[Q_j]_j$ in the basis $\mathcal{B}$; namely,

$$\widehat{Q}_i = \sum_{j \leqslant \ell} f_{ij} Q_j \quad \text{with} \quad f_{ij} = \binom{j}{i} L^{j-i}$$

We have in particular $f_{ij} = 0$ for $i > j$ and $f_{jj} = 1$. From the equations (6), we thus obtain the system of linear modular equations

$$\sum_{j \leqslant \ell} f_{ij} Q_j \;=\; 0 \bmod M^{\mu - i} \quad \text{for} \;\; i < \mu.$$

In particular, due to the nature of these equations, the coefficients f_{ij}'s can be computed modulo M^μ without loss of generality.

This set of equations, along with the degree constraints on the coefficients Q_j given by the condition (2.ii), is precisely the *simultaneous polynomial approximations* problem tackled in [14] or the *minimal solution basis* problem studied in [33]. We give an overview of this in Section 3.3.

3.2 Reduction in the general case $d \geqslant 1$

In the specific case $F = Y - L$, we can summarize the reduction presented above as follows: first, decompose Q in the basis of $\mathcal{P}[Y]_{\leqslant \ell}$ formed by the polynomials $\{1, F, \ldots, F^\mu, YF, \ldots, Y^{\ell-\mu}F^\mu\}$, and second, express the fact that the coefficients in this decomposition must vanish modulo some powers of M, so that Q is actually in the module generated by $\{M^\mu, M^{\mu-1}F, \ldots, MF^{\mu-1}, F^\mu\}$. In the rest of this section, we will extend this idea to the general case $d \geqslant 1$.

The change of basis is now from the basis $\{1, Y, \ldots, Y^\ell\}$ of $\mathcal{P}[Y]_{\leqslant \ell}$ to the basis

$$\mathcal{B}' \;=\; \{Y^r F^i,\; r < d, i < \mu\} \;\cup\; \{Y^r F^\mu, r \leqslant \ell - \mu\}.$$

In this more general context, we will not be able to give an explicit formula for the coefficients $[\widehat{Q}_i]_i$ of Q in $\mathcal{B}'$. Still, expressing them as linear combinations of $[Q_j]_j$, we will show that the coefficients in these combinations can be computed efficiently. Let us make this more precise by properly defining the problem we are faced with: Problem 3 below.

Before working on solving Problem 3 efficiently, we explain our interest in this problem: its solution helps us to rewrite the fact that Q belongs to the ideal $\mathcal{I}$ into a set of divisibility properties involving the coefficients $\widehat{Q}_i$ for $i < \mu$. These divisibility properties give us the approximation equations we are looking for.

Lemma 1. *Let $Q = Q_0 + Q_1 Y + \cdots + Q_\ell Y^\ell$ in $\mathcal{P}/(M^\mu)[Y]$. Then, the polynomials $\widehat{Q}_0, \widehat{Q}_1, \ldots, \widehat{Q}_{\mu-1}$ defined for all $i < \mu$ by $\widehat{Q}_i = \sum_{r<d} \sum_{j \leqslant \ell} f_{ij}^{(r)} Q_j Y^r$ satisfy*

$$Q = \widehat{Q}_0 + \widehat{Q}_1 F + \cdots + \widehat{Q}_{\mu-1} F^{\mu-1} + \widehat{Q} F^\mu$$

Problem 3 (Change of basis)**.**
Input:

- $M \in \mathcal{P}$ of degree n,
- $F \in \mathcal{P}[Y]$ monic of degree d with coefficients in $\mathcal{P}$ of degree $< n$,
- ℓ, μ positive integers,

Output: polynomials $\{f_{ij}^{(r)} \in \mathcal{P}/(M^\mu),\ i < \mu, j \leqslant \ell, r < d\}$ such that for each $j \leqslant \ell$, the vector $[f_{ij}^{(r)}]_{i,r}$ gives the first μd coefficients in the decomposition of Y^j in the basis $\mathcal{B}'$:

$$Y^j \ = \sum_{i<\mu} \sum_{r<d} f_{ij}^{(r)} Y^r F^i \ + \ \hat{f}_j F^\mu \ \text{ for some } \hat{f}_j \in \mathcal{P}/(M^\mu)[Y]\,. \qquad (7)$$

in $\mathcal{P}/(M^\mu)[Y]$*, for some* $\widehat{Q} \in \mathcal{P}/(M^\mu)[Y]$.

Besides, Q *is in the ideal* $\mathcal{I}$ *of* $\mathcal{P}[Y]$ *generated by* $\{M^\mu, M^{\mu-1}F, \ldots, MF^{\mu-1}, F^\mu\}$ *if and only if for every* $i < \mu$ *and* $r < d$*,* $M^{\mu-i}$ *divides* $\widehat{Q}_i^{(r)}$*. Equivalently, writing* $\widehat{Q}_i^{(r)} = \sum_{j\leqslant\ell} f_{ij}^{(r)} Q_j$ *for all* $i < \mu$ *and* $r < d$*, then* Q *satisfies the system of linear modular equations*

$$\sum_{j\leqslant\ell} f_{ij}^{(r)} Q_j = 0 \bmod M^{\mu-i} \quad \textit{for all } i < \mu \textit{ and } r < d. \qquad (8)$$

We note that here we are working modulo M^μ. This is sufficient for our purpose, since all approximation equations we are going to focus on are modulo powers of M which do not exceed μ. Besides, this helps to keep the size of the elements of $\mathcal{P}$ we are manipulating reasonable. In the rest of this section, we show how to solve Problem 3 efficiently, namely using $\widetilde{\mathcal{O}}(\ell\mu^2 nd)$ operations in $\mathbb{K}$, which is quasi-linear in the number of field elements used to represent the output $\{f_{ij}^{(r)},\ i < \mu, j \leqslant \ell, r < d\}$.

Theorem 2 (Change of basis)**.** *Problem 3 can be solved using* $\widetilde{\mathcal{O}}(\ell\mu^2 nd)$ *operations in* $\mathbb{K}$.

When applying this change of basis to solve Problem 2, the coefficients Q_j's of Q are unknowns: this is why we focus on computing the coefficients $f_{ij}^{(r)}$'s of the linear combinations $\widehat{Q}_i$. (In the case where the Q_j's are known, one may directly compute the $\widehat{Q}_i$'s as well as $\widehat{Q}$ using $\widetilde{\mathcal{O}}(\ell\mu n)$ operations in $\mathbb{K}$.)

Furthermore, we remark that we are not interested in computing the polynomial $\widehat{Q}$, although Algorithm 2 below could be easily adapted to include the computation of $\widehat{Q}$. The reason is that the mere representation of all coefficients $f_{ij}^{(r)}$, if also computing $\widehat{Q}$, uses $\widetilde{\mathcal{O}}(\ell^2\mu n)$ coefficients from $\mathbb{K}$, which is beyond our target cost bound $\widetilde{\mathcal{O}}(\ell^{\omega-1}\mu^2 nd)$ for solving Problem 2.

We will use the definition of the sought coefficients $f_{ij}^{(r)}$ to compute them incrementally for $j \in \{0, \ldots, \ell\}$, using the relations between the decomposition of Y^j and $Y^{j+1} = Y \cdot Y^j$. Thus, we first focus on how to use the knowledge of the decomposition of some $P \in \mathcal{P}/(M^\mu)[Y]_{<\mu d}$ in $\mathcal{B}'$ to compute the decomposition of YP in $\mathcal{B}'$.

Lemma 3. *Write $F = F_0 + \cdots + F_{d-1}Y^{d-1} + Y^d$, with the coefficients $F_0, F_1, \ldots, F_{d-1}$ of degree less than n. Let $P \in \mathcal{P}/(M^\mu)[Y]$ of degree less than μd with coefficients $P_i^{(r)} \in \mathcal{P}/(M^\mu)$ in $\mathcal{B}'$, that is,*

$$P \;=\; \sum_{i<\mu} \sum_{r<d} P_i^{(r)} Y^r F^i.$$

Then, Algorithm 1 computes coefficients $\{g_i^{(r)},\ i < \mu, r < d\}$ in $\mathcal{P}/(M^\mu)$ such that

$$YP \;=\; \sum_{i<\mu} \sum_{r<d} g_i^{(r)} Y^r F^i \;+\; \hat{g}F^\mu \quad \text{for some } \hat{g} \in \mathcal{P}/(M^\mu) \tag{9}$$

using $\widetilde{\mathcal{O}}(\mu^2 nd)$ operations in $\mathbb{K}$.

Algorithm 1 (Change of basis: from P to YP)**.**
Input:

- $M \in \mathcal{P}$ of degree n,
- $F = F_0 + \cdots + F_{d-1}Y^{d-1} + Y^d$ in $\mathcal{P}[Y]$ with coefficients of degree less than n,
- positive integer μ,
- $P = \sum_{i<\mu} \sum_{r<d} P_i^{(r)} Y^r F^i$ in $\mathcal{P}/(M^\mu)[Y]$.

Output: polynomials $\{g_i^{(r)},\ i < \mu, r < d\}$ in $\mathcal{P}/(M^\mu)$ such that (9).

1. $g_0^{(0)} \leftarrow -F_0 P_0^{(d-1)}$

2. $g_i^{(0)} \leftarrow P_{i-1}^{(d-1)} - F_0 P_i^{(d-1)}$ for $0 < i < \mu$

3. $g_i^{(r)} \leftarrow P_i^{(r-1)} - F_r P_i^{(d-1)}$ for $0 < r < d, 0 < i < \mu$

4. Return $\{g_i^{(r)},\ i < \mu, r < d\}$

Proof of Lemma 3. Using $Y^d = F - F_0 - F_1 Y - \cdots - F_{d-1}Y^{d-1}$, we have

$$\begin{aligned}
YP \;&=\; \sum_{i<\mu} \left(P_i^{(d-1)} Y^d + \sum_{0<r<d} P_i^{(r-1)} Y^r \right) F^i \\
&=\; \sum_{i<\mu} \left(P_i^{(d-1)} (F - F_0 - \cdots - F_{d-1}Y^{d-1}) + \sum_{0<r<d} P_i^{(r-1)} Y^r \right) F^i \\
&=\; \sum_{i<\mu} \left(-F_0 P_i^{(d-1)} + \sum_{0<r<d} (P_i^{(r-1)} - F_r P_i^{(d-1)}) Y^r \right) F^i \;+\; \sum_{0<i<\mu} P_{i-1}^{(d-1)} F^i \;+\; P_{\mu-1}^{(d-1)} F^\mu,
\end{aligned}$$

which gives the correctness of Algorithm 1. Now, for each $i < \mu$ and $r < d$ the computation of $g_i^{(r)}$ involves one subtraction and one multiplication in $\mathcal{P}/(M^\mu)$. Altogether Step 1 to Step 3 do at most

μd subtractions and μd multiplications in $\mathcal{P}/(M^\mu)$, which can be done using $\widetilde{\mathcal{O}}(\mu^2 nd)$ operations in $\mathbb{K}$. □

We have now all the tools to give the main algorithm and prove Theorem 2.

Proof of Theorem 2. Algorithm 2 below computes the polynomials $\{f_{ij}^{(r)},\ j \leqslant \ell, i < \mu, r < d\}$, starting from the $F_{0i}^{(r)}$'s that are known and using ℓ calls to Algorithm 1. Its cost is thus bounded by $\widetilde{\mathcal{O}}(\ell\mu^2 nd)$, and its correctness follows from that of Algorithm 1. □

Algorithm 2 (Change of basis)**.**
Input:

- $M \in \mathcal{P}$ of degree n,
- $F \in \mathcal{P}[Y]$ monic of degree d with coefficients of degree less than n,
- positive integers μ, ℓ.

Output: polynomials $\{f_{ij}^{(r)},\ j \leqslant \ell, i < \mu, r < d\}$ in $\mathcal{P}/(M^\mu)$ such that (7) for $j \leqslant \ell$.

1. Set $F_{00}^{(0)} \leftarrow 1$ and $F_{0i}^{(r)} \leftarrow 0$ for $i > 0$ or $r > 0$

2. For j from 1 to ℓ do

a. Define $P = \sum_{i<\mu} \sum_{r<d} F_{j-1,i}^{(r)} Y^r$

b. $\{f_{ij}^{(r)},\ i < \mu, r < d\} \leftarrow$ Algorithm 1 on input M, F, μ, P

3. Return $\{f_{ij}^{(r)},\ j \leqslant \ell, i < \mu, r < d\}$

Now, having Lemma 1 and Theorem 2 it is easy to reduce the interpolation step of the Coppersmith technique (Problem 2) to a system of linear modular equations.

3.3 Solving systems of linear modular equations

Here, we give an overview of the existing fast algorithms for solving systems of linear modular equations. This includes the well-known problem of Hermite-Padé approximation.

The problem. Hereafter, $\mathcal{P}_{\neq 0}$ denotes the set of nonzero univariate polynomials in $\mathcal{P}$. We fix some moduli $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathcal{P}_{\neq 0}^n$, and for two matrices $\mathbf{A}, \mathbf{B} \in \mathcal{P}^{m\times n}$ we write $\mathbf{A} = \mathbf{B} \bmod \mathfrak{M}$ if there exists a quotient matrix $\mathbf{Q} \in \mathcal{P}^{m\times n}$ such that $\mathbf{A} = \mathbf{B} + \mathbf{Q}\mathrm{Diag}\mathfrak{M}$. Then, given a matrix $\mathbf{F} \in \mathcal{P}^{m\times n}$ which specifies the equations of our system, we call *solution for* $(\mathfrak{M}, \mathbf{F})$ any row vector $\mathbf{p} \in \mathcal{P}^{1\times m}$ such that $\mathbf{pF} = 0 \bmod \mathfrak{M}$. Writing $\mathbf{p} = [p_1 \cdots p_m]$ and $\mathbf{F} = [f_{ij}]$, this matrix equation corresponds to the system of linear modular equations

$$p_1 f_{1j} + \cdots + p_m f_{mj} = 0 \bmod \mathfrak{m}_j \quad \text{for all } 1 \leqslant j \leqslant n.$$

The set of all such solutions $\mathbf{p}$ for $(\mathfrak{M}, \mathbf{F})$ is a $\mathbb{K}[X]$-submodule of $\mathcal{P}^{1\times m}$, which obviously contains $\operatorname{lcm}(\mathfrak{m}_1, \ldots, \mathfrak{m}_n)\mathcal{P}^{1\times m}$, and is thus free of rank m [29, p. 146]. Then, any basis of this module can be represented as the rows of a square matrix $\mathbf{P} \in \mathcal{P}^{m\times m}$, called a *solution basis for* $(\mathfrak{M}, \mathbf{F})$.

Furthermore, in our case we are interested in such a $\mathbf{P}$ which is $\mathbf{s}$-reduced, so that it contains a vector $\mathbf{p}$ of minimal $\mathbf{s}$-degree. In this case, $\mathbf{P}$ is said to be an $\mathbf{s}$-*minimal* solution basis for $(\mathfrak{M}, \mathbf{F})$.

Problem 4 (Minimal solution basis)**.**
Input:

- the base field $\mathbb{K}$,
- dimensions m and n,
- polynomials $\mathfrak{M} = (\mathfrak{m}_1, \ldots, \mathfrak{m}_n) \in \mathcal{P}^n_{\neq 0}$,
- a matrix $\mathbf{F} \in \mathcal{P}^{m\times n}$ with $\deg(\mathbf{F}_{*,j}) < \deg(\mathfrak{m}_j)$,
- a shift $\mathbf{s} \in \mathbb{Z}^m$.

Output: an $\mathbf{s}$-minimal solution basis for $(\mathfrak{M}, \mathbf{F})$.

Hermite-Padé approximation. A well-known particular case of this problem is Hermite-Padé approximation. In this context, the moduli are powers of X, namely, $\mathfrak{M} = (X^{\sigma_1}, \ldots, X^{\sigma_n})$. In what follows, we write $\sigma = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$.

When $\sigma_1 = \cdots = \sigma_n = \sigma/n$, fast algorithms include

- [47] which gives an iterative solution in $\mathcal{O}(m^2\sigma^2)$, using a mix between Gaussian elimination and multiplications by X;
- [4] which proposes a divide-and-conquer version in $\widetilde{\mathcal{O}}(m^\omega\sigma)$ operations relying on polynomial matrix multiplication;
- [20] which better exploits Gaussian elimination over $\mathbb{K}$ at the leaves of the recursion and runs in $\widetilde{\mathcal{O}}(m^\omega\sigma/n)$ operations;
- [45, 50] improved this for small n, using transformations which reduce to the case $n \approx m$ and degree of the moduli about $\sigma n/m$, leading to the cost bound $\widetilde{\mathcal{O}}(m^{\omega-1}\sigma)$ independently of n; this however requires assumptions on $\mathbf{s}$.

All these algorithms are deterministic.

In [27, Theorem 1.4], we removed the assumption on $\mathbf{s}$ and dealt with any powers $(X^{\sigma_1}, \ldots, X^{\sigma_n})$, returning a basis in $\mathbf{s}$-Popov form in time $\widetilde{\mathcal{O}}(m^{\omega-1}\sigma)$ (as said above, this is a canonical form for the $\mathbf{s}$-reduced bases of the module of solutions).

M-Padé approximation. More generally, when the moduli in $\mathfrak{M}$ are products of known linear factors, Problem 4 is known as M-Padé approximation [3, 47, 5]. This is the case for example in the list-decoding of Reed-Solomon codes with the Guruswami-Sudan algorithm: as shown in Section 3.1,

the equations are modulo powers of M which itself is known as $M = (X - x_1) \cdots (X - x_n)$, where $x_1, \ldots, x_n$ are the evaluation points of the code.

For this problem, fast algorithms include

- [3, 47] which give an iterative solution in $\mathcal{O}(m^2\sigma^2)$, using a mix between Gaussian elimination and multiplications by the linear factors $X - x_i$ of the moduli;
- [5] which give a similar solution that is fraction-free and returns a basis in $\mathbf{s}$-Popov form, at the price of a cost increase;
- [14] which is probabilistic and computes a single solution with small $\mathbf{s}$-degree in $\widetilde{\mathcal{O}}(m^{\omega-1}\sigma)$, relying on the fast structured linear system solver in [11];
- [26, 27] which is deterministic and returns a basis of solutions in $\mathbf{s}$-Popov form, for an arbitrary $\mathbf{s}$, using $\widetilde{\mathcal{O}}(m^{\omega-1}\sigma)$ operations.

Arbitrary moduli. For the Coppersmith technique, we may encounter situations where the moduli do not split into linear factors, or where these linear factors are not known. Problem 4 with arbitrary moduli has been studied in [33] (with the purpose of giving a fast algorithm for $\mathbf{s}$-reduction for an arbitrary shift $\mathbf{s}$), leading to the following result.

Theorem 4. *Assuming $n \in \mathcal{O}(m)$, there is a deterministic algorithm which solves Problem 4 using $\widetilde{\mathcal{O}}(m^{\omega-1}\sigma)$ operations in $\mathbb{K}$, with $\sigma = \deg(\mathfrak{m}_1) + \cdots + \deg(\mathfrak{m}_n)$, and returns the $\mathbf{s}$-Popov solution basis for $(\mathfrak{M}, \mathbf{F})$.*

It should be noted that a similar cost bound (with less logarithmic factors) for computing a single solution of small $\mathbf{s}$-degree was previously obtained in [14, Theorem 2], with a probabilistic algorithm based on fast structured linear algebra over $\mathbb{K}$.

It follows from this theorem and from the system of equations derived in the previous subsection that the interpolation step can be solved with the announced cost bound.

Corollary 5. *The interpolation step of the Coppersmith technique can be solved using $\widetilde{\mathcal{O}}(\ell^{\omega-1}\mu^2 nd)$ operations.*

References

[1] M. Alekhnovich. Linear Diophantine equations over polynomials and soft decoding of Reed-Solomon codes. In *FOCS'02*, pages 439–448. IEEE, 2002.

[2] M. Alekhnovich. Linear Diophantine equations over polynomials and soft decoding of Reed-Solomon codes. *IEEE Trans. Inf. Theory*, 51(7):2257–2265, July 2005.

[3] B. Beckermann. A reliable method for computing M-Padé approximants on arbitrary staircases. *J. Comput. Appl. Math.*, 40(1):19–42, 1992.

[4] B. Beckermann and G. Labahn. A uniform approach for the fast computation of matrix-type Padé approximants. *SIAM J. Matrix Anal. Appl.*, 15(3):804–823, July 1994.

[5] B. Beckermann and G. Labahn. Fraction-free computation of matrix rational interpolants and matrix gcds. *SIAM J. Matrix Anal. Appl.*, 22(1):114–144, 2000.

[6] B. Beckermann, G. Labahn, and G. Villard. Normal forms for general polynomial matrices. *J. Symbolic Comput.*, 41(6):708–737, 2006.

[7] P. Beelen and K. Brander. Key equations for list decoding of Reed-Solomon codes and how to solve them. *J. Symbolic Comput.*, 45(7):773–786, 2010.

[8] Th.G.J. Beelen, G.J. van den Hurk, and C. Praagman. A new method for computing a column reduced polynomial matrix. *Systems and Control Letters*, 10(4):217 – 224, 1988.

[9] D. J. Bernstein. Simplified high-speed high-distance list decoding for alternant codes. In *PQCrypto'11*, volume 7071 of *LNCS*, pages 200–216. Springer, 2011.

[10] Dan Boneh. Finding smooth integers in short intervals using CRT decoding. *J. Comput. Syst. Sci.*, 64(4):768–784, June 2002.

[11] A. Bostan, C.-P. Jeannerod, and É. Schost. Solving structured linear systems with large displacement rank. *Theor. Comput. Sci.*, 407(1-3):155–181, 2008.

[12] K. Brander. *Interpolation and List Decoding of Algebraic Codes*. PhD thesis, Technical University of Denmark, 2010.

[13] P. Busse. *Multivariate List Decoding of Evaluation Codes with a Gröbner Basis Perspective*. PhD thesis, University of Kentucky, 2008.

[14] M. Chowdhury, C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard. Faster algorithms for multivariate interpolation with multiplicities and simultaneous polynomial approximations. *IEEE Trans. Inf. Theory*, 61(5):2370–2387, 2015.

[15] H. Cohn and N. Heninger. Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding. In *Innovations in Computer Science*, pages 298–308. Tsinghua University Press, 2011. Extended version available at http://arxiv.org/pdf/1008.1284.

[16] H. Cohn and N. Heninger. Approximate common divisors via lattices. In *Tenth Algorithmic Number Theory Symposium*, pages 271–293. Mathematical Sciences Publishers (MSP), 2012-2013.

[17] H. Cohn and N. Heninger. Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding. *Advances in Mathematics of Communications*, 9(3):311–339, 2015.

[18] Don Coppersmith. Finding a small root of a univariate modular equation. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 155–165. Springer Berlin / Heidelberg, 1996.

[19] C. Devet, I. Goldberg, and N. Heninger. Optimally robust private information retrieval. Cryptology ePrint Archive, Report 2012/083, 2012.

[20] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In *ISSAC'03*, pages 135–142. ACM, 2003.

[21] S. Gupta. Hermite forms of polynomial matrices. Master's thesis, University of Waterloo, 2011.

[22] S. Gupta, S. Sarkar, A. Storjohann, and J. Valeriote. Triangular x-basis decompositions and derandomization of linear algebra algorithms over $K[x]$. *J. Symbolic Comput.*, 47(4):422–453, 2012.

[23] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Trans. Inf. Theory*, 54(1):135–150, 2008.

[24] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Trans. Inf. Theory*, 45(6):1757–1767, 1999.

[25] Nick Howgrave-Graham. Approximate integer common divisors. In *Revised Papers from the International Conference on Cryptography and Lattices*, CaLC'01, pages 51–66, London, UK, 2001. Springer-Verlag.

[26] C.-P. Jeannerod, V. Neiger, E. Schost, and G. Villard. Computing minimal interpolation bases, 2015.

[27] C.-P. Jeannerod, V. Neiger, E. Schost, and G. Villard. Fast computation of minimal interpolation bases in popov form for arbitrary shifts, 2016.

[28] T. Kailath. *Linear Systems*. Prentice-Hall, 1980.

[29] S. Lang. *Algebra (Revised Third Edition)*. Springer-Verlag New-York Inc., 2002.

[30] K. Lee and M. E. O'Sullivan. List decoding of Reed-Solomon codes from a Gröbner basis perspective. *J. Symbolic Comput.*, 43(9):645–658, 2008.

[31] R. J. McEliece. The Guruswami-Sudan decoding algorithm for Reed-Solomon codes, 2003. IPN Progress Report 42-153.

[32] T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. *J. Symb. Comput.*, 35:377–401, 2003.

[33] V. Neiger. Fast computation of shifted popov forms of polynomial matrices via systems of modular polynomial equations, 2016. http://arxiv.org/abs/1602.00710.

[34] J. S. R. Nielsen. *List Decoding of Algebraic Codes*. PhD thesis, Technical University of Denmark, 2013.

[35] J. S. R. Nielsen. Fast Kötter-Nielsen-Høholdt interpolation in the Guruswami-Sudan algorithm. In *ACCT'14*, 2014.

[36] R. R. Nielsen and T. Høholdt. Decoding Reed-Solomon codes beyond half the minimum distance. In *Coding Theory, Cryptography and Related Areas*, pages 221–236. Springer, 2000.

[37] V. Olshevsky and M. A. Shokrollahi. A displacement approach to efficient decoding of algebraic-geometric codes. In *STOC'99*, pages 235–244. ACM, 1999.

[38] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *FOCS'05*, pages 285–294. IEEE, 2005.

[39] V. M. Popov. Invariant description of linear, time-invariant controllable systems. *SIAM Journal on Control*, 10(2):252–264, 1972.

[40] J.-R. Reinhard. Algorithme LLL polynomial et applications. Master's thesis, École Polytechnique, Paris, France, 2003.

[41] R. M. Roth. *Introduction to Coding Theory*. Cambridge University Press, 2007.

[42] R. M. Roth and G. Ruckenstein. Efficient decoding of Reed-Solomon codes beyond half the minimum distance. *IEEE Trans. Inf. Theory*, 46(1):246–257, 2000.

[43] S. Sarkar and A. Storjohann. Normalization of row reduced matrices. In *ISSAC'11*, pages 297–304. ACM, 2011.

[44] A. Storjohann. High-order lifting and integrality certification. *J. Symbolic Comput.*, 36(3-4):613–648, 2003.

[45] A. Storjohann. Notes on computing minimal approximant bases. In *Dagstuhl Seminar Proceedings*, 2006.

[46] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *J. Complexity*, 13(1):180–193, 1997.

[47] M. Van Barel and A. Bultheel. A general module theoretic framework for vector M-Padé and matrix rational interpolation. *Numer. Algorithms*, 3:451–462, 1992.

[48] G. Villard. Computing Popov and Hermite forms of polynomial matrices. ISSAC'96, pages 250–258. ACM, 1996.

[49] A. Zeh, C. Gentner, and D. Augot. An interpolation procedure for list decoding Reed-Solomon codes based on generalized key equations. *IEEE Trans. Inf. Theory*, 57(9):5946–5959, 2011.

[50] W. Zhou and G. Labahn. Efficient algorithms for order basis computation. *J. Symbolic Comput.*, 47(7):793–819, 2012.