Certification of minimal approximant bases

Pascal Giorgi, Vincent Neiger

Certification of minimal approximant bases

Pascal Giorgi
Université Montpellier - CNRS, LIRMM
Montpellier, France
pascal.giorgi@lirmm.fr

Vincent Neiger
Univ. Limoges, CNRS, XLIM, UMR 7252
F-87000 Limoges, France
vincent.neiger@unilim.fr

ABSTRACT

Considering a given computational problem, a certificate is a piece of additional data that one attaches to the output in order to help verifying that this output is correct. Certificates are often used to make the verification phase significantly more efficient than the whole (re-)computation of the output. Here, we consider the minimal approximant basis problem, for which the fastest known algorithms compute a polynomial matrix of dimensions $m \times m$ and average degree D/m using $O^{\sim}(m^{\omega}\frac{D}{m})$ field operations. In the usual setting where the matrix to approximate has n columns with $n \leq m$, we provide a certificate of size mn, which can be computed in $O(m^{\omega}\frac{D}{m})$ operations and which allows us to verify an approximant basis by a Monte Carlo algorithm with cost bound $O(m^{\omega} + mD)$.

Besides theoretical interest, our motivation also comes from the fact that approximant bases arise in most of the fastest known algorithms for linear algebra over the univariate polynomials; thus, this work may help in designing certificates for other polynomial matrix computations. Furthermore, cryptographic challenges such as breaking records for discrete logarithm computations or for integer factorization rely in particular on computing minimal approximant bases for large instances: certificates can then be used to provide reliable computation on outsourced and error-prone clusters.

KEYWORDS

Certification; minimal approximant basis; order basis; polynomial matrix; truncated product.

1 INTRODUCTION

Context. For a given tuple $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{Z}_{>0}^n$ called *order*, we consider an $m \times n$ matrix $\mathbf{F}$ of formal power series with the column j truncated at order d_j. Formally, we let $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ be a matrix over the univariate polynomials over a field $\mathbb{K}$, such that the column j of $\mathbf{F}$ has degree less than d_j. Then, we consider the classical notion of minimal approximant bases for $\mathbf{F}$ [1, 26]. An approximant is a polynomial row vector $\mathbf{p} \in \mathbb{K}[X]^{1\times m}$ such that

$$\mathbf{pF} = \mathbf{0} \bmod \mathbf{X^d}, \quad \text{where } \mathbf{X^d} = \operatorname{diag}(X^{d_1}, \ldots, X^{d_n}); \qquad (1)$$

here $\mathbf{pF} = \mathbf{0} \bmod \mathbf{X^d}$ means that $\mathbf{pF} = \mathbf{qX^d}$ for some $\mathbf{q} \in \mathbb{K}[X]^{1\times n}$. The set of all approximants forms a (free) $\mathbb{K}[X]$-module of rank m,

$$\mathcal{A}_{\mathbf{d}}(\mathbf{F}) = \left\{ \mathbf{p} \in \mathbb{K}[X]^{1\times m} \;\middle|\; \mathbf{pF} = \mathbf{0} \bmod \mathbf{X^d} \right\}.$$

A basis of this module is called an *approximant basis* (or sometimes an *order basis* or a *σ-basis*); it is a nonsingular matrix in $\mathbb{K}[X]^{m\times m}$ whose rows are approximants in $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ and generate $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$.

The design of fast algorithms for computing approximant bases has been studied throughout the last three decades [1, 13, 14, 25–27]. Furthermore, these algorithms compute *minimal* bases, with respect to some degree measure specified by a shift $\mathbf{s} \in \mathbb{Z}^m$. The best known cost bound is $O^{\sim}(m^{\omega-1}D)$ operations in $\mathbb{K}$ [14] where D is the sum $D = |\mathbf{d}| = d_1 + \cdots + d_n$. Throughout the paper, our complexity estimates will fit the algebraic RAM model counting only operations in $\mathbb{K}$, and we will use $O(n^{\omega})$ to refer to the complexity of the multiplication of two $m \times m$ matrices, where $\omega < 2.373$ [4, 20].

Here, we are interested in the following question:

> *How to efficiently certify that some approximant basis algorithm indeed returns an* $\mathbf{s}$*-minimal basis of* $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$*?*

Because the known fast algorithms for approximant bases are deterministic, it might seem at first that a posteriori certification is pointless. In fact, it remains an essential tool in the context of unreliable computations that may arise when one delegates the processing to outsourced servers or to some large infrastructure that may be error-prone. In such a situation, and maybe before concluding a commercial contract to which this computing power is attached, one wants to ensure that he will be able to guarantee the correctness of the result of these computations. Of course, to be worthwhile, the verification procedure must be drastically faster than the original computation.

Resorting to such computing power is indeed necessary in the case of large instances of approximant bases, which are a key tool within challenging computations that try to tackle the hardness of some cryptographic protocols, for instance those based on the discrete logarithm problem (e.g. El Gamal) or integer factorization (e.g. RSA). The computation of a discrete logarithm over a 768-bits prime field, presented in [19], required to compute an approximant basis that served as input for a larger computation which took a total time of 355 core years on a 4096-cores cluster. The approximant basis took also 1 core year to be computed. For this kind of computations, it is of great interest to be able to guarantee the correctness of the approximant basis.

Linear algebra is a good candidate for designing fast verification algorithms since many problems have a cost related to matrix multiplication while their inputs only use a quadratic space. The first example one may think of is linear system solving. Indeed, given a solution vector $\mathbf{x} \in \mathbb{K}^n$ to a system $\mathbf{Ax} = \mathbf{b}$ defined by $\mathbf{A} \in \mathbb{K}^{n\times n}$ and $\mathbf{b} \in \mathbb{K}^n$, one can directly verify the correctness by checking the equations at a cost of $O(n^2)$ operations in $\mathbb{K}$. This procedure is deterministic and significantly faster than the current fastest known algorithm for solving the system, which costs $O(n^{\omega})$.

Another famous result, due to Freivalds [10], gives a probabilistic method to certify a matrix product. Given matrices $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{K}^{n\times n}$, the idea is to check $\mathbf{uC} = (\mathbf{uA})\mathbf{B}$ rather than $\mathbf{C} = \mathbf{AB}$, for a random row vector $\mathbf{U} \in \{0, 1\}^n$. This Monte-Carlo verification uses $O(n^2)$ operations and the probability of error can be made arbitrarily small by picking several vectors $\mathbf{u}$. Remark that this verification is always correct when it answers "false": this is called a *false-biased one-sided* Monte-Carlo algorithm.

A more general framework for certification has been introduced in [16]. The idea is to incorporate a piece of additional data to the output, called *certificate*, that helps to reduce the complexity of the verification. Since the certificate is produced together with the output, its correctness cannot be assumed either. Following this, efficient certificates have been designed for many linear algebra problems (see for example [7, 8, 17]).

In this context, one of the main challenges is to design *optimal* certificates, that is, ones which are verifiable in linear time. In this work, we seek such an optimal certificate for the problem of computing shifted minimal approximant bases.

Here, an instance is given by the input $(\mathbf{d}, \mathbf{F}, \mathbf{s})$ which is of size $O(mD)$: each column j of $\mathbf{F}$ contains at most md_j elements of $\mathbb{K}$, and the order sums to $d_1 + \cdots + d_n = |\mathbf{d}| = D$. We neglect the size of the shift $\mathbf{s}$, since one may always assume that it is nonnegative and such that $\max(\mathbf{s}) < mD$ (see [14, App. A]). Thus, ideally one would like to have a certificate which can be verified in time $O(mD)$. Comparing this to the cost $O\tilde{}(m^{\omega-1}D)$ of the fastest known approximant basis algorithms shows that such a certificate would be extremely valuable in practice, where $\omega \approx 2.78$ in the current best implementations of matrix multiplication over a finite field [3].

Here, we provide a non-interactive certification protocol which use the input $(\mathbf{d}, \mathbf{F}, \mathbf{s})$, the output $\mathbf{P}$, and a certificate which is a constant matrix in $\mathbf{C} \in \mathbb{K}^{m\times n}$. We design a Monte-Carlo verification algorithm with cost bound $O(mD + m^{\omega-1}(m+n))$; this is optimal as soon as D is large compared to m and n (e.g. when $D > m^2 + mn$), which is most often the case of interest. We also show that the certificate $\mathbf{C}$ can be computed in $O(m^{\omega-1}D)$ operations in $\mathbb{K}$, which is faster than known approximant basis algorithms.

Before describing our contribution in more detail, we recall basic facts on approximant bases, notably their main degree properties.

Degrees and size of approximant bases. For $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$, we denote the row degree of $\mathbf{P}$ as $\mathrm{rdeg}(\mathbf{P}) = (r_1, \ldots, r_m)$ where $r_i = \deg(\mathbf{P}_{i,*})$ is the degree of the row i of $\mathbf{P}$ for $1 \le i \le m$. The column degree $\mathrm{cdeg}(\mathbf{P})$ is defined similarly. More generally, we will consider row degrees shifted by some additive column weights: for a shift $\mathbf{s} = (s_1, \ldots, s_m) \in \mathbb{Z}^m$ the $\mathbf{s}$-row degree of $\mathbf{P}$ is $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}) = (r_1, \ldots, r_m)$ where $r_i = \max(\deg(\mathbf{P}_{i,1}) + s_1, \ldots, \deg(\mathbf{P}_{i,m}) + s_m)$.

As above, we will use $|\cdot|$ to denote the sum of integer tuples: for example $|\mathrm{rdeg}(\mathbf{P})|$ is the sum of the degrees of the rows of $\mathbf{P}$. The comparison of integer tuples is made entrywise: $\mathrm{cdeg}(\mathbf{F}) < \mathbf{d}$ means that the column j of $\mathbf{F}$ has degree less than d_j, for $1 \le j \le n$. When adding a constant to a tuple, say for example $\mathbf{s} - \min(\mathbf{s})$, this stands for the tuple $(s_1 - \mu, \ldots, s_m - \mu)$ where $\mu = \min(\mathbf{s})$.

In existing approximant basis algorithms, the output bases may take different forms: essentially, they can be $\mathbf{s}$-minimal (or $\mathbf{s}$-reduced [26]), $\mathbf{s}$-weak Popov [22], or $\mathbf{s}$-Popov [2]. For formal definitions, we direct the reader to these references as well as those above concerning approximant basis algorithms; here the precise form of the basis will not play an important role. What is however at the core of the efficiency of our algorithms is the impact of these forms on the degrees in the basis.

In what follows, by *size* of a matrix we mean the number of field elements used for its dense representation. We define the quantity

$$\mathrm{Size}(\mathbf{P}) = m^2 + \sum_{1\le i,j\le m} \max(0, \deg(p_{ij}))$$

for a matrix $\mathbf{P} = [p_{ij}] \in \mathbb{K}[X]^{m\times m}$. In the next paragraph, we discuss degree bounds on $\mathbf{P}$ when it is the output of any of the approximant basis algorithms mentioned above; note that these bounds all imply that $\mathbf{P}$ has size in $O(mD)$.

There is no general degree bound for approximant bases, even though a basis $\mathbf{P}$ of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ always satisfies $\deg(\det(\mathbf{P})) \le D$. For example, any unimodular matrix is a basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{0}) = \mathbb{K}[X]^{1\times m}$. Now, for an $\mathbf{s}$-minimal $\mathbf{P}$, we have $|\mathrm{rdeg}(\mathbf{P})| \in O(D)$ as soon as $|\mathbf{s} - \min(\mathbf{s})| \in O(D)$, then [26, Thm. 4.1], and it was shown in [27] that $\mathbf{P}$ has size in $O(D/m)$ if $|\max(\mathbf{s}) - \mathbf{s}| \in O(D)$. Yet, without such assumptions on the shift, there are $\mathbf{s}$-minimal bases whose size is in $\Theta(m^2D)$ [14, App. B], ruling out the feasibility of finding them in time $O\tilde{}(m^{\omega-1}D)$. In this case, the fastest known algorithms return the more constrained $\mathbf{s}$-Popov basis $\mathbf{P}$, for which $|\mathrm{cdeg}(\mathbf{P})| \le D$ holds independently of $\mathbf{s}$.

Problem and contribution. In order to certify that a given matrix $\mathbf{P}$ is an $\mathbf{s}$-minimal approximant basis for a given instance $(\mathbf{d}, \mathbf{F}, \mathbf{s})$, one needs to verify the following three characteristics on $\mathbf{P}$:

(1) *Minimal:* $\mathbf{P}$ is in $\mathbf{s}$-reduced form. This amounts to retrieving the so-called $\mathbf{s}$-leading matrix of $\mathbf{P}$ and testing its invertibility. The cost bound for this is in $O(m^\omega)$ operations in $\mathbb{K}$.

(2) *Approximant:* the rows of $\mathbf{P}$ are approximants. That is, we should check that $\mathbf{PF} = \mathbf{0} \bmod \mathbf{X}^{\mathbf{d}}$. The difficulty is to avoid computing the full truncated product $\mathbf{PF} \bmod \mathbf{X}^{\mathbf{d}}$, since this costs $O\tilde{}(m^{\omega-1}D)$. In Section 3, we give a probabilistic algorithm which verifies more generally $\mathbf{PF} = \mathbf{G} \bmod \mathbf{X}^{\mathbf{d}}$ using $O(\mathrm{Size}(\mathbf{P}) + mD)$ operations, without requiring a certificate.

(3) *Basis:* the rows of $\mathbf{P}$ generate the approximant module[1]. For this, we show that it suffices to verify first that $\det(\mathbf{P})$ is of the form cX^δ for some $c \in \mathbb{K}\setminus\{0\}$ and where $\delta = |\mathrm{rdeg}(\mathbf{P})|$, and second that some constant $m \times (m+n)$ matrix has full rank; this matrix involves $\mathbf{P}(0)$ and the coefficient $\mathbf{C}$ of degree 0 of $\mathbf{PFX}^{-\mathbf{d}}$. In Section 2, we provide new material proving that $\mathbf{C}$ can serve as a certificate, and that a probabilistic algorithm can assess its correctness at a suitable cost.

Our single-round certification protocol is as follows. The *Prover* sends the matrix $\mathbf{P}$, supposedly an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$, along with a constant matrix $\mathbf{C} \in \mathbb{K}^{m\times n}$, supposedly the coefficient of degree 0 of the product $\mathbf{PFX}^{-\mathbf{d}}$. Then, the *Verifier* must solve Problem 1 within a cost asymptotically better than $O\tilde{}(m^{\omega-1}D)$.

Problem 1: Approximant basis certification
Input:
- order $\mathbf{d} \in \mathbb{Z}^n_{>0}$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\mathrm{cdeg}(\mathbf{F}) < \mathbf{d}$,
- shift $\mathbf{s} \in \mathbb{Z}^m$,
- matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$,
- certificate matrix $\mathbf{C} \in \mathbb{K}^{m\times n}$.

Output:
- *True* if $\mathbf{P}$ is an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ and $\mathbf{C}$ is the coefficient of degree 0 of $\mathbf{PFX}^{-\mathbf{d}}$, otherwise *False*.

The main result in this paper is the following.

[1] This is not implied by (1) and (2): for $d = \max(\mathbf{d})$, then $X^d\mathbf{I}_m$ is $\mathbf{s}$-reduced and $X^d\mathbf{I}_m\mathbf{F} = \mathbf{0} \bmod \mathbf{X}^{\mathbf{d}}$ holds; yet, $X^d\mathbf{I}_m$ is not a basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ for most $(\mathbf{F}, \mathbf{d})$.

Theorem 1.1. *There is a Monte-Carlo algorithm which solves Problem 1 using $O(mD + m^{\omega-1}(m+n))$ operations in $\mathbb{K}$, assuming* $\text{Size}(\mathbf{P}) \in O(mD)$. *It chooses $m+2$ elements uniformly at random from a finite subset $S \subset \mathbb{K}$. If S has cardinality at least $2(D+1)$, then the probability that a* True *answer is incorrect is less than $1/2$, while a* False *answer is always correct.*

For the reader interested in a more precise cost bound showing the constant factors, we refer to Proposition 2.5. Under the assumptions of the theorem, the size of the input and output of the problem is in $O(mD)$; the cost bound above is therefore optimal (up to constant factors) as soon as $m^{\omega-2}(m+n) \in O(D)$.

In the case of small finite fields, there could be no such subset S with $\#S \geq 2(D+1)$, writing $\#S$ for the cardinality of S. Then, our approach would still work by performing the probabilistic part of the computation over a field extension of $\mathbb{K}$ which sufficiently large cardinality. Note that an extension of degree about $1 + \lceil \log_2(D) \rceil$ would be sufficient; this augments our complexity estimate by a factor of $O^{\sim}(\log_2(D))$, which remains acceptable in our context.

Our second result is the efficient computation of the certificate.

Theorem 1.2. *Let $\mathbf{d} \in \mathbb{Z}^n_{>0}$, let $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\text{cdeg}(\mathbf{F}) < \mathbf{d}$ and $m \in O(D)$, and let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$. If $|\text{rdeg}(\mathbf{P})| \in O(D)$ or $|\text{cdeg}(\mathbf{P})| \in O(D)$, there is a deterministic algorithm which computes the coefficient of degree 0 of $\mathbf{PFX}^{-\mathbf{d}}$ using $O(m^{\omega-1}D)$ operations in $\mathbb{K}$ if $m \geq n$ and $O(m^{\omega-1}D\log(n/m))$ operations in $\mathbb{K}$ if $m < n$.*

Note that the assumption $m \in O(D)$ in this theorem is commonly made in approximant basis algorithms, since when $D \leq m$ most entries of a minimal approximant basis have degree in $O(1)$ and the algorithms then rely on classical dense linear algebra methods.

2 CERTIFYING APPROXIMANT BASES

In this section we present our certification algorithm, which solves Problem 1 with the cost bound and probability of success announced in Theorem 1.1. One of its core components is the verification of truncated polynomial matrix products, the details of whom are given in Section 3 and are taken for granted here.

First, we give the basic properties behind the correctness of this algorithm, which are summarized in the following result.

Theorem 2.1. *Let $\mathbf{d} \in \mathbb{Z}^n_{>0}$, let $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$, and let $\mathbf{s} \in \mathbb{Z}^m$. A matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ is an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ if and only if the following properties are all satisfied:*

(i) $\mathbf{P}$ *is* $\mathbf{s}$*-reduced;*
(ii) $\det(\mathbf{P})$ *is a nonzero monomial in* $\mathbb{K}[X]$*;*
(iii) the rows of $\mathbf{P}$ *are in* $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$*, that is,* $\mathbf{PF} = \mathbf{0} \bmod \mathbf{X}^{\mathbf{d}}$*;*
(iv) $[\mathbf{P}(0) \;\; \mathbf{C}] \in \mathbb{K}^{m\times(m+n)}$ *has full rank, where* $\mathbf{C}$ *is the coefficient of degree 0 of* $\mathbf{PFX}^{-\mathbf{d}}$.

We remark that having both $\mathbf{PF} = \mathbf{0} \bmod \mathbf{X}^{\mathbf{d}}$ and $\mathbf{C}$ the constant coefficient of $\mathbf{PFX}^{-\mathbf{d}}$ is equivalent to the single truncated identity $\mathbf{PF} = \mathbf{X}^{\mathbf{d}}\mathbf{C} \bmod \mathbf{X}^{\mathbf{t}}$, where $\mathbf{t} = (d_1+1, \ldots, d_n+1)$.

As mentioned above, the details of the certification of the latter identity is deferred to Section 3, where we present more generally the certification for truncated products of the form $\mathbf{PF} = \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$.

The combination of Items (i) and (iii) describes the set of matrices $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ which are $\mathbf{s}$-reduced and whose rows are in $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$. For $\mathbf{P}$ to be an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$, its rows should further form a generating set for $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$; thus, our goal here is to prove that this property is realized by the combination of Items (ii) and (iv).

For this, we will rely on a link between approximant bases and kernel bases, given in Lemma 2.3. We recall that, for a given matrix $\mathbf{M} \in \mathbb{K}[X]^{\mu\times\nu}$ of rank r,

- a *kernel basis* for $\mathbf{M}$ is a matrix in $\mathbb{K}[X]^{(\mu-r)\times\mu}$ whose rows form a basis of the left kernel $\{\mathbf{p} \in \mathbb{K}[X]^{1\times\mu} \mid \mathbf{pM} = \mathbf{0}\}$,
- a *column basis* for $\mathbf{M}$ is a matrix in $\mathbb{K}[X]^{\mu\times r}$ whose columns form a basis of the column space $\{\mathbf{Mp}, \mathbf{p} \in \mathbb{K}[X]^{\nu\times 1}\}$.

In particular, by definition a kernel basis has full row rank and a column basis has full column rank. We will use the next result, which states that the column space of a kernel basis is the whole space, spanned by the identity matrix.

Lemma 2.2. *Let $\mathbf{M} \in \mathbb{K}[X]^{\mu\times\nu}$ and let $\mathbf{B} \in \mathbb{K}[X]^{k\times\mu}$ be a kernel basis for $\mathbf{M}$. Then, any column basis for $\mathbf{B}$ is unimodular. Equivalently, $\mathbf{BU} = \mathbf{I}_k$ for some $\mathbf{U} \in \mathbb{K}[X]^{\mu\times k}$.*

Proof. Let $\mathbf{S} \in \mathbb{K}[X]^{k\times k}$ be a column basis for $\mathbf{B}$. By definition, $\mathbf{B} = \mathbf{S}\hat{\mathbf{B}}$ for some $\hat{\mathbf{B}} \in \mathbb{K}[X]^{k\times\mu}$. Then $\mathbf{0} = \mathbf{BM} = \mathbf{S}\hat{\mathbf{B}}\mathbf{M}$, hence $\hat{\mathbf{B}}\mathbf{M} = \mathbf{0}$ since $\mathbf{S}$ is nonsingular. Thus, $\mathbf{B}$ being a kernel basis for $\mathbf{M}$, we have $\hat{\mathbf{B}} = \mathbf{TB}$ for some $\mathbf{T} \in \mathbb{K}[X]^{k\times k}$. We obtain $(\mathbf{ST} - \mathbf{I}_k)\mathbf{B} = \mathbf{0}$, hence $\mathbf{ST} = \mathbf{I}_k$ since $\mathbf{B}$ has full row rank. Thus, $\mathbf{S}$ is unimodular. □

This property arises for example in the computation of column bases and unimodular completions in [28, 29]. (While Lemma 2.2 can be derived from the material in these references, in particular from [28, Lem. 3.1], we wrote a direct proof for better clarity.)

We will use the property of Lemma 2.2 for a specific kernel basis, built from an approximant basis as follows.

Lemma 2.3. *Let $\mathbf{d} \in \mathbb{Z}^n_{>0}$, $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$, and $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$. Then, $\mathbf{P}$ is a basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ if and only if there exists $\mathbf{Q} \in \mathbb{K}[X]^{m\times n}$ such that $[\mathbf{P} \;\; \mathbf{Q}]$ is a kernel basis for $[\mathbf{F}^{\mathsf{T}} \;\; -\mathbf{X}^{\mathbf{d}}]^{\mathsf{T}}$. If this is the case, then we have $\mathbf{Q} = \mathbf{PFX}^{-\mathbf{d}}$ and there exist $\mathbf{V} \in \mathbb{K}[X]^{m\times m}$ and $\mathbf{W} \in \mathbb{K}[X]^{n\times m}$ such that $\mathbf{PV} + \mathbf{QW} = \mathbf{I}_m$.*

Proof. The equivalence is straightforward; a detailed proof can be found in [23, Lem. 8.2]. If $[\mathbf{P} \;\; \mathbf{Q}]$ is a kernel basis for $[\mathbf{F}^{\mathsf{T}} \;\; -\mathbf{X}^{\mathbf{d}}]^{\mathsf{T}}$, then we have $\mathbf{PF} = \mathbf{QX}^{\mathbf{d}}$, hence the explicit formula for $\mathbf{Q}$. Besides, the last claim is a direct consequence of Lemma 2.2. □

This leads us to the following result, which forms the main ingredient that was missing in order to prove Theorem 2.1.

Lemma 2.4. *Let $\mathbf{d} \in \mathbb{Z}^n_{>0}$ and let $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$. Let $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ be such that $\mathbf{PF} = \mathbf{0} \bmod \mathbf{X}^{\mathbf{d}}$ and $\det(\mathbf{P})$ is a nonzero monomial, and let $\mathbf{C} \in \mathbb{K}^{m\times(m+n)}$ be the constant coefficient of $\mathbf{PFX}^{-\mathbf{d}}$. Then, $\mathbf{P}$ is a basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ if and only if $[\mathbf{P}(0) \;\; \mathbf{C}] \in \mathbb{K}^{m\times(m+n)}$ has full rank.*

Proof. First, assume that $\mathbf{P}$ is a basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$. Then, defining $\mathbf{Q} = \mathbf{PFX}^{-\mathbf{d}} \in \mathbb{K}[X]^{m\times n}$, Lemma 2.3 implies that $\mathbf{PV} + \mathbf{QW} = \mathbf{I}_m$ for some $\mathbf{V} \in \mathbb{K}[X]^{m\times m}$ and $\mathbf{W} \in \mathbb{K}[X]^{n\times m}$. Since $\mathbf{Q}(0) = \mathbf{C}$, this yields $\mathbf{P}(0)\mathbf{V}(0) + \mathbf{CW}(0) = \mathbf{I}_m$, and thus $[\mathbf{P}(0) \;\; \mathbf{C}]$ has full rank.

Now, assume that $\mathbf{P}$ is not a basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$. If $\mathbf{P}$ has rank $< m$, then $[\mathbf{P}(0) \;\; \mathbf{C}]$ has rank $< m$ as well. If $\mathbf{P}$ is nonsingular, $\mathbf{P} = \mathbf{UA}$ for some basis $\mathbf{A}$ of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ and some $\mathbf{U} \in \mathbb{K}[X]^{m\times m}$ which is nonsingular but not unimodular. Then, $\det(\mathbf{U})$ is a nonconstant divisor of the nonzero monomial $\det(\mathbf{P})$; hence $\det(\mathbf{U})(0) = 0 = \det(\mathbf{U}(0))$, and

thus $\mathbf{U}(0)$ has rank $< m$. Since $[\mathbf{P} \;\; \mathbf{Q}] = \mathbf{U}[\mathbf{A} \;\; \mathbf{AFX}^{-\mathbf{d}}]$, it directly follows that $[\mathbf{P}(0) \;\; \mathbf{C}]$ has rank $< m$. □

Proof of Theorem 2.1. If $\mathbf{P}$ is an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$, then by definition Items (i) and (iii) are satisfied. Since the rows of $X^{\max(\mathbf{d})}\mathbf{I}_m$ are in $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ and $\mathbf{P}$ is a basis, the matrix $X^{\max(\mathbf{d})}\mathbf{I}_m$ is a left multiple of $\mathbf{P}$ and therefore the determinant of $\mathbf{P}$ divides $X^{m\max(\mathbf{d})}$: it is a nonzero monomial. Then, according to Lemma 2.4, $[\mathbf{P}(0) \;\; \mathbf{C}]$ has full rank. Conversely, if Items (ii) to (iv) are satisfied, then Lemma 2.4 states that $\mathbf{P}$ is a basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$; thus if furthermore Item (i) is satisfied then $\mathbf{P}$ is an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$. □

Algorithm 1: CertifApproxBasis
Input:

- order $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{Z}^n_{>0}$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\mathrm{cdeg}(\mathbf{F}) < \mathbf{d}$,
- shift $\mathbf{s} = (s_1, \ldots, s_m) \in \mathbb{Z}^m$,
- matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$,
- certificate matrix $\mathbf{C} \in \mathbb{K}^{m\times n}$.

Output: *True* if $\mathbf{P}$ is an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$ and $\mathbf{C}$ is the constant term of $\mathbf{PFX}^{-\mathbf{d}}$, otherwise *True* or *False*.

1. `/* P not in s-reduced form ⇒ False */`
   $\mathbf{L} \leftarrow$ the matrix in $\mathbb{K}^{m\times m}$ whose entry i,j is the coefficient of degree $\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P}_{i,*}) - s_j$ of the entry i,j of $\mathbf{P}$
   If $\mathbf{L}$ is not invertible *then return False*
2. `/* rank([P(0) C]) not full rank ⇒ False */`
   If $\mathrm{rank}([\mathbf{P}(0) \;\; \mathbf{C}]) < m$ *then return False*
3. `/* det(P) not a nonzero monomial ⇒ False */`
   $S \leftarrow$ a finite subset of $\mathbb{K}$
   $\Delta \leftarrow |\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})| - |\mathbf{s}|$
   $\alpha \leftarrow$ chosen uniformly at random from S
   If $\det(\mathbf{P}(\alpha)) \neq \det(\mathbf{P}(1))\alpha^{\Delta}$ *then return False*
4. `/* certify truncated product PF = CX^d mod X^t */`
   $\mathbf{t} \leftarrow (d_1 + 1, \ldots, d_n + 1)$
   Return VerifTruncMatProd$(\mathbf{t}, \mathbf{P}, \mathbf{F}, \mathbf{CX}^{\mathbf{d}})$

In order to provide a sharp estimate of the cost of Algorithm 1, we recall the best known cost bound with constant factors of the LQUP factorization of an $m \times n$ matrix over $\mathbb{K}$, which we use for computing ranks and determinants. Assuming $m \leq n$, we have:

$$C(m,n) = \left(\left\lceil \frac{n}{m} \right\rceil \frac{1}{2^{\omega-1}-2} - \frac{1}{2^{\omega}-2}\right) MM(m)$$

operations in $\mathbb{K}$ [6, Lem 5.1], where $MM(m)$ is the cost for the multiplication of $m \times m$ matrices over $\mathbb{K}$.

Proposition 2.5. *Algorithm 1 uses at most*

$$\begin{aligned} &5\mathrm{Size}(\mathbf{P}) + 2m(D + \max(\mathbf{d})) + 3C(m,m) + C(m,m+n) \\ &\qquad + (4m+1)n + 4\log_2(Dd_1\cdots d_n) \\ &\in O(\mathrm{Size}(\mathbf{P}) + mD + m^{\omega-1}(m+n)) \end{aligned}$$

operations in $\mathbb{K}$, where $D = |\mathbf{d}|$. It is a false-biased Monte Carlo algorithm. If $\mathbf{P}$ is not an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$, then the probability that it outputs True *is less than $\frac{D+1}{\#S}$, where S is the finite subset of $\mathbb{K}$ from which random field elements are drawn.*

Proof. By definition, $\mathbf{P}$ is $\mathbf{s}$-reduced if and only if its so-called $\mathbf{s}$-leading matrix $\mathbf{L}$ computed at Step **1** is invertible. Thus, Step **1** correctly tests the property in Item (i) of Theorem 2.1. It uses at most $C(m,m)$ operations in $\mathbb{K}$.

Step **2** correctly tests the first part of Item (iv) of Theorem 2.1. It uses at most $C(m, m+n)$ operations in $\mathbb{K}$.

Step **3** performs a false-biased Monte Carlo verification of Item (ii) of Theorem 2.1. Indeed, since $\mathbf{P}$ is $\mathbf{s}$-reduced (otherwise the algorithm would have exited at Step **1**), we know that

$$\deg(\det(\mathbf{P})) = \Delta = |\mathrm{rdeg}_{\mathbf{s}}(\mathbf{P})| - |\mathbf{s}|$$

(see for example [15, Section 6.3.2]). Thus, $\det(\mathbf{P})$ is a nonzero monomial if and only if $\det(\mathbf{P}) = \det(\mathbf{P}(1))X^{\Delta}$. Step **3** tests the latter equality by evaluation at a random point α. The algorithm only returns *False* if $\det(\mathbf{P}(\alpha)) \neq \det(\mathbf{P}(1))\alpha^{\Delta}$, in which case $\det(\mathbf{P})$ is indeed not a nonzero monomial. Furthermore, if we have $\det(\mathbf{P}) \neq \det(\mathbf{P}(1))X^{\Delta}$, then the probability that the algorithm fails to detect this, meaning that $\det(\mathbf{P}(\alpha)) = \det(\mathbf{P}(1))\alpha^{\Delta}$, is at most $\frac{\Delta}{\#S}$. Since $\Delta \leq D$ according to [26, Thm. 4.1], this is also at most $\frac{D}{\#S} < \frac{D+1}{\#S}$.

The evaluations $\mathbf{P}(\alpha)$ and $\mathbf{P}(1)$ are computed using respectively at most $2(\mathrm{Size}(\mathbf{P}) - m^2)$ operations and at most $\mathrm{Size}(\mathbf{P}) - m^2$ additions. Then, computing the two determinants $\det(\mathbf{P}(\alpha))$ and $\det(\mathbf{P}(1))$ uses at most $2C(m,m) + 2m$ operations. Finally, computing $\det(\mathbf{P}(1))\alpha^{\Delta}$ uses at most $2\log_2(\Delta) + 1 \leq 2\log_2(D) + 1$ operations.

Summing the cost bounds for the first three steps gives

$$\begin{aligned} &3(\mathrm{Size}(\mathbf{P}) - m^2) + 3C(m,m) + C(m,m+n) + 2m + 2\log_2(D) + 1 \\ &\leq 3\mathrm{Size}(\mathbf{P}) + 3C(m,m) + C(m,m+n) + 2\log_2(D), \end{aligned} \tag{2}$$

which is in $O(\mathrm{Size}(\mathbf{P}) + m^{\omega-1}(m+n) + \log_2(D))$.

Step **4** tests the identity $\mathbf{PF} = \mathbf{CX}^{\mathbf{d}} \bmod \mathbf{X}^{\mathbf{t}}$, which corresponds to both Item (iii) of Theorem 2.1 and the second part of Item (iv). Proposition 3.2 ensures that:

- If the call to VerifTruncMatProd returns *False*, we have $\mathbf{PF} \neq \mathbf{CX}^{\mathbf{d}} \bmod \mathbf{X}^{\mathbf{t}}$, and Algorithm 1 correctly returns *False*.
- If $\mathbf{PF} \neq \mathbf{CX}^{\mathbf{d}} \bmod \mathbf{X}^{\mathbf{t}}$ holds, the probability that Algorithm 1 fails to detect this (that is, the call at Step **4** returns *True*) is less than $\frac{\max(\mathbf{d})+1}{\#S}$.

A cost bound for Step **4** is given in Proposition 3.2, with a minor improvement for the present case given in Remark 3.3. Summing it with the bound in Eq. (2) gives a cost bound for Algorithm 1, which is at most that in the proposition.

Thanks to Theorem 2.1, the above considerations show that when the algorithm returns *False*, then $\mathbf{P}$ is indeed not an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$. On the other hand, if $\mathbf{P}$ is not an $\mathbf{s}$-minimal basis of $\mathcal{A}_{\mathbf{d}}(\mathbf{F})$, the algorithm returns *True* if and only of one of the randomized verifications in Steps **3** and **4** take the wrong decision. According to the probabilities given above, this may happen with probability less than $\max(\frac{D+1}{\#S}, \frac{\max(\mathbf{d})+1}{\#S}) = \frac{D+1}{\#S}$. □

3 VERIFYING A TRUNCATED PRODUCT

In this section, we focus on the verification of truncated products of polynomial matrices, and we give the corresponding algorithm VerifTruncMatProd used in Algorithm 1.

Given a truncation order $\mathbf{t}$ and polynomial matrices $\mathbf{P}$, $\mathbf{F}$, $\mathbf{G}$, our goal is to verify that $\mathbf{PF} = \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$ holds with good probability.

Without loss of generality, we assume that the columns of $\mathbf{F}$ and $\mathbf{G}$ are already truncated with respect to the order $\mathbf{t}$, that is, $\text{cdeg}(\mathbf{F}) < \mathbf{t}$ and $\text{cdeg}(\mathbf{G}) < \mathbf{t}$. Similarly, we assume that $\mathbf{P}$ is truncated with respect to $\delta = \max(\mathbf{t})$, that is, $\deg(\mathbf{P}) < \delta$.

Problem 2: Truncated matrix product verification
Input:

- truncation order $\mathbf{t} \in \mathbb{Z}_{>0}^n$,
- matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ with $\deg(\mathbf{P}) < \max(\mathbf{t})$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m\times n}$ with $\text{cdeg}(\mathbf{F}) < \mathbf{t}$,
- matrix $\mathbf{G} \in \mathbb{K}[X]^{m\times n}$ with $\text{cdeg}(\mathbf{G}) < \mathbf{t}$.

Output:

- *True* if $\mathbf{PF} = \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$, otherwise *False*.

Obviously, our aim is to obtain a verification algorithm which has a significantly better cost than the straightforward approach which computes the truncated product $\mathbf{PF} \bmod \mathbf{X}^{\mathbf{t}}$ and compares it with the matrix $\mathbf{G}$. To take an example: if we have $n \in O(m)$ as well as $|\text{rdeg}(\mathbf{P})| \in O(|\mathbf{t}|)$ or $|\text{cdeg}(\mathbf{P})| \in O(|\mathbf{t}|)$, as commonly happens in approximant basis computations, then this truncated product $\mathbf{PF} \bmod \mathbf{X}^{\mathbf{t}}$ can be computed using $O\tilde{}(m^{\omega-1}|\mathbf{t}|)$ operations in $\mathbb{K}$.

If our problem was to verify the full product $\mathbf{PF} = \mathbf{G}$, the classical approach would be to use evaluation at a random point, following ideas from [5, 24, 31]. However, evaluation does not behave well with regards to truncation. A similar issue was tackled in [12] for the verification of the middle product and the short products of univariate polynomials. The algorithm of [12] can be adapted to work with polynomial matrices by writing them as univariate polynomials with matrix coefficents; for example, $\mathbf{P}$ is a polynomial $\mathbf{P} = \sum_{0\le i<\delta} \mathbf{P}_i X^i$ with coefficients $\mathbf{P}_i \in \mathbb{K}^{m\times m}$. While this approach does lead to a verification of $\mathbf{PF} = \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$ with a good probability of success, it has a cost which is close to that of computing $\mathbf{PF} \bmod \mathbf{X}^{\mathbf{t}}$.

To further lower down the cost, we will combine the evaluation of truncated product from [12] with Freivalds' technique [10]. The latter consists in left-multiplying the matrices by some random vector $\mathbf{u} \in \mathbb{K}^{1\times m}$, and rather checking whether $\mathbf{uPF} = \mathbf{uG} \bmod \mathbf{X}^{\mathbf{t}}$; this effectively reduces the row dimension of the manipulated matrices, hence faster computations. Furthermore, this does not harm the probability of success of the verification, as we detail now.

In what follows, given a matrix $\mathbf{A} \in \mathbb{K}[X]^{m\times n}$ and an order $\mathbf{t} \in \mathbb{Z}_{>0}^n$, we write $\mathbf{A} \operatorname{rem} \mathbf{X}^{\mathbf{t}}$ for the (unique) matrix $\mathbf{B} \in \mathbb{K}[X]^{m\times n}$ such that $\mathbf{B} = \mathbf{A} \bmod \mathbf{X}^{\mathbf{t}}$ and $\text{cdeg}(\mathbf{B}) < \mathbf{t}$. For simplicity, we will often write $\mathbf{A}_1\mathbf{A}_2 \operatorname{rem} \mathbf{X}^{\mathbf{t}}$ to actually mean $(\mathbf{A}_1\mathbf{A}_2) \operatorname{rem} \mathbf{X}^{\mathbf{t}}$.

Lemma 3.1. *Let S be a finite subset of $\mathbb{K}$. Let $\mathbf{u} \in \mathbb{K}^{1\times m}$ with entries chosen uniformly and independently at random from S, and let $\alpha \in \mathbb{K}$ be chosen uniformly at random from S. Assuming $\mathbf{PF} \neq \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$, the probability that $(\mathbf{uPF} \operatorname{rem} \mathbf{X}^{\mathbf{t}})(\alpha) = \mathbf{uG}(\alpha)$ is less than $\frac{\delta}{\#S}$.*

Proof. Let $\mathbf{A} = (\mathbf{PF} - \mathbf{G}) \operatorname{rem} \mathbf{X}^{\mathbf{t}}$. By assumption, there exists a pair (i, j) such that the entry (i, j) of $\mathbf{A}$ is nonzero. Since this entry is a polynomial in $\mathbb{K}[X]$ of degree less than δ, the probability that α is a root of this entry is at most $\frac{\delta-1}{\#S}$. As a consequence, we have $\mathbf{A}(\alpha) \neq 0 \in \mathbb{K}^{m\times n}$ with probability at least $1 - \frac{\delta-1}{\#S}$. In this case, $\mathbf{uA}(\alpha) = 0$ occurs with probability at most $\frac{1}{\#S}$ (see [21, Sec. 7.1]).

Thus, altogether the probability that $\mathbf{uA}(\alpha) = 0$ is at most

$$\frac{\delta-1}{\#S} + \left(1 - \frac{\delta-1}{\#S}\right)\frac{1}{\#S} < \frac{\delta}{\#S},$$

which concludes the proof. □

We readily deduce an approach to verify the truncated product: compute $\mathbf{uA}(\alpha) = ((\mathbf{uPF} - \mathbf{uG}) \operatorname{rem} \mathbf{X}^{\mathbf{t}})(\alpha)$ and check whether it is zero or nonzero. The remaining difficulty is to compute this quantity efficiently: we will show that this can be done in $O(\text{Size}(\mathbf{P}) + m|\mathbf{t}|)$ operations in $\mathbb{K}$, where $\text{Size}(\mathbf{P})$ is the size of $\mathbf{P}$ as defined in Section 1.

For this, we use a strategy similar to that in [12, Lem. 4.1] and essentially based on the following formula for the truncated product. Consider a positive integer $t \le \delta$ and a vector $\mathbf{f} \in \mathbb{K}[X]^{m\times 1}$ of degree less than t; one may think of $\mathbf{f}$ as a column $\mathbf{F}_{*,j}$ of $\mathbf{F}$ and of t as the corresponding order t_j. Writing $\mathbf{f} = \sum_{0\le k<t} \mathbf{f}_k X^k$ with $\mathbf{f}_k \in \mathbb{K}^{m\times 1}$ and $\mathbf{uP} = \sum_{0\le k<\delta} \mathbf{p}_k X^k$ with $\mathbf{p}_k \in \mathbb{K}^{1\times m}$, we have

$$\begin{aligned}\mathbf{uPf} \operatorname{rem} X^t &= \sum_{k=0}^{t-1}\left(\sum_{i=0}^{t-1-k} \mathbf{p}_i X^i\right)\mathbf{f}_k X^k \\ &= X^{t-1}\sum_{k=0}^{t-1}\left(\sum_{i=0}^{t-1-k} \mathbf{p}_{t-1-k-i} X^{-i}\right)\mathbf{f}_k.\end{aligned}$$

Thus, the evaluation can be expressed as

$$(\mathbf{uPf} \operatorname{rem} X^t)(\alpha) = \alpha^{t-1}\sum_{k=0}^{t-1} \mathbf{c}_{t-1-k}\mathbf{f}_k, \tag{3}$$

where we define, for $0 \le k < \delta$,

$$\mathbf{c}_k = (\mathbf{uP} \operatorname{rem} X^{k+1})(\alpha^{-1}) = \sum_{i=0}^{k} \mathbf{p}_{k-i}\alpha^{-i} \in \mathbb{K}^{1\times m}. \tag{4}$$

These identities give an algorithm to compute the truncated product evaluation $(\mathbf{uPf} \operatorname{rem} X^t)(\alpha)$, which we sketch as follows:

- apply Horner's method to the reversal of $\mathbf{uP} \operatorname{rem} X^t$ at the point α^{-1}, storing the intermediate results which are exactly the t vectors $\mathbf{c}_0, \ldots, \mathbf{c}_{t-1}$;
- compute the scalar products $\lambda_k = \mathbf{c}_{t-1-k}\mathbf{f}_k$ for $0 \le k < t$;
- compute α^{t-1} and then $\alpha^{t-1}\sum_{0\le k<t} \lambda_k$.

The last step gives the desired evaluation according to Eq. (3). In our case, this will be applied to each column $\mathbf{f} = \mathbf{F}_{*,j}$ for $1 \le j \le n$. We will of course perform the first item only once to obtain the δ vectors $\mathbf{c}_0, \ldots, \mathbf{c}_{\delta-1}$, since they do not depend on $\mathbf{f}$.

Proposition 3.2. *Algorithm 2 uses at most*

$$2\text{Size}(\mathbf{P}) + (6m+1)|\mathbf{t}| + 2n\log_2(\delta) \in O(\text{Size}(\mathbf{P}) + m|\mathbf{t}| + n\log_2(\delta))$$

operations in $\mathbb{K}$, where $\delta \le |\mathbf{t}|$ is the largest of the truncation orders. It is a false-biased Monte Carlo algorithm. If $\mathbf{PF} \neq \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$, the probability that it outputs True *is less than $\frac{\delta}{\#S}$, where S is the finite subset of $\mathbb{K}$ from which random field elements are drawn.*

Proof. The discussion above shows that this algorithm correctly computes the evaluations

$$[e_j]_{1\le j\le n} = \mathbf{uG}(\alpha) \quad \text{and} \quad [e'_j]_{1\le j\le n} = (\mathbf{uPF} \operatorname{rem} \mathbf{X}^{\mathbf{t}})(\alpha).$$

If it returns *False*, then there is at least one column for which the evaluations of $\mathbf{uG}$ and $\mathbf{uPF} \operatorname{rem} \mathbf{X}^{\mathbf{t}}$ do not match; thus we must

Algorithm 2: VerifTruncMatProd
Input:

- truncation order $\mathbf{t} = (t_1, \ldots, t_n) \in \mathbb{Z}^n_{>0}$,
- matrix $\mathbf{P} \in \mathbb{K}[X]^{m\times m}$ such that $\deg(\mathbf{P}) < \delta = \max(\mathbf{t})$,
- matrix $\mathbf{F} = [f_{ij}] \in \mathbb{K}[X]^{m\times n}$ with $\mathrm{cdeg}(\mathbf{F}) < \mathbf{t}$,
- matrix $\mathbf{G} \in \mathbb{K}[X]^{m\times n}$ with $\mathrm{cdeg}(\mathbf{G}) < \mathbf{t}$.

Output: *True* if $\mathbf{PF} = \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$, otherwise *True* or *False*.

1. `/* Main objects for verification */`
   $S \leftarrow$ a finite subset of $\mathbb{K}$
   $\alpha \leftarrow$ element of $\mathbb{K}$ chosen uniformly at random from S
   $\mathbf{u} \leftarrow$ vector in $\mathbb{K}^{1\times m}$ with entries chosen uniformly and independently at random from S
2. `/* Freivalds: row dimension becomes 1 */`
   $\mathbf{p} \leftarrow \mathbf{uP}$ `// in` $\mathbb{K}[X]^{1\times m}$, `degree` $< \delta$
   $\mathbf{g} \leftarrow \mathbf{uG}$ `// in` $\mathbb{K}[X]^{1\times n}$, $\mathrm{cdeg}(\mathbf{g}) < \mathbf{t}$
3. `/* Evaluation of right-hand side:` $\mathbf{uG}(\alpha)$ `*/`
   write $\mathbf{g} = [g_1 \ \cdots \ g_n]$ with $g_j \in \mathbb{K}[X]$ of degree $< t_j$
   For j from 1 *to n*:
   $e_j \leftarrow g_j(\alpha)$
4. `/* Truncated evaluations` $\mathbf{c}_0, \ldots, \mathbf{c}_{\delta-1}$ `*/`
   write $\mathbf{p} = \sum_{0\le k<\delta} \mathbf{p}_k X^k$ with $\mathbf{p}_k \in \mathbb{K}^{1\times m}$
   $\mathbf{c}_0 \leftarrow \mathbf{p}_0$
   For k from 1 *to* $\delta - 1$:
   $\mathbf{c}_k \leftarrow \mathbf{p}_k + \alpha^{-1}\mathbf{c}_{k-1}$
5. `/* Evaluation of left-hand side:` $(\mathbf{uPF} \operatorname{rem} \mathbf{X}^{\mathbf{t}})(\alpha)$ `*/`
   For j from 1 *to n*: `// process column` $\mathbf{F}_{*,j}$
   write $\mathbf{F}_{*,j} = \sum_{0\le k<t_j} \mathbf{f}_k X^k$
   $(\lambda_k)_{0\le k<t_j} \leftarrow (\mathbf{c}_{t_j-1-k} \cdot \mathbf{f}_k)_{0\le k<t_j}$
   $e'_j \leftarrow \alpha^{t_j-1} \sum_{0\le k<t_j} \lambda_k$
6. *If* $e_j \neq e'_j$ for some $j \in \{1, \ldots, n\}$ *then return False*
   Else return True

have $\mathbf{uPF} \operatorname{rem} \mathbf{X}^{\mathbf{t}} \neq \mathbf{uG}$ and therefore $\mathbf{PF} \neq \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$. Besides, the algorithm correctly returns *True* if $\mathbf{PF} = \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$.

The analysis of the probability of failure (the algorithm returns *True* while $\mathbf{PF} \neq \mathbf{G} \bmod \mathbf{X}^{\mathbf{t}}$) is a direct consequence of Lemma 3.1.

Step **2** uses at most $2\mathrm{Size}(\mathbf{P}) + (2m-1)|\mathbf{t}|$ operations in $\mathbb{K}$. The Horner evaluations at Steps **3** and **4** require at most $2(|\mathbf{t}| - n)$ and at most $1 + 2m(\delta - 1)$ operations, respectively. Now, we consider the j-th iteration of the loop at Step **5**. The scalar products $(\lambda_k)_{0\le k<t_j}$ are computed using at most $(2m-1)t_j$ operations; the sum and multiplication by α^{t_j-1} giving e'_j use at most $t_j + 2\log_2(t_j - 1)$ operations. Summing over $1 \le j \le n$, this gives a total of at most $2m|\mathbf{t}| + 2\log_2((t_1 - 1)\cdots(t_n - 1))$ operations for Step **5**. Finally, Step **6** uses at most n comparisons of two field elements. Summing these bounds for each step yields the cost bound

$$2\mathrm{Size}(\mathbf{P}) + (4m+1)|\mathbf{t}| + 2m(\delta-1) - n + 2\log_2((t_1-1)\cdots(t_n-1)), \quad (5)$$

which is at most the quantity in the proposition. □

In the certification of approximant bases, we want to verify a truncated matrix product in the specific case where each entry in the column j of $\mathbf{G}$ is simply zero or a monomial of degree $t_j - 1$. Then, a slightly better cost bound can be given, as follows.

Remark 3.3. Assume that $\mathbf{t} = (d_1 + 1, \ldots, d_n + 1)$ and $\mathbf{G} = \mathbf{CX}^{\mathbf{d}}$, for some $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{Z}^n_{>0}$, and some constant $\mathbf{C} \in \mathbb{K}^{m\times n}$. Then, the computation of $\mathbf{uG}$ at Step **2** uses at most $(2m-1)n$ operations in $\mathbb{K}$. Besides, since the polynomial g_j at Step **3** is either zero or a monomial of degree d_j, its evaluation e_j is computed using at most $2\log_2(d_j) + 1$ operations via repeated squaring [11, Sec. 4.3]. Thus, Step **3** uses at most $2\log_2(d_1 \cdots d_n) + n$ operations. As a result, the cost bound in Eq. (5) is lowered to

$$\begin{aligned} &2\mathrm{Size}(\mathbf{P}) + 2m(|\mathbf{t}| + \delta - 1 + n) + n + 4\log_2(d_1 \cdots d_n) + 1 \\ &= 2\mathrm{Size}(\mathbf{P}) + 2m(D + \max(\mathbf{d}) + 2n) + n + 4\log_2(d_1 \cdots d_n) + 1 \end{aligned}$$

where $D = |\mathbf{d}|$. □

4 COMPUTING THE CERTIFICATE

4.1 Context

In this section, we show how to efficiently compute the proposed certificate $\mathbf{C} \in \mathbb{K}^{m\times n}$. As stated above, $\mathbf{C}$ is the term of degree 0 of the product $\mathbf{PFX}^{-\mathbf{d}}$, whose entries are Laurent polynomials a priori (they are in $\mathbb{K}[X]$ if and only if $\mathbf{P}$ is an approximant basis). In other words, the column j of $\mathbf{C}$ is the term of degree d_j of the column j of $\mathbf{PF}$, where we write $\mathbf{d} = (d_1, \ldots, d_n)$.

We recall the notation $D = d_1 + \cdots + d_n$. Note that, without loss of generality, we may truncate $\mathbf{P}$ so that $\deg(\mathbf{P}) \le \max(\mathbf{d})$.

For example, suppose that the dimensions and the order are balanced: $m = n$ and $\mathbf{d} = (D/m, \ldots, D/m)$. Then, $\mathbf{C} \in \mathbb{K}^{m\times m}$ is the coefficient of degree D/m of the product $\mathbf{PF}$, where $\mathbf{P}$ and $\mathbf{F}$ are $m \times m$ matrices over $\mathbb{K}[X]$. Thus $\mathbf{C}$ can be computed using D/m multiplications of $m \times m$ matrices over $\mathbb{K}$, at a total cost $O(m^{\omega-1}D)$.

Going back to the general case, the main obstacle to obtain similar efficiency is that both the degrees in $\mathbf{P}$ and the order $\mathbf{d}$ (hence the degrees in $\mathbf{F}$) may be unbalanced. Still, we have $\mathrm{cdeg}(\mathbf{F}) < \mathbf{d}$ with sum $|\mathbf{d}| = D$, and as we have described in the introduction, we may assume that either $|\mathrm{rdeg}(\mathbf{P})| \in O(D)$ or $|\mathrm{cdeg}(\mathbf{P})| \le D$ holds. In this context, both $\mathbf{F}$ and $\mathbf{P}$ are represented by $O(mD)$ field elements.

Our goal is to generalize the method above for the balanced case to this general situation with unbalanced degrees, while keeping the same cost bound $O(m^{\omega-1}D)$. This implies that computing the certificate $\mathbf{C}$ has negligible cost compared to the fastest known algorithms for computing an approximant basis. Indeed, the latter are in $O^{\sim}(m^{\omega-1}D)$, involving logarithmic factors related both to polynomial arithmetic and to divide and conquer approaches.

Note that fast approximant basis algorithms usually involve the computation of a so-called residual, which is a truncated product $\mathbf{PF} \bmod \mathbf{X}^{\mathbf{d}}$ which involves the same profiles of unbalanced degrees as the ones here. The techniques used in these algorithms allow one to compute the full product $\mathbf{PF} \bmod \mathbf{X}^{\mathbf{d}+1}$ by relying essentially on a multiplication of two $m \times m$ polynomial matrices of degree in $O(D/m)$. This gives in particular the certificate matrix $\mathbf{C}$; our aim here is to give a more efficient approach when only $\mathbf{C}$ is needed.

We first remark that $\mathbf{C}$ can be computed by naive linear algebra using $O(m^2D)$ operations. Indeed, writing $\mathrm{rdeg}(\mathbf{P}) = (r_1, \ldots, r_m)$, we have the following explicit formula for each entry in $\mathbf{C}$:

$$\mathbf{C}_{i,j} = \sum_{k=1}^{\min(r_i, d_j)} \mathbf{P}_{i,*,k}\, \mathbf{F}_{*,j,d_j-k},$$

where $\mathbf{P}_{i,*,k}$ is the coefficient of degree k of the row i of $\mathbf{P}$ and similar notation is used for $\mathbf{F}$. Then, since $\min(r_i, d_j) \le d_j$, the column j of $\mathbf{C}$ is computed via md_j scalar products of length m, at a cost of $O(m^2 d_j)$ operations in $\mathbb{K}$. Summing this for the n columns yields the desired cost.

This approach considers each column of $\mathbf{F}$ separately, allowing us to truncate at precision $d_j + 1$ for the column j and thus to rule out the issue of the unbalancedness of the degrees in $\mathbf{P}$. However, this also prevents us from incorporating fast matrix multiplication into the computation of $\mathbf{C}$. In our efficient method, we rather avoid considering columns or rows separately, and at the same time we take into account the unbalancedness of the degrees in both $\mathbf{P}$ and $\mathbf{F}$. Our approach bears some similarities with algorithms for polynomial matrix multiplication with unbalanced degrees (see for example [30, Sec. 3.6]), yet without following them strictly since the full product is not needed here.

4.2 Sparsity and degree structure

For simplicity, we explain our method assuming $|\mathrm{rdeg}(\mathbf{P})| \in O(D)$*; in Sections 4.2 and 4.3,* $\gamma \ge 1$ *is a real number such that* $|\mathrm{rdeg}(\mathbf{P})| \le \gamma D$*. There is no difficulty in adapting the arguments and the algorithm to the other case of interest, that is, when* $|\mathrm{cdeg}(\mathbf{P})| \le D$*.*

To simplify the exposition, we start by replacing the tuple $\mathbf{d}$ by the uniform bound $d = \max(\mathbf{d})$. To achieve this, we consider the matrix $\mathbf{H} = \mathbf{F}\mathbf{X}^{d-\mathbf{d}}$, where $d - \mathbf{d}$ stands for $(d - d_1, \ldots, d - d_n)$: then, $\mathbf{C}$ is the coefficient of degree d in $\mathbf{PH}$.

Since $\mathrm{cdeg}(\mathbf{F}) < \mathbf{d}$, we now have $\deg(\mathbf{H}) < d$. The fact that $\mathbf{F}$ has column degree less than $\mathbf{d}$ translates into the fact that $\mathbf{H}$ has column valuation at least $d - \mathbf{d}$ (and degree less than d); like $\mathbf{F}$, this matrix $\mathbf{H}$ is represented by only mD field elements. Recalling that we have assumed $\deg(\mathbf{P}) \le d$, we can write

$$\mathbf{P} = \sum_{k=0}^{d} \mathbf{P}_k X^k \quad \text{and} \quad \mathbf{H} = \sum_{k=0}^{d} \mathbf{H}_k X^k.$$

where $\mathbf{P}_k \in \mathbb{K}^{m \times m}$ and $\mathbf{H}_k \in \mathbb{K}^{m \times n}$ for all k (note that $\mathbf{H}_d = \mathbf{0}$). Then, our goal is to compute the matrix

$$\mathbf{C} = \sum_{k=1}^{d} \mathbf{P}_k \mathbf{H}_{d-k}. \tag{6}$$

The essential remark which will lead us to an efficient algorithm is that each matrix $\mathbf{P}_k$ has only few nonzero rows when k becomes large, and each matrix $\mathbf{H}_{d-k}$ has only few nonzero columns when k becomes large. To state this formally, we define the set of indices of the rows of degree at least k in $\mathbf{P}$ and the set of indices of the orders at least k in $\mathbf{d}$:

$$\mathcal{R}_k = \{i \in \{1, \ldots, m\} \mid \mathrm{rdeg}(\mathbf{P}_{i,*}) \ge k\},$$
$$\mathcal{D}_k = \{j \in \{1, \ldots, n\} \mid d_j \ge k\}.$$

The latter corresponds to the set of indices of columns of $\mathbf{F}$ which are allowed to have degree $\ge k - 1$ or, equivalently, to the set of indices of columns of $\mathbf{H}$ which are allowed to have valuation $\le d - k$.

Lemma 4.1. *For a given* $k \in \{1, \ldots, d\}$*:*

- *If* $i \notin \mathcal{R}_k$*, then the row* i *of* $\mathbf{P}_k$ *is zero.*
- *If* $j \notin \mathcal{D}_k$*, then the column* j *of* $\mathbf{H}_{d-k}$ *is zero.*

In particular, $\mathbf{P}_k$ *has at most* $\#\mathcal{R}_k \le \gamma D/k$ *nonzero rows and* $\mathbf{H}_{d-k}$ *has at most* $\#\mathcal{D}_k \le D/k$ *nonzero columns.*

Proof. The row i of $\mathbf{P}_k$ is the coefficient of degree k of the row i of $\mathbf{P}$. If it is nonzero, we must have $i \in \mathcal{R}_k$. Similarly, the column j of $\mathbf{H}_{d-k}$ is the coefficient of degree $d-k$ of the column j of $\mathbf{H} = \mathbf{F}\mathbf{X}^{d-\mathbf{d}}$. If it is nonzero, we must have $d - k \ge d - d_j$, hence $k \in \mathcal{D}_k$.

The upper bounds on the cardinalities of $\mathcal{R}_k$ and $\mathcal{D}_k$ follow by construction of these sets: we have $k \cdot \#\mathcal{D}_k \le |\mathbf{d}| = D$, and also $k \cdot \#\mathcal{R}_k \le |\mathrm{rdeg}(\mathbf{P})|$ with $|\mathrm{rdeg}(\mathbf{P})| \le \gamma D$ by assumption. □

4.3 Algorithm and cost bound

As a result of Lemma 4.1, in the computation of $\mathbf{C}$ based on Eq. (6), we may restrict our view of the matrix $\mathbf{P}_k$ to its submatrix formed by the rows with index in $\mathcal{R}_k$, and our view of the matrix $\mathbf{H}_k$ to its submatrix formed by the columns with index in $\mathcal{D}_k$. For example, if $k > \gamma D/m$ and $k > D/n$, the matrices in the product $\mathbf{P}_k\mathbf{H}_k$ have dimensions at most $\lfloor \gamma D/k \rfloor \times m$ and $m \times \lfloor D/k \rfloor$, respectively.

These remarks on the sparsity of the matrices $\mathbf{P}_k$ and $\mathbf{H}_k$, and on the structure of this sparsity given by the easily computed sets $\mathcal{R}_k$ and $\mathcal{D}_k$, lead to the following efficient algorithm.

Algorithm 3: CertificateComp
Input:

- order $\mathbf{d} \in \mathbb{Z}_{>0}^n$,
- matrix $\mathbf{F} \in \mathbb{K}[X]^{m \times n}$ such that $\mathrm{cdeg}(\mathbf{F}) < \mathbf{d}$,
- matrix $\mathbf{P} \in \mathbb{K}[X]^{m \times m}$ such that $\deg(\mathbf{P}) \le \max(\mathbf{d})$.

Output: the coefficient $\mathbf{C} \in \mathbb{K}^{m \times n}$ of degree 0 of $\mathbf{P}\mathbf{F}\mathbf{X}^{-\mathbf{d}}$.

1. $(r_1, \ldots, r_m) \leftarrow \mathrm{rdeg}(\mathbf{P})$
2. $\mathbf{C} \leftarrow \mathbf{0} \in \mathbb{K}^{m \times n}$
3. *For* k *from* 1 *to* $\max(\mathbf{d})$:
   - $\mathcal{R} \leftarrow \{i \in \{1, \ldots, m\} \mid r_i \ge k\}$
   - $\mathcal{D} = \{c_1, \ldots, c_t\} \leftarrow \{j \in \{1, \ldots, n\} \mid d_j \ge k\}$
   - $\mathbf{A} \in \mathbb{K}^{\#\mathcal{R} \times m} \leftarrow$ coefficient of degree k of $\mathbf{P}_{\mathcal{R},*}$
   - $\mathbf{B} \in \mathbb{K}^{m \times t} \leftarrow$ for all $1 \le j \le t$, $\mathbf{B}_{*,j}$ is the coefficient of degree $d_j - k$ of $\mathbf{F}_{*,c_j}$
   - $\mathbf{C}_{\mathcal{R},\mathcal{D}} \leftarrow \mathbf{C}_{\mathcal{R},\mathcal{D}} + \mathbf{A}\mathbf{B}$
4. *Return* $\mathbf{C}$

Proposition 4.2. *Algorithm 3 is correct. Assuming that* $m \in O(D)$ *and* $|\mathrm{rdeg}(\mathbf{P})| \in O(D)$*, where* $D = |\mathbf{d}|$*, it uses* $O(m^{\omega-1}D)$ *operations in* $\mathbb{K}$ *if* $n \le m$ *and* $O(m^{\omega-1}D\log(n/m))$ *operations in* $\mathbb{K}$ *if* $n > m$*.*

Proof. For the correctness, note first that for $1 \le j \le n$, the coefficient of degree $d_j - k$ of $\mathbf{F}_{*,j}$ is the coefficient of degree $d - k$ of $\mathbf{H}_{*,j}$. Thus, with the notation from Section 4.2, the matrix $\mathbf{B}$ at the iteration k of the loop is exactly the submatrix of $\mathbf{H}_{d-k}$ formed by its columns with index in $\mathcal{D}_k$. Therefore, the loop in Algorithm 3 simply applies Eq. (6), discarding from $\mathbf{P}_k$ and $\mathbf{F}_{d-k}$ the rows and columns which are known to be zero.

It remains to estimate the cost of the update of $\mathbf{C}$ in each iteration of the loop. Precisely, the main task is to compute the product $\mathbf{AB}$, where the matrices $\mathbf{A}$ and $\mathbf{B}$ have dimensions $\#\mathcal{R} \times m$ and $m \times t$, respectively. Then, adding this product to the submatrix $\mathbf{C}_{\mathcal{R},\mathcal{D}}$ only costs $\#\mathcal{R} \cdot t$ additions in $\mathbb{K}$. Note that for each $(\mathbf{AB})_{i,j}$ we need to

keep track of its corresponding row and column indices in $\mathcal{R}$ and $\mathcal{D}$, which can be done by storing these indices into two arrays.

Consider $\gamma = \lceil |\mathrm{rdeg}(\mathbf{P})|/D \rceil \geq 1$ (indeed, if $|\mathrm{rdeg}(\mathbf{P})| = 0$, then $\mathbf{P}$ is constant and $\mathbf{C} = \mathbf{0}$). By Lemma 4.1, at the iteration k we have

$$\#\mathcal{R} \leq \min(m, \gamma D/k) \quad \text{and} \quad t = \#\mathcal{D} \leq \min(n, D/k).$$

We will consider the cases $n \leq m$ and $n > m$ separately. We will repeatedly use the bound $\lceil \gamma D/m \rceil \in O(D/m)$, which holds thanks to our assumptions $m \in O(D)$ and $\gamma \in O(1)$.

First, suppose $n \leq m$. At the iterations $k < \lceil \gamma D/m \rceil$ the matrices $\mathbf{A}$ and $\mathbf{B}$ both have dimensions at most $m \times m$, hence their product can be computed in $O(m^\omega)$ operations. These iterations have a total cost of $O(m^\omega \lceil \gamma D/m \rceil) \subseteq O(m^{\omega-1} D)$. At the iterations $k \geq \lceil \gamma D/m \rceil$ the matrices $\mathbf{A}$ and $\mathbf{B}$ have dimensions at most $(\gamma D/k) \times m$ and $m \times (D/k)$, with $D/k \leq \gamma D/k \leq m$; computing their product uses

$$O\Big((D/k)^{\omega-1} m\Big) \subseteq O\Big(m D^{\omega-1} k^{1-\omega}\Big)$$

operations. Thus, the total cost for these iterations is in

$$O\left(m D^{\omega-1} \sum_{k=\lceil \gamma D/m \rceil}^{\max(\mathbf{d})} k^{1-\omega}\right)$$
$$\subseteq O\left(m D^{\omega-1} (\lceil \gamma D/m \rceil)^{2-\omega} \sum_{i=0}^{+\infty} 2^{i(2-\omega)}\right) \subseteq O(m^{\omega-1} D).$$

The first inclusion follows by applying Lemma 4.3 with parameters $\mu = \lceil \gamma D/m \rceil$, $\nu = \max(\mathbf{d})$, and $\theta = 1 - \omega$; note that the sum is finite since $0 < 2^{2-\omega} < 1$. Grouping both sets of iterations, this proves that Algorithm 3 uses $O(m^{\omega-1} D)$ operations in $\mathbb{K}$.

Now, suppose $n > m$. At the iterations $k < \lceil D/n \rceil$ the matrices $\mathbf{A}$ and $\mathbf{B}$ have dimensions at most $m \times m$ and $m \times n$, hence their product can be computed in $O(m^{\omega-1} n)$. The total cost is in $O(m^{\omega-1} D)$ since there are $\lceil D/n \rceil - 1 < D/n$ iterations (with $n \leq D$ by definition).

For the iterations $k \geq \lceil \gamma D/m \rceil$, we can just repeat the analysis above for the same values of k when $m \leq n$: this shows that these iterations cost $O(m^{\omega-1} D)$ here as well.

Finally, for the iterations $\lceil D/n \rceil \leq k < \lceil \gamma D/m \rceil$, $\mathbf{A}$ and $\mathbf{B}$ have dimensions at most $m \times m$ and $m \times (D/k)$, with $D/k \leq n$. Thus the product $\mathbf{AB}$ can be computed in $O(m^\omega + m^{\omega-1} D/k)$ operations. Summing the term m^ω over these $O(D/m)$ iterations yields a cost of $O(m^{\omega-1} D)$ operations. Summing the other term gives

$$\sum_{k=\lceil D/n \rceil}^{\lceil \gamma D/m \rceil - 1} m^{\omega-1} D/k \quad \in O(m^{\omega-1} D \log(n/m))$$

since, by the last claim of Lemma 4.3, we have

$$\sum_{k=\lceil D/n \rceil}^{\lceil \gamma D/m \rceil - 1} k^{-1} \;\leq\; 1 + \left\lfloor \log_2 \left(\frac{\lceil \gamma D/m \rceil - 1}{\lceil D/n \rceil} \right) \right\rfloor \;\leq\; 1 + \log_2(\gamma n/m).$$

Thus, summing the costs for the three considered sets of iterations, we obtain the announced cost for Algorithm 3 when $n > m$. □

In the proof above, we use the following straightforward bound.

Lemma 4.3. *Given integers $0 < \mu < \nu$ and a real number $\theta \leq 0$, we have the following bound:*

$$\sum_{k=\mu}^{\nu} k^\theta \;\leq\; \mu^{\theta+1} \sum_{i=0}^{\ell-1} 2^{i(\theta+1)},$$

where $\ell = \lfloor \log_2(\nu/\mu) \rfloor + 1$. In particular, $\sum_{k=\mu}^{\nu} k^{-1} \leq \ell$.

Proof. Note that ℓ is chosen such that $2^\ell \mu - 1 \geq \nu$. Then, the upper bound is obtained by splitting the sum as follows:

$$\sum_{k=\mu}^{\nu} k^\theta \leq \sum_{i=0}^{\ell-1} \sum_{k=2^i\mu}^{2^{i+1}\mu-1} k^\theta \leq \sum_{i=0}^{\ell-1} \sum_{k=2^i\mu}^{2^{i+1}\mu-1} (2^i \mu)^\theta = \sum_{i=0}^{\ell-1} (2^i \mu)^{\theta+1},$$

where the second inequality comes from the fact that $x \mapsto x^\theta$ is decreasing on the positive real numbers. □

Finally, we describe minor changes in Algorithm 3 which allow us to deal with the case with small average *column* degree; precisely, we now assume that $\mathrm{cdeg}(\mathbf{P}) \in O(D)$. Then, instead of using the set $\mathcal{R}_k$ considered above, we rather define the set

$$C_k = \{j \in \{1, \ldots, m\} \mid \mathrm{cdeg}(\mathbf{P}_{*,j}) \geq k\}.$$

Then we have the following lemma, analogous to Lemma 4.1.

Lemma 4.4. *For $k \in \{1, \ldots, m\}$ and $j \notin C_k$, the column j of $\mathbf{P}_k$ is zero. In particular, $\mathbf{P}_k$ has at most $\#C_k \leq \gamma D/k$ nonzero columns.*

Thus, we can modify Algorithm 3 to take into account the column degree of $\mathbf{P}$ instead of its row degree. This essentially amounts to redefining the matrices $\mathbf{A}$ and $\mathbf{B}$ in the loop as follows:

- $\mathbf{A} \in \mathbb{K}^{m \times \#C_k}$ is the coefficient of degree k of $\mathbf{P}_{*,C_k}$.
- $\mathbf{B} \in \mathbb{K}^{\#C_k \times t}$ is such that for all $i \in C_k$ and $1 \leq j \leq t$, $\mathbf{B}_{i,j}$ is the coefficient of degree $d_j - k$ of $\mathbf{F}_{i,c_j}$

These modifications have obviously no impact on the correctness. Furthermore, it is easily verified that the same cost bound holds since we obtain a similar matrix multiplication cost at each iteration.

5 PERSPECTIVES

As noted in the introduction, our certificate is almost optimal since we can verify it at a cost $O(mD + m^{\omega-1}(m+n))$ while the input size is mD. One should notice that the extra term $O(m^{\omega-1}(m+n))$ corresponds to certifying problems of linear algebra over $\mathbb{K}$, namely the rank and the determinant. One may easily imagine that these could be verified in $O(m(m+n))$ operations using interactive certificates built upon the results in [7, 9, 17], thus yielding an optimal certificate. Still, for practical applications, our simpler certificate should already be negligible compared to the approximant basis computation, since the constants involved in the cost are very small, as we have observed in our estimates above. We plan to confirm this for the approximant bases implementations in the LinBox library[2].

Finally, our verification protocol needs $(m+2)\log_2(\#S)$ random bits, yielding a probability of failure less than $\frac{D+1}{\#S}$. The majority of these bits is required by Algorithm 2 when choosing m random elements for the vector $\mathbf{u}$. As proposed in [18], it may be worthwhile to pick a single random value ζ and to use $\mathbf{u} = [1 \;\; \zeta \;\; \cdots \;\; \zeta^{m-1}]$. In the case where $\max(\mathbf{d}) < D/2$, this choice would not affect the

[2] https://github.com/linbox-team/linbox

probability of failure while decreasing the number of random bits to $3\log_2(\#S)$. In particular, at the price of the same number of bits as we currently use in our algorithm, we could run our verification $(m+2)/3$ times and decrease the probability of failure to $(\frac{D+1}{\#S})^{\frac{m+2}{3}}$.

REFERENCES

[1] 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.

[2] B. Beckermann, G. Labahn, and G. Villard. Shifted normal forms of polynomial matrices. In *ISSAC'99*, pages 189–196. ACM, 1999.

[3] B. Boyer and J.-G. Dumas. Matrix multiplication over word-size modular rings using approximate formulas. *ACM Trans. Math. Softw.*, 42(3):20:1–20:12, 2016.

[4] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symbolic Comput.*, 9(3):251–280, 1990.

[5] R. A. DeMillo and R. J. Lipton. A probabilistic remark on algebraic program testing. *Inform. Process. Lett.*, 7(4):193–195, 1978.

[6] J.-G. Dumas, P. Giorgi, and C. Pernet. Dense linear algebra over word-size prime fields: The FFLAS and FFPACK packages. *ACM Trans. Math. Softw.*, 35(3):19:1–19:42, 2008.

[7] J.-G. Dumas and E. Kaltofen. Essentially optimal interactive certificates in linear algebra. In *ISSAC'14*, pages 146–153. ACM, 2014.

[8] J-G. Dumas, E. Kaltofen, E. Thomé, and G. Villard. Linear time interactive certificates for the minimal polynomial and the determinant of a sparse matrix. In *ISSAC'16*, pages 199–206. ACM, 2016.

[9] J.-G. Dumas, D. Lucas, and C. Pernet. Certificates for triangular equivalence and rank profiles. In *ISSAC'17*, pages 133–140. ACM, 2017.

[10] R. Freivalds. Fast probabilistic algorithms. In *Mathematical Foundations of Computer Science*, volume 74, pages 57–69. Springer Berlin Heidelberg, 1979.

[11] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra (third edition)*. Cambridge University Press, 2013.

[12] P. Giorgi. Certification of polynomial middle product, 2017. Available online at https://hal-lirmm.ccsd.cnrs.fr/lirmm-015384532 (accessed in February 2018).

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

[14] C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard. Fast computation of minimal interpolation bases in Popov form for arbitrary shifts. In *ISSAC'16*, pages 295–302. ACM, 2016.

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

[16] E. Kaltofen, B. Li, Z. Yang, and L. Zhi. Exact certification in global polynomial optimization via sums-of-squares of rational functions with rational coefficients. *Journal of Symbolic Computation*, 47(1):1 – 15, 2012.

[17] E. Kaltofen, M. Nehring, and B. D. Saunders. Quadratic-time certificates in linear algebra. In *ISSAC'11*, pages 171–176. ACM, 2011.

[18] T. Kimbrel and R. K. Sinha. A probabilistic algorithm for verifying matrix products using $O(n^2)$ time and $\log_2(n) + O(1)$ random bits. *Information Processing Letters*, 45(2):107 – 110, 1993.

[19] T. Kleinjung, C. Diem, A. K. Lenstra, C. Priplata, and C. Stahlke. Computation of a 768-bit prime field discrete logarithm. In *Eurocrypt 2017*, pages 185–201. Springer International Publishing, 2017.

[20] F. Le Gall. Powers of tensors and fast matrix multiplication. In *ISSAC'14*, pages 296–303. ACM, 2014.

[21] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1995.

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

[23] V. Neiger. *Bases of relations in one or several variables: fast algorithms and applications*. PhD thesis, École Normale Supérieure de Lyon, November 2016.

[24] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

[25] A. Storjohann. Notes on computing minimal approximant bases. In *Challenges in Symbolic Computation Software*, Dagstuhl Seminar Proceedings, 2006.

[26] 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.

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

[28] W. Zhou and G. Labahn. Computing column bases of polynomial matrices. In *ISSAC'13*, pages 379–386. ACM, 2013.

[29] W. Zhou and G. Labahn. Unimodular completion of polynomial matrices. In *ISSAC'14*, pages 413–420. ACM, 2014.

[30] W. Zhou, G. Labahn, and A. Storjohann. Computing minimal nullspace bases. In *ISSAC'12*, pages 366–373. ACM, 2012.

[31] R. Zippel. Probabilistic algorithms for sparse polynomials. In *EUROSAM'79*, volume 72 of *LNCS*, pages 216–226. Springer, 1979.