HAL open science

High-order lifting for polynomial Sylvester matrices

Clément Pernet, Hippolyte Signargout, Gilles Villard

▶ **To cite this version:**

Clément Pernet, Hippolyte Signargout, Gilles Villard. High-order lifting for polynomial Sylvester matrices. Journal of Complexity, 2024, 80, pp.101803. 10.1016/j.jco.2023.101803 . hal-03740320v3

HAL Id: hal-03740320

https://hal.science/hal-03740320v3

Submitted on 5 Oct 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

High-order lifting for polynomial Sylvester matrices

Clément Pernet[b], Hippolyte Signargout[a,b], Gilles Villard[a]

[a] *Univ. Lyon, CNRS, ENS de Lyon, Inria, UCBL, LIP UMR 5668 Lyon, France*
[b] *Univ. Grenoble Alpes, CNRS, LJK UMR 5224 Grenoble, France*

Abstract

A new algorithm is presented for computing the resultant of two "sufficiently generic" bivariate polynomials over an arbitrary field. For such p and q in $\mathsf{K}[x, y]$ of degree d in x and n in y, the resultant with respect to y is computed using $O(n^{1.458}d)$ arithmetic operations as long as $d = O(n^{1/3})$. For $d = 1$, the complexity estimate is therefore essentially reconciled with the best known estimates of Neiger et al. 2021 for the related problems of modular composition and characteristic polynomial in a univariate quotient algebra. This allows to cross the $3/2$ barrier in the exponent of n for the first time in the case of the resultant. More generally, our algorithm improves on best previous algebraic ones as long as $d = O(n^{0.47})$.

The resultant is the determinant of the associated univariate polynomial Sylvester matrix of degree d, the problem is therefore intimately related to that of computing determinants of structured polynomial matrices. We first identify new advanced aspects of structure specific to the polynomial Sylvester matrix. This enables to compute the determinant by successfully mixing the block baby steps/giant steps approach of Kaltofen and Villard 2005, until then restricted to the case $d = 1$ for characteristic polynomials, and the high-order lifting strategy of Storjohann 2003 usually reserved for dense polynomial matrices.

Keywords: complexity, algorithm, computer algebra, resultant, polynomial structured matrix, displacement rank.

1. Introduction

In this paper, we propose a new algorithm for computing the resultant of two generic bivariate polynomials over a commutative field K. For two polynomials $p, q \in \mathsf{K}[x, y]$, the resultant $\mathrm{Res}_y(p, q)$ with respect to y is the determinant of the Sylvester matrix associated to p and q over $\mathsf{K}[x]$ (see Eq. (2)). The reader may refer to the books [2, 10], and to [11, 31] and references therein on the whole subject.

As well as for many fundamental operations on univariate polynomials over K (multiplication, division with remainder, multipoint evaluation, greatest common divisor, etc.), the resultant of two polynomials of degree at most n in $\mathsf{K}[x]$ can be computed using $\tilde{O}(n)$ arithmetic operations in K [10, Chap. 11]. (The soft-O notation is used to omit logarithmic factors: $c' = \tilde{O}(c)$ if there exists $k \in \mathbb{N}$ for which $c' = O(c \log^k c)$.) In the bivariate case, and since the early 1970's, the best known complexity bound for the computation of the resultant in the general case is $\tilde{O}(n^2 d)$ for p, q of degree bounded by d in x and n in y [10, Chap. 11]. The resultant with respect to y is a polynomial of degree at most $2nd$ in $\mathsf{K}[x]$, we therefore see that the latter bound is within a factor of the order of n from the input/output size.

Usual solutions for the resultant of two polynomials are most often based on the extended Euclidean algorithm and polynomial remainder sequences [11, 31].

By going beyond this path, complementary reductions of the complexity gap with respect to the input/output size were recently obtained [40, 16, 41]. These approaches exploit the properties of appropriate

families of polynomials in the ideal $\mathcal{I} = \langle p, q\rangle$ in $\mathsf{K}[x, y]$ or structured matrix operations (see Section 1.3), and rely on genericity assumptions on p and q in the Zariski sense. Throughout the paper, a property is generic if it holds except on a hypersurface of the corresponding parameter space.

Given bivariate polynomials p, q of degree d in x and n in y, it is shown in [40] that the resultant $\mathrm{Res}_y(p, q)$ can be computed generically with respect to p and q using $\tilde{O}(n^{2-1/\omega}d)$ arithmetic operations, where $\omega \leq 3$ is a feasible exponent for the cost of square matrix multiplication (two $n \times n$ matrices over a ring can be multiplied using $O(n^\omega)$ arithmetic operations). For example in the case $d = n$ this gives the first subcubic complexity estimate for the problem. The algorithm is mainly based on polynomial matrix operations and our work builds upon it (see Section 1.1.1).

On the other hand, using a bit complexity model and in the specific case of a finite field $\mathbb{F}$, consider p and q of respective total degrees $n_1 \geq n_2$. If p and q are sufficiently generic then $\mathrm{Res}_y(p, q)$ can be computed in expected time $O((n_1 n_2 \log|\mathbb{F}|)^{1+\epsilon}) + \tilde{O}(n_1^2 \log|\mathbb{F}|)$ using a randomized Las Vegas algorithm [16] (a few more details are given in Section 1.3). Even if limited to certain fields, the latter bound is a major milestone since it is quasi-linear in the input/output size. It has been extended in [41] to the case of degree conditions on x and y individually (other situations than the one with the total degree). However, it is unclear to us whether these approaches, which use a bit complexity model, could be exploited for general fields.

Despite these recent advances, we see that the algebraic complexity question of lowering the exponent of the complexity estimate for the resultant over a general field K remains a long-standing open problem.

The starting point of our progress is to notice that, at least for $d = 1$ and p and q with a particular shape, the approach of [40] can be improved, and a better complexity bound can be obtained. Indeed if $p = x - a$ and $q = f$ for a, f univariate in $\mathsf{K}[y]$, then the resultant can be obtained from the characteristic polynomial χ_a of the multiplication by a modulo f. For such a and f of degree n, let $c = (-1)^n e^n$ where e is the leading coefficient of f, then we have (see for instance [2, Thms 4.26 (resultant from the roots of p and q) & 4.69 (Stickelberger)]:

$$\mathrm{Res}_y(p, q) = c\,\chi_a. \tag{1}$$

It follows that the resultant can be computed generically with respect to a using $\tilde{O}(n^{(\omega+2)/3})$ arithmetic operations from the characteristic polynomial algorithm of [34, Sec. 10.1]. In this special case, the latter algorithm is the first one that allows to break the barrier $3/2$ in the exponent of n. The cost bound can be sligthly improved using rectangular matrix multiplication [34].

A key ingredient for obtaining the better estimate $\tilde{O}(n^{(\omega+2)/3})$ compared to $\tilde{O}(n^{2-1/\omega})$ in [40] for $d = 1$, is the possibility of setting up a baby steps/giant steps strategy from the powers of a modulo f [37]. One of the main difficulties that we overcome is the development of such a strategy for polynomials p and q having degree d and no special shape.

When d is not too large in relation to n our new algorithm also allows to cross the $3/2$ barrier in the exponent of n for the general resultant. As long as $d = O(n^{1/3})$, what we get is reconciled with the case $d = 1$, indeed we establish that the resultant of two sufficiently generic polynomials can be computed using $\tilde{O}(n^{(\omega+2)/3}d)$ arithmetic operations. One might also expect a slight improvement by using fast rectangular matrix multiplication [30, 29, 42]. This would require technical developments for adapting the core arithmetic on structured matrices on which we rely [6].

More precisely, we prove the following (Section 8):

Theorem 1.1. *Let $p, q \in \mathsf{K}[x, y]$ be of degree d in x and n in y. Except if the coefficients of p and q are on a certain hypersurface of $\mathsf{K}^{2(n+1)(d+1)}$, Algorithm* StructuredResultant *computes the resultant of p and q with respect to y using:*

- *$\tilde{O}(n^{(\omega+2)/3}d)$ arithmetic operations in K if $d = O(n^{1/3})$;*
- *$\tilde{O}(n^\theta d^\tau)$ arithmetic operations with $\theta = \frac{\omega^2-2}{3\omega-4}$ and $\tau = \frac{5\omega-6}{3\omega-4}$, otherwise.*

With the known bound $\omega < 2.372$ [1, 9, 42] and $d = O(n^{1/3})$, the cost of the algorithm is $O(n^{1.458}d)$. In particular, as long as $d = O(n^{0.47})$ our complexity estimate improves on the best previous one for generic polynomials over an arbitrary field, namely $\tilde{O}(n^{2-1/\omega}d)$ [40] (see Fig. 1 in Section 8).

1.1. Tools from previous works

We elaborate our algorithm from three complementary points of view. In this section we present the algorithmic ideas they each bring and that we combine.

1.1.1. Minor of the inverse and matrix fraction reconstruction

From this point on, we will rather use n to denote the dimension of the Sylvester matrix, which corresponds, in the context of Theorem 1.1, to polynomials p and q of y-degree $n/2$. In a more general way, let p, $q \in \mathsf{K}[x, y]$ be polynomials of x-degree bounded by d, and respective y-degrees n_p and n_q, with $n = n_p + n_q$. Given a polynomial $t \in \mathsf{K}[y]$, we denote by $t^{(j)}$ its coefficient in y^j (and by t_j the coefficient in x^j of a polynomial $t \in \mathsf{K}[x]$). The Sylvester matrix

$$S = \begin{bmatrix} p^{(n_p)} & & & q^{(n_q)} & & \\ p^{(n_p-1)} & \ddots & & q^{(n_q-1)} & \ddots & \\ \vdots & & p^{(n_p)} & \vdots & & q^{(n_q)} \\ p^{(0)} & & p^{(n_p-1)} & q^{(0)} & & q^{(n_q-1)} \\ & \ddots & \vdots & & \ddots & \vdots \\ & & p^{(0)} & & & q^{(0)} \end{bmatrix} \in \mathsf{K}[x]^{n\times n} \tag{2}$$

associated to p and q is formed by two adjacent Toeplitz matrices such that with $i = 1, \dots, n$, $S_{i,j} = p^{(n_p+j-i)}$ for $j = 1, \dots, n_q$, $S_{i,j+n_q} = q^{(n_q+j-i)}$ for $j = 1, \dots, n_p$, and the remaining entries are zero. The resultant $\mathrm{Res}_y(p, q) \in \mathsf{K}[x]$ of p and q is the determinant of S. In the same vein as [40] our algorithm reduces the computation of this determinant to the computation of the determinant of a smaller matrix, while controlling the x-degree of the latter. The Sylvester matrix is assumed to be invertible for $x = 0$.

Given a parameter $m \ll n$, chosen at the end for optimizing the overall cost, consider the projections $X = [\mathrm{I}_m\ 0]^\mathsf{T}$ and $Y = [0\ \mathrm{I}_m]^\mathsf{T}$ in $\mathsf{K}^{n\times m}$ where $\mathrm{I}_m \in \mathsf{K}^{m\times m}$ is the identity matrix (its dimension will be omitted when clear from context). The first step of the algorithm of [40] consists in computing sufficiently many terms of the power series expansion of $X^\mathsf{T} S^{-1} Y$. Coprime matrices $N, D \in \mathsf{K}[x]^{m\times m}$, with D nonsingular, such that

$$X^\mathsf{T} S^{-1} Y = ND^{-1} \tag{3}$$

are then deduced in a second step using matrix fraction reconstruction [3, 12]. For $n_p = n_q = n/2$, let $\lambda = 4\lceil n/(2m)\rceil$. Generically in p and q, λd terms of the expansion of $X^\mathsf{T} S^{-1} Y$ are sufficient for the computation of an $m\times m$ denominator matrix D of degree $\lambda d/2$ (Section 8.1). The total size of D is therefore $O(mnd)$, which for $m \ll n$ is below the previously known complexity bound $\tilde{O}(n^2 d)$ for the resultant. The computation of the truncated expansion of $X^\mathsf{T} S^{-1} Y$ can be obtained from four solutions of Sylvester linear systems modulo $x^{\lambda d}$, using $\tilde{O}(n \times \lambda d) = \tilde{O}(n^2 d/m)$ arithmetic operations [40]. By looking at the determinants of the denominator matrices in Eq. (3), exploiting genericity, and by involving appropriate properties of irreducible fractions [20, Lem. 6.5-9, p. 446], the desired resultant is then obtained from

$$\mathrm{Res}_y(p, q) = \det S = c \det D$$

for some nonzero $c \in \mathsf{K}$ (a scalar obtained separately at a negligible cost). The determinant of D is computed using a fast algorithm on polynomial matrices [38, 28]. This general strategy is suitable for handling more general structures than that of the Sylvester matrix, such as that of Toeplitz-like and Hankel-like matrices [25].

1.1.2. Characteristic polynomials and baby steps/giant steps approach

Equations of the type of Eq. (3) are at the heart of block Krylov-Wiedemann schemes for the computation of minimal or characteristic polynomials of scalar matrices, see [23] and references therein. For $A \in \mathsf{K}^{n\times n}$ and projections $U, V \in \mathsf{K}^{n\times m}$ ($1 \le m \le n$), the central core of these schemes is the computation of coprime matrices $N, D \in \mathsf{K}[x]^{m\times m}$, with D nonsingular, such that

$$U^\mathsf{T}(x\mathrm{I} - A)^{-1} V = \sum_{k\ge 0} U^\mathsf{T} A^k V x^{-k-1} = ND^{-1}. \tag{4}$$

The matrix fraction ND^{-1} is obtained from a truncated expansion using fraction reconstruction as mentioned previously, or by computing minimal polynomials of matrix sequences [23, 24]. The invariant factors of the denominator matrix D then provide information on those of $x\mathrm{I} - A$ [23, Thm 2.12]. In particular, for generic U and V, the characteristic polynomial $\det(x\mathrm{I} - A)$ can be recovered from the determinant of D as soon as it coincides with the minimal polynomial.

An advantage here is given by the shape of $x\mathrm{I} - A$ compared to the situation of Eq. (3) where the entries of S are general polynomials. Using the explicit form of the expansion of $(x\mathrm{I}-A)^{-1}$, a baby steps/giant steps strategy can be used from the powers of A to compute sufficiently many $U^\mathsf{T}A^kV$ terms. For $\lambda = 2\lceil n/m\rceil$ terms, consider $r = \lceil\lambda^{1/2}\rceil$, $s = \lceil\lambda/r\rceil$ and the precomputation of A^r. The $U^\mathsf{T}A^{i+rj}V$ terms for $0 \le i < r$ and $0 \le j < s$ can be computed by [23]:

- getting $U^\mathsf{T}A^i$ for $i = 0, 1, \ldots, r-1$ by repeated multiplications by A^T (baby steps); (5a)
- getting $A^{jr}V$ for $j = 0, 1, \ldots, s-1$ by repeated multiplications by A^r (giant steps); (5b)
- multiplying $(U^\mathsf{T}A^i)(A^{jr}V)$ for $i = 0, 1, \ldots, r-1$ and $j = 0, 1, \ldots, s-1$. (5c)

In relation with the special resultant case with $d = 1$ seen at Eq. (1), let $a, f \in \mathsf{K}[y]$ with $\deg f = n$, and let $A \in \mathsf{K}^{n\times n}$ be the matrix of multiplication by a modulo f in the basis $(1, y, \ldots, y^{n-1})$. The above baby steps/giants steps approach can be made efficient using modular polynomial operations. Generically in a (A is invertible, and its minimal polynomial and characteristic polynomial coincide), and using for technical reasons an expansion

$$U^\mathsf{T}(x\mathrm{I} - A)^{-1}V = \sum_{k\ge 0} -U^\mathsf{T}A^{-k-1}Vx^k = ND^{-1} \tag{6}$$

at zero rather than at infinity, this leads to a fast algorithm for computing the characteristic polynomial of a modulo f [34, Sec. 10.1]. It can be shown that λ terms of the expansion in Eq. (6) are sufficient for the reconstruction of a suitable description ND^{-1}. Further, these terms can be computed using $\tilde{O}(m^{(1-\omega)/2}n^{(\omega+1)/2})$ arithmetic operations, which is less than the estimate $\tilde{O}(n^2/m)$ of Section 1.1.1 for $d = 1$ as soon as $\omega < 3$.

1.1.3. Series solutions of polynomial linear systems

Consider $M \in \mathsf{K}[x]^{n\times n}$, $V \in \mathsf{K}[x]^{n\times m}$ $(1 \le m \le n)$ and $z \in \mathsf{K}[x]$ such that $\gcd(\det M, z) = 1$. For any given integer $\lambda \ge 0$, the high-order lifting method of [38] allows to compute the truncated z-adic expansion of $M^{-1}V$ modulo z^λ. For an integer $k \ge 0$, the z-adic expansion of $M^{-1}V$ is written in the form

$$M^{-1}V = \text{lower order terms} + z^kM^{-1}R_k \tag{7}$$

where R_k is a polynomial matrix called *residue* of V at order k. Noticing that computing the z-adic expansion of $M^{-1}R_k$ is a problem of the same type as computing that of $M^{-1}V$, the idea is to compute the expansion of $M^{-1}V$ recursively using residues of V at various orders.

We denote by $\mathsf{K}[x]_{<d}$ the set of polynomials in $\mathsf{K}[x]$ and degree less than d. If $\deg M \le \deg z = d$ and $V \in \mathsf{K}[x]^{n\times m}_{<d}$, then the K-linear map

$$\rho : \mathsf{K}[x]^{n\times m}_{<d} \to \mathsf{K}[x]^{n\times m}_{<d}$$

that sends V to $\rho(V) = R_1$ is well defined, and R_k is obtained from the functional power ρ^k as $\rho^k(V)$ (Lemmas 2.2 and 2.3).

A central point of the high-order lifting method is that the order of a residue can be efficiently increased from k to $k + i$, for an integer $i \ge 0$, using only two consecutive terms of the z-adic expansion of M^{-1}. These two terms, whose orders depend on i only, form a matrix $E^{(i)} \in \mathsf{K}[x]^{n\times n}_{<2d}$ called *high-order component* of M^{-1} that can be computed using $\tilde{O}(n^\omega d)$ arithmetic operations if $i \in O(n)$ [38, Prop. 12]. For two polynomials $f, g \in \mathsf{K}[x]$ with $\deg f \le 2d$, $\deg g \le d$ and $fg = \sum_{k=0}^{3d} h_kx^k$ (recall we write indices for coefficients in x of polynomials, as a distinction with superscripts for coefficients in y), let us denote by

$$f \odot g = h_d + h_{d+1}x + \ldots + h_{2d-1}x^{d-1} \tag{8}$$

the *middle product* operation. For univariate polynomial matrices F and G with appropriate dimensions and degrees at most $2d$ and d respectively, the middle product $F \odot G$ is defined to be the matrix obtained by extracting the middle coefficients of the entries of FG. Then we have (Lemma 2.5):

$$\rho^i(\rho^k(V)) = \rho^{k+i}(V) \equiv M(E^{(i)} \odot \rho^k(V)) \bmod z. \tag{9}$$

The ability to increase the residue orders thanks to high-order components leads to the following iteration [38, Sec. 8], in the style of the one of Keller-Gehrig for Krylov subspaces [27, Sec. 3]:

$$\rho^{2^i}([V, \rho(V), \rho^2(V), \ldots, \rho^{2^i-1}(V)]) = [\rho^{2^i}(V), \rho^{2^i+1}(V), \rho^{2^i+2}(V), \ldots, \rho^{2\cdot 2^i-1}(V)],\ i = 0, 1, \ldots \tag{10}$$

By considering $i = 0, 1, \ldots \lceil \log \lambda \rceil - 1$, this iteration allows to compute the residues $R_k = \rho^k(V)$ of V at all orders up to $\lambda - 1$. According to Eq. (9) only $\lceil \log \lambda \rceil$ high-order components of M^{-1} are required. Finally, we see from Eq. (7) that the coefficients of the z-adic expansion of $M^{-1}V$ modulo z^λ are obtained from the residues as $M^{-1}R_k \bmod z$ for $k = 0, 1, \ldots, \lambda - 1$. All the ingredients of the lifting needed for our algorithm are recalled in Section 2.

1.2. Overview of the contribution

Our resultant algorithm follows the line of Section 1.1.1 and is completely described and analyzed in Section 8. For technical reasons (simplification of the giant steps) we slightly modify the projections and rather consider the fraction $Y^\mathsf{T} S^{-1} X$. The different aspects of our contribution are about the reduction of the cost of the first step that computes $\lambda d = 4\lceil n/(2m) \rceil d$ terms of the x-adic expansion of $Y^\mathsf{T} S^{-1} X$. The other steps are treated in the same way as in [40] and are not discussed in this section.

Assuming that $\det S(0) \neq 0$ (see Section 8), we use the high-order lifting tools with $z = x^d$ and compute the z-adic expansion of $Y^\mathsf{T} S^{-1} X$ modulo z^λ. For $M = S$ and $V = X$, the high-order lifting method focuses on an expansion of $S^{-1}X$. If we consider that each coefficient of a power of x in the expansion has size $\Omega(n)$ elements in K, then the output has size $\Omega(n^2 d/m)$ and this lower bound will also apply to the time complexity of this approach. This lower bound on complexity is reached by the algorithm of [40], up to logarithmic factors. In order to get a better complexity estimate, we design a baby steps/giant steps version of the lifting which takes into consideration the left projection by Y. Based on the role played by A in Section 1.1.2, we introduce the high-order component $E^{(r)}$ of S^{-1} for $r = \lceil \lambda^{1/2} \rceil$. Rather than handling all the residues up to order $\lambda - 1$ as in Section 1.1.3, we consider $s = \lceil \lambda / r \rceil$ of them. We first implement *giant steps* and compute the residues

$$\rho^r(X), \rho^{2r}(X), \ldots, \rho^{(s-1)r}(X) \tag{11}$$

using a Keller-Gehrig iteration of the type of the one of Eq. (10) (Lemma 2.7). From Eq. (7) and for $0 \leq j \leq s - 1$, these residues satisfy

$$Y^\mathsf{T} S^{-1} X = \text{lower order terms} + z^{jr} Y^\mathsf{T} S^{-1} \rho^{jr}(X), \tag{12}$$

where we have taken $\rho^0(X) = X$. Our algorithm then obtain the target z-adic expansion of $Y^\mathsf{T} S^{-1} X$ in the form of s successive pieces of length r. Following Eq. (12), the piece that gives the coefficients of $z^{jr}, z^{jr+1}, \ldots, z^{(j+1)r-1}$ in the expansion of $Y^\mathsf{T} S^{-1} X$ is indeed the result of the multiplication modulo z^r, of the projection $Y^\mathsf{T} S^{-1}$ by the residue $\rho^{jr}(X)$. For taking advantage of fast polynomial matrix multiplication, these multiplications for $0 \leq j \leq s-1$ are performed by cutting $Y^\mathsf{T} S^{-1} \bmod z^r$ itself into r pieces. Obtaining these pieces corresponds to the *baby steps*.

This approach is detailed in Section 2, by highlighting the relations between block-Krylov and high-order lifting points of view. We give a template of the final expansion algorithm which at this stage does not take into account the structure of the matrices that are manipulated (Algorithm ProjectedExpansion).

Next, a main contribution is to exploit the fact that S is a polynomial Sylvester matrix and its consequences on the other matrices involved. The class of structure that we are facing is the one of *Toeplitz-like polynomial matrices*, we recall all necessary tools around this class in Section 3. Toeplitz-like matrices over a field are commonly handled using the notion of *displacement rank* [21]. The notion allows to have a concise matrix representation through which matrix arithmetic can be implemented efficiently [36]. Typically,

by extending the ΣLU form defined over fields [22], a polynomial Toeplitz-like matrix $T \in \mathsf{K}[x]^{n\times n}$ can be represented as

$$T(x) = \sum_{i=1}^{\alpha} L_i(x)U_i(x) \tag{13}$$

for a parameter α "small" compared to n, and where the L_i's and the U_i's are respectively lower and upper triangular polynomial Toeplitz matrices (Section 3). If α is minimal then α is precisely what is called displacement rank of T, and corresponds to the displacement rank over the field of rational fractions. We show in Section 4 that under genericity conditions on p and q (assumptions (A1) to (A3) in Section 8.2), the polynomial matrices involved in the high-order lifting with S are Toeplitz-like (the residues and the high-order components). They further have displacement rank at most $d + 2$ with a ΣLU representation as in Eq. (13) of degree at most d. Since these matrices have dimension n, using the structure is cost-effective when $d = o(n)$. This leads us to consider that $d < n$ for what follows.

The ΣLU form with factored terms in Eq. (13) is however not fully appropriate for the reduction modulo x^d (remember that $z = x^d$) and middle product operations as in Eq. (9). For example, deriving the ΣLU representation of T modulo x^d from the one of T may necessitate explicit matrix products to reconstitute the summands in Eq. (13). We show that a stronger representation can actually be used in our case, especially for the high-order components of S^{-1} (Section 4.2) and the residues (Section 4.3). Truncations and middle products are facilitated by the fact that for every i in Eq. (13), either L_i ot U_i can be chosen as a scalar matrix. Under the genericity conditions we have mentioned above, this representation is defined uniquely and is qualified as *canonical*. Moreover, a Toeplitz-like by Toeplitz-like matrix product generally leads to an increase in displacement rank (see Lemma 3.1). In contrast, the matrices involved in the lifting maintain the same structure and displacement rank. Canonical representations can thus be either computed directly or recovered by compressing results of products, thereby allowing to keep the costs contained. We describe in Section 5 how we can rely on canonical representations for all matrix operations involved.

Once efficient matrix arithmetic is available, we implement the giant steps for the computation of the residues of Eq. (11) in Section 6, then the expansion algorithm in Section 7.

From there, the complexity estimate for the whole resultant algorithm is established in Section 8, where we also specify the genericity hypotheses. The displacement rank that quantifies the structure on which we rely and from which we can benefit, is a function of the degree bound d. This is what determines the range $d = O(n^{0.471})$ for which our complexity estimate improves over previous ones.

1.3. Related questions: resultants, characteristic polynomials and bivariate ideals

As seen in a special case with Eq. (1), the resultant problem is related to the problem of computing characteristic polynomials in quotient algebras. This relation has been used in particular for the diamond product in [5], and for the resultant algorithms in [16, 41] which have already been quoted in the introduction.

The algorithm in [16] resulted in the first quasi-linear complexity bound over finite fields $\mathbb{F}$ for sufficiently generic p and q, with respect to the total degree. It relies on the *concise representation* of a Gröbner basis of the bivariate ideal $\mathcal{I} = \langle p, q\rangle$ [15]. Such a representation has size $\tilde{O}(n^2)$ elements in $\mathbb{F}$, and allows multiplication in $\mathsf{K}[x, y]/\mathcal{I}$ in quasi-linear time [15]. This fast multiplication then leads to an efficient reduction of the resultant problem to a bivariate modular composition problem, in turn reduced to a multivariate multipoint evaluation problem [16]. Thanks to efficient multipoint evaluation algorithms over finite fields [26, 4], a quasi-linear bit complexity bound is established for the resultant. This work has been extended to the case of generic polynomials of degree d in x and n in y in [41], by developing a method of multiplication in $\mathsf{K}[x, y]/\mathcal{I}$ which uses polynomial matrix division.

Our improvement for a general field K is based on a structured polynomial matrix formalism. We note however that the general approach we follow, as well as the characteristic polynomial algorithm of the case $d = 1$, can be interpreted in terms of operations on bivariate polynomials, see [40, Sec. 7] and [34, Sec. 1.6.2]. The denominator matrices D in Eqs. (3), (4) and (6) indeed give sets of m small degree polynomials in the corresponding ideals $\mathcal{I}$.

1.4. Model of computation and notations

Throughout this paper, K is an effective field. We analyze our algorithms by bounding the number of arithmetic operations from K required for large enough inputs. Addition, subtraction, multiplication and nonzero division are considered as unit cost operations. Our complexity bounds are often given as a function of ω, which is a feasible exponent for square matrix multiplication [1, 9, 42].

Given a bivariate polynomial $t \in \mathsf{K}[x, y]$, we can either view it as a polynomial in y, $t = \sum_j t^{(j)} y^j$ with $t^{(j)} \in \mathsf{K}[x]$ or as a polynomial in x, $t = \sum_i t_i x^i$ where $t_i \in \mathsf{K}[y]$. In what follows, $t_i^{(j)}$ denotes the constant coefficient for monomial $x^i y^j$. As we will sometimes use simultaneously x-adic and x^d-adic representations, we introduce in such situations a dot over the coefficient variable when it refers to the x-adic representation (Sections 4.2 and 4.3). If t is a power series in $\mathsf{K}[[x]]$ then the notation $t \bmod x^k$ for $k \geq 0$ indicates that the terms of degree k or higher are ignored.

The m-th canonical vector will be denoted by e_m, as its dimension is always clear from the context. For a matrix $M \in \mathsf{K}^{m\times n}$, and two sets of row and column indices $I \subseteq \{1 \dots m\}, J \subseteq \{1 \dots n\}$, we denote by $M_{I,J}$ the submatrix of M formed by the rows of indices in I and columns of indices in J. Similarly, $M_{i,k..l}$ will denote the submatrix of M formed by row i and columns comprised between k and l.

2. Baby steps/giant steps for high-order lifting

A key ingredient of our resultant algorithm is the high-order lifting method of Storjohann [38]. Throughout this section, M is a nonsingular matrix in $\mathsf{K}[x]^{n\times n}$, and $z \in \mathsf{K}[x]$ is of degree $d > 0$ such that $\gcd(z, \det M) = 1$ (the resultant algorithm uses $z = x^d$).

For a given $V \in \mathsf{K}[x]^{n\times m}$, the aim of the lifting is the efficient computation of parts of the z-adic expansion

$$M^{-1}V = B_0 + B_1 z + B_2 z^2 + \dots.$$

Two types of matrices play a central role for this computation: the residue terms (Definition 2.1) and the high-order components of the inverse (Eq. (20)). These notions, as well as all the elements which are the basis of the approach and that we need later on, are detailed in Section 2.1 where we essentially follow [38]. We however propose an adaptation of the definition of the residues, based on a linear map (Lemma 2.3), which allows us to highlight the relations between high-order lifting and Krylov iteration points of view.

For efficiency reasons we then also consider a left projection $U \in \mathsf{K}^{n\times m}$, and focus on computing parts of the expansion

$$U^{\mathsf{T}} M^{-1} V = H_0 + H_1 z + H_2 z^2 + \dots.$$

In section Section 2.2 we introduce Algorithm ProjectedExpansion for computing this expansion up to an arbitrary order, by using a baby steps/giant steps strategy. Our approach somewhat interpolates between the power series expansion algorithms of [38], and the block Krylov-Wiedemann approaches mentioned in Section 1.1.2.

At this stage we do not take into account the structure of the matrices that are manipulated. Algorithm ProjectedExpansion should be seen as a template of which our structured expansion algorithm of Section 7 is a specialization. We therefore delay the complexity analyses of the algorithms presented here to Sections 5, 6 and 7 where the structure of the matrices is detailed.

2.1. High-order lifting

Given any $h \in \mathsf{K}(x)$ whose denominator is coprime with z, we consider its z-adic expansion $h = h_0 + h_1 z + h_2 z^2 + \dots$ and define two operations. For an integer $k \geq 0$,

$$\lceil h \rceil_k = h_0 + h_1 z + \dots + h_{k-1} z^{k-1}$$

corresponds to the truncation operation, and

$$\lfloor h \rfloor_k = h_k + h_{k+1} z + h_{k+2} z^2 \dots.$$

denotes the quotient of the division of h by z^k. These notations are extended to matrices entry-wise. The core of high-order lifting is to compute the expansion of $M^{-1}V$ recursively from intermediate terms called *residues* (the residues play a role analogous to the one of residual terms in e.g. numerical iterative refinement [14, Chap. 12]).

Definition 2.1. *([38, Dfn. 5]) For $V \in \mathsf{K}[x]^{n\times m}$ and an integer $k \geq 0$, the matrix $R_k \in \mathsf{K}[x]^{n\times m}$ such that*

$$M^{-1}V = \lceil M^{-1}V \rceil_k + z^k M^{-1} R_k \tag{14}$$

is called the residue of V at order k.

With appropriate degree conditions, obtaining the residue at order $k = 1$ may be viewed as the application of a linear map ρ.

Lemma 2.2. *If $d = \deg z$ and $\deg M \leq d$ then the residue at order 1 induces the K-linear map*

$$\begin{array}{rccl} \rho: & \mathsf{K}[x]^{n\times m}_{<d} & \to & \mathsf{K}[x]^{n\times m}_{<d} \\ & V & \mapsto & \lfloor V - M\lceil M^{-1}V\rceil_1 \rfloor_1. \end{array} \tag{15}$$

Proof. From Definition 2.1, $\rho(V)$ is the residue of V at order 1. The map is well defined since from [38, Cor. 10] we know that with the assumptions the residue has degree less than d; ρ is a K-linear map by linearity of the operations $\lceil\cdot\rceil_1$ and $\lfloor\cdot\rfloor_1$. □

Then, the residue at order k is obtained from the functional power ρ^k.

Lemma 2.3. *Assume $\deg M \leq \deg z = d$. If $k \geq 0$ and $V \in \mathsf{K}[x]^{n\times m}_{<d}$ then $\rho^k(V)$ is the residue of V at order k.*

Proof. We have $\rho^0(V) = V$ and $\rho(V)$ is the residue at order 1. We proceed by induction and assume that ρ_k is the residue at order k. For $k+1$ we get $M^{-1}\rho^{k+1}(V) = M^{-1}\rho(\rho^k(V)) = \lfloor M^{-1}\rho^k(V)\rfloor_1$, where the second equality is from Eq. (14) (which can be written $\lfloor M^{-1}V\rfloor_k = M^{-1}R_k$) with $k = 1$. The induction hypothesis and Eq. (14) at order k then leads to $M^{-1}\rho^{k+1}(V) = \lfloor\lfloor M^{-1}V\rfloor_k\rfloor_1 = \lfloor M^{-1}V\rfloor_{k+1}$, which proves the assertion. □

If $M(x) = x\mathrm{I} - A$ with A nonsingular in $\mathsf{K}^{n\times n}$, $z = x$, and $V \in \mathsf{K}^{n\times m}$, then the residue of V is $\rho(V) = A^{-1}V$ and $\rho^k(V) = A^{-k}V$. The expansion (see Eq. (6) in Section 1.1.2)

$$(x\mathrm{I} - A)^{-1}V = \sum_{k\geq 0} -A^{-k-1}Vx^k$$

is generalized as follows. Taking $C_0 = \lceil M^{-1}\rceil_1$, for $V \in \mathsf{K}[x]^{n\times m}_{<d}$ the z-adic expansion of $M^{-1}V$ is

$$M^{-1}V = \sum_{k\geq 0} \lceil C_0\rho^k(V)\rceil_1 z^k. \tag{16}$$

Since $\rho^k(V)$ is the residue at order k, we indeed know from Eq. (14) that $\lceil C_0\rho^k(V)\rceil_1 = \lceil M^{-1}\rho^k(V)\rceil_1$ is the coefficient of z^k in the z-adic expansion of $M^{-1}V$.

We see from Eq. (16) that the role of the matrix powers in Krylov type methods can now be assigned to the powers of ρ, hence we now focus on how to increase the order of a given residue. Using the notation

$$M^{-1}V = \sum_{k\geq 0} B_k z^k$$

for the z-adic expansion of $M^{-1}V$, from Eq. (16) we obtain

$$\rho^k(V) = \lceil MB_k\rceil_1, \tag{17}$$

which reduces the computation of $\rho^{k+i}(V)$ for a given $i \geq 0$, to the computation of B_{k+i}. Note that another formulation of Eq. (17) could be [38, Thm. 9]:

$$\rho^k(V) = \lfloor -MB_{k-1}\rfloor_1. \tag{18}$$

Using Eq. (14) for writing the z-adic expansion of $M^{-1}\rho^k(V)$ at order $k+i$ we further have that B_{k+i} satisfies

$$M^{-1}\rho^k(V) = \lceil M^{-1}\rho^k(V)\rceil_{k+i} + z^{k+i}B_{k+i} + \dots. \tag{19}$$

Then the ingredient given by Lemma 2.4 below is that, knowing $\rho^k(V)$, only a few terms of the expansion of M^{-1} are sufficient in Eq. (19) for computing B_{k+i}, and therefore then $\rho^{k+i}(V)$. These few terms form what is called a *high-order component* of M^{-1} [38, Sec. 6]. A high-order component is a piece of length 2 of the z-adic expansion of M^{-1} defined as follows. Writing $M^{-1} = \sum_{i\geq 0} C_i z^i$, we let $E^{(0)} = zC_0$ and for $i \geq 1$ we take

$$E^{(i)} = C_{i-1} + C_i z \in \mathsf{K}[x]^{n\times n}_{<2d}. \tag{20}$$

For two polynomial matrices F and G with appropriate dimensions, remember also the definition of the middle product operation $F \odot G = \lceil\lfloor FG\rfloor_1\rceil_1$ (see Eq. (8)). Then, using $E^{(i)}$, B_{k+i} is computed as follows from $\rho^k(V)$.

Lemma 2.4. *Assume* $\deg M \leq \deg z = d$, *and* $\deg V < d$. *Let* $M^{-1}V = \sum_{k\geq 0} B_k z^k$. *For* $k, i \geq 0$ *we have* $E^{(i)} \odot \rho^k(V) = B_{k+i}$.

Proof. We first claim that $E^{(i)} \odot \rho^k(V)$ is the coefficient of z^i in the z-adic expansion of $M^{-1}\rho^k(V)$. For $i = 0$, $E^{(0)} \odot \rho^k(V) = \lceil C_0\rho^k(V)\rceil_1$ is the coefficient of z^0. For $i \geq 1$, the claim follows from [38, Thm. 8]. Then we conclude using Eq. (14) since indeed, the coefficient of z^i in the expansion of $M^{-1}\rho^k(V)$ is the coefficient of z^{k+i} in the expansion of $M^{-1}V$. □

In the way we have anticipated we can now compute $\rho^{k+i}(V)$ from $\rho^k(V)$, this operation is a main brick in [38, Algo. 3] for computing selected parts of the expansion of $M^{-1}V$. Note that from Lemma 2.4 we could write $B_{k+i} = E^{(0)} \odot \rho^{k+i}(V)$, hence the effect of the multiplication by M at Step 2 of Algorithm 2.1 is to "discard" $E^{(0)}$ from B_{k+i}.

Algorithm 2.1 FurtherResidue

Input: The high-order component $E^{(i)}$ of M^{-1} and the residue $\rho^k(V)$ for some $V \in \mathsf{K}[x]^{n\times m}_{<d}$, with $k, i \geq 0$
Output: The residue $\rho^{k+i}(V)$
1: $B \leftarrow E^{(i)} \odot \rho^k(V)$
2: **return** $\lceil MB\rceil_1$

Lemma 2.5. *Assume* $\deg M \leq \deg z = d$, *and* $\deg V < d$. *Given the residue* $\rho^k(V)$ *of order* $k \geq 0$, $i \geq 0$, *and the high-order component* $E^{(i)}$ *of* M^{-1}, *Algorithm* FurtherResidue *computes the residue* $\rho^{k+i}(V)$ *of order* $k+i$.

Proof. From Lemma 2.4 we know that B is the coefficient of z^{k+i} in the z-adic expansion of $M^{-1}V$, and we conclude using Eq. (17). □

For computing the expansion of $M^{-1}V$ efficiently as in [38, Algo. 3] as well as in our baby steps/giant steps approach, the application of Lemma 2.5 relies on the availability of high-order components of few selected orders (an amount logarithmic in the length of the expansion). The high-order component at some order can be computed by a sort of binary powering from components at lower orders [38, Algo. 1]. Consider indeed two high-order components $E^{(i)}$ and $E^{(j)}$. The residue ρ^i of the identity matrix involved in $E^{(i)} = (E^{(0)} \odot \rho^{i-1}(\mathrm{I})) + (E^{(0)} \odot \rho^i(\mathrm{I}))$ (Lemma 2.4), combined with ρ^{j-1}, ρ^j that are found in $E^{(j)}$ (after having discarded $E^{(0)}$), allows to construct the high-order component at order $i+j$.

Lemma 2.6. *Assume* $\deg M \leq \deg z = d$. *Given the high-order components* $E^{(i)}$ *and* $E^{(j)}$ *of* M^{-1}, *with* $i \geq 0$ *and* $j \geq 1$, *Algorithm* ComponentProduct *computes the high-order component* $E^{(i+j)}$ *of* M^{-1}.

Proof. From Eq. (17) we have $R_{j-1} = \rho^{j-1}(\mathrm{I})$ and $R_j = \rho^j(\mathrm{I})$, then Lemma 2.4 allows to conclude. □

It can be noticed that Algorithm 2.2 is slightly different from the procedure of Storjohann in [38, Algo. 1]. The application of Eq. (18) rather than Eq. (17) at Step 1 could be used in order to compute $E^{(i+j+1)}$ from $E^{(i)}$ and $E^{(j)}$.

Algorithm 2.2 ComponentProduct

Input: Two high-order components $E^{(i)}$ and $E^{(j)} = C_{j-1} + C_j z$ of M^{-1}, with $i \geq 0$ and $j \geq 1$
Output: The high-order component $E^{(i+j)}$ of M^{-1}
1: $R_{j-1} \leftarrow \lceil MC_{j-1} \rceil_1$
2: $R_j \leftarrow \lceil MC_j \rceil_1$
3: **return** $(E^{(i)} \odot R_{j-1}) + (E^{(i)} \odot R_j)z$

2.2. *Baby steps/giant steps*

We now apply the tools of Section 2.1 for computing parts of the z-adic expansion $U^\mathsf{T} M^{-1} V = H_0 + H_1 z + H_2 z^2 + \dots$, where the left projection U is in $\mathsf{K}^{n\times m}$. Algorithm ProjectedExpansion is designed as an extension to the lifting context of the three phases (5a)-(5c) of the block Krylov approach.

First we focus on the giant steps, that is on the extension of step (5b). For some given $r, s \geq 0$, the purpose is to compute the residues $\rho^{jr}(V)$ for $j = 0, 1, \dots s-1$. Following Storjohann [38, Sec. 8], the combination of Algorithm FurtherResidue and Algorithm ComponentProduct allows to compute such a sequence of residues *à la* Keller-Gehrig for the computation of Krylov subspaces [27, Sec. 3]. This is what Algorithm FurtherResidues does, computing s residues in $\lceil \log_2 s \rceil$ recursive steps. Taking $\varrho = \rho^r$, we proceed with an iteration of the type:

$$\varrho^{2^i}([V, \varrho(V), \varrho^2(V), \dots, \varrho^{2^i-1}(V)]) = [\varrho^{2^i}(V), \varrho^{2^i+1}(V), \varrho^{2^i+2}(V), \dots, \varrho^{2\cdot 2^i-1}(V)],\ i = 0, \dots, l-1, \tag{21}$$

which generalizes the Krylov iteration

$$A^{2^i}[v, Av, A^2 v, \dots, A^{2^i-1} v] = [A^{2^i} v, A^{2^i+1} v, A^{2^i+2} v, \dots, A^{2\cdot 2^i-1} v]$$

for $A \in \mathsf{K}^{n\times n}$ and $v \in \mathsf{K}^n$.

Algorithm 2.3 FurtherResidues

Input: The high-order component $E^{(r)}$ of order r, $V \in \mathsf{K}[x]^{n\times m}_{<d}$, and $s \in \mathbb{N}_{>0}$
Output: $\mathcal{R} = [V\, \rho^r(V)\, \rho^{2r}(V)\ \dots \rho^{(s-1)r}(V)] \in \mathsf{K}[x]^{n\times(sm)}_{<d}$
1: $E \leftarrow E^{(r)}$
2: **for** $i = 0, \dots, \lceil \log_2 s \rceil - 1$ **do**
3: $\quad k \leftarrow 2^i$
4: $\quad$ ▹ *New coefficients of the z-adic expansion of $M^{-1}V$, Lemma 2.4*
$\quad \mathcal{B} \leftarrow E \odot [V\, \rho^r(V)\, \rho^{2r}(V)\ \dots, \rho^{(k-1)r}(V)]$
5: $\quad$ ▹ *New residues, Eq. (17)*
$\quad [\rho^{kr}(V)\, \rho^{(k+1)r}(V)\, \rho^{(k+2)r}(V)\ \dots\ \rho^{(2k-1)r}(V)] \leftarrow \lceil M\mathcal{B} \rceil_1$
6: $\quad$ ▹ *Obtaining $E^{(2^{i+1}r)}$*
$\quad$ **if** $2k < s$ **then** $E \leftarrow$ ComponentProduct(E, E)
7: **return** $[V\, \rho^r(V)\, \rho^{2r}(V)\ \dots \rho^{(s-1)r}(V)]$

Lemma 2.7. *Assume* $\deg M \leq \deg z = d$. *Algorithm* FurtherResidues *is correct.*

Proof. From Lemma 2.6, for any given $0 \leq i \leq \lceil \log_2 s \rceil - 1$, at Step 4 we have that E is the high-order component computed at Step 6 for $i-1$, hence $E = E^{(2^{(i-1)}r + 2^{(i-1)}r)} = E^{(kr)}$. Then from Lemma 2.5, considering $B_{kr+jr} = E \odot \rho^{jr}(V)$ for $0 \leq j \leq k-1$, we know that $\lceil MB_{kr+jr} \rceil_1$ is $\rho^{kr+jr}(V)$, which shows that the residues of orders $kr, kr+r, \dots, kr+(k-1)r$ are computed from those of orders $0, r, \dots, (k-1)r$ at Step 5. This gives the residues $\rho^{jr}(V)$ for every $0 \leq j \leq (s-1)$ as soon as $k \geq s/2$. □

With $r = 1$, the iteration of Eq. (21) is dual to the one used by Storjohann for computing a truncated expansion of $M^{-1}V$ [38, Sec. 8]. As soon as $\rho^j(V)$ is known for every $0 \leq j \leq (s-1)$, then the truncated expansion $M^{-1}V \bmod z^s$ can be deduced using Eq. (16).

Algorithm 2.4 ProjectedExpansion

Input: $M \in \mathsf{K}[x]^{n\times n}$, $z \in \mathsf{K}[x]$ with $\deg M \le \deg z = d$ and $\gcd(z, \det M) = 1$,
$U \in \mathsf{K}^{n\times m}$, $V \in \mathsf{K}[x]^{n\times m}_{<d}$, $r, s \in \mathbb{N}_{>0}$
Output: $\lceil U^{\mathsf{T}} M^{-1}(x) V \rceil_{rs}$

1: **for** $i = 0, \ldots, r-1$ **do** ▷ *Baby steps, compare to (5a)*
2: $\quad D^{(i)} \leftarrow U^{\mathsf{T}} E^{(i)}$
3: Compute $E^{(r)}$
4: $[P^{(0)}, \ldots, P^{(s-1)}] \leftarrow$ FurtherResidues$(E^{(r)}, V, s)$ ▷ *Giant steps, compare to (5b)*
5: **for** $i = 0, \ldots, r-1$ **do** ▷ *Final products, compare to (5c)*
6: $\quad$ **for** $j = 0, \ldots, s-1$ **do**
7: $\quad\quad H_{i+rj} \leftarrow D^{(i)} \odot P^{(j)}$
8: **return** $H_0 + zH_1 + z^2 H_2 + \ldots + z^{rs-1} H_{rs-1}$

By relying on Algorithm FurtherResidues for the giant steps, we now have all the necessary ingredients for combining the approach of [23] for Krylov sequences with lifting techniques.

Proposition 2.8. *Let M be a nonsingular matrix in $\mathsf{K}[x]^{n\times n}$, and $z \in \mathsf{K}[x]$ be of degree d such that* $\deg M \le d$ *and* $\gcd(z, \det M) = 1$. *Given block projections $U \in \mathsf{K}^{n\times m}$, $V \in \mathsf{K}[x]^{n\times m}_{<d}$, and positive integers r, s, Algorithm* ProjectedExpansion *computes the expansion of $U^{\mathsf{T}} M^{-1}(x) V$ truncated at order rs.*

Proof. From $P^{(0)} = V = \rho^0(V)$ and using Lemma 2.5, arriving at Step 5 the algorithm has computed $D^{(i)} = U^{\mathsf{T}} E^{(i)}$ for $0 \le i \le r-1$, and $P^{(j)} = \rho^{rj}(V)$ for $0 \le j \le s-1$. Since U has scalar coefficients we have $(U^{\mathsf{T}} E^{(i)}) \odot \rho^{rj}(V) = U^{\mathsf{T}}(E^{(i)} \odot \rho^{rj}(V))$, hence from Lemma 2.4 we know that $H_{i+rj} = U^{\mathsf{T}} B_{i+rj}$. With $0 \le i \le r-1$ and $0 \le j \le s-1$ the output gives therefore the rs appropriate terms of the expansion of $U^{\mathsf{T}} M^{-1} V$. □

Algorithm 2.4 mimics for a general $M \in \mathsf{K}[x]^{n\times n}$ what has been seen in Section 1.1.2 for $M = x\mathrm{I} - A$. Every step will be detailed in next sections and optimized by taking into considerations the specific structure and properties of Sylvester matrices.

3. Matrices with a displacement structure

We characterize the structure of the matrices which we handle using the customary notion of displacement rank [21]; the reader may refer e.g. to [13] and [36] for comprehensive overviews of the domain, and detailed introductions to the tools we use. We especially rely on [6] for the integration of asymptotically fast matrix multiplication in the algorithms.

In this paper we define the displacement rank of a matrix from the specific Stein operator

$$\Delta_{m,n} : A \in \mathsf{K}^{m\times n} \mapsto A - Z_m A Z_n^{\mathsf{T}} \tag{22}$$

where $Z_m \in \mathsf{K}^{m\times m}$ is the lower shift matrix $(\delta_{i,j+1})_{1\le i,j\le m}$ ($\delta_{a,b}$ is 1 if $a = b$ and 0 otherwise), and Z_n^{T} is the transpose of $Z_n \in \mathsf{K}^{n\times n}$ (we will simply write Δ and Z when the dimensions are clear from the context). The *displacement rank* of A is defined as the rank of $\Delta(A)$, and A is called *Toeplitz-like* if its displacement rank is "small" compared to m and n. In particular, Toeplitz and Sylvester matrices are Toeplitz-like matrices of displacement rank at most 2 (see Section 4 for the Sylvester case).

If A has displacement rank α (or bounded by α), then a pair of matrices $G \in \mathsf{K}^{m\times\alpha}$ and $H \in \mathsf{K}^{n\times\alpha}$ such that

$$\Delta_{m,n}(A) = GH^{\mathsf{T}} \tag{23}$$

is called a *generator of length* α for A, and G and H are respectively called *left and right generator*. As soon as α is small enough, such generators (G, H) with size $(m+n)\,\alpha \ll mn$ are used as concise representations of Toeplitz-like matrices. Our central procedures for matrix operations take generators as input and return

generators, in place of the corresponding matrices (Sections 5 and 6). When needed, matrices can be explicitly and uniquely recovered from their generator based representation. Indeed, the displacement operator Δ is invertible [36, Thm. 4.3.2], which means that for given G and H, Eq. (23) viewed as an equation in A has a unique solution

$$A = \sum_{i=1}^{\alpha} \mathcal{L}(G_{*,i})\mathcal{U}(H_{*,i}) \tag{24}$$

where for $u \in \mathsf{K}^m$ and $v \in \mathsf{K}^n$, $\mathcal{L}(u) \in \mathsf{K}^{m\times m}$ is the lower triangular Toeplitz matrix whose first column is u, and $\mathcal{U}(v) \in \mathsf{K}^{m\times n}$ is the upper triangular Toeplitz matrix whose first row is v^T. The expression in Eq. (24) is called a ΣLU representation of A [22].

The multiplication of a Toeplitz matrix in $\mathsf{K}^{n\times n}$ by a vector in K^n reduces to univariate polynomial multiplication [36, Sec. 2.4], and polynomial multiplication is computed in softly linear time over any algebra [7]. From the ΣLU representation in Eq. (24) we therefore deduce that the multiplication of A or A^T by a scalar vector may be computed from 2α products of Toeplitz matrices by vectors. The resulting cost is $\tilde{O}(\max(m,n)\alpha)$.

An important property of Toeplitz-like matrices that we use for our algorithm is the fact that their product remains Toeplitz-like. As shown by the following, if A and B have respective displacement ranks bounded by α and β then AB has displacement rank at most $\alpha+\beta+1$.

Lemma 3.1 ([36, Thm 1.5.4])**.** *For $A \in \mathsf{K}^{l\times m}$ and $B \in \mathsf{K}^{m\times n}$ the product AB satisfies*

$$\Delta(AB) = \Delta(A)B + Z_l A Z_m^\mathsf{T}\Delta(B) - Z_l A e_m {e_m}^\mathsf{T} B Z_n^\mathsf{T}.$$

Proof. From the second item in [36, Thm 1.5.4] we have $\Delta_{l,n}(AB) = \Delta_{l,m}(A)B + Z_l A\nabla(B)$ where $\nabla(B) = Z_m^\mathsf{T}B - BZ_n^\mathsf{T}$. The assertion of the lemma follows by noticing that $\nabla(B) = Z_m^\mathsf{T}\Delta(B) - e_m e_m^\mathsf{T} B Z_n^\mathsf{T}$. □

Lemma 3.1 may not lead to a generator of minimal length for AB. When AB is known to have displacement rank less than $\alpha+\beta+1$, a shorter generator can be recovered by a compression mechanism [36, Sec. 4.6]. This will be the case in Section 5 where the compression method we use on polynomial generators also guarantees specific properties for the resulting shorter generator.

We rely on the following complexity bound for the cost of applying a Toeplitz-like matrix to a dense matrix.

Theorem 3.2 ([6, Theorem 1.2])**.** *Let $A \in \mathsf{K}^{l\times m}$ be Toeplitz-like given by a generator of length $\alpha \le \min(l,m)$ and $B \in \mathsf{K}^{m\times\beta}$. The product $AB \in \mathsf{K}^{m\times\beta}$ can be computed using $\tilde{O}(\max(l,m)\max(\alpha,\beta)\min(\alpha,\beta)^{\omega-2})$ arithmetic operations in K.*

The following corollary expands the scope of Theorem 3.2 to the cases of structured matrix by dense matrix products when the structured matrix is applied on the right and/or when the generator representation is larger than the size of the matrix (which may happen in our algorithms for extreme choices of parameters).

Corollary 3.3. *Let $A \in \mathsf{K}^{l\times m}$ be Toeplitz-like given by a generator of length $\alpha = O(\max(l,m))$, $B \in \mathsf{K}^{m\times\beta}$ and $C \in \mathsf{K}^{\beta\times l}$. The products $AB \in \mathsf{K}^{l\times\beta}$ and $CA \in \mathsf{K}^{\beta\times m}$ can be computed using*

$$\tilde{O}\left(\max(l,m)\max(\alpha,\beta)\min(\alpha,\beta)^{\omega-2}\right)$$

arithmetic operations in K.

Proof. The conditions of Theorem 3.2 are recovered by padding the generators to appropriate dimensions. Let A be represented by (G,H) such that $GH^\mathsf{T} = \Delta(A)$ as in Eq. (23), and $\gamma = \max(l,m,\alpha)$. Construct $G' = [G^\mathsf{T}\ 0]^\mathsf{T} \in \mathsf{K}^{\gamma\times\alpha}, H' = [H^\mathsf{T}\ 0]^\mathsf{T} \in \mathsf{K}^{\gamma\times\alpha}, B' = [B^\mathsf{T}\ 0]^\mathsf{T} \in \mathsf{K}^{\gamma\times\beta}$ and consider $A' \in \mathsf{K}^{\gamma\times\gamma}$ such that $\Delta(A') = G'(H')^\mathsf{T}$. Then since $\gamma = O(\max(l,m))$, $AB = (A'B')_{1..l,*}$ can be computed from G', H' and B' in $\tilde{O}(\max(l,m)\max(\alpha,\beta)\min\alpha,\beta)^{\omega-2})$ by Theorem 3.2. Note that G', H', and B' are free to construct, and that there is no need to explicitly have A'.

If A is Toeplitz-like with generator (G,H), then A^T has a similar structure with generator (H,G), hence the product CA can be performed in a similar way. □

Note that by taking $\beta = 1$ in Corollary 3.3 we find the cost given earlier for the matrix-vector multiplication.

The previous definitions and properties are valid for any commutative field and can thus be applied to polynomial matrices, if seen over the field $\mathsf{K}(x)$. Note that in this case, generators of minimal length can always be taken as polynomial matrices themselves. Indeed, if $T \in \mathsf{K}[x]^{m\times n}$ of degree has displacement rank α (over the field of rational functions), then there exists a unimodular matrix $U \in \mathsf{K}[x]^{n\times n}$ such that $\Delta(T)U = [G\ 0]$ with $G \in \mathsf{K}[x]^{m\times\alpha}$ (consider for example the Hermite normal form of $\Delta(T)$, see e.g. [35, Ch. II, Sec. 6]). Hence a polynomial generator of minimal length for T is given by $\Delta(T) = G(U^{-1})_{1..\alpha,*}$.

Throughout the paper, the Toeplitz-like polynomial matrices we handle are represented using polynomial generators. When the degree of the matrix is less than d, the generator we use will also be of degree less than d (see Sections 4.2 and 4.3). By extending Eqs. (23) and (24) we thus consider matrices $T \in \mathsf{K}[x]^{m\times n}_{<d}$ such that

$$\Delta_{m,n}(T) = G(x)H(x)^{\mathsf{T}},$$

where $G \in \mathsf{K}[x]^{m\times\alpha}_{<d}$ and $H \in \mathsf{K}[x]^{n\times\alpha}_{<d}$, whose ΣLU representation is

$$T = \sum_{i=1}^{\alpha} \mathcal{L}(G_{*,i}(x))\mathcal{U}(H_{*,i}(x)).$$

The products involving such matrices are treated from Theorem 3.2 and Corollary 3.3 as follows.

Theorem 3.4. *Let $T \in \mathsf{K}[x]^{l\times m}_{<d}$ be Toeplitz-like represented by a generator of degree less than d and length $\alpha = O(n)$ with $n = \max(l, m)$. For matrices $V \in \mathsf{K}[x]^{m\times\beta}_{<d}$ and $W \in \mathsf{K}[x]^{\beta\times l}_{<d}$, the products TV and WT can be computed using $\tilde{O}(\mathsf{MM}_{n,d}(\alpha,\beta))$ arithmetic operation in K with*

$$\mathsf{MM}_{n,d}(\alpha,\beta) = nd\max(\alpha,\beta)\min(\alpha,\beta)^{\omega-2}. \tag{25}$$

Proof. The matrix products can be computed by running the algorithm which underpins Corollary 3.3 over $\mathsf{K}[x]/(x^{2d-1})$. In our case, [6, Thm. 1.2] relies on the algorithm `mul` of [6, Section 5.2], followed by the polynomial products in [6, Thm. 3.1, Cor. 3.2]. The only arithmetic operations performed in the case of the displacement operator $\Delta_{l,m}$ from Eq. (22) are additions, subtractions, multiplications, and possibly tests to zero, which can be handled in $\mathsf{K}[x]/(x^{2d-1})$ using $\tilde{O}(d)$ operations in K [7]. □

In particular, taking $\beta = 1$ in *Eq.* (25), we consider that the product of T or T^{T} by a vector with entries in $\mathsf{K}[x]_{<d}$ has cost $\tilde{O}(\mathsf{MM}_{n,d}(\alpha, 1)) = \tilde{O}(n\alpha d)$. We will keep using the notation MM in the complexity analyses when computing structured matrix products. Beyond the product, the class of nonsingular Toeplitz-like matrices is closed under inversion [21], [36, Thm. 1.5.3]. Our approach exploits this property in the special case of Sylvester matrices that are studied in detail in Section 4.

4. Displacement structure of Sylvester matrices and its residues and high-order components

For $p, q \in \mathsf{K}[x, y]$ the resultant is $\mathrm{Res}_y(p, q) = \det(S)$ where S is the associated Sylvester matrix with entries in $\mathsf{K}[x]$. By taking $z = x^d$, the first step of our resultant algorithm proceeds to do high-order lifting by implementing Algorithm ProjectedExpansion of Section 2 with $M = S$ and z-adic expansions. The algorithms in Section 2 are given for general polynomial matrices. In this section we show that in the case of a polynomial Sylvester matrix of degree at most d, the residues (Definition 2.1) and high-order components (Eq. (20)) involved are Toeplitz-like with displacement rank at most $d + 2$. This allows later in the paper to represent these matrices by their generators, as we explained in Section 3. Since matrices have dimension n, the bound on the displacement rank is useful when d is relatively smaller than n; the content of this section remains however correct for arbitrary degrees.

The polynomial case relies on properties of scalar Sylvester matrices and their inverses that are given in Section 4.1. However, operations such as truncation and middle product need a special attention. Especially consider the question of truncating a matrix given by its generator. Let typically $A \in \mathsf{K}[x]^{n\times n}$ be of degree $2d$

and represented by a generator (G, H), with G and H of degree d in $\mathsf{K}[x]^{n\times\alpha}$ such that $\Delta(A) = GH^{\mathsf{T}}$. Without additional assumptions, we are not aware of a way to compute a generator for the truncation $\lceil A\rceil_1$ of A modulo x^d which does not involve the reconstruction of a dense $n \times n$ matrix, and/or an expensive compression mechanism.

Our solution is obtained thanks to the fact that, in the Sylvester case, high-order components and residues can be represented by generators with a specific shape that makes the operations much easier. Indeed, it turns out that we can use a variation of the simple fact that if e.g. G (resp. H) is scalar, then a generator for $\lceil A\rceil_1$ is $(G, \lceil H\rceil_1)$ (resp. $(\lceil G\rceil_1, H)$). These specific generators which we call *canonical* are introduced in Section 4.2 for the coefficients of the z-adic expansion of S^{-1} and the high-order components, and in Section 4.3 for the residues.

4.1. Sylvester matrices over K

Here we detail the structure properties we need for Sylvester matrices and their inverses over K, the polynomial matrix case is treated using this in next sections. We consider polynomials $p, q \in \mathsf{K}[y]$ of respective degrees n_p and n_q, with $n = n_p + n_q$. The entries of the Sylvester matrix $S \in \mathsf{K}^{n\times n}$ associated to p and q are, in row $1 \le i \le n$: $S_{i,j} = p^{(n_p+j-i)}$ for $j = 1, \dots, n_q$, $S_{i,j+n_q} = q^{(n_q+j-i)}$ for $j = 1, \dots, n_p$, and zero otherwise (see Eq. (2)).

We study the structure of S by noticing that it can be viewed as a matrix of multiplication in a quotient algebra (we do not know whether this remark has been used previously). We omit the elementary proof of the following.

Lemma 4.1. *Consider the reverse polynomials $a = y^{n_p}p(1/y)$ and $b = y^{n_q}q(1/y)$, and define $f = b - y^{n_q}a$. If $p^{(0)} \neq 0$ then* $\deg f = n$ *and S is the matrix of multiplication by a modulo f in the basis* $(1, y, \dots, y^{n-1})$.

This characterization of Sylvester matrices allows us to highlight what makes these matrices special in the class of Toeplitz-like matrices. The fact that their displacement rank does not increase by raising to power makes the connection with the algorithm in [34] for $d = 1$ (see Eq. (1)), and gives an intuition about the displacement structure of high-order components and residues in the next sections. The characterization also allows us to directly deduce the structure of the inverse matrix, and more easily write the generators of Propositions 4.4 and 4.7 in terms of polynomial coefficients.

Matrices of modular multiplication are in particular Toeplitz-like matrices, the following recalls recurrence relations on its rows and columns, and the form of their generators (see Eq. (23) in Section 3). For $t \in \mathsf{K}[y]$, we let $v(t) = [t^{(0)}\ t^{(1)}\ \dots\ t^{(n-1)}]^{\mathsf{T}} \in \mathsf{K}^n$ be the vector of the coefficients of $t \bmod y^n$ in the basis $(1, y, \dots, y^{n-1})$.

If T is the matrix of multiplication by t modulo f then the j-th column of T is $v(y^{j-1}t \operatorname{rem} f)$ (the rem operation returns the remainder of the Euclidean division). Further, if $f^{(0)} \neq 0$ then for any integer i one has

$$(t \operatorname{rem} f)^{(i)} = (yt \operatorname{rem} f)^{(i+1)} - c(f^{(i+1)}/f^{(0)}) \tag{26}$$

where c is the coefficient of degree 0 of $yt \operatorname{rem} f$. We can first deduce by replacing t with $y^j t$ for $j = 0, \dots, n-2$ in Eq. (26) that the rows of T follow the recursion

$$e_i^{\mathsf{T}}T - e_n^{\mathsf{T}}T_{i,n} = e_{i+1}^{\mathsf{T}}TZ - \frac{f^{(i)}}{f^{(0)}}e_1^{\mathsf{T}}TZ. \tag{27}$$

This recursion is used in Section 7 for reconstructing a whole submatrix of a multiplication matrix from only two of its rows. On other hand, we have a similar relation between the columns of T:

$$Te_{j+1} = ZTe_j - \frac{T_{n,j}}{f^{(n)}}v(f)\,,\ j = 1, \dots, n-1. \tag{28}$$

From there we deduce a generator expression for T [36, Sec. 2.7], which can therefore also be applied to S with a and f as in Lemma 4.1.

Lemma 4.2. *Given $f \in \mathsf{K}[y]$ of degree n and $t \in \mathsf{K}[y]_{<n}$, the matrix $T \in \mathsf{K}^{n\times n}$ of multiplication by t modulo f satisfies*

$$\Delta(T) = v(t)\, e_1^\mathsf{T} - v(f)\, w_t^\mathsf{T}, \tag{29}$$

where $w_t = ZT^\mathsf{T} e_n / f^{(n)}$.

Proof. The first column of $\Delta(T)$ is equal to the first column of T, that is $v(t)$. By definition, w_t^T is the last row of T, shifted to the right and divided by $f^{(n)}$; its first entry is thus 0, which ensures that the first column of $v(t)\, e_1^\mathsf{T} - v(f)\, w_t^\mathsf{T}$ is $v(t)$. For $j = 1, \dots, n-1$, the $(j+1)$-th column of $\Delta(T)$ is such that $\Delta(T)e_{j+1} = Te_{j+1} - ZTe_j$, hence from Eq. (28) we have $\Delta(T)e_{j+1} = -T_{n,j}v(f)\,/f^{(n)}$ and $T_{n,j}/f^{(n)}$ is precisely the $(j+1)$-th entry of w_t. □

Since S is a multiplication matrix, we know that it is invertible if and only if a is invertible modulo f, and in that case S^{-1} is the matrix of multiplication by $g \in \mathsf{K}[y]_{<n}$ such that $ag \equiv 1 \bmod f$. The following then gives useful relations between the rows 1, $n_q + 1$ and n of S^{-1}.

Lemma 4.3. *Assume that S is invertible with $p^{(0)} \neq 0$, and note that $q^{(n_q)} \neq 0$ by degree hypothesis. Viewing S as the matrix of multiplication by a modulo f, let $g \in \mathsf{K}[y]_{<n}$ be such that $ag \equiv 1 \bmod f$. We have*

$$e_1^\mathsf{T} S^{-1} = g^{(0)} e_1^\mathsf{T} - f^{(0)} w^\mathsf{T} \tag{30}$$

$$e_{n_q+1}^\mathsf{T} S^{-1} = \frac{1 - a^{(0)} g^{(0)}}{f^{(0)}} e_1^\mathsf{T} + a^{(0)} w^\mathsf{T} \tag{31}$$

$$e_n^\mathsf{T} S^{-1} Z^\mathsf{T} = f^{(n)} w^\mathsf{T}, \tag{32}$$

where $w = ZS^{-\mathsf{T}} e_n / f^{(n)}$ with the notation $S^{-\mathsf{T}}$ for the transpose of S^{-1}.

Proof. Since $f^{(0)} = q^{(n_q)}$ and $f^{(n)} = -p^{(0)}$, all the quantities are well defined. As the first row of $\Delta(S^{-1})$ is the first row of S^{-1}, Eq. (30) is a consequence of Lemma 4.2, and Eq. (32) is the definition of w in the same lemma. For Eq. (31) we use the fact that S is a Sylvester matrix. Considering $e_1^\mathsf{T} S S^{-1} = e_1^\mathsf{T}$ we have $a^{(0)} e_1^\mathsf{T} S^{-1} + b^{(0)} e_{n_q+1}^\mathsf{T} S^{-1} = e_1^\mathsf{T}$, then we note that $b^{(0)} = f^{(0)}$ and conclude using Eq. (30). □

4.2. Structure of high-order components

We move to the structured polynomial matrix case. Let now $p, q \in \mathsf{K}[x, y]$ be of degree at most $d > 0$ in x and respective degrees n_q and n_p in y, with $n = n_q + n_p$. The associated Sylvester matrix S is in $\mathsf{K}[x]^{n\times n}_{\leq d}$. Taking $z = x^d$ and assuming that $\gcd(\det S, z) = 1$, we write $S^{-1} = \sum_{k\geq 0} C_k z^k$ with $C_k \in \mathsf{K}[x]^{n\times n}_{<d}$. We thereafter also use the name *slices* for the coefficients of the z-adic expansion. From Eq. (20), the high-order components are formed by two consecutive slices. This section is devoted to the description of the slices, and in doing so, the one of the high-order components.

Whenever an ambiguity between x-adic and z-adic expansions may appear, we distinguish them by denoting with a dot the x-adic coefficients of a series (hence of a polynomial). In particular we have $S^{-1} = \sum_{i\geq 0} \dot{C}_i x^i$ with $\dot{C}_i \in \mathsf{K}^{n\times n}$ and $C_k = \sum_{i=0}^{d-1} \dot{C}_{kd+i} x^i$. Further, we use a bar notation for the scalar matrices involved in the generators to mark their difference from general polynomial matrices.

Rather than polynomials of $\mathsf{K}[y]$ as in Lemma 4.1 we now have bivariate polynomials

$$a = y^{n_p} p(1/y),\ b = y^{n_q} q(1/y),\ f = b - y^{n_q} a, \tag{33}$$

and we write e.g. $f = \sum_{j=0}^{d} \dot{f}_j x^j$ with $\dot{f}_j \in \mathsf{K}[y]_{\leq n}$. Recall that for bivariate polynomials, indices name coefficients in x while superscripts are used for coefficients in y. Assuming that $p^{(0)}$ is nonzero, S is the matrix of multiplication by a modulo f. Then the identities of Lemma 4.3 are considered on power series, assuming $f^{(0)}$ is invertible. Using that S is invertible we know there exists $g \in \mathsf{K}(x)[y]_{<n}$ such that

$$ag \equiv 1 \bmod f.$$

Since $\gcd(\det S, z) = 1$, the x-adic and z-adic expansions of g can be considered in the form $g = \sum_{k\geq 0} g_k z^k = \sum_{i\geq 0} \dot{g}_i x^i$ where the g_k's in $\mathsf{K}[x, y]$ have respective degrees less than d in x and n in y, and the $\dot{g}_i$ are in $\mathsf{K}[y]_{<n}$.

We also assume that the constant terms $q_0^{(n_q)}$ and $p_0^{(0)}$, of $q^{(n_q)}$ and $p^{(0)}$ respectively, are nonzero. Hence the expansions of $f^{(0)}$ and $f^{(n)}$ are well defined and we write

$$w = ZS^{-\mathsf{T}}e_n/f^{(n)} = \sum_{k\geq 0} w_k z^k = \sum_{i\geq 0} \dot{w}_i x^i \in \mathsf{K}[[x]]^n \quad (\dot{w}_i = 0 \text{ for } i < 0). \tag{34}$$

The following proposition describes the displacement structure of S^{-1} and the specific representation we take for its z-adic slices. We keep the notation previously used by associating with a polynomial t in y over an arbitrary domain of coefficients, the vector $v(t)$ of the coefficients of t mod y^n in the basis $(1, y, \ldots, y^{n-1})$.

Proposition 4.4. *Assume that the constant terms of* $\det S$, $p^{(0)}$ *and* $q^{(n_q)}$ *in* $\mathsf{K}[x]$ *are nonzero. Let* $\bar{F} \in \mathsf{K}^{n\times(d+1)}$ *be the matrix whose* j*-th column is* $v(\dot{f}_j)$*. For any* $k \geq 0$*, the slice* C_k *of* S^{-1} *is Toeplitz-like (over the field* $\mathsf{K}(x)$*) with displacement rank at most* $d+2$ *and one of its generators is given by*

$$\Delta(C_k) = v(g_k)\, e_1^{\mathsf{T}} - \bar{F}\, W_k^{\mathsf{T}} \in \mathsf{K}[x]_{<d}^{n\times n}, \tag{35}$$

where the j*-th column of* $W_k \in \mathsf{K}[x]_{<d}^{n\times(d+1)}$ *is* $\sum_{i=0}^{d-1} \dot{w}_{kd-(j-1)+i} x^i$*. A generator in this form is called canonical. The matrix* W_k *can be fully constructed in* $O(nd^2)$ *operations from its first and last column, which are the coefficients* w_k *and* w_{k-1} *of the* z*-adic expansion of* w*.*

Proof. Since S^{-1} is the matrix of multiplication by g modulo f, Eq. (29) gives $\Delta(S^{-1}) = v(g)\, e_1^{\mathsf{T}} - v(f)\, w^{\mathsf{T}}$, hence for $i \geq 0$ we get $\Delta(\dot{C}_i) = v(\dot{g}_i)e_1^{\mathsf{T}} - \bar{F}[\dot{w}_i\ \dot{w}_{i-1}\ \cdots\ \dot{w}_{i-d}]^{\mathsf{T}}$. Combining the $\dot{C}_j$'s into slices of size d, we obtain the z-adic coefficient in Eq. (35). We can check that W_k is fully determined by its first and last column. Indeed, every column of W_k is a sum of d vectors among the $\dot{w}_{(k-1)d+i}$ for $0 \leq i \leq 2d-1$, each multiplied by a distinct power of x. All these vectors appear as coefficient vectors either in the first column of W_k which is the coefficient w_k of the z-adic expansion of w, or in the last column which is w_{k-1}; the cost bound is given by the size of W_k. □

High-order components have generators directly given by those of the slices. For $k \geq 0$, the high-order component $E^{(k)}$ of S^{-1} for $z = x^d$ is indeed a sum of two slices (Eq. (20)). From Proposition 4.4 we can write

$$\Delta(E^{(k)}) = v(g_{k-1} + g_k x^d)e_1^{\mathsf{T}} - \bar{F}\,(W_{k-1}^{\mathsf{T}} + W_k^{\mathsf{T}} x^d) \in \mathsf{K}[x]_{<2d}^{n\times n}, \tag{36}$$

which gives a generator in canonical form for E. Generators in canonical form are uniquely defined and have properties that will be useful for lowering the computational cost. We remark that the first entry of w is zero (see Eq. (34)) and the first column of a canonical generator is the first column of the matrix itself. This will be exploited by separating the computation of first columns from the computation of the remaining parts of the generators. The fact that generators are polynomials only on one side allows to directly represent a truncated matrix using truncations of parts of its generator. Further, we are going to take advantage of the structure of the W_k's by restricting computations to only two of their columns.

Remark 4.5. *Note that* $\bar{F}$ *and* W_k *in Eq.* (35) *do not necessarily have full rank, which means that the slice may have displacement rank less than* $d+2$*. Genericity assumptions on* p *and* q *ensure that* $\bar{F}$ *has rank* $d+1$ *(see Section 8), which is used in Section 5 for the efficient computation of canonical generators for matrix middle products.*

4.3. Structure of residues

We work under the same assumptions as in Section 4.2 and study the displacement structure of residues (Definition 2.1). Since we apply high-order lifting for an expansion of $Y^{\mathsf{T}}S^{-1}X$, where $Y = [0\ \ \mathrm{I}_m]^{\mathsf{T}}$, $X = [\mathrm{I}_m\ \ 0]^{\mathsf{T}}$ and $1 \leq m \leq n$ (see Section 7), only residues of the type $\rho^k(\mathrm{I})$ as in Algorithm ComponentProduct and $\rho^k(V) = \rho^k(X)$ as in Algorithm FurtherResidues are involved for integers $k \geq 0$. From Definition 2.1 we also see that $\rho^k(X) = \rho^k(\mathrm{I})X$, therefore noticing that any $n\times n$ matrix M satisfies $\Delta(MX) = \Delta(M)X$ (Eq. (22)) we deduce that

$$\Delta(\rho^k(X)) = \Delta(\rho^k(\mathrm{I}))X.$$

This allows us to limit ourselves in this section, to the description of canonical generators for residues $\rho^k(\mathrm{I})$ of the identity matrix. From Eq. (17), these matrices are obtained as truncated products of S and slices of S^{-1}: $\rho^k(\mathrm{I}) = \lceil S\, C_k \rceil_1$. The following describes the displacement structure for the scalar summands of this product. We use the notation of Section 4.2 for $a, b \in \mathsf{K}[x, y]$ as in Eq. (33) and w as in Eq. (34), as well as the dot convention for the x-adic coefficients of expansions and polynomials.

Lemma 4.6. *Assume that the constant terms of* $\det S$, $p^{(0)}$ *and* $q^{(n_q)}$ *in* $\mathsf{K}[x]$ *are nonzero. For* $i, j \in \mathbb{N}$ *the product* $\dot{S}_j \dot{C}_i \in \mathsf{K}^{n\times n}$ *is Toeplitz-like with displacement rank at most* $d+1$. *One of its generators is given by*

$$\Delta(\dot{S}_j \dot{C}_i) = \left(\dot{S}_j \dot{C}_i e_1\right) e_1^{\mathsf{T}} + \sum_{\substack{l=0\\ l\neq j}}^{d} v^{(l,j)} \dot{w}_{i-l}^{\mathsf{T}} \tag{37}$$

where $v^{(l,j)} = v(\dot{a}_l \dot{b}_j - \dot{a}_j \dot{b}_l) \in \mathsf{K}^n$ *for* $l \in \mathbb{N}$.

Proof. We describe the generator for the product using Lemma 3.1:

$$\Delta(\dot{S}_j \dot{C}_i) = \Delta(\dot{S}_j)\dot{C}_i + Z\dot{S}_j \left(Z^{\mathsf{T}} \Delta(\dot{C}_i) - e_n e_n^{\mathsf{T}} \dot{C}_i Z^{\mathsf{T}}\right). \tag{38}$$

From Lemma 4.2 we have that $\Delta(S) = v(a)\, e_1^{\mathsf{T}} + v(f)\, e_{n_q+1}^{\mathsf{T}}$ and hence $\Delta(\dot{S}_j) = v(\dot{a}_j) e_1^{\mathsf{T}} + v(\dot{f}_j) e_{n_q+1}^{\mathsf{T}}$. Since S^{-1} is the matrix of multiplication by g modulo f, one can derive the x-adic coefficients $e_1^{\mathsf{T}} \dot{C}_i$ and $e_{n_q+1}^{\mathsf{T}} \dot{C}_i$, of $e_1^{\mathsf{T}} S^{-1}$ and $e_{n_q+1}^{\mathsf{T}} S^{-1}$, by applying Lemma 4.2 on power series. Here we have used the assumptions for having nonzero coefficients at $x = 0$ hence the existence of the expansions. From Eqs. (30) and (31) and for some $\bar{v} \in \mathsf{K}^n$, the first term of the sum in Eq. (38) can be written as

$$\Delta(\dot{S}_j)\dot{C}_i = \bar{v} e_1^{\mathsf{T}} - v(\dot{a}_j) \sum_{l=0}^{d} \dot{f}_l^{(0)} \dot{w}_{i-l}^{\mathsf{T}} + v(\dot{f}_j) \sum_{l=0}^{d} \dot{a}_l^{(0)} \dot{w}_{i-l}^{\mathsf{T}}.$$

On other hand, Lemma 4.2 on power series gives $\Delta(\dot{C}_i) = v(\dot{g}_i)\, e_1^{\mathsf{T}} + \sum_{l=0}^{d} v(\dot{f}_l) \dot{w}_{i-l}^{\mathsf{T}}$. Therefore from Eq. (32), the term being multiplied by $Z\dot{S}_j$ in Eq. (38) is

$$Z^{\mathsf{T}} \Delta(\dot{C}_i) - e_n e_n^{\mathsf{T}} \dot{C}_i Z^{\mathsf{T}} = Z^{\mathsf{T}} v(\dot{g}_i)\, e_1^{\mathsf{T}} - \sum_{l=0}^{d} Z^{\mathsf{T}} v(\dot{f}_l) \dot{w}_{i-l}^{\mathsf{T}} - \sum_{l=0}^{d} \dot{f}_l^{(n)} e_n \dot{w}_{i-l}^{\mathsf{T}},$$

and Eq. (38) becomes

$$\Delta(\dot{S}_j \dot{C}_i) = \bar{G}\, [e_1\ \dot{w}_i\ \dot{w}_{i-1}\ \ldots\ \dot{w}_{i-d}]^{\mathsf{T}}$$

with a matrix $\bar{G} \in \mathsf{K}^{n\times(d+2)}$ that we now study. The first column of $\bar{G}$ is the first column of $\Delta(\dot{S}_j \dot{C}_i)$, that is $\dot{S}_j \dot{C}_i e_1$. For $l = 0, \ldots d$, the remaining columns of $\bar{G}$ can be expressed as

$$\bar{G} e_{l+2} = -\dot{f}_l^{(0)} v(\dot{a}_j) + \dot{a}_l^{(0)} v(\dot{f}_j) - Z\dot{S}_j \left(Z^{\mathsf{T}} v(\dot{f}_l) + \dot{f}_l^{(n)} e_n\right).$$

From

$$Z\dot{S}_j Z^{\mathsf{T}} = \dot{S}_j - \left(v(\dot{a}_j) e_1^{\mathsf{T}} + v(\dot{f}_j) e_{n_q+1}^{\mathsf{T}}\right),$$

we get

$$\bar{G} e_{l+2} = -\dot{f}_l^{(0)} v(\dot{a}_j) + \dot{a}_l^{(0)} v(\dot{f}_j) - \dot{S}_j v(\dot{f}_l) + v(\dot{a}_j) f_l^{(0)} + v(\dot{f}_j) e_{n_q+1}^{\mathsf{T}} v(\dot{f}_l) - \dot{f}_l^{(n)} Z\dot{S}_j e_n,$$

and since $e_{n_q+1}^{\mathsf{T}} v(\dot{f}_l) = \dot{b}_l^{(n_q)} - \dot{a}_l^{(0)}$ (one has $f = b - y^{n_q} a$), we arrive at

$$\bar{G} e_{l+2} = \dot{b}_l^{(n_q)} v(\dot{f}_j) - \dot{S}_j v(\dot{f}_l) - \dot{f}_l^{(n)} Z\dot{S}_j e_n. \tag{39}$$

Then consider each of the three summand vectors in this equation from the corresponding polynomials modulo y^n:

$$\begin{aligned}
\dot{b}_l^{(n_q)} v(\dot{f}_j) &= v\Big(\dot{b}_l^{(n_q)}\left(\dot{b}_j - y^{n_q}(\dot{a}_j \mod y^{n_p})\right)\Big),\\
\dot{S}_j v(\dot{f}_l) &= v\Big(\dot{a}_j\left(\dot{b}_l \mod y^{n_q}\right) + \dot{b}_j\left(\dot{b}_l^{(n_q)} - (\dot{a}_l \mod y^{n_p})\right)\Big),\\
\dot{f}_l^{(n)} Z \dot{S}_j e_n &= v\Big(-\dot{a}_l^{(n_p)} y^{n_p}(\dot{b}_j \mod y^{n_q})\Big).
\end{aligned}$$

By viewing Eq. (39) on polynomials modulo y^n this allows to assert that $\bar{G}e_{l+2} = v(\dot{a}_l\dot{b}_j - \dot{a}_j\dot{b}_l)$. Hence $\bar{G} = [\dot{S}_j\dot{C}_i e_1 \; v^{(0,j)} \; v^{(1,j)} \; \cdots \; v^{(d,j)}]$, which by noticing that $v^{(j,j)} = 0$ concludes the proof. □

Lemma 4.6 reveals a structure similar to the one of the slices in Proposition 4.4: the first column can be separated from the remaining ones; the left remaining part of the generator does not depend on the considered order in the expansion of S^{-1}; the right part of the generator is given by the z-adic coefficient w_k of w. From there we define a canonical representation for the residues which retains these properties.

Proposition 4.7. *Assume that the constant terms of* $\det S$, $p^{(0)}$ *and* $q^{(n_q)}$ *in* $\mathsf{K}[x]$ *are nonzero, and for indices* $i, j \in \mathbb{N}$ *consider the* $v^{(i,j)}$*'s as in Lemma 4.6. For any* $k \geq 0$ *and* $z = x^d$*, the residue* $\rho^k(\mathrm{I}) = \lceil SC_k \rceil_1$ *(see Eq. (17)) is Toeplitz-like (over the field* $\mathsf{K}(x)$*) with displacement rank at most* $d+1$*. One of its generators is given by*

$$\Delta(\rho^k(\mathrm{I})) = \lceil SC_k \rceil_1 e_1 e_1^\mathsf{T} + L\bar{W}_{k-1}^\mathsf{T} \in \mathsf{K}[x]_{<d}^{n\times n}, \tag{40}$$

where the l-th column of $L \in \mathsf{K}[x]_{<d}^{n\times d}$ *is* $\sum_{i=0}^{d-1} x^i \sum_{j=0}^{i} v^{(i+l-j,j)}$ *and the l-th column of* $\bar{W}_{k-1} \in \mathsf{K}^{n\times d}$ *is* $\dot{w}_{kd-l}$*. A generator in this form is called canonical. The left part L of the generator does not depend on k and can be computed using* $\tilde{O}(nd^2)$ *operations.*

Proof. By looking at the x-adic coefficients of $\lceil SC_k \rceil_1$ we can write

$$\Delta(\rho^k(\mathrm{I})) = \sum_{i=0}^{d-1} x^i \sum_{j=0}^{i} \Delta\left(\dot{S}_j\dot{C}_{kd+i-j}\right).$$

From Lemma 4.6 and since w has its first entry zero we have

$$\Delta(\rho^k(\mathrm{I})) = \rho^k(\mathrm{I}) e_1 e_1^\mathsf{T} + \delta,$$

where $\delta = \Delta(\rho^k(\mathrm{I}))ZZ^\mathsf{T}$ has first column zero. Hence it remains to study the structure of δ which gives the last $n-1$ columns of $\Delta(\rho^k(\mathrm{I}))$. From Lemma 4.6 (omitting to write $l \neq j$ since $v^{(j,j)} = 0$) and by substituting l by $i-j-l$ we obtain

$$\delta = \sum_{i=0}^{d-1} x^i \sum_{j=0}^{i} \sum_{l=0}^{d} v^{(l,j)} \dot{w}_{kd+i-j-l}^\mathsf{T} = \sum_{i=0}^{d-1} x^i \sum_{j=0}^{i} \sum_{l=i-j-d}^{i-j} v^{(i-j-l,j)} \dot{w}_{kd+l}^\mathsf{T},$$

then by swapping the sums the contribution of w can be factored out:

$$\delta = \sum_{l=-d}^{d-2} \left(\sum_{i=0}^{d-1} x^i \sum_{\substack{j=0\\ j\leq i-l, j\geq i-l-d}}^{i} v^{(i-j-l,j)} \right) \dot{w}_{kd+l}^\mathsf{T}. \tag{41}$$

This sum can be divided into two parts, for $l < 0$ and $l \geq 0$. We show that the latter sum is zero. It is indeed given by

$$\sum_{l=0}^{d-2} \left(\sum_{i=0}^{d-1} x^i \sum_{\substack{j=0\\ j\leq i-l,\; j\geq i-l-d}}^{i} v^{(i-j-l,j)} \right) \dot{w}_{kd+l}^\mathsf{T} = \sum_{l=0}^{d-2} \left(\sum_{i=l}^{d-1} x^i \sum_{j=0}^{i-l} v^{(i-j-l,j)} \right) \dot{w}_{kd+l}^\mathsf{T}. \tag{42}$$

Now notice that $v^{(i,i)} = 0$ and $v^{(i,j)} + v^{(j,i)} = 0$. It follows that for all l and i the summands of $\sum_{j=0}^{i-l} v^{(i-j-l,j)}$ cancel each other out (one summand is zero for even values of $i - l$), and the sum in Eq. (42) is zero.

By substituting l by $-l$, the nonzero terms in Eq. (41) then give $\delta = L\bar{W}_{k-1}^{\mathsf{T}}$ where for $l = 1, \ldots, d$ the l-th column of $L \in \mathsf{K}[x]_{<d}^{n\times d}$ is

$$\sum_{i=0}^{d-1} x^i \sum_{j=0, j\geq i+l-d}^{i} v^{(i+l-j,j)},$$

and $\bar{W}_{k-1}$ is as asserted (the constraints in the sum over j have vanished as $v^{(i+l-j,j)} = 0$ for $i + l - j > d$). The matrix L can then be computed in time $\tilde{O}(nd^2)$ from $O(d^2)$ products of the $\dot{a}_i$'s by the $\dot{b}_j$'s modulo y^n. □

Remark 4.8. *The columns of $\bar{W}_{k-1}$ in Proposition 4.7 are the scalar coefficient vectors of the z-adic coefficient w_{k-1}. It follows from Proposition 4.4 that the last d columns (the first one is e_1) of the right generator for $\rho^k(\mathrm{I})$ are the linearization of the last column of the right generator for the slice C_k. As L can be precomputed once and used for all residues (Section 6.2), the computation of the canonical generator for ρ^k (I) from the one for C_k is essentially the computation of its first column.*

5. Structured middle and truncated products

Our specialization of high-order lifting to the Sylvester case represents all high-order components of S^{-1} and residues by their canonical generators as in Sections 4.2 and 4.3. In this section we show that these representations allow to lower the cost of the two central matrix operations on which we rely: middle and truncated products. We work with $z = x^d$ under the assumptions of Propositions 4.4 and 4.7, and consider that $0 < d < n$ (the displacement rank structure does not directly enable faster operations for degrees that reach the dimension). We keep using a bar notation for the scalar matrices involved in the generators.

Typically, consider a high-order component E of S^{-1} at some arbitrary order. From Eq. (36) we know that E satisfies

$$\Delta(E) = v_E e_1^{\mathsf{T}} + \bar{F}\, W_E^{\mathsf{T}} \in \mathsf{K}[x]_{<2d}^{n\times n}, \tag{43}$$

with $v_E \in \mathsf{K}[x]^n$, $\bar{F} \in \mathsf{K}^{n\times(d+1)}$ and $W_E \in \mathsf{K}[x]^{n\times(d+1)}$. Consider also a residue $R = \rho^k(\mathrm{I})$ at some other arbitrary order k, from Eq. (40) we have

$$\Delta(R) = v_R e_1^{\mathsf{T}} + L\, \bar{W}_R^{\mathsf{T}} \in \mathsf{K}[x]_{<d}^{n\times n}, \tag{44}$$

where $v_R \in \mathsf{K}[x]^n$, $L \in \mathsf{K}[x]^{n\times d}$, and $\bar{W}_R \in \mathsf{K}^{n\times d}$. Then a central brick of the high-order lifting approach is the computation of the middle product

$$C = E \odot R \in \mathsf{K}[x]_{<d}^{n\times n},$$

where, according to Lemma 2.4, C is a coefficient of the z-adic expansion of S^{-1}. Hence using the canonical form Eq. (35) again, we know there exists a matrix $W_C \in \mathsf{K}[x]_{<d}^{n\times(d+1)}$ such that

$$\Delta(C) = v_C e_1^{\mathsf{T}} + \bar{F} W_C^{\mathsf{T}} \in \mathsf{K}[x]_{<d}^{n\times n}, \tag{45}$$

with $v_C \in \mathsf{K}[x]_{<d}^n$. For the computation of C, we exploit the fact that the generator parts $\bar{F}$ and $\bar{W}_R^{\mathsf{T}}$ are scalar matrices, which allows us to perform the middle product without resorting to a change of representation (Lemma 5.2). Further, several middle products will follow one another. We therefore ensure that the resulting C is itself represented by its canonical generator, which by the way also avoids the increase of the sizes of the representations (inherent, in general, to the product of structured matrices, see Section 3).

We note that the first column of a left canonical generator (e.g. v_E, v_R, v_C) is the first column of the matrix itself. In addition, the last $n - 1$ columns of the displaced matrices of Eqs. (43) to (45) are associated with the "W" parts $W_E, \bar{W}_R, W_C$ of the generators (whose first rows are zero). This leads us to separate the computation of first columns of generators from the computation of their W parts. In Section 5.1 we start by focusing on the computation of the right generator part W_C of the middle product as in Eq. (45);

Lemma 5.1 actually deals with a slightly more general situation that is required later for the concatenation of several products. An additional advantage is that, according to Proposition 4.4, the right generator for a slice is determined by its first and last column. This allows us to further decrease the exponent of d in the complexity bound, and compute these two columns of W_C using $\tilde{O}(\mathsf{MM}_{n,d}(d,1))$ operations with $\mathsf{MM}_{n,d}$ from Eq. (25). This is essentially the cost of multiplying a matrix of displacement rank d in $\mathsf{K}[x]^{n\times n}_{<d}$ by a vector in $\mathsf{K}[x]^{n}_{<d}$. Here the left part $\bar{F}$ of the generator for the slices will be assumed to have rank $d+1$, which corresponds to generic situations (see Remark 4.5).

The whole generator for C is deduced in Section 5.2, also in time $\tilde{O}(\mathsf{MM}_{n,d}(d,1))$ (Lemma 5.2). We will use for that (computation of the first column), and in some other places, the fact that for a vector b, the middle product $E \odot b$ does not cause any difficulty compared to the matrix middle product. Indeed, $E \odot b$ can simply be computed from the regular product Eb, by extracting the middle coefficients.

Besides the middle product, high-order lifting involves the truncated product operation as for instance at Step 5 of Algorithm FurtherResidues or Steps 1 and 2 of Algorithm ComponentProduct. These truncated products are used for the computation of residues. From Remark 4.8 the right generator for a residue $\lceil SC \rceil_1$ is directly known from that of C, and in the same way as for a slice of the inverse, the first column can be computed separately. Combining this with the middle product, in Section 5.2 we derive the cost bound $\tilde{O}(\mathsf{MM}_{n,d}(d,1))$ for high-order components handling (Lemma 5.3).

5.1. Middle product: computation of the right generator

Given a high-order component $E \in \mathsf{K}[x]^{n\times n}_{<2d}$ of S^{-1}, we consider the computation of a right generator for a middle product $E \odot \mathcal{R}$, for $\mathcal{R} \in \mathsf{K}[x]^{n\times c}_{<d}$ a residue or a concatenation or several residues. The latter case is addressed for performing the giant steps which we will discuss in Section 6.

We let $\mathcal{E}$ be a matrix in $\mathsf{K}^{c\times s}$ whose columns are s distinct canonical vectors $e_{i_1}, \ldots, e_{i_s}$ such that $1 \le i_1, \ldots i_s \le c$, and $\bar{\mathcal{W}}_{\mathcal{R}}$ be a matrix in $\mathsf{K}^{c\times d}$ whose submatrix $(\bar{\mathcal{W}}_{\mathcal{R}})_{I,*} \in \mathsf{K}^{s\times d}$ is zero for $I = \{i_1, \ldots, i_s\}$. Then generalizing Eq. (44), we consider $\mathcal{R}$ such that

$$\Delta(\mathcal{R}) = V_{\mathcal{R}}\mathcal{E}^{\mathsf{T}} + L\bar{\mathcal{W}}_{\mathcal{R}}^{\mathsf{T}} \in \mathsf{K}[x]^{n\times c}_{<d}, \tag{46}$$

with $V_{\mathcal{R}} \in \mathsf{K}[x]^{n\times s}_{<d}$. For $c = n$ and $s = 1$ we are in the case of a unique residue as in Eq. (44). We assume that the middle product $\mathcal{B} = E \odot \mathcal{R}$ satisfies

$$\Delta(E \odot \mathcal{R}) = V_{\mathcal{B}}\mathcal{E}^{\mathsf{T}} + \bar{F}\mathcal{W}_{\mathcal{B}}^{\mathsf{T}} \tag{47}$$

with $V_{\mathcal{B}} \in \mathsf{K}[x]^{n\times s}_{<d}$ and $\mathcal{W}_{\mathcal{B}} \in \mathsf{K}[x]^{c\times(d+1)}_{<d}$, and focus on the computation of the first and last column of $\mathcal{W}_{\mathcal{B}}$. Note that assuming the generator form as in Eq. (47) is appropriate for covering the case $s = 1$. This indeed generalizes the canonical form for a single slice as in Eq. (45). Focusing on the first and last column of the generator part is to be put in correspondence with the last assertion of Proposition 4.4.

From Lemma 3.1, for the product $E\mathcal{R}$ we have

$$\Delta(E\mathcal{R}) = \Delta(E)\mathcal{R} + ZEZ^{\mathsf{T}}\Delta(\mathcal{R}) - ZEe_n e_n^{\mathsf{T}}\mathcal{R}Z^{\mathsf{T}},$$

which can also be decomposed according to

$$\Delta(E\mathcal{R}) = v_E(e_1^{\mathsf{T}}\mathcal{R}) + \bar{F}(W_E^{\mathsf{T}}\mathcal{R}) + (ZEZ^{\mathsf{T}}V_{\mathcal{R}})\mathcal{E}^{\mathsf{T}} + (ZEZ^{\mathsf{T}}L)\bar{\mathcal{W}}_{\mathcal{R}}^{\mathsf{T}} - (ZEe_n)(e_n^{\mathsf{T}}\mathcal{R}Z^{\mathsf{T}}), \tag{48}$$

where we have kept the notation $\Delta(E) = v_E e_1^{\mathsf{T}} + \bar{F}\, W_E^{\mathsf{T}}$ introduced in Eq. (43).

We then focus on the $c - s$ columns of $\Delta(E \odot \mathcal{R})$ with indices in $\bar{I} = \{1, \ldots, c\} \setminus I$. Since $\bar{F}$ and $\bar{\mathcal{W}}_{\mathcal{R}}$ have entries in K, and using that $\Delta(E \odot \mathcal{R}) = \lceil \lfloor \Delta(E\mathcal{R}) \rfloor_1 \rceil_1$, these columns can be deduced from Eq. (48) in the following form:

$$\Delta(E \odot \mathcal{R})_{*,\bar{I}} = v_E \odot (e_1^{\mathsf{T}}\mathcal{R})_{\bar{I}} + \bar{F} \cdot (W_E^{\mathsf{T}} \odot \mathcal{R}_{*,\bar{I}}) + (ZEZ^{\mathsf{T}} \odot L) \cdot (\bar{\mathcal{W}}_{\mathcal{R}}^{\mathsf{T}})_{*,\bar{I}} - (ZEe_n) \odot (e_n^{\mathsf{T}}\mathcal{R}Z^{\mathsf{T}})_{\bar{I}}. \tag{49}$$

The right hand side of Eq. (49) can be rewritten as a generator. For the second term we simply let $\bar{G}_1 = \bar{F} \in \mathsf{K}^{n\times(d+1)}$ and $H_1^{\mathsf{T}} = \lceil \lfloor W_E^{\mathsf{T}}\mathcal{R}_{*,\bar{I}} \rfloor_1 \rceil_1 \in \mathsf{K}[x]^{(d+1)\times(c-s)}$. Then the first and last terms in Eq. (49) are partially

linearized. For a polynomial vector $r = \sum_{i=0}^{2d-1} r_i x^i \in \mathsf{K}[x]^n_{<2d}$, we denote by M_r the matrix in $\mathsf{K}[x]^{n\times d}_{<d}$ whose j-th column is $\sum_{i=0}^{d-1} r_{d+i-j+1} x^i$. Likewise, for $t = \sum_{i=0}^{d-1} t_i x^i \in \mathsf{K}[x]^{c-s}_{<d}$ we denote by $\bar{M}_t$ the matrix in $\mathsf{K}^{(c-s)\times d}$ whose j-th column is t_{j-1}. We these notations we have

$$r \odot t^\mathsf{T} = M_r \cdot \bar{M}_t^\mathsf{T}. \tag{50}$$

Applying Eq. (50) allows to write the last $c-s$ columns of $v_E \odot (e_1^\mathsf{T}\mathcal{R})$ and $-(ZEe_n) \odot (e_n^\mathsf{T}\mathcal{R}Z^\mathsf{T})$ as $G_2\bar{H}_2^\mathsf{T}$ and $G_3\bar{H}_3{}^\mathsf{T}$, with $G_2, G_3 \in \mathsf{K}[x]^{n\times d}_{<d}$ and $\bar{H}_2, \bar{H}_3 \in \mathsf{K}^{(c-s)\times d}$. Finally, we let $G_4 = \lceil \lfloor ZEZ^\mathsf{T}L \rfloor_1 \rceil_1 \in \mathsf{K}[x]^{n\times d}_{<d}$ and take $\bar{H}_4^\mathsf{T} = (\bar{\mathcal{W}}_\mathcal{R}^\mathsf{T})_{*,\bar{I}}$. The construction leads to the generator expression:

$$\Delta(E \odot \mathcal{R})_{*,\bar{I}} = \left(\begin{array}{c|c|c|c} \bar{G}_1 & G_2 & G_3 & G_4 \end{array} \right) \left(\begin{array}{c} H_1{}^\mathsf{T} \\ \hline \bar{H}_2^\mathsf{T} \\ \hline \bar{H}_3^\mathsf{T} \\ \hline \bar{H}_4^\mathsf{T} \end{array} \right) \in \mathsf{K}[x]^{n\times(c-s)}_{<d}. \tag{51}$$

Now, $\mathcal{W}_\mathcal{B}$ as in Eq. (47) can be obtained by compression of above right-hand side matrices. Let $G = [G_2, G_3, G_4] \in \mathsf{K}[x]^{n\times 3d}_{<d}$ and $\bar{H} = [\bar{H}_2, \bar{H}_3, \bar{H}_4]^\mathsf{T} \in \mathsf{K}^{(c-s)\times 3d}$. From Eq. (51) we indeed have

$$\bar{F}\,(\mathcal{W}_\mathcal{B}^\mathsf{T})_{*,\bar{I}} = \bar{F}\,H_1^\mathsf{T} + G\,\bar{H}^\mathsf{T},$$

hence

$$(\mathcal{W}_\mathcal{B}^\mathsf{T})_{*,\bar{I}} = H_1^\mathsf{T} + \bar{U}^{-1} G_{J,*}\,\bar{H}^\mathsf{T}, \tag{52}$$

where we assume that $\bar{F}$ has rank $d+1$ and $\bar{U}$ is a $(d+1)\times(d+1)$ nonsingular submatrix of $\bar{F}$ constructed from row indices forming the set J. Since the rows of $\mathcal{W}_\mathcal{B}$ whose indices are in I are zero, the first and last columns of $\mathcal{W}_\mathcal{B}$ can be fully obtained from Eq. (52), the nonzero rows of these columns are given by:

$$e_i^\mathsf{T}(\mathcal{W}_\mathcal{B}^\mathsf{T})_{*,\bar{I}} = e_i^\mathsf{T} H_1^\mathsf{T} + e_i^\mathsf{T}\bar{U}^{-1} G_{J,*}\,\bar{H}^\mathsf{T} \in \mathsf{K}[x]^{c-s}_{<d}, \text{ for } i \in \{1, d+1\}. \tag{53}$$

Lemma 5.1. *For $0 < d < n$, assume that the inverse of a $(d+1)\times(d+1)$ submatrix of $\bar{F}$ as in Proposition 4.4 is given, with the corresponding set of row indices J. From a high-order component E of S^{-1} and $\mathcal{R} \in \mathsf{K}[x]^{n\times c}_{<d}$ with $c = O(n)$, respectively given by their generators as in Eq. (43) and Eq. (46), one can compute the first and the last column of $\mathcal{W}_\mathcal{B}$ for $E \odot \mathcal{R}$ as in Eq. (47) using $\tilde{O}(\mathsf{MM}_{n,d}(s+d,1)) = \tilde{O}(nd(s+d))$ operations.*

Proof. For $i = 1$, the first term $e_1^\mathsf{T} H_1^\mathsf{T}$ in Eq. (53) is computed from $e_1^\mathsf{T} W_E^\mathsf{T} \in \mathsf{K}[x]^n$ by multiplication by $\mathcal{R}$, then extraction of the middle coefficients. From Eq. (46), $\mathcal{R}$ is seen as a Toeplitz-like matrix of degree less than d and displacement rank bounded by $s+d$, which from Theorem 3.4 gives a cost $\tilde{O}(\mathsf{MM}_{n,d}(s+d,1))$ for those first computations.

The target cost bound is valid for obtaining the generators $G_2, \bar{H}_2, G_3$ and $\bar{H}_3$ from E and $\mathcal{R}$. This is indeed equivalent to having the first (v_E is part of the generators) and last column of E, and the first and last row of $\mathcal{R}$. The required products involving canonical vectors and E and $\mathcal{R}$ can be computed in time $\tilde{O}(\mathsf{MM}_{n,d}(s+d,1))$.

Then, let $u \in \mathsf{K}^{d+1}$ be the first row of $\bar{U}^{-1}$. From $u^\mathsf{T}(G_2)_{J,*}$ and $u^\mathsf{T}(G_3)_{J,*}$ in $\mathsf{K}[x]^d$, $u^\mathsf{T}(G_2)_{J,*}\bar{H}_2^\mathsf{T}$ and $u^\mathsf{T}(G_3)_{J,*}\bar{H}_3^\mathsf{T}$ are deduced in time $O(nd^2)$ by matrix times vector products using that $\bar{H}_2^\mathsf{T}$ and $\bar{H}_3^\mathsf{T}$ are matrices in $\mathsf{K}^{d\times(c-s)}$ and $c = O(n)$. Here, recall that $\mathsf{MM}_{n,d}(d,1) = nd^2$. It thus remains to verify the cost bound for the computation of $u^\mathsf{T}(G_4)_{J,*}\bar{H}_4^\mathsf{T}$. Since u has scalar entries we can first compute $u^\mathsf{T}(ZEZ^\mathsf{T})_{J,*} \in \mathsf{K}[x]^n$, then multiply the result by $L \in \mathsf{K}[x]^{n\times d}_{<d}$, and multiply the middle coefficients of the latter by $\bar{H}_4$ using a total of $\tilde{O}(nd^2)$ operations. What we have just said with e_1 is also valid with e_{d+1}, which concludes the proof. □

The inversion of an appropriate submatrix of $\bar{F}$ that is required for Lemma 5.1 will be done only once for all products in Section 6.2.

5.2. High-order components

By combining Lemma 5.1 in the case $s = 1$ and a direct computation of the first column we can perform the middle product of a high-order component by a residue as shown by next lemma.

Lemma 5.2. *Under the assumptions of Lemma 5.1 for $\bar{F}$, consider a high-order component E of S^{-1} and a residue $R = \rho^k(\mathrm{I})$ for some $k \geq 0$, both represented by their generators as in Eq. (43) and Eq. (44). A generator for $E \odot R$ as in Eq. (45) can be computed using $\tilde{O}(\mathsf{MM}_{n,d}(d, 1)) = \tilde{O}(nd^2)$ operations.*

Proof. The first column $v_C = \lceil \lfloor E v_R \rfloor_1 \rceil_1$ of C can be computed by applying E to v_R in $\tilde{O}(\mathsf{MM}_{n,d}(d, 1))$ operations, and by extracting the middle coefficients. The assertion of the lemma then follows from Lemma 5.1 with $s = 1$ and $i_1 = 1$ for the computation of the first and last column of W_C, and from Proposition 4.4 for the whole generator. □

Given a slice C of S^{-1}, the right generator for the residue at the same order is obtained using Remark 4.8 and allows to manipulate high-order components in the following way.

Lemma 5.3. *Under the assumptions of Lemma 5.1 for $\bar{F}$, we consider further that L as in Proposition 4.7 is given. For two high-order components $E^{(i)}$ and $E^{(j)}$ with $i \geq 0$ and $j \geq 1$, both represented by their generators as in Eq. (43), Algorithm* ComponentProduct *computes generators having the same shape for the high-order component $E^{(i+j)}$ using $\tilde{O}(\mathsf{MM}_{n,d}(d, 1)) = \tilde{O}(nd^2)$ operations.*

Proof. We keep the notation used for Algorithm ComponentProduct. Since $\bar{F}$ is a scalar matrix, the generators for $E^{(j)}$ directly give generators for C_{j-1} and C_j, which from Proposition 4.4 give in particular w_{j-2} and w_{j-1}. Therefore from Remark 4.8 we have right generators for R_{j-1} and R_j, which are residues of order $j-1$ and j, respectively. The first columns of C_{j-1} and C_j are available from the generators, by truncated multiplication by S these two columns give the first columns of R_{j-1} and R_j. Hence the whole generators for R_{j-1} and R_j are known and we finally apply Lemma 5.2 twice. □

6. Giant steps

In order to perform the giant steps (Step 4 in Algorithm ProjectedExpansion) we specialize Algorithm FurtherResidues to the Sylvester case. The successive products that involve high-order components and concatenated residues as in Eq. (21) are implemented thanks to middle and truncated products. As before, high-order components and residues are represented by their generators as those in Sections 4.2 and 4.3. Their respective orders do not matter in this section. We work under the assumptions of Propositions 4.4 and 4.7, and following Section 5 we take $0 < d < n$. The representation for concatenated matrices, which for technical reasons is a little bit different, is specified in Section 6.1.

Taking advantage of the special shape of the generators we can split up the middle products into regular matrix products for obtaining their left parts, and apply the strategy of Section 5.1 for computing their right parts. Then, truncated products are used for the computation of residues from slices of the inverse according to Eq. (17). Remark 4.8 actually implies that the right generator parts for residues are directly deduced from those of the slices; in the same way as for middle products we compute their left generator parts by regular matrix product.

We proceed to high-order lifting with the projection $V = X = [\mathrm{I}_m\ 0]^\mathsf{T}$, where $1 \leq m \leq n$. Considering a number s of giant steps, our purpose in this section is to bound the cost of a call to FurtherResidues with input the high-order component E of order r of S^{-1}, which computes the matrix

$$\mathcal{R} = \left[X\ \rho^r(X)\ \rho^{2r}(X)\ \ldots \rho^{(s-1)r}(X)\right] \in \mathsf{K}[x]^{n\times(sm)}_{<d}. \tag{54}$$

We first study in Section 6.1 the two central products at Steps 4 and 5 of Algorithm FurtherResidues, and then bound the overall cost in Section 6.2.

6.1. Concatenated middle and truncated products

Let us first specify the representation we use for the block residue matrices involved. Noting that the residue map satisfies $\rho^k(X) = \rho^k(\mathrm{I})X$, for $k = 1, 2, 3, \ldots, 2^{l-1}$ the right operand at Step 4 of Algorithm FurtherResidues is of the type

$$\mathcal{P} = \left[P^{(0)}X \;\; P^{(1)}X \;\; \cdots \;\; P^{(k-1)}X\right] \in \mathsf{K}[x]^{n\times km}_{<d}, \tag{55}$$

where for $j \geq 0$ each $P^{(j)} \in \mathsf{K}[x]^{n\times n}$ is some residue of the identity. The displacement operator applied to such a $\mathcal{P}$ gives

$$\Delta(\mathcal{P}) = \mathcal{P} - Z_n\mathcal{P}Z_{km}^{\mathsf{T}} = \mathcal{P} - \left[(Z_nP^{(0)}XZ_m^{\mathsf{T}}) \quad \ldots \quad (Z_nP^{(k-1)}XZ_m^{\mathsf{T}})\right] - \sum_{j=1}^{k-1} Z_nP^{(j-1)}e_m\epsilon_{1+jm}^{\mathsf{T}}, \tag{56}$$

here, in order to avoid confusion, we have $e_m \in \mathsf{K}^n$ and we use ϵ_{1+jm} for the canonical vectors in K^{km}. On the other hand, from Eq. (40) we can write

$$\Delta(P^{(j)}X) = \Delta(P^{(j)})X = p_je_1^{\mathsf{T}} + L\bar{W}_{P^{(j)}}^{\mathsf{T}}, \tag{57}$$

where p_j is the first column of $P^{(j)}$, $L \in \mathsf{K}[x]^{n\times d}_{<d}$, and $\bar{W}_{P^{(j)}} \in \mathsf{K}^{m\times d}$. Equations (56) and (57) then give

$$\Delta(\mathcal{P}) = V_{\mathcal{P}}\mathcal{E}^{\mathsf{T}} + L\bar{\mathcal{W}}_{\mathcal{P}}^{\mathsf{T}}, \tag{58}$$

such that: $V_{\mathcal{P}} \in \mathsf{K}[x]^{n\times k}$ has first column p_0 and column j being $p_{j-1} - Z_nP^{(j-2)}e_m$ for $2 \leq j \leq k$; $\mathcal{E} \in \mathsf{K}^{km\times k}$ has column j being $\epsilon_{1+(j-1)m}$; $\bar{\mathcal{W}}_{\mathcal{P}} \in \mathsf{K}^{km\times d}$ has k blocks of rows with j-th block being $\bar{W}_{P^{(j)}}$.

For a high-order component E of S^{-1}, let $\mathcal{B} = E \odot \mathcal{P}$. Then the resulting matrix $\mathcal{Q} = \lceil S\mathcal{B}\rceil_1$ at Step 5 of Algorithm FurtherResidues involves residues Q^j at further orders for $j = 0, \ldots, k-1$ such that

$$\Delta(Q^{(j)}X) = q_je_1^{\mathsf{T}} + L\bar{W}_{Q^{(j)}}^{\mathsf{T}}, \tag{59}$$

where q_j is the first column of $Q^{(j)}$ and $\bar{W}_{Q^{(j)}} \in \mathsf{K}^{m\times d}$. In accordance with Eq. (58) we have

$$\Delta(\mathcal{Q}) = V_{\mathcal{Q}}\mathcal{E}^{\mathsf{T}} + L\bar{\mathcal{W}}_{\mathcal{Q}}^{\mathsf{T}}, \tag{60}$$

with: $V_{\mathcal{Q}} \in \mathsf{K}[x]^{n\times k}$ has first column q_0 and column j being $q_{j-1} - Z_nQ^{(j-2)}e_m$ for $2 \leq j \leq k$; $\bar{\mathcal{W}}_{\mathcal{Q}} \in \mathsf{K}^{km\times d}$ has k blocks of rows with j-th block being $\bar{W}_{Q^{(j)}}$.

Equations (58) and (60) lead us to represent $\mathcal{P}$ and $\mathcal{Q}$ as follows. To ensure that a canonical representation is maintained throughout the algorithm, hence following our approach in Section 5, we separate out left and right generator parts. The right parts are given by $\bar{\mathcal{W}}_{\mathcal{P}}$ and $\bar{\mathcal{W}}_{\mathcal{Q}}$. Besides L and from the characterizations of $V_{\mathcal{P}}$ and $V_{\mathcal{Q}}$, we slightly modify the representation of the left generator parts. They are represented by the first and m-th columns of the $P^{(j)}$'s and $Q^{(j)}$'s. This is temporarily slightly different from the canonical representation with $V_{\mathcal{P}}$ and $V_{\mathcal{Q}}$ in order to simplify the following statement.

Lemma 6.1. *For $d < n$, assume that the inverse of a $(d+1)\times(d+1)$ submatrix of $\bar{F}$ is given, with the corresponding row indices set J, also assume that L and the canonical generator for E is given. Consider $\mathcal{P}$ represented by $\{P^{(j)}e_1\}_{j=0,\ldots k-1}$, $\{P^{(j)}e_m\}_{j=0,\ldots k-2}$ and $\bar{\mathcal{W}}_{\mathcal{P}}$ as in Eq. (58). If $\mathcal{P}$ has $km = O(n)$ columns, then Steps 4 and 5 of Algorithm FurtherResidues compute $\{Q^{(j)}e_1\}_{j=0,\ldots k-1}$, $\{Q^{(j)}e_m\}_{j=0,\ldots k-2}$ and $\bar{\mathcal{W}}_{\mathcal{Q}}$ using $\tilde{O}(\mathsf{MM}_{n,d}(d,k))$ operations.*

Proof. The specification of the output is from Lemma 2.7, we have to prove the cost bound. The matrix $\mathcal{B} = E \odot \mathcal{P}$ has k blocks of columns $E \odot P^{(j)}X$, each of which is the projection of a slice $C^{(j)} = E \odot P^{(j)}$ of S^{-1} (Lemma 2.4). From Eq. (35) we can thus write:

$$\Delta(E \odot P^{(j)}X) = \Delta(C^{(j)}X) = c_je_1^{\mathsf{T}} + \bar{F}W_{C^{(j)}}^{\mathsf{T}}, \tag{61}$$

where c_j is the first column of $C^{(j)}$, and $W_{C^{(j)}} \in \mathsf{K}[x]^{m\times(d+1)}_{<d}$. Hence by doing the same manipulation as for $\mathcal{P}$ and $\mathcal{Q}$, we arrive at

$$\Delta(\mathcal{B}) = V_{\mathcal{B}}\mathcal{E}^{\mathsf{T}} + \bar{F}\mathcal{W}_{\mathcal{B}}^{\mathsf{T}}, \tag{62}$$

involving matrices such that: $V_{\mathcal{B}} \in \mathsf{K}[x]^{n\times k}$ has first column c_0 and column j being $c_{j-1} - Z_n C^{(j-2)} e_m$ for $2 \le j \le m$; $\mathcal{W} \in \mathsf{K}^{km\times(d+1)}$ has k blocks of rows with j-th block being $W_{C^{(j)}}$.

Using that $\mathcal{Q} = \lceil S\mathcal{B}\rceil_1$, we first detail the whole computation of the first and m-th columns of the $\mathcal{Q}^{(j)}$'s from the representations of $\mathcal{P}$ and E. We then conclude with the right generator part for $\mathcal{Q}$.

For $j = 0, \dots k-1$ and $i = 1, m$ we have $C^{(j)}e_i = E \odot (P^{(j)}e_i)$. All these middle products can be computed from the regular product $E \cdot [P^{(0)}e_i \ \dots \ P^{(k-1)}e_i]$ by extraction of the middle coefficients. From Theorem 3.4 this can be performed using $\tilde{O}(\mathsf{MM}_{n,d}(d,k))$ operations. For all j we then have $\mathcal{Q}^{(j)}e_i = \lceil S\,C^{(j)}e_i\rceil_1$, products which can computed by extraction of the coefficients of $S \cdot [C^{(0)}e_i \ \dots \ C^{(k-1)}e_i]$ within the same cost bound.

We now deduce the right generator part $\bar{\mathcal{W}}_{\mathcal{Q}}$ as in Eq. (60) from the corresponding $\mathcal{W}_{\mathcal{B}}$ of Eq. (62). The last column of $\mathcal{W}_{\mathcal{B}}$ can indeed be computed using Lemma 5.1: the generator for $\mathcal{P}$ given by Eq. (58) has the shape of the one for $\mathcal{R}$ as in Eq. (46); the generator for $\mathcal{B}$ in Eq. (62) corresponds to the one for $E \odot \mathcal{R}$ in Eq. (47). Remark that the application of Lemma 5.1 explicitly requires the generator part $V_{\mathcal{P}}$ for $\mathcal{P}$ as in Eq. (58): $V_{\mathcal{P}}$ can be reconstructed from $\{P^{(j)}e_1\}_{j=0,\dots,k-1}$ and $\{P^{(j)}e_m\}_{j=0,\dots,k-2}$ in time $O(nkd)$. Taking $c = km$ in Lemma 5.1, the last column of $\mathcal{W}_{\mathcal{B}}$ is therefore obtained in time $\tilde{O}(\mathsf{MM}_{n,d}(k+d,1))$, which is $\tilde{O}(\mathsf{MM}_{n,d}(d,k))$.

By definition of $\mathcal{W}_{\mathcal{B}}$, we now know for $j = 0, \dots, k-1$ the last column of $W_{C^{(j)}}$ given by Eq. (61) for the projected slice $C^{(j)}X$ of S^{-1}. For any fixed j let κ be the z-adic order of $C^{(j)}X$, and remark that $W_{C^{(j)}}$ must be the projection of the full right generator part for $C^{(j)}$. It follows from Proposition 4.4 that the last column of $W_{C^{(j)}}$ provides us with the z-adic coefficient $X^{\mathsf{T}}w_{\kappa-1}$ of the first m entries $X^{\mathsf{T}}w$ of w. We conclude using Remark 4.8. Indeed, from Eq. (17), $\mathcal{Q}^{(j)} = \lceil S\,C^{(j)}\rceil_1$ is the residue $\rho^{\kappa}(\mathrm{I})$, hence its right generator part is directly deduced from $w_{\kappa-1}$. Equivalently, by projection using X, $\bar{W}_{\mathcal{Q}^{(j)}}$ as in Eq. (59) is deduced from $X^{\mathsf{T}}w_{\kappa-1}$ in $O(md^2)$ operations. Finally, since we have been working for an arbitrary $0 \le j \le k-1$, all the blocks of rows for $\bar{\mathcal{W}}_{\mathcal{Q}}$ as in Eq. (60) are obtained in $O(kmd^2)$, which does not dominate the cost. □

6.2. *Cost bound for the giant steps*

We can now bound the cost of Algorithm FurtherResidues in the case of the Sylvester matrix, with input some high-order component E of the inverse and $V = X$.

Lemma 6.2. *Let E be the high-order component E of order r of S^{-1} represented by its canonical generator, and assume that $\bar{F}$ has rank $d+1 \le n$. With input E and the projection $X = [\mathrm{I}_m \ \ 0]^{\mathsf{T}}$ for $1 \le m \le n$, Algorithm* FurtherResidues *computes a generator as in Eq. (46) for the matrix $\mathcal{R} = [X\ \rho^r(X)\ \rho^{2r}(X)\ \dots \rho^{(s-1)r}(X)] \in \mathsf{K}[x]^{n\times(sm)}_{<d}$. If $sm = O(n)$ the cost of the computation is $\tilde{O}\,(\mathsf{MM}_{n,d}(d,s))$.*

Proof. From Lemma 2.7 we know that the algorithm correctly computes $\mathcal{R}$. The cost bound comes from $\log s$ applications of Lemma 6.1 for the representations of the new residues at Steps 4 and 5 with $k \le s$ and $(\log s) - 1$ applications of Lemma 5.3 for the generators for the new high-order components at Step 6. For the first application of Lemma 6.1, $P^{(0)} = X$ is represented using $\Delta(X) = e_1{\epsilon_1}^{\mathsf{T}}$ (here ϵ_1 is the canonical vector in K^m). Both Lemma 6.1 and Lemma 5.3 require the preliminary computation of L whose cost is $\tilde{O}(nd^2)$ operations from Proposition 4.7. They also rely on the inverse of a $(d+1)\times(d+1)$ submatrix of $\bar{F}$; such an inverse can be computed in $O(nd^{\omega-1})$ operations [17, 19], which does not dominate since for $d < n$ and $s \ge 1$ we have $\mathsf{MM}_{n,d}(d,s) \in \Omega(nd^2)$.

Lemma 6.1 uses a representation of the matrices slightly different from here. The left generator part $V_{\mathcal{R}}$ of Eq. (46) still needs to be recovered from $\{\rho^{jr}(e_1)\}_{j=0,\dots s-1}$, $\{\rho^{jr}(e_m)\}_{j=0,\dots s-2}$ and the right generator. From the shape of the left generator given by Eq. (58), we deduce that the latter can be reconstructed from $O(s)$ polynomial vector additions with cost $O(nsd)$. □

7. Complete expansion algorithm

From the preceding sections we have all the ingredients for giving the cost of the specialization of Algorithm ProjectedExpansion to the Sylvester case. For $X = [\mathrm{I}_m\ 0]^{\mathsf{T}}$ and $Y = [0\ \mathrm{I}_m]^{\mathsf{T}}$ in $\mathsf{K}^{n\times m}$, Algorithm

StructuredExpansion computes the truncated expansion of $Y^\mathsf{T} S^{-1} X$ at the z-adic order rs, following the three main phases of the general approach of Section 2.2.

First, the baby steps are performed based on two linear system solutions computed using the seminal lifting methods of [32, 8]. (The high-order lifting as in Section 2 is not required for this step.) The loop at Step 1 of Algorithm ProjectedExpansion can indeed be implemented by computing the expansion of $Y^\mathsf{T} S^{-1}$ at the order r, of which coefficients give the projections of the high-order components. We actually compute only truncations of $e_1^\mathsf{T} S^{-1}$ (at order $r+1$ as it will also be used to compute a high-order component at order r) and of $e_n^\mathsf{T} S^{-1}$, since by using the recursion of Eq. (27), these two rows are sufficient to recover the expansion of $Y^\mathsf{T} S^{-1} X$ at the end. These polynomial vectors of degree rd are linearized as r vectors of degree d in order to take advantage of fast structured matrix multiplication.

The special structures we have identified for the high-order components and the residues are then used in the giant steps as well as in the third phase which computes the final product. This product is applied on the linearized rows of the baby steps, which are combined together in the following step. In anticipation of the reconstruction of the expansion of $Y^\mathsf{T} S^{-1} X$, the two latter phases actually use a modified projection $X' = [\mathrm{I}_{2m-1}\ 0]^\mathsf{T}$ on the right, and give the first and last row of the expansion of $S^{-1} X'$ (this trick is taken from [34, Sec. 3.4.3]). Finally, the whole target expansion of $Y^\mathsf{T} S^{-1} X$ is reconstructed using Eq. (27) as mentioned above.

Algorithm 7.1 StructuredExpansion

Input: $p, q \in \mathsf{K}[x, y]$ of respective y-degrees n_p and n_q and of x-degree at most $d < n = n_p + n_q$, $m \leq (n+1)/2$, $r, s \in \mathbb{N}^*$

Assumptions: the constant terms of $p^{(0)}$, $q^{(n_q)}$ and $\det S$ in $\mathsf{K}[x]$ are nonzero, where $S \in \mathsf{K}[x]^{n\times n}_{\leq d}$ is the Sylvester matrix associated to p and q, and $\bar{F} \in \mathsf{K}^{n\times(d+1)}$ as defined in Proposition 4.4 has rank $d+1$.

Output: $\lceil Y^\mathsf{T} S^{-1} X \rceil_{rs}$ where $Y = [0\ \ \mathrm{I}_m]^\mathsf{T}$ and $X = [\mathrm{I}_m\ \ 0]^\mathsf{T}$

1: ▹ *Baby steps*
$z \leftarrow x^d$; $a \leftarrow e_1^\mathsf{T} S^{-1} \bmod z^{r+1}$; $b \leftarrow e_n^\mathsf{T} S^{-1} \bmod z^r$
$A \leftarrow [a_0^\mathsf{T}\ a_1^\mathsf{T}\ \dots\ a_{r-1}^\mathsf{T}]^\mathsf{T}$; $B \leftarrow [b_0^\mathsf{T}\ b_1^\mathsf{T}\ \dots\ b_{r-1}^\mathsf{T}]^\mathsf{T}$ ▹ *Both in* $\mathsf{K}[x]^{r\times n}_{<d}$
2: ▹ *Generator for* $E = E^{(r)} \in \mathsf{K}[x]^{r\times n}_{<2d}$ *of length* $d+2$: $\Delta(E) = v e_1^\mathsf{T} + \bar{F}\, W^\mathsf{T}$
$f \leftarrow y^{n_q} q(1/y) - y^{n_q} y^{n_p} p(1/y)$ ▹ $f = \sum_{j=0}^{d} \sum_{i=0}^{n} \dot{f}_j^{(i)} x^j y^i$
$\bar{F} \leftarrow [\dot{f}_{j-1}^{(i-1)}]_{1..n, 1..d+1}$ ▹ *In* $\mathsf{K}^{n\times(d+1)}$
$v \leftarrow S^{-1} e_1 \bmod z^{r+1}$; $v \leftarrow \lfloor v \rfloor_{r-1}$
$w \leftarrow -a/f^{(0)} \bmod z^{r+1}$; $w \leftarrow [0\ w_{2..n}]$ ▹ *See Eq.* (30)
Construct W from $w_{r-2} + z w_{r-1}$ and $w_{r-1} + z w_r$ ▹ *See Proposition 4.4 and Eq.* (36)
3: ▹ *Giant steps*
$m' \leftarrow 2m - 1$; $X' \leftarrow [\mathrm{I}_{m'}\ 0]^\mathsf{T}$
$\mathcal{R} \leftarrow (\text{FurtherResidues}(E, X', s))_{*, 1..sm'}$ ▹ *In* $\mathsf{K}[x]^{n\times(sm')}_{<d}$, *see Lemma 6.2*
4: ▹ *Final products*
$A' \leftarrow A \cdot \mathcal{R}$; $A'_{r,*} \leftarrow A'_{r,*} \bmod z$ ▹ *Matrices in* $\mathsf{K}[x]^{r\times sm'}_{<2d}$
$B' \leftarrow B \cdot \mathcal{R}$; $B'_{r,*} \leftarrow B'_{r,*} \bmod z$
5: ▹ *Reconstruction of the first and last row of* $\lceil S^{-1} X' \rceil_{rs} \in \mathsf{K}[x]^{n\times m'}$
$H' \leftarrow 0 \in \mathsf{K}[x]^{m'}$; $H \leftarrow 0 \in \mathsf{K}[x]^{m\times m'}$
for $i = 0, \dots, r-1$ ▹ *See Proposition 7.1 and proof thereof*
  for $j = 0, \dots, s-1$
    $H'_{1..m'} \leftarrow H'_{1..m'} + z^{i+rj} A'_{i+1,(jm'+1)..(jm'+m')}$
    $H_{m,1..m'} \leftarrow H_{m,1..m'} + z^{i+rj} B'_{i+1,(jm'+1)..(jm'+m')}$ ▹ *Remaining rows left to* 0
6: ▹ *Obtaining* $\lceil Y^\mathsf{T} S^{-1} X \rceil_{rs} \in \mathsf{K}[x]^{m\times m}$ *using truncated power series operations*
for $i = m-1, \dots, 1$ ▹ *See Proposition 7.1 and proof thereof*
  $c \leftarrow m + i - 1$
  $H_{i,1..c} \leftarrow H_{i+1,2..c+1} - f^{(i)} (H'_{2..c+1}/f^{(0)}) \bmod z^{rs}$ ▹ *See Eq.* (63)
7: **return** $H_{1..m,1..m}$

Proposition 7.1. *Let p and q in $\mathsf{K}[x,y]$ be of respective y-degrees n_p and n_q, and of x-degree at most $d < n = n_p + n_q$. Assume that the constant terms of $p^{(0)}$, $q^{(n_q)}$ and $\det S$ in $\mathsf{K}[x]$ are nonzero, where $S \in \mathsf{K}[x]^{n\times n}_{\le d}$ is the Sylvester matrix associated to p and q, and also that $\bar{F} \in \mathsf{K}^{n\times(d+1)}$ as defined in Proposition 4.4 has rank $d+1$. For $X = [\mathrm{I}_m\ 0]^\mathsf{T}$ and $Y = [0\ \mathrm{I}_m]^\mathsf{T}$ with $2m-1 \le n$, $z = x^d$, and positive integers r, s, Algorithm* StructuredExpansion *computes the expansion of $Y^\mathsf{T}S^{-1}X$ modulo z^{rs}. If $s = O(r)$ and $mr = O(n)$, then it uses $\tilde{O}(\mathsf{MM}_{n,d}(r+d,r) + m^2rsd)$ arithmetic operations in K.*

Proof. Step 1 computes truncated expansions of $e_1^\mathsf{T}S^{-1}$ and $e_n^\mathsf{T}S^{-1}$ which are used for constructing the generators for $E^{(r)}$ and for the final products. This can be done by x-adic lifting using that $S(0)$ is nonsingular [32, 8]. We follow the description of the method in [8], carried over to the case $\mathsf{K}[x]$. The cost is essentially that of $O(rd)$ multiplications of the transpose inverse $S(0)^{-\mathsf{T}}$ by a vector. Using a displacement rank-based representation of $S(0)^{-\mathsf{T}}$— see e.g. [40, Sec. 5] and references therein, such a multiplication costs $\tilde{O}(n)$; this gives a total of $\tilde{O}(nrd)$ operations for Step 1. The rows of A and B are the z-adic coefficients of the computed expansions.

Step 2 computes the canonical generator for $E^{(r)} \equiv \lfloor S^{-1}\rfloor_{r-1} \bmod z^2$. The first column v is computed in the same way as in Step 1. (Note that the multiplication of $S(0)^{-1}$ by a vector could also be computed by using the extended Euclidean algorithm in $\mathsf{K}[x]$ [10, Sec. 4.5].) Then according to Eq. (36), the left generator part $\bar{F}$ is given by f, and the right generator requires w_{r-2}, w_{r-1} and w_r. The first entry of w is zero and from Eq. (30) the remaining ones modulo z^{r+1} are deduced as $(e_1^\mathsf{T}S^{-1})_{2\ldots n}/f^{(0)}$ using the first row of S^{-1} obtained at previous step (the inverse of $f^{(0)}$ exists from $f(0,0) = q^{(n_q)}(0) \neq 0$). From Proposition 4.4, this leads to the wanted z-adic coefficients of w and the generator for $E^{(r)}$ in time $\tilde{O}(nrd) + O(nd^2)$.

We then deduce from Lemma 6.2 that a generator for $\mathcal{R} = [X'\ \rho^r(X')\ \rho^{2r}(X')\ \ldots \rho^{(s-1)r}(X')]$ with $X' = [\mathrm{I}_{m'}\ 0]^\mathsf{T}$ is correctly computed in allotted time at Step 3 as $s = O(r)$. This generator is as in Eq. (46), hence of length at most $s+d$. Using the bounds $s = O(r)$ and thus $sm' = O(n)$ in Theorem 3.4, the matrix products $A\cdot\mathcal{R}$ and $B\cdot\mathcal{R}$ are computed using $\tilde{O}(\mathsf{MM}_{n,d}(r+d,r))$ operations.

Then let $a'_{i,j}$ be the $(i+1)$-th row of $A'_{*,(jm'+1)..(jm'+m')}$ at Step 4. From the product with the block of columns corresponding to $\rho^{rj}(X')$ in $\mathcal{R}$ and Definition 2.1 we have

$$\sum_{i=0}^{r-1} a'_{i,j}z^{i+rj} \equiv (\sum_{i=0}^{r-1} a_i z^i)\rho^{rj}(X') \bmod z^r \equiv (e_1^\mathsf{T}S^{-1} \bmod z^r)\rho^{rj}(X') \bmod z^r \equiv \lfloor e_1^\mathsf{T}S^{-1}X'\rfloor_{rj} \bmod z^r,$$

hence the sums at Step 5 give $H'_{1..m'} \equiv e_1^\mathsf{T}S^{-1}X' \bmod z^{rs}$; in an equivalent manner, with B' we get that the last row of H is $H_{m,1..m'} \equiv e_n^\mathsf{T}S^{-1}X' \bmod z^{rs}$. The cost is bounded by the one of rs additions of polynomial vectors of dimension m' and degree d, which is dominated by the previous step.

We conclude at Step 6 by following the trick in [34, Sec. 3.4.3], which consists in using the recursion given by Eq. (27) for reconstructing the whole expansion of $Y^\mathsf{T}S^{-1}X$ from those of $e_1^\mathsf{T}S^{-1}X'$ and $e_n^\mathsf{T}S^{-1}X'$. For the inverse C of S which is a matrix of multiplication and $1 \le i, c < n$, Eq. (27) indeed leads to:

$$C_{i,1..c} = C_{i+1,2..c+1} - f^{(i)}(C_{1,2..c+1}/f^{(0)}). \tag{63}$$

From $C_{1,2..2m-1}$ and $C_{m,1..2m-1}$, given by $e_1^\mathsf{T}S^{-1}X'$ and $e_n^\mathsf{T}S^{-1}X'$, the application of Eq. (63) for $i = m-1, m-2, \ldots 1$ on truncated power series modulo z^{rs} provides with the m^2 entries of $Y^\mathsf{T}S^{-1}X$ using $\tilde{O}(m^2rsd)$ operations. □

8. Resultant algorithm

Following the previous works in Section 1.1.1 and Section 1.1.2, once sufficiently many terms of the expansion of $Y^\mathsf{T}S^{-1}X \in \mathsf{K}(x)^{m\times m}$ are known then a matrix fraction description ND^{-1} with coprime matrices $N, D \in \mathsf{K}[x]^{m\times m}$ is computed. For generic polynomials p and q of degree d in x and $n_p = n_q = n/2$ in y, we recall in Section 8.1 how such a fraction description can be reconstructed from only $O(n/m)$ terms of the x^d-adic expansion of $YS^{-1}X$. Furthermore, the denominator matrix D that is obtained is such that its determinant is the resultant of p and q up to a scalar factor. Together with Algorithm StructuredExpansion this leads us to the resultant algorithm given in Section 8.2, and to the proof of Theorem 1.1.

8.1. Matrix fraction reconstruction

The number of terms sufficient for reconstructing a matrix fraction depends on the degrees of its possible descriptions. First, let us recall a few notions on matrix fractions (the reader may refer to the comprehensive material in [20, Chap. 6] and its applications in [40], [34, Sec. 5]). For a matrix $F \in \mathsf{K}(x)^{m\times m}$, a description $F = ND^{-1}$ with $N, D \in \mathsf{K}[x]^{m\times m}$ is said to be minimal if N and D are right coprime (have unimodular right matrix gcd's), and D has minimal column degrees among all possible denominators. The fraction F is said to be describable in degree δ if it admits both a left description $F = D_L^{-1}N_L$ and a right description $F = ND^{-1}$ with denominators D_L and D of degree at most δ [34, Sec. 5.1.1]. Generically, we have the following for the fraction $Y^\mathsf{T}S^{-1}X$ we are interested in. This is an adaptation of [40, Prop. 4.1] which used slightly different projections. Indeed we chose to switch the role of the projections X and Y, in order to make the giant steps simpler.

Proposition 8.1. *For any even n and integers $d, m \in \{1, \dots, n\}$ there exists a nonzero polynomial Φ in $2(n/2+1)(d+1)$ variables over K and of degree $O(n^3d^2)$ such that for $p = \sum_{0\le i\le d, 0\le j\le n/2} p_i^{(j)} x^i y^j$ and $q = \sum_{0\le i\le d, 0\le j\le n/2} q_i^{(j)} x^i y^j$ of y-degree $n/2$ in $\mathsf{K}[x, y]$, if $\Phi(p_0^{(0)}, \dots, p_d^{(n/2)}, q_0^{(0)}, \dots, q_d^{(n/2)}) \neq 0$ then:*

i) S is invertible and S^{-1} is strictly proper (each entry has its numerator degree less than its denominator degree);

ii) $Y^\mathsf{T}S^{-1}X$ is describable in degree $\delta = 2\lceil n/(2m)\rceil d$;

iii) if $Y^\mathsf{T}S^{-1}X = ND^{-1}$ is a minimal description then $\det D = c\,Res_y(p, q)$ for some nonzero $c \in \mathsf{K}$.

Proof. Consider $\hat{q} = y^{n/2}q(1/y)$, $\hat{p} = y^{n/2}p(1/y)$, and the associated Sylvester matrix $\hat{S} \in \mathsf{K}[x]^{n\times n}$. From [40, Sec. 4], there exists a nonzero polynomial $\hat{\Phi}$ in $2(n/2+1)(d+1)$ variables and of degree $O(n^3d^2)$, such that if the coefficients of $\hat{q}$ and $\hat{p}$ do not form a zero of $\hat{\Phi}$, then: $\hat{q}$ and $\hat{p}$ have degree $n/2$; $\hat{S}$ is invertible and $\hat{S}^{-1}$ is strictly proper; $X^\mathsf{T}\hat{S}^{-1}Y$ is describable in degree δ; a minimal description

$$X^\mathsf{T}\hat{S}^{-1}Y = \hat{N}\hat{D}^{-1} \tag{64}$$

has a denominator that satisfies $\det \hat{D} = \hat{c}\,\mathrm{Res}_y(\hat{q}, \hat{p})$ for some $\hat{c} \in \mathsf{K}^*$.

Let Φ be the polynomial obtained from $\hat{\Phi}$ by swapping variables so that evaluating Φ at the coefficients of p and q is evaluating $\hat{\Phi}$ at the coefficients of $\hat{q}$ and $\hat{p}$. We show that Φ is appropriate.

Assume that the coefficients of p and q do not form a zero of Φ. We have $\hat{S} = J_nSJ_n$, where J_n is the reversal matrix of dimension n. Since *i*) is satisfied with $\hat{S}$ as $\hat{q}$ and $\hat{p}$ do not form a zero of $\hat{\Phi}$, we have that *i*) is also satisfied with S.

We then show that if $X^\mathsf{T}\hat{S}^{-1}Y$ is describable in degree δ, then $Y^\mathsf{T}S^{-1}X$ is also describable in degree δ. From Eq. (64) and using $\hat{S}^{-1} = J_nS^{-1}J_n$ we get

$$Y^\mathsf{T}S^{-1}X = J_m\hat{N}(J_m\hat{D})^{-1}, \tag{65}$$

which shows the existence of an appropriate right description for $Y^\mathsf{T}S^{-1}X$. Indeed, $\deg \hat{D} \le \delta$ since $\hat{D}$ is a minimal denominator of $X^\mathsf{T}\hat{S}^{-1}Y$. In a similar way, a left denominator of degree at most δ for $Y^\mathsf{T}S^{-1}X$ is obtained from a left denominator of degree at most δ for $X^\mathsf{T}\hat{S}^{-1}Y$.

Item *iii*) is finally proved by noticing that $J_m\hat{D}$ is minimal in Eq. (65) if and only if $\hat{D}$ is minimal in Eq. (64). A minimal denominator D for $Y^\mathsf{T}S^{-1}X$ hence gives a minimal denominator $\hat{D} = J_mD$ for $X^\mathsf{T}\hat{S}^{-1}Y$, and $\det D = \pm\det\hat{D} = \pm\hat{c}\,\mathrm{Res}_y(\hat{q}, \hat{p}) = c\mathrm{Res}_y(p, q)$. □

If $Y^\mathsf{T}S^{-1}X$ satisfies the first two items in Proposition 8.1, then a minimal description ND^{-1} can be computed from 2δ terms of its expansion. We follow [12, 34], and perform the reconstruction of the fraction using minimal approximant bases [3, 39]. We especially refer to [34, Sec. 5.2, 5.3] for a detailed treatment of the reconstruction, which we do not repeat here, Algorithm FractionReconstruction being exactly Step 2 of [34, Algorithm 5.1]. Note that the latter algorithm is applied to a fraction that is constructed in a different way than $Y^\mathsf{T}S^{-1}X$ but this does not intervene for the reconstruction itself.

Algorithm 8.1 FractionReconstruction

Input: $\delta \in \mathbb{N}, H \in \mathsf{K}[x]^{m\times m}_{<2\delta}$
Output: $(N, D) \in \mathsf{K}[x]^{m\times m}_{\leq\delta}$ such that $ND^{-1} \equiv H \mod x^{2\delta}$
1: $F \leftarrow [H \quad -\mathrm{I}_m] \in \mathsf{K}[x]^{m\times 2m}$
2: ▹ *Computation of a minimal approximant basis $P \in \mathsf{K}[x]^{2m\times 2m}_{\leq 2\delta}$, see [12, Thm. 2.4], [18, Prop. 3.2]*
$P \leftarrow$ PM-Basis$(F^\mathsf{T}, 2\delta, 0)$, with P in weak Popov Form; $P \leftarrow P^\mathsf{T}$ ▹ PM-Basis *from [12]*
3: **return** $(P_{m+1..2m,1..m}, P_{1..m,1..m})$

Lemma 8.2. *Assume that $Y^\mathsf{T}S^{-1}X$ satisfies i) and ii) in Proposition 8.1. Given $H = \lceil Y^\mathsf{T}S^{-1}X\rceil_{2\delta/d}$ (x^d-adic notation here), Algorithm* FractionReconstruction *computes a minimal description $Y^\mathsf{T}S^{-1}X = ND^{-1}$ using $\tilde{O}(m^\omega\delta)$ arithmetic operations in K.*

Proof. Item *(iii)* of [34, Proposition 5.4] proves the correctness as soon as a correct approximant basis is computed at Step 2. This basis is obtained using $\tilde{O}(m^\omega\delta)$ operations [12, Thm. 2.4], [18, Prop. 3.2]. Following [34], transposes are used at Step 2 because in [12, 18] approximant bases are considered row-wise rather than column-wise. □

8.2. Resultant algorithm

We now present our Algorithm StructuredResultant that computes the resultant of two generic polynomials p and q whose Sylvester matrix is S. Once sufficiently many terms of the expansion of $Y^\mathsf{T}S^{-1}X$ are computed with Algorithm StructuredExpansion, Algorithm FractionReconstruction is called to compute a fraction description $Y^\mathsf{T}S^{-1}X = ND^{-1}$ whose denominator's determinant is the resultant up to a constant factor. This determinant is obtained using dense polynomial linear algebra [28], and the multiplicative constant is retrieved by comparing the determinant with the resultant at $x = 0$.

Our improved complexity bound is proved for generic p and q of degree $d < n$ in x and $n_p = n_q = n/2$ in y. More precisely, the resultant algorithm is correct with the prescribed cost if the following assumptions are satisfied.

(A1) $\det S(0) \neq 0$. This allows to choose $x = 0$ as expansion point. (Note that a truncated resultant algorithm which avoids this hypothesis is studied in [33].)

(A2) $p^{(0)}(0) \neq 0$ and $q^{(n_q)}(0) \neq 0$. These conditions are introduced in Section 4 in order to identify the structure of the high-order components and residues.

(A3) $\bar{F} \in \mathsf{K}^{n\times(d+1)}$ as defined in Proposition 4.4 has rank $d + 1$. This is assumed in Lemma 5.1 in order to compress the results of middle products into canonical generators.

(A4) The coefficients of p and q do not form a zero of the polynomial Φ of Proposition 8.1. This assumption ensures that an appropriate description of $Y^\mathsf{T}S^{-1}X$ can be recovered from a small number of terms in the expansion of the matrix fraction.

Algorithm 8.2 StructuredResultant

Input: $p, q \in \mathsf{K}[x, y]$ of degree $d < n$ in x and degree $n/2$ in y (n even), and $S \in \mathsf{K}[x]^{n\times n}_{\leq d}$ their associated Sylvester matrix; $m \leq n/2, r, s \in \mathbb{N}^*$ such that $rs \geq 4\lceil n/(2m)\rceil$
Genericity assumptions: (A1) to (A4)
Output: $\mathrm{Res}_y(p, q) \in \mathsf{K}[x]_{\leq nd}$
1: $H \leftarrow$ StructuredExpansion(p, q, m, r, s) ▹ $H = \lceil Y^\mathsf{T}S^{-1}X\rceil_{rs}$
2: $(N, D) \leftarrow$ FractionReconstruction(rsd, H) ▹ $ND^{-1} = Y^\mathsf{T}S^{-1}X$
3: $t \leftarrow \det D \in K[x]_{\leq nd}$ ▹ *Determinant computation from [28, Thm. 1.1]*
$c \leftarrow \det(S(0))/t(0) \in \mathsf{K}^*$ ▹ *Nonzero scalar to obtain the resultant*
4: **return** ct

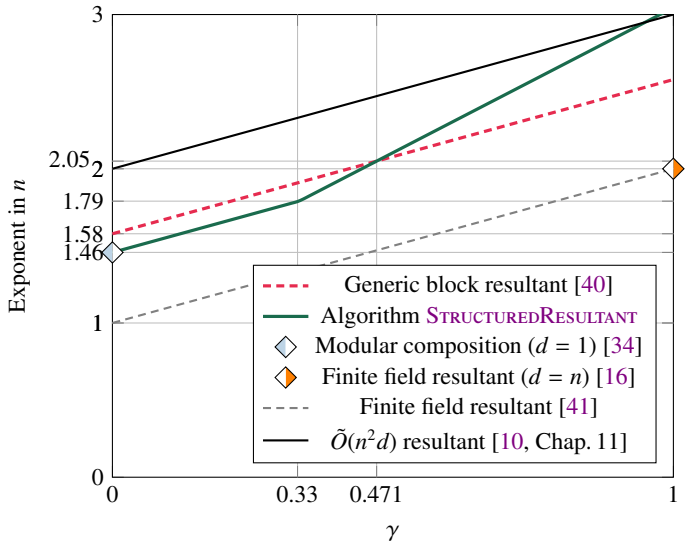
Lemma 8.3. *Let $p, q \in \mathsf{K}[x, y]$ of degree $d < n$ in x and degree $n/2$ in y (n even) and $m \leq n/2$, $r, s \in \mathbb{N}^*$. If the assumptions (A1) to (A4) hold and $rs \geq 4\lceil n/(2m)\rceil$ then Algorithm* StructuredResultant *computes the resultant of p and q with respect to y. With $s = O(r)$, $mr = O(n)$ the algorithm uses $\tilde{O}\left(\mathsf{MM}_{n,d}(r + d, r) + m^{\omega} rsd\right)$ arithmetic operations in K.*

Proof. The truncated expansion $H = \lceil Y^{\mathsf{T}} S^{-1} X\rceil_{rs}$ is computed at Step 1 in $\tilde{O}(\mathsf{MM}_{n,d}(r + d, r) + m^2 rsd)$ from Proposition 7.1, here we have used assumptions (A1) to (A3). Assumption (A4) ensures that Proposition 8.1 can be applied. Items *i*) and *ii*) of the latter proposition and Lemma 8.2 ensure that Step 2 is performed in time $\tilde{O}(m^{\omega} rsd)$ since $rs \geq 4\lceil n/(2m)\rceil = 2\delta/d$. Items *iii*) of the same proposition shows that $\mathrm{Res}_y(p, q) = ct$ for some nonzero $c \in \mathsf{K}$, which is computed by considering the constant terms of both polynomials at negligible cost. Note that by (A1) we know that $\det(S(0)) \neq 0$, hence $t(0) = \det(D(0)) \neq 0$. From *ii*) in Proposition 8.1, the degree of $D \in \mathsf{K}[x]^{m\times m}$ is bounded by δ, its determinant is also computed in time $\tilde{O}(m^{\omega} rsd)$ [28, Thm. 1.1]. □

The input parameters m, r and s of Algorithm StructuredResultant can be optimized with respect to n and d, which allows us to prove Theorem 1.1.

Proof of Theorem 1.1. Now taking p and q of y-degree n, for every $m \in \{1, \ldots, n\}$ we can associate to each of the assumptions (A1) to (A4) a non identically zero polynomial in $2(n + 1)(d + 1)$ variables over K, whose zeros are the inputs which do not meet the conditions. An appropriate hypersurface is defined by the product of these polynomials for a well chosen value of m. Consider $s = r$ and m such that $mr^2 \sim 2n$. From Lemma 8.3 and Eq. (25), the resultant can be computed generically using $\tilde{O}(nd(m^{\omega-1} + r^{\omega-1} + dr^{\omega-2}))$ operations. When $d \leq r$ the optimal choice for the parameters leads to $m = r$, and we arrive to the announced bound in the case $d = O(n^{1/3})$. For greater values of d, we consider $m = n^{\frac{\omega-2}{3\omega-4}} d^{\frac{2}{3\omega-4}}$ and keep the same relations between r, s and m. □

Figure 1. Comparison of known asymptotic cost bounds for the resultant of generic polynomials. Exponents in n with $d = n^{\gamma}$ and $\omega = 2.372$.



On Fig. 1 we compare the asymptotic exponent of Algorithm StructuredResultant to the exponents of existing algorithms for d varying in comparison to n. Note that the genericity conditions (in the Zariski sense) differ for each algorithm.

When $d = O(n^{1/3})$, our algorithm has an exponent in n that coincides with the one of [34] (without fast rectangular matrix multiplication), and allows to be essentially linear in the degree. In that case, Algorithm StructuredResultant compares favorably to [40] as soon as $\omega < 3$: the cost is $\tilde{O}(n^{1.458}d)$ with $\omega < 2.372$ [1, 9, 42]. Our new estimate breaks $\tilde{O}(n^{1.5}d)$, which is the cost estimate for [40] in the best possible case where ω would be 2. The cost bound in the second item of Theorem 1.1 has a stronger dependence in d, hence our new algorithm is not better for large values of d compared to n. However, it remains faster as long as $d = O(n^{(\omega-1)(4-\omega)/(2\omega)})$, hence $d = O(n^{0.47})$ for $\omega = 2.372$.

Algorithm StructuredResultant may be viewed as a generalization of [34, Sec. 10.1] for structured polynomial matrices with arbitrary degrees. On the other hand it also generalizes the resultant algorithm of [40] (up to a minor technical change in projections), which can be revealed by taking $s = 1$. With this parameter choice we have no more giant steps, neither application of high-order lifting. The computation of the expansion is essentially done in the baby steps at Step 1, this may be compared to the use of [40, Prop. 5.1] with truncated power series. For $s = 1$, the displacement rank of $\mathcal{R}$ at Step 3 is constant, and our bound $s+d$ for this rank (proof of Proposition 7.1) leads to an overestimation of the cost of the reconstruction from Step 4 to Step 6. Nevertheless, by taking into account the displacement rank simplification in this degenerate case, the resultant algorithm in [40] is recovered.

Acknowledgments

We thank the anonymous reviewers for detailed lists of comments and useful suggestions that helped us improve the quality of the manuscript.

References

[1] J. Alman and V. V. Williams. A Refined Laser Method and Faster Matrix Multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539. SIAM, 2021.
[2] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer, 1rst edition, 2003.
[3] B. Beckermann and G. Labahn. A Uniform Approach for the Fast Computation of Matrix-Type Padé Approximants. *SIAM J. Matrix Analysis and Applications*, 15(3):804–823, 1994.
[4] V. Bhargava, S. Ghosh, Z. Guo, M. Kumar, and C. Umans. Fast multivariate multipoint evaluation over all finite fields. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 221–232, 2022.
[5] A. Bostan, P. Flajolet, B. Salvy, and E. Schost. Fast computation of special resultants. *J. Symb. Comput.*, 41(1):1–29, 2006.
[6] A. Bostan, C.-P. Jeannerod, C. Mouilleron, and E. Schost. On Matrices With Displacement Structure: Generalized Operators and Faster Algorithms. *SIAM J. Matrix Anal. Appl.*, 38(3):733–775, 2017.
[7] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28(7):693–701, 1991.
[8] J. D. Dixon. Exact solution of linear equations using p-adic expansions. *Numer. Math.*, 40(1):137–141, 1982.
[9] R. Duan, H. Wu, and R. Zhou. Faster Matrix Multiplication via Asymmetric Hashing. arXiv:2210.10173, 2022.
[10] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999. Third edition 2013.
[11] J. von zur Gathen and T. Lücking. Subresultants revisited. *Theoretical Computer Science*, 297(1-3):199–239, 2003.
[12] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 135–142. ACM Press, 2003.
[13] G. Heinig and K. Rost. *Algebraic Methods for Toeplitz-like Matrices and Operator*. Springer, 1984.
[14] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.
[15] J. van der Hoeven and R. Larrieu. Fast Gröbner basis computation and polynomial reduction for generic bivariate ideals. *Appl. Algebr. Eng. Comm.*, 30(6):509–539, 2019.
[16] J. van der Hoeven and G. Lecerf. Fast computation of generic bivariate resultants. *J. of Complexity*, 62, 2021.
[17] O. Ibarra, S. Moran, and R. Hui. A generalization of the fast LUP matrix decomposition algorithm and applications. *Journal of Algorithms*, 3(1):45–56, 1982.
[18] C.-P. Jeannerod, V. Neiger, and G. Villard. Fast computation of approximant bases in canonical form. *J. Symb. Comput.*, 98:192–224, 2020.
[19] C.-P. Jeannerod, C. Pernet, and A. Storjohann. Rank-profile revealing Gaussian elimination and the CUP matrix decomposition. *J. Symb. Comput.*, 56:46–68, 2013.
[20] T. Kailath. *Linear Systems*. Prentice-Hall, 1980.
[21] T. Kailath, S. Y. Kung, and M. Morf. Displacement ranks of matrices and linear equations. *J. Math. Anal. Appl.*, 68(2):395–407, 1979.
[22] E. Kaltofen. Asymptotically fast solution of Toeplitz-like singular linear systems. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 297–304. ACM Press, 1994.
[23] E. Kaltofen and G. Villard. On the complexity of computing determinants. *Comput. Complex.*, 13(3):91–130, 2005.

[24] E. Kaltofen and G. Yuhasz. On the matrix Berlekamp-Massey algorithm. *ACM Trans. Algorithms*, 9(4):33:1–33:24, 2013.
[25] P. Karpman, C. Pernet, H. Signargout, and G. Villard. Computing the Characteristic Polynomial of Generic Toeplitz-like and Hankel-like Matrices. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 249–256. ACM Press, 2021.
[26] K. S. Kedlaya and C. Umans. Fast Polynomial Factorization and Modular Composition. *SIAM J. on Computing*, 40(6):1767–1802, 2011.
[27] W. Keller-Gehrig. Fast algorithms for the characteristic polynomial. *Theoretical computer science*, 36:309–317, 1985.
[28] G. Labahn, V. Neiger, and W. Zhou. Fast, deterministic computation of the Hermite normal form and determinant of a polynomial matrix. *J. Complexity*, 42:44–71, 2017.
[29] F. Le Gall. Faster Rectangular Matrix Multiplication by Combination Loss Analysis. arXiv:2307.06535, 2023.
[30] F. Le Gall and F. Urrutia. Improved Rectangular Matrix Multiplication using Powers of the Coppersmith-Winograd Tensor. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1029–1046. SIAM, 2018.
[31] G. Lecerf. On the complexity of the Lickteig-Roy subresultant algorithm. *J. Symb. Comput.*, 92:243–268, 2019.
[32] R. T. Moenck and J. H. Carter. Approximate algorithms to derive exact solutions to systems of linear equations. In *International Symposium on Symbolic and Algebraic Manipulation (Eurosam)*, LNCS 72, pages 65–73, 1979.
[33] G. Moroz and E. Schost. A Fast Algorithm for Computing the Truncated Resultant. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 341–348. ACM Press, 2016.
[34] V. Neiger, B. Salvy, É. Schost, and G. Villard. Faster Modular Composition. arXiv:2110.08354, to appear in J. ACM, 2021.
[35] M. Newman. *Integral Matrices*. Academic Press, 1972. First edition.
[36] V. Y. Pan. *Structured Matrices and Polynomials: Unified Superfast Algorithms*. Springer, 2001.
[37] M. Paterson and L. J. Stockmeyer. On the Number of Nonscalar Multiplications Necessary to Evaluate Polynomials. *SIAM J. Comput.*, 2(1):60–66, 1973.
[38] A. Storjohann. High-order lifting and integrality certification. *J. Symb. Comput.*, 36(3-4):613–648, 2003.
[39] 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.
[40] G. Villard. On Computing the Resultant of Generic Bivariate Polynomials. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 391–398. ACM Press, 2018.
[41] G. Villard. Elimination ideal and bivariate resultant over finite fields. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 526–534. ACM Press, 2023.
[42] V. V. Williams, Y. Xu, Z. Xu, and R. Zhou. New Bounds for Matrix Multiplication: from Alpha to Omega. arXiv:2307.07970, 2023.