A probabilistic numerical algorithm based on Terracini's lemma for proving nondefectivity of Segre varieties

Nick Vannieuwenhoven *Raf Vandebril*
Karl Meerbergen

Report TW 622, December 2012



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

A probabilistic numerical algorithm based on Terracini's lemma for proving nondefectivity of Segre varieties

Nick Vannieuwenhoven *Raf Vandebril*
Karl Meerbergen

Report TW 622, December 2012

Department of Computer Science, K.U.Leuven

Abstract

The CANDECOMP/PARAFAC (CP) tensor decomposition is a crucial data analysis and processing technique in applications and sciences. Tensors admitting a CP decomposition of rank at most s can be considered points of the sth order secant variety of a Segre variety, yet the most basic invariant of this variety—its dimension—is not yet fully understood. A conjecture was nevertheless proposed in [H. Abo, G. Ottaviani, and C. Peterson, Induction for secant varieties of Segre varieties. *Trans. Amer. Math. Soc.*, 361:767–792, 2009], proved to be correct for $s \leq 6$. We propose a memory efficient probabilistic numerical algorithm based on Terracini's Lemma to verify with high probability whether this conjecture is true. The proposed method can geometrically decrease the probability of incorrectly classifying a defective Segre variety as nondefective by performing more statistically independent checks. While increasing the precision of the calculations is not required to decrease the probability of an incorrect classification, a meticulous numerical analysis illustrates that the computations must, nevertheless, be performed with sufficiently high precision. The numerical experiments in this paper establish that the Segre varieties $\mathbb{PC}^{n_1} \times \mathbb{PC}^{n_2} \times \cdots \times \mathbb{PC}^{n_d}$ embedded in $\mathbb{PC}^{n_1 \times \cdots \times n_d}$ with generic rank $s \leq 55$ obey the conjecture; the probability that the obtained results are due to chance is less than 10^{-55}.

Keywords : Algebraic geometry, dimensions of secant varieties of Segre varieties, probabilistic algorithm, Terracini's lemma, finite precision calculations, error analysis
MSC : Primary : 14Q15, 14Q20, 65C05, 65F10, 14A25.

Noname manuscript No.
(will be inserted by the editor)

A probabilistic numerical algorithm based on Terracini's lemma for proving nondefectivity of Segre varieties

Nick Vannieuwenhoven · Raf Vandebril · Karl Meerbergen

the date of receipt and acceptance should be inserted later

Abstract The CANDECOMP/PARAFAC (CP) tensor decomposition is a crucial data analysis and processing technique in applications and sciences. Tensors admitting a CP decomposition of rank at most s can be considered points of the sth order secant variety of a Segre variety, yet the most basic invariant of this variety—its dimension—is not yet fully understood. A conjecture was nevertheless proposed in [H. Abo, G. Ottaviani, and C. Peterson, Induction for secant varieties of Segre varieties. *Trans. Amer. Math. Soc.*, 361:767–792, 2009], proved to be correct for $s \leq 6$. We propose a memory efficient probabilistic numerical algorithm based on Terracini's Lemma to verify with high probability whether this conjecture is true. The proposed method can geometrically decrease the probability of incorrectly classifying a defective Segre variety as nondefective by performing more statistically independent checks. While increasing the precision of the calculations is not required to decrease the probability of an incorrect classification, a meticulous numerical analysis illustrates that the computations must, nevertheless, be performed with sufficiently high precision. The numerical experiments in this paper establish that the Segre varieties $\mathbb{PC}^{n_1} \times \mathbb{PC}^{n_2} \times \cdots \times \mathbb{PC}^{n_d}$ embedded in $\mathbb{PC}^{n_1 \times \cdots \times n_d}$ with generic rank $s \leq 55$ obey the conjecture; the probability that the obtained results are due to chance is less than 10^{-55}.

Keywords Algebraic geometry · dimensions of secant varieties of Segre varieties · probabilistic algorithm · Terracini's lemma · finite precision calculations · error analysis

Mathematics Subject Classification (2000) 14Q15 · 14Q20 · 65C05 · 65F10 · 65G50 · 14A25

Nick Vannieuwenhoven · Raf Vandebril · Karl Meerbergen
Department of Computer Science, KU Leuven, B-3001 Heverlee, Belgium

Nick Vannieuwenhoven
E-mail: nick.vannieuwenhoven@cs.kuleuven.be

Raf Vandebril
E-mail: raf.vandebril@cs.kuleuven.be

Karl Meerbergen
E-mail: karl.meerbergen@cs.kuleuven.be

1 Introduction

Tensor decompositions [20, 26, 44, 55] have become a valuable data analysis and processing technique in various applications and sciences [37, 42]. The CANDECOMP/PARAFAC (CP) tensor decomposition [10,31], for instance, is particularly suited to applications arising from psychometrics, chemometrics, and signal processing [37, 42]. We say that a multidimensional array $\mathscr{A} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d}$, assuming without loss of generality that $n_1 \geq n_2 \geq \cdots \geq n_d \geq 2$, admits a rank s CP decomposition if

$$\mathscr{A} = \sum_{i=1}^{s} \lambda_i \mathbf{a}_i^1 \otimes \mathbf{a}_i^2 \otimes \cdots \otimes \mathbf{a}_i^d, \tag{1}$$

with $\lambda_i \in \mathbb{C}$, $\mathbf{a}_i^k \in \mathbb{C}^{n_k}$ of unit norm[1], and $\otimes$ the tensor product. It is assumed s is minimal. A tensor admitting a rank s CP decomposition may thus be parameterized by the λ_i and

$$A^{(k)} := \begin{bmatrix} \mathbf{a}_1^k & \mathbf{a}_2^k & \dots & \mathbf{a}_s^k \end{bmatrix}, \quad \text{for } 1 \leq k \leq d.$$

In the context of algebraic geometry, (1) may be considered a parameterization of an element of the sth order *secant variety* $\sigma_s(\mathscr{S})$ of the Segre variety $\mathscr{S} = \mathbb{PC}^{n_1} \times \mathbb{PC}^{n_2} \times \cdots \times \mathbb{PC}^{n_d} \subseteq \mathbb{PC}^{n_1 n_2 \cdots n_d}$, after choosing bases for the $(n_k - 1)$-dimensional projective spaces $\mathbb{PC}^{n_k}$.[2] The secant variety $\sigma_s(\mathscr{S})$ is defined as the Zariski closure of the union of the linear span of s points on $\mathscr{S}$ [38]:

$$\sigma_s(\mathscr{S}) = \overline{\bigcup_{p_1, p_2, \dots, p_s \in \mathscr{S}} \operatorname{span}(p_1, p_2, \dots, p_s)},$$

i.e., the Zariski closure of all tensors satisfying (1). The set of tensors satisfying (1) is not closed; see [21] for an extended discussion and its implications on the calculation of a CP decomposition. Letting

$$\Pi := \prod_{k=1}^{d} n_i \quad \text{and} \quad \Sigma := \sum_{k=1}^{d} n_i,$$

the *expected dimension* of $\sigma_s(\mathscr{S})$ is $\min\{s(\Sigma - d + 1) - 1, \Pi - 1\}$; i.e., the minimum of s times the dimension of the Segre variety $\mathscr{S}$, and the dimension of the ambient space. Unfortunately, the dimension of $\sigma_s(\mathscr{S})$ can be strictly less than this expected dimension, in which case we call $\mathscr{S}$ *s-defective*. A Segre variety is called *defective* if it is s-defective for some s. The defective secant varieties are presently not known in all cases, notwithstanding their intrinsic importance; for instance, a proof of nondefectivity for a Segre variety entails the nonexistence of the Schmidt–Eckart–Young decomposition for generic tensors in the ambient space [54]. Additionally, a defective sth order secant variety implies that a generic rank s tensor does not admit a unique CP decomposition, which is nonetheless a crucial prerequisite in data analysis applications for interpreting unambiguously the rank-one terms in a CP decomposition.

Defects of secant varieties of algebraic varieties have been studied classically, and recently much progress has been made for Segre [5, 11, 13, 16, 17, 39, 50], Segre–Veronese

[1] If $\mathbf{a} \in \mathbb{C}^n$ we define $\|\mathbf{a}\|^2 := \sum_{i=1}^{n} a_i \bar{a}_i$ as the squared 2-norm.

[2] In algebraic geometry it is usual to work over projective spaces, hereby considering $\mathscr{A} = \lambda \mathscr{A}$ if $\lambda \neq 0$, because the quantities of interest—e.g., degree, dimension, and rank—are invariant to scaling. In this paper, quotienting out this scalar factor will sometimes be referred to as "projectivization". In the numerical multilinear algebra community, projectivization is usually implicitly assumed in papers dealing with the uniqueness of the CP decomposition, when "scaling and permutation indeterminacies" are ignored [37]. These scaling indeterminacies are removed by defining the Segre variety in terms of projective spaces. See [38] for more connections between algebraic geometry and multilinear algebra.

[3, 4, 12, 15], and Grassmann [6, 14, 18, 40] varieties. The defective secant varieties of the Veronese variety were conjectured at least 150 years ago [9], and the conjecture was proven to be correct in 1995 by Alexander and Hirschowitz [7]. For the Segre variety, only a few defective cases are known, and it is conjectured, e.g., in [2, 5], that the following list of defective cases is complete:

Conjecture 1 (Abo, Ottaviani, and Peterson [5]) The Segre variety $\mathscr{S} = \mathbb{PC}^{n_1} \times \mathbb{PC}^{n_2} \times \cdots \times \mathbb{PC}^{n_d}$ is nondefective, unless one of the following holds:

1. $n_1 > 1 + \prod_{k=2}^{d} n_k - \sum_{k=2}^{d}(n_k - 1)$, i.e., $\mathscr{S}$ is unbalanced,
2. $\mathscr{S} = \mathbb{PC}^4 \times \mathbb{PC}^4 \times \mathbb{PC}^3$,
3. $\mathscr{S} = \mathbb{PC}^{2n+1} \times \mathbb{PC}^{2n+1} \times \mathbb{PC}^3$, $n \in \mathbb{N}$,
4. $\mathscr{S} = \mathbb{PC}^{n} \times \mathbb{PC}^{n} \times \mathbb{PC}^2 \times \mathbb{PC}^2$, $n \in \mathbb{N}$.

In [5], Abo, Ottaviani and Peterson verified that the sth order secant varieties of the Segre varieties are not defective when $s \leq 6$, except for the known defective cases in the above list. They proposed an inductive proof strategy amendable for automated computer processing, manually proving a number of base cases from which the induction can start. In theory, this approach can prove the nondefectivity of any Segre variety; however, the core reduction process employed is combinatorial in nature, thus limiting the practical applicability of this method for large ranks and large dimensions.

An approach complementary to the inductive procedure of [5] is also founded on Terracini's lemma [38, 52, 57]; this classic lemma provides a local characterization of the dimensions of secant varieties:

Theorem 1 (Terracini's Lemma) *Let $\mathscr{S} \subseteq \mathbb{PC}^N$ be a Segre variety, and let $p_1, p_2, \ldots, p_s \in \mathscr{S}$ be generic points. Let $T_p\mathscr{A}$ denote the projectivized tangent space of an algebraic variety $\mathscr{A} \subseteq \mathbb{PC}^N$ at $p \in \mathscr{A}$. If p is general in the linear span $\langle p_1, p_2, \ldots, p_s \rangle$, then*

$$\dim \sigma_s(\mathscr{S}) = \dim T_p\sigma_s(\mathscr{S}) = \dim\langle T_{p_1}\mathscr{S}, T_{p_2}\mathscr{S}, \ldots, T_{p_s}\mathscr{S}\rangle.$$

The utility of this lemma as a computational tool arises from the observation that it requires but *one* set of s points on the Segre variety, and one general point in their span where the dimension of the tangent space is maximal to *prove* that the sth order secant variety has the expected dimension. A well-known probabilistic algorithm, see, e.g., [2], to check whether the sth order secant variety of the Segre variety is nondefective consists of using a computer algebra system to implement a Monte Carlo algorithm that randomly selects s points on the Segre variety and verifies whether the rank of the matrix representing the tangent space is the expected one. The projectivized tangent space to $\mathscr{S} = \mathbb{PC}^{n_1} \times \mathbb{PC}^{n_2} \times \cdots \times \mathbb{PC}^{n_d}$ at $p_s = v_s^1 \otimes v_s^2 \otimes \cdots \otimes v_s^d$ is given by [5, 38]:

$$\begin{aligned} T_{p_s}\mathscr{S} := \mathbb{PC}^{n_1} \otimes v_s^2 \otimes \ldots \otimes v_s^d + v_s^1 \otimes \mathbb{PC}^{n_2} \otimes v_s^3 \otimes \cdots \otimes v_s^d + \cdots \\ + v_s^1 \otimes \cdots \otimes v_s^{d-1} \otimes \mathbb{PC}^{n_d}. \end{aligned} \tag{2}$$

One may then choose bases for the $\mathbb{PC}^{n_i}$ and parameterize point p_s by $\mathbf{a}_s^1 \otimes \mathbf{a}_s^2 \otimes \cdots \otimes \mathbf{a}_s^d$, where $\mathbf{a}_s^k \in \mathbb{C}^{n_k}$ is of norm unity;[3] herein, $\otimes$ denotes the Kronecker product. We note that the parameters may be chosen in any way, including from a strict subset of $\mathbb{C}^{n_k}$, e.g., $\mathbb{R}^{n_k}$, without affecting the correctness of the procedure;[4] for instance, the parameters in the parameterization may be drawn independently and identically distributed (i.i.d.) from a real

[3] As we are working over a projective variety, $\mathbf{a}^k$ is projectively equivalent with any other $\lambda\mathbf{a}^k$, $\lambda \neq 0$. We fix a representative by requiring that $\|\mathbf{a}^k\| = 1$.

[4] Choosing the subset too small may entail that true generic points of the variety may never be sampled.

standard normal distribution, *which we will assume throughout the paper*: all points will be sampled from the real Segre variety $\mathbb{PR}^{n_1} \times \mathbb{PR}^{n_2} \times \cdots \times \mathbb{PR}^{n_d}$ for reasons of efficiency. The span of the kth linear space appearing in (2) can be represented by the column span of the $\Pi \times n_k$ matrix

$$T_s^k := \mathbf{a}_s^1 \otimes \cdots \otimes \mathbf{a}_s^{k-1} \otimes I \otimes \mathbf{a}_s^{k+1} \otimes \cdots \otimes \mathbf{a}_s^d, \tag{3}$$

having chosen the standard basis for $\mathbb{R}^{n_k}$ for simplicity. As a consequence of Terracini's lemma, the span of $T_p\sigma_s(\mathscr{S})$ is given by the column span of the matrix

$$T_{(s)} := \left[T_{(s)}^1\ T_{(s)}^2\ \cdots\ T_{(s)}^d\right], \quad \text{where} \quad T_{(s)}^k := \left[T_1^k\ T_2^k\ \cdots\ T_s^k\right]. \tag{4}$$

Given this matrix representation of the tangent space, the dimension of the latter equals the rank of the former. It can be determined by computing the number of nonzero singular values [22]. If the rank of this matrix is the expected dimension, we have found a set of *witness points*: s points on the Segre variety where a general point in their span has the maximum dimension on the secant variety; these points are testimony to the maximal dimensionality of the variety and compose a proof of nondefectivity. Otherwise, no conclusions can be drawn; the variety may be defective, or the chosen points are not generic. Letting

$$r = \left\lfloor \frac{\Pi}{\Sigma - d + 1} \right\rfloor \quad \text{and} \quad R = \left\lceil \frac{\Pi}{\Sigma - d + 1} \right\rceil,$$

it suffices to check that the rth and Rth order secant varieties have the expected dimension to show that the Segre variety is *nondefective* [5]: we say that the Segre variety is *subabundant* if the rth order secant variety has the expected dimension, and, similarly, the Segre variety is *superabundant* if the Rth order secant variety fills the ambient space.

The above algorithm, which we refer to as the *classic algorithm*, can be implemented readily in any computer algebra system; see [2]. A major disadvantage, however, is that it should be executed in exact arithmetic. This is expensive on a computer system, as it requires symbolic calculation, the cost of which, in terms of memory and execution time, is not constant but increases with the complexity of the symbolic expression. Numerical calculations in floating point arithmetic, on the other hand, offer a constant memory and execution time cost, and are efficiently implemented on modern computer systems; however, the obvious disadvantage of any numerical algorithm is loss of exactness. The difficulties introduced by inexactness are subtle; for instance, due to rounding errors the matrix representation of the tangent space to a point on the secant variety, i.e., $T_{(r)}$, may not even correspond to the matrix representation of the tangent space of some other point on the secant variety. We say that computing $T_{(r)}$ is not *backward stable* [36].[5] Furthermore, due to roundoff errors, $T_{(r)}$ is almost surely of full rank: none of the exact singular values of $T_{(r)}$ will be precisely zero. The computation of the singular values introduces additional computational errors. It is presently not well understood insofar these issues influence the correctness of the dimension calculation. In [8], for instance, it is stated that:

> "In a floating point setting, one may compute the numerical rank of [the numerical approximation of $T_{(r)}$] by counting the number of singular values larger than some tolerance ε. Thus the numerical rank is a function of ε. This raises the natural question: *How to choose ε?* A general precise answer to this question is both unclear and application-dependent."

[5] If the matrix computed in floating point arithmetic were an actual matrix representation of the tangent space at some point on the secant variety, the situation would be much less severe, as the classic approach allows for freedom in the selection of the points on the Segre variety.

This paper describes a reliable implementation of the classic Monte Carlo algorithm in fixed-precision floating point arithmetic. Its contribution is twofold: first, the aforementioned difficulties in a straightforward numerical implementation are circumvented by considering a *probabilistic method* to determine the rank of a matrix; and, second, the probability of failure can be decreased geometrically to zero with *fixed memory demands*.

We propose a probabilistic algorithm based on Terracini's lemma that is well-suited for an implementation using fixed precision floating point arithmetic. In exact arithmetic, it almost never fails, i.e., it can incorrectly conclude that a defective variety is nondefective only with probability zero. Through a numerical analysis we demonstrate that the proposed algorithm can fail only with a small and quantifiable probability in a numerical setting. The unique feature of our algorithm is that this probability of failure can be decreased geometrically by increasing the number of statistically independent trials of which the algorithm is composed. It is thus not required to increase the precision of the numerical computations, which would increase execution times and memory costs, to improve certainty. In this manner, the probability of failure may be reduced to any user-specified threshold, without increasing memory demands.

In the next section, an alternative verification approach based on Terracini's lemma is proposed in the simple setting of exact arithmetic. Subsequently, modifications to the basic algorithm are proposed to handle the theoretical difficulties arising in the more involved case of inexact arithmetic. The basic algorithm is, in particular, modified so as to provide reliable error estimates. In section 4, we propose structure-exploiting and memory efficient operations for implementing the main steps of the aforementioned algorithm. Through the detailed numerical analysis of these structured operations, in section 5, we may practically implement, on a computer system, the proposed algorithm. We present the main results of our numerical experiments in section 6, and draw conclusions in section 7.

Concerning notation, the following conventions apply throughout. Matrices are typeset in uppercase (A, B), vectors in a bold face $(\mathbf{a}, \mathbf{b})$. The tensor product is denoted by $\otimes$; this notation is also used for the Kronecker product, which operates on a parameterization. The transpose of a matrix is denoted by $\cdot^T$. The Hadamard, or componentwise, product of matrices is defined as $(A \odot B)_{ij} = a_{ij} b_{ij}$. The standard inner product of $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ is denoted by $\langle \mathbf{a}, \mathbf{b} \rangle$, which immediately defines the Euclidian norm $\| \cdot \|$.

2 A probabilistic algorithm: exact arithmetic

In this section, we propose a variant of the classic algorithm based on Terracini's lemma that combines the checks for subabundance and superabundance. The outline of our method is as follows. First, an auxiliary $\Pi \times \Pi$ matrix is constructed whose range coincides with $T_{(R)}$ with probability one. Then, a Schur complement of the Gram matrix of this auxiliary matrix is considered, which admits an exploitable structure. Finally, a linear system is solved using a Krylov subspace method exploiting the structured Schur complement matrix; the norm of the obtained solution will reveal whether the matrix is nonsingular with probability one.

We note that the parameterization of the tangent space at point $p_i \in \mathscr{S}$, i.e., the $\Pi \times \Sigma$ matrix

$$T_i := \left[T_i^1 \; T_i^2 \; \cdots \; T_i^d \right],$$

cannot be of full rank because the dimension of the Segre variety $\mathscr{S}$ is at most $\Sigma - d + 1$. However, we can select $\Sigma - d + 1$ columns from T_i spanning its range by constructing a $\Sigma \times (\Sigma - d + 1)$ permutation matrix[6] P_i so that $\text{range}\,(T_i) = \text{range}\,(T_i P_i)$. If the rank of T_i is strictly smaller than $\Sigma - d + 1$, then we have selected a singular point of the Segre variety $\mathscr{S}$.

[6] In this context, a permutation matrix is a binary matrix whose columns contain precisely one 1.

In this case, we must select a different point $p_i \in \mathscr{S}$ and proceed as before; in the remainder it is assumed that all selected points are nonsingular. The matrix P_i can be chosen to be block diagonal permutation matrix:

$$P_i = \text{block-diag}\left(I_{n_1}, P_i^2, P_i^3, \ldots, P_i^d\right) \tag{5}$$

where P_i^k is of size $n_k \times n_k - 1$ and I_{n_1} the $n_1 \times n_1$ identity matrix. Indeed, T_i^1 is a matrix with orthonormal columns, hence I_{n_1} as the first diagonal block. For the other blocks we note that

$$\mathbf{a}_i^1 \otimes \cdots \otimes \mathbf{a}_i^d = T_i^1 \mathbf{a}_i^1 = T_i^k \mathbf{a}_i^k, \quad \text{for all } k \geq 2,$$

because of the multilinear property of the tensor product. Letting j be such that the jth component of $\mathbf{a}_i^k$ is nonzero, we conclude that P_i^k may be chosen as the I_{n_k} identity matrix whose jth column has been removed. Given the permutation matrices P_i for every point p_i, $1 \leq i \leq r \leq R$, we can set

$$\begin{aligned} P &= \text{block-diag}\left(I, \text{block-diag}\left(P_1^2, \ldots, P_r^2\right), \ldots, \text{block-diag}\left(P_1^d, \ldots, P_r^d\right)\right) \\ &:= \text{block-diag}\left(I, P_{(r)}^2, P_{(r)}^3, \ldots, P_{(r)}^d\right) \\ P &:= \text{block-diag}\left(I, P_\star\right), \end{aligned} \tag{6}$$

where I is of size $rn_1 \times rn_1$, and

$$P_\star := \text{block-diag}\left(P_{(r)}^2, P_{(r)}^3, \ldots, P_{(r)}^d\right), \tag{7}$$

so that $\text{range}\left(T_{(r)}\right) = \text{range}\left(T_{(r)}P\right)$. P is an $r\Sigma \times r(\Sigma - d + 1)$ matrix, so that the rth order secant variety is nondefective if $T_{(r)}P$ is of full column rank.[7] In the remainder, $T_{(r)}$ will be denoted as $\mathscr{T}$. An essential observation is that subabundance and superabundance may be verified simultaneously.

Theorem 2 *Let $c := \Pi - r(\Sigma - d + 1)$ and T_R be the parameterization of the tangent space of $p_R \in \mathscr{S}$. If there exists a matrix $C_R \in \mathbb{R}^{\Sigma - d + 1 \times c}$ of full column rank such that*

$$\text{rank}\left(\begin{bmatrix}\mathscr{T}P & T_R P_R C_R\end{bmatrix}\right) = \text{rank}\left(\begin{bmatrix}\mathscr{T} & T_R\end{bmatrix}\right) = \Pi, \tag{8}$$

then the Segre variety $\mathscr{S}$ is nondefective, i.e., the rth order secant variety is subabundant and the Rth order secant variety is superabundant.

Conversely, if $\mathscr{S}$ is nondefective and the points $p_i \in \mathscr{S}$, $i = 1, \ldots, R$ are generic, then a matrix $C_R \in \mathbb{R}^{\Sigma - d + 1 \times c}$ whose entries are drawn i.i.d. from the standard normal distribution $N(0,1)$ satisfies (8) with probability one.

Proof Assume there exists a matrix C_R such that (8) holds. From

$$\begin{aligned} \Pi = \text{rank}\left(\begin{bmatrix}\mathscr{T}P & T_R P_R C_R\end{bmatrix}\right) &\leq \text{rank}\left(\mathscr{T}P\right) + \text{rank}\left(T_R P_R C_R\right) \\ &\leq r(\Sigma - d + 1) + \Pi - r(\Sigma - d + 1) = \Pi, \end{aligned}$$

it is clear that the rth order secant variety is subabundant because $\mathscr{T}P$ has to be of full rank for the left equality to hold. Because $\text{range}\left(T_R P_R C_R\right) \subseteq \text{range}\left(T_R\right)$ and the left equality holds, we know that $\begin{bmatrix}\mathscr{T}P & T_R\end{bmatrix}$ is also of full rank, so that the Rth order secant variety is superabundant. This proves the first part of the theorem.

[7] The converse is not true, because the selected points on $\mathscr{S}$ are not known to be generic.

Conversely, as the points on the Segre variety are generic, and the variety is nondefective, the last equality in (8) holds vacuously. Let Q_1 be an orthogonal basis for range$(\mathscr{T}) =$ range$(\mathscr{T}P)$, and let Q_2 be a matrix with orthonormal columns such that $\begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$ is an orthogonal basis of $\mathbb{R}^{\Pi}$. We may then write

$$T_R P_R = Q_1 Q_1^T T_R P_R + Q_2 Q_2^T T_R P_R := Q_1 Q_1^T T_R P_R + Q_2 B,$$

where $B = Q_2^T T_R P_R$. By assumption, B is of full rank; otherwise, adding p_R would not cause $\mathscr{S}$ to be superabundant, which is contradictory. Thus, the singular value decomposition of B may be written as

$$B = USV^T \quad \text{where} \quad U \in \mathbb{R}^{c\times c},\ S \in \mathbb{R}^{c\times c}, \text{ and } \quad V \in \mathbb{R}^{\Sigma-d+1\times c}.$$

Next, we note that both

$$Q_2' := Q_2 U S \quad \text{and} \quad C_R' := V^T C_R$$

are, with probability one, matrices of rank c: Q_2' is a matrix with c orthogonal columns; and $C_R' \in \mathbb{R}^{c\times c}$ is an orthogonal transformation of Gaussian i.i.d. random variables, which is again standard normally distributed [23, §4.2], i.e., $(C_R')_{ij} \sim N(0,1)$. Consequently, applying the transformation C_R on the right of $(I - Q_1 Q_1^T) T_R P_R$ does not decrease its rank; hence,

$$\begin{aligned} \Pi = \operatorname{rank}\left(\begin{bmatrix}\mathscr{T} & T_R\end{bmatrix}\right) &= \operatorname{rank}(\mathscr{T}) + \operatorname{rank}\left((I - Q_1 Q_1^T) T_R\right) \\ &= \operatorname{rank}(\mathscr{T}P) + \operatorname{rank}\left((I - Q_1 Q_1^T) T_R P_R C_R\right) \\ &= \operatorname{rank}\left(\begin{bmatrix}\mathscr{T}P & T_R P_R C_R\end{bmatrix}\right), \end{aligned}$$

with probability one, completing the proof. □

The above theorem halves the required number of rank computations in the classic algorithm based on Terracini's lemma. Consider now the matrix

$$\begin{bmatrix} T^1 & T_R & T^2 & T^3 & \cdots & T^d \end{bmatrix} \cdot \text{block-diag}\left(I, P_R C_R, P_{(r)}^2, P_{(r)}^3, \ldots, P_{(r)}^d\right),$$

whose Gram matrix is

$$G = \begin{bmatrix} D_1 & L_{1,2}^T & L_1^T \\ L_{1,2} & D_2 & L_2^T \\ L_1 & L_2 & S \end{bmatrix}, \tag{9}$$

where the matrices on the diagonal are given by

$$\begin{aligned} D_1 &= {T^1}^T T^1, \\ D_2 &= C_R^T P_R^T T_R^T T_R P_R C_R, \\ S := P_\star^T S^\star P_\star &= P_\star^T \begin{bmatrix} {T^2}^T T^2 & {T^2}^T T^3 & \cdots & {T^2}^T T^d \\ {T^3}^T T^2 & {T^3}^T T^3 & \cdots & {T^3}^T T^d \\ \vdots & \vdots & \ddots & \vdots \\ {T^d}^T T^2 & {T^d}^T T^3 & \cdots & {T^d}^T T^d \end{bmatrix} P_\star, \end{aligned}$$

and the off-diagonal matrices are:

$$L_{1,2} := C_R^T P_R^T L_{1,2}^\star = C_R^T P_R^T T_R^T T^1,$$

$$L_1 := P_\star^T L_1^\star = P_\star^T \begin{bmatrix} {T^2}^T T^1 \\ {T^3}^T T^1 \\ \vdots \\ {T^d}^T T^1 \end{bmatrix}, \text{ and } L_2 := P_\star^T L_2^\star P_R C_R = P_\star^T \begin{bmatrix} {T^2}^T T_R \\ {T^3}^T T_R \\ \vdots \\ {T^d}^T T_R \end{bmatrix} P_R C_R.$$

For future reference, we define

$$G = \begin{bmatrix} D & L^T \\ L & S \end{bmatrix}, \quad \text{where} \quad D = \begin{bmatrix} D_1 & L_{1,2}^T \\ L_{1,2} & D_2 \end{bmatrix} \quad \text{and} \quad L = \begin{bmatrix} L_1 & L_2 \end{bmatrix}.$$

We may then apply the Guttman–Haynsworth rank additivity formula [30, 32] twice to the Gram matrix G:[8]

$$\begin{aligned} \operatorname{rank}\left(\begin{bmatrix} \mathscr{T} & T_R \end{bmatrix}\right) &= \operatorname{rank}(G) \\ &= \operatorname{rank}(D) + \operatorname{rank}(G/D) \\ &= \operatorname{rank}(D_1) + \operatorname{rank}(D/D_1) + \operatorname{rank}(G/D), \end{aligned}$$

where the Schur complements are defined by

$$D/D_1 := D_2 - L_{1,2} D_1^{-1} L_{1,2}^T \quad \text{and} \quad G/D := S - L D^{-1} L^T.$$

It is straightforward to show that

$$G/D = S - L_1 D_1^{-1} L_1^T - (L_2 - L_1 D_1^{-1} L_{1,2}^T)(D/D_1)^{-1}(L_2 - L_1 D_1^{-1} L_{1,2}^T)^T, \tag{10}$$

using, e.g., the direct inversion formula of a partitioned block matrix [30]. Note that inertia[9] is additive on the Schur complement [32]; thus, the eigenvalues of G, D and G/D are all positive and real. In section 5, it will be shown that explicitly computing the rank of D_1 and D/D_1 is not necessary in a numerical setting, because a possible singularity of these matrices can be detected while computing the rank of G/D. For establishing that G/D is of *full rank*, or, alternatively, whether G/D has a zero-dimensional null space, we propose to verify that

$$(G/D)\mathbf{x} = 0 \tag{11}$$

admits only the trivial solution. This *symmetric positive semidefinite*[10] system can be solved using a Krylov subspace method with a well-chosen starting vector. The key idea of these iterative methods for solving a linear system $A\mathbf{x} = \mathbf{b}$, with $A \in \mathbb{R}^{n\times n}$ and symmetric positive semidefinite, and $\mathbf{b} \in \operatorname{range}(A)$, consists of generating iteratively a sequence of Krylov spaces

$$\mathscr{K}_k(A, \mathbf{r}) := \operatorname{range}\left(\mathbf{r}, A\mathbf{r}, A^2\mathbf{r}, \ldots, A^{k-1}\mathbf{r}\right) \subseteq \operatorname{range}(A),$$

with $\mathbf{r} = \mathbf{b} - A\mathbf{x}^{(0)}$ and $\mathbf{x}^{(0)}$ any *starting vector*, and choosing an approximation $\mathbf{x}^{(k)}$ from the affine space $\mathbf{x}^{(0)} + \mathscr{K}_k(A, \mathbf{r})$; see, e.g., [47]. Krylov spaces are nested:

$$\mathscr{K}_1(A, \mathbf{r}) \subset \mathscr{K}_2(A, \mathbf{r}) \subset \cdots \subset \mathscr{K}_m(A, \mathbf{r}) = \mathscr{K}_{m+1}(A, \mathbf{r}) = \ldots = \mathscr{K}_\infty(A, \mathbf{r});$$

[8] In principle, the first equality holds only with probability one. With probability zero, the rank of G may be strictly less than that of $\mathscr{T}$; however, it can be verified that this situation does not alter the probability of failure of the proposed algorithm (see Algorithm 2).

[9] The triplet of the number of negative, zero, and positive eigenvalues of a matrix.

[10] All eigenvalues of a matrix are consequently real and positive.

m is called the *grade* of $\mathbf{b}$ with respect to A, which is bounded from above by n in exact arithmetic. There are various Krylov methods differing in the way the approximation $\mathbf{x}^{(k)}$ is chosen; one consists of choosing

$$\mathbf{b} - A\mathbf{x}^{(k)} \perp \mathscr{K}_k(A, \mathbf{r}),$$

where orthogonality with respect to the standard inner product is implied: $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_i x_i y_i$. This method is known as the conjugate gradient (CG) method [33]; it is presented as Algorithm 1.

Algorithm 1: CONJUGATEGRADIENT(A, $\mathbf{b}$, $\mathbf{x}^{(0)}$).
input : Symmetric positive semidefinite matrix $A \in \mathbb{R}^{n\times n}$, right hand side $\mathbf{b} \in \text{range}\,(A)$, and a starting vector $\mathbf{x}^{(0)}$.
output: The CG solution of $A\mathbf{x} = \mathbf{b}$ with starting vector $\mathbf{x}^{(0)}$.
$\mathbf{r}^{(0)} \leftarrow \mathbf{b} - A\mathbf{x}^{(0)}$
$\mathbf{p}^{(0)} \leftarrow \mathbf{r}^{(0)}$
for $k \leftarrow 1$ *to* n **do**

$$\gamma_{k-1} \leftarrow \frac{\|\mathbf{r}^{(k-1)}\|^2}{\left\langle \mathbf{p}^{(k-1)}, A\mathbf{p}^{(k-1)} \right\rangle}$$
$$\mathbf{x}^{(k)} \leftarrow \mathbf{x}^{(k-1)} + \gamma_{k-1}\mathbf{p}^{(k-1)}$$
$$\mathbf{r}^{(k)} \leftarrow \mathbf{r}^{(k-1)} - \gamma_{k-1}A\mathbf{p}^{(k-1)}$$
$$\beta_k \leftarrow \frac{\|\mathbf{r}^{(k)}\|^2}{\|\mathbf{r}^{(k-1)}\|^2}$$
$$\mathbf{p}^{(k)} \leftarrow \mathbf{r}^{(k)} + \beta_k\mathbf{p}^{(k-1)}$$

end

It is well-established that the CG algorithm ignores the null space of A, which we can exploit to probabilistically verify whether the dimension of the null space of G/D is nonzero:

Lemma 1 *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive semidefinite matrix and $\mathbf{x}^{(0)} \in \mathbb{R}^n$ a vector whose components were drawn i.i.d. from a standard normal distribution. Let $\mathbf{x}^*$ be the conjugate gradient solution to $A\mathbf{x} = 0$ with starting vector $\mathbf{x}^{(0)}$. If $\|\mathbf{x}^*\| = 0$ then A is nonsingular with probability one.*

Proof As A is real symmetric, it admits an eigenvalue decomposition

$$A = VDV^T,$$

with $V \in \mathbb{R}^{n\times n}$ orthogonal and $D \in \mathbb{R}^{n\times n}$ a diagonal matrix whose diagonal elements are sorted in order of decreasing magnitude. We may then write

$$\mathbf{x}^{(0)} = VV^T\mathbf{x}^{(0)} = \sum_{i=1}^{s} \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle \mathbf{v}_i + \sum_{i=s+1}^{n} \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle \mathbf{v}_i$$

where s is the rank of A, i.e., the number of nonzero eigenvalues. Because the components of $\mathbf{x}^{(0)}$ were drawn statistically independent from A and because Gaussian distributions are invariant under orthogonal transformations [23, §4.2], it follows that

$$\left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle \sim N(0,1), \tag{12}$$

i.e., they are also standard normally distributed. Consequently, if $s \neq n$, $\mathbf{x}^{(0)}$ has a nonzero coefficient with respect to $\mathbf{v}_i$, $s+1 \leq i \leq n$, with probability one. It is well-known that, in exact arithmetic, Algorithm 1 produces a solution $\mathbf{x}^*$ such that $A\mathbf{x}^* = 0$ when it is applied to the (possibly singular) linear system $A\mathbf{x} = 0$. It is furthermore trivial to prove by induction that

$$\mathbf{r}^{(k)} \in \operatorname{range}(A) \quad \text{and} \quad \mathbf{p}^{(k)} \in \operatorname{range}(A)$$

for all $0 \leq k \leq n$. Letting

$$\mathbf{p}^{(k)} := \sum_{i=1}^{s} \phi_i^{(k)} \mathbf{v}_i$$

we find by definition of the CG algorithm that

$$\mathbf{x}^{(k)} = \sum_{i=1}^{s} \left(\left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle + \sum_{j=1}^{k} \gamma_{j-1} \phi_i^{(j-1)} \right) \mathbf{v}_i + \sum_{i=s+1}^{n} \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle \mathbf{v}_i,$$

i.e., components of $\mathbf{x}^{(0)}$ in the null space of A remain unaffected by the CG algorithm. Finally, because $A\mathbf{x}^{(n)} = 0$, it follows that

$$\mathbf{x}^{(n)} = \sum_{i=s+1}^{n} \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle \mathbf{v}_i \quad \text{and} \quad \|\mathbf{x}^{(n)}\|^2 = \sum_{i=s+1}^{n} \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle^2.$$

The squared norm of the CG solution $\mathbf{x}^* = \mathbf{x}^{(n)}$ thus satisfies the chi-squared distribution with $n-s$ degrees of freedom. Consequently, if A is nonsingular then $\|\mathbf{x}^{(n)}\| = 0$ deterministically; if A is singular $\|\mathbf{x}^{(n)}\| = 0$ with probability zero, and $\|\mathbf{x}^{(n)}\| > 0$ with probability one. □

The above theorem allows us to design a statistical hypothesis test. Assume as null hypothesis that G/D is a singular matrix, i.e., the Segre variety is defective, and thus $\|\mathbf{x}^*\|^2$ is chi-squared distributed with $n-s$ degrees of freedom. The null hypothesis is rejected when the probability of observing the data is below a user-specified threshold α; i.e., if

$$\mathrm{P}\left(x \leq \|\mathbf{x}^*\|^2 \mid G/D \text{ is singular}\right) \leq \alpha,$$

where $x \sim \chi^2_{n-s}$ with $n-s \geq 1$ under the null hypothesis. In the exact arithmetic setting, we propose to take $\alpha = 0$, which immediately entails that the type I error[11] of this test is zero. Indeed, $\|\mathbf{x}^*\| = 0$ can occur only with probability zero if G/D is singular, because only starting vectors $\mathbf{x}^{(0)}$ that have no components in the null space, i.e., $\langle \mathbf{x}^{(0)}, \mathbf{v}_i \rangle = 0$ with $i \geq s+1$, can cause the solution $\mathbf{x}^* = 0$ for a singular matrix; however, these inner products are standard normally distributed, hence the probability of selecting such a vector is zero. Interestingly, the type II error[12] of this hypothesis test is also zero: if the null hypothesis is false, i.e., G/D is nonsingular, $\|\mathbf{x}^*\| = 0$ deterministically, so the hypothesis test rejects the null hypothesis. This implies that the described hypothesis test correctly distinguishes between singular and nonsingular G/D with probability one. It should be stressed that the test is *not deterministically correct*. With probability zero, it can incorrectly decide that G/D is nonsingular if the starting vector $\mathbf{x}^{(0)}$ has no components in the null space of G/D. The aforementioned hypothesis test is the basis for the proposed probabilistic algorithm in exact arithmetic. It is presented as Algorithm 2.

If the Segre variety is nondefective and the selected points are generic, the algorithm will output NONDEFECTIVE with probability one, and UNKNOWN with probability zero. If

[11] Rejecting the null hypothesis when it is true.

[12] Not rejecting the null hypothesis when it is false.

Algorithm 2: A probabilistic projection algorithm based on Terracini's lemma

input : A Segre variety $\mathscr{S} = \mathbb{P}^{n_1} \times \cdots \times \mathbb{P}^{n_d}$.

Set $\Sigma \leftarrow \sum_{k=1}^{d} n_k$, $\Pi \leftarrow \prod_{k=1}^{d} n_k$, $r \leftarrow \lfloor \Pi/(\Sigma - d + 1) \rfloor$ and $R \leftarrow \lceil \Pi/(\Sigma - d + 1) \rceil$
for $i \leftarrow 1$ *to* R **do**
    Generate random smooth point $p_i \in \mathscr{S}$ and compute T_i
    Compute a $\Sigma \times (\Sigma - d + 1)$ matrix P_i such that range(T_i) = range$(T_i P_i)$
end
if rank$(D_1) \neq n_1 r$ ***or*** rank$(D/D_1) \neq \Pi - r(\Sigma - d + 1)$ **then**
    return UNKNOWN
end
Generate $\mathbf{x}^{(0)}$ with $x_i^{(0)} \sim N(0,1)$ standard normally distributed
$\mathbf{x}^* \leftarrow$ CONJUGATEGRADIENT$(G/D, 0, \mathbf{x}^{(0)})$
if $\|\mathbf{x}^*\| \neq 0$ **then**
    return UNKNOWN
end
return NONDEFECTIVE

the variety is nondefective but the points are nongeneric, the algorithm outputs UNKNOWN. If the variety is defective, the algorithm outputs UNKNOWN with probability one, and can output NONDEFECTIVE with probability zero. The proposed algorithm may thus incorrectly decide that a defective variety is nondefective; we say that the algorithm *fails*. The algorithm *almost never* fails: a failure occurs with probability zero. The classic algorithm, on the other hand, *never* fails. We remark that an incorrect classification is due to a unfortunate choice of $\mathbf{x}^{(0)}$, occurring with probability zero, which can be alleviated by rerunning the algorithm with a different $\mathbf{x}^{(0)}$.

3 A probabilistic algorithm: inexact arithmetic

In this section, we discuss modifications to Algorithm 2 for making it trustworthy in a situation where the calculations are performed in inexact arithmetic; for example, fixed precision floating point arithmetic. Essentially, it will no longer be possible to choose the α-value of the statistical test equal to zero, so that there is a nonzero type I error associated with the test. The key feature of the modified algorithm is that this type I error can be decreased by performing more tests, rather than by requiring more precise arithmetic that would increase memory demands.

Assume that we apply the conjugate gradient method in inexact arithmetic for finding an approximate solution of $(G/D)\mathbf{x} = 0$ with starting vector $\mathbf{x}^{(0)}$. Our key observation is that if both the norm of the obtained inexact approximation and the calculation errors are small, the actual starting vector $\mathbf{x}^{(0)}$ must have had small components in the null space, i.e., the norm of the projection of the starting vector onto the null space of G/D is small. This is presented as

Lemma 2 *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive semidefinite matrix and $\mathbf{x}^{(0)} \in \mathbb{R}^n$ a vector whose components were drawn i.i.d. from a standard normal distribution. Let $\underline{\mathbf{x}}^{(m)}$ be the approximate solution to $A\mathbf{x} = 0$ obtained by applying the conjugate gradient method in inexact arithmetic with starting vector $\mathbf{x}^{(0)}$, and let $\mathbf{x}$ be the true solution obtained by applying the CG method in exact arithmetic to the starting vector $\mathbf{x}^{(0)}$. Let $A = VDV^T$ be the eigenvalue decomposition of A where the diagonal elements of D are sorted by decreasing magnitude, so that*

$$\underline{\mathbf{x}}^{(m)} = \sum_{i=1}^{n} \left\langle \underline{\mathbf{x}}^{(m)}, \mathbf{v}_i \right\rangle \mathbf{v}_i \quad \textit{and} \quad \mathbf{x} = \sum_{i=s+1}^{n} \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle \mathbf{v}_i,$$

where s is the rank of A. Let

$$\eta^{(m)} := \|\underline{\mathbf{x}}^{(m)}\| \quad \textit{and} \quad \chi^{(m)} := \sqrt{\sum_{i=s+1}^{n} \left(\left\langle \underline{\mathbf{x}}^{(m)}, \mathbf{v}_i \right\rangle - \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle\right)^2}.$$

Then,

$$\left|\left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle\right| \leq \eta^{(m)} + \chi^{(m)}, \quad \textit{for } i \geq s+1.$$

Proof Let $\underline{\mathbf{x}}^{(m)} := \mathbf{x} + \mathbf{f}$. If $i \geq s+1$, then

$$\left|\left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle\right| = |\left\langle \mathbf{x}, \mathbf{v}_i \right\rangle| = \left|\left\langle \underline{\mathbf{x}}^{(m)}, \mathbf{v}_i \right\rangle - \left\langle \mathbf{f}, \mathbf{v}_i \right\rangle\right| \leq \left|\left\langle \underline{\mathbf{x}}^{(m)}, \mathbf{v}_i \right\rangle\right| + |\left\langle \mathbf{f}, \mathbf{v}_i \right\rangle| \leq \eta^{(m)} + \chi^{(m)},$$

where the last inequality follows from the Cauchy–Schwarz inequality and

$$\left\langle \mathbf{f}, \mathbf{v}_i \right\rangle^2 = \left\langle \underline{\mathbf{x}}^{(m)} - \mathbf{x}, \mathbf{v}_i \right\rangle^2 = \left(\left\langle \underline{\mathbf{x}}^{(m)}, \mathbf{v}_i \right\rangle - \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle\right)^2 \leq \left(\chi^{(m)}\right)^2.$$

□

In Theorem 3, we propose a computable three-term recurrence that bounds $\chi^{(m)}$ from above; however, that analysis depends on the characteristics of the underlying inexact arithmetic scheme and is therefore delayed.

The *a posteriori* bound on the magnitude of the components of the starting vector in the null space of G/D presented in the previous theorem can then be used to design a statistical hypothesis test as in the previous section. As null hypothesis, we assume that G/D is singular, i.e., the Segre variety is defective. We thus know that $\langle \mathbf{x}^{(0)}, \mathbf{v}_i \rangle$ is standard normally distributed for $i \geq s+1$. The null hypothesis will again be rejected if the probability of observing the data under the assumption of the null hypothesis is below a user-specified threshold α, ensuring that the type I error of the hypothesis test is α. Ideally, we would thus verify whether

$$\mathrm{P}\left(|x_1| \leq |\langle \mathbf{x}^{(0)}, \mathbf{v}_{s+1} \rangle| \wedge \ldots \wedge |x_{n-s}| \leq |\langle \mathbf{x}^{(0)}, \mathbf{v}_n \rangle| \mid G/D \text{ is singular}\right) \leq \alpha,$$

where $x_i \sim N(0,1)$ and statistically independent because the standard normal distribution is invariant to orthogonal transformations [23]. Because of the latter observation, we may write

$$\begin{aligned}\mathrm{P}\left(|x_1| \leq |\langle \mathbf{x}^{(0)}, \mathbf{v}_{s+1} \rangle| \wedge \ldots \wedge |x_{n-s}| \leq |\langle \mathbf{x}^{(0)}, \mathbf{v}_n \rangle| \mid G/D \text{ is singular}\right) = \\ \mathrm{P}\left(|x_1| \leq |\langle \mathbf{x}^{(0)}, \mathbf{v}_{s+1} \rangle| \mid G/D \text{ is singular}\right) \cdots \mathrm{P}\left(|x_{n-s}| \leq |\langle \mathbf{x}^{(0)}, \mathbf{v}_n \rangle| \mid G/D \text{ is singular}\right).\end{aligned}$$

Unfortunately, this probability cannot be computed as it involves quantities that cannot be computed exactly in inexact arithmetic. However, from the previous lemma it is clear that the above probability can be bounded from above by

$$\mathrm{P}\left(|x| \leq \eta^{(m)} + \chi^{(m)} \mid G/D \text{ is singular}\right)^{n-s} \leq \mathrm{P}\left(|x| \leq \eta^{(m)} + \chi^{(m)} \mid G/D \text{ is singular}\right),$$

with $x \sim N(0,1)$, and where the weaker upper bound is also necessary in practice because the value of s is unknown; assuming the null hypothesis, however, it is known to be at most $n-1$. The weaker upper bound can then itself be bounded by $\eta^{(m)} + \chi^{(m)}$, which is readily computable. We thus propose to reject the null hypothesis whenever

$$\eta^{(m)} + \chi^{(m)} \leq \alpha.$$

Otherwise, the null hypothesis is retained, and the test is inconclusive. In this manner, the type I error will be bounded from above by α; however, in practice the type I error will be substantially smaller if $n-s>1$. Note that the type II error could be large with this test.

To decrease the type I error of the aforementioned hypothesis test, we could simply reduce α. This is an effective technique, however, it cannot be applied indefinitely in a practical manner. Because the upper bound in Lemma 2 involves the magnitude of the computation errors, i.e., $\chi^{(m)}$, decreasing α usually requires us to decrease the computation errors. We pursue a complementary strategy; provided that the computation errors are sufficiently small—a notion we will investigate in section 5—we may design an alternative test with a vastly reduced type I error by performing additional experiments with different starting vectors. Let k be a user-specified number of trials, then we generate k starting vectors $\{\mathbf{x}_j^{(0)}\}_{j=1}^k$. For each of these k starting vectors, we solve the system $(G/D)\mathbf{x}=0$ using the conjugate gradient method in inexact arithmetic. If we then reject the null hypothesis only if *all* of the obtained upper bounds $\{\eta_j^{(m)}+\chi_j^{(m)}\}_{j=1}^k$, as in Lemma 2, are smaller than α, then it is clear that the probability of obtaining a random observation[13] in the rejection region by chance is at most α^k. We thus propose to reject the null hypothesis if *all* of the obtained upper bounds in Lemma 2 for the k starting vectors are smaller than α. The type I error of this hypothesis test is then α^k; by increasing k, the type I error can be reduced geometrically to zero. This proves:

Lemma 3 *Let $\alpha<1$ be fixed. Assume that k approximate solutions of $(G/D)\mathbf{x}=0$ were obtained by the conjugate gradient method in inexact arithmetic with k starting vectors whose components were drawn i.i.d. from a standard normal distribution. Consider Lemma 2 and let*

$$\delta := \max_{1\le j\le k}\left\{\eta_j^{(m)}+\chi_j^{(m)}\right\}.$$

If we reject the null hypothesis that G/D is singular whenever $\delta\le\alpha$, then the type I error is at most α^k.

The above discussion suggests minor modifications to Algorithm 2 to make it reliable in inexact arithmetic. First, in line 10, the conjugate gradient algorithm should return the *a posteriori* bound presented in Lemma 2. Lines 9 and 10 should then be repeated k times, and the test in line 11 should test whether any of the returned *a posteriori* bounds is larger in magnitude than the user-imposed threshold α. Finally, in line 14, the algorithm should additionally report the type I error rate: α^k.

It is well-known that the conjugate gradient algorithm usually does not converge within n steps for an $n\times n$ symmetric positive semidefinite matrix in inexact arithmetic [41]. It is thus necessary to include an appropriate stopping criterion. In our case, we propose to stop the conjugate gradient algorithm at step t if the computation errors have become greater than the norm of the current approximation, i.e., if $\chi^{(t)}\ge\eta^{(t)}$, or if the upper bound on the components in the null space is sufficiently small to reject the null hypothesis, i.e., if $\eta^{(t)}+\chi^{(t)}\le\alpha$. Note that the latter approach is valid, because Lemma 2 applies for any t; thus, the most restrictive bound applies, and we may stop if we encounter a bound that is sufficiently restrictive.

4 Structure exploiting operations

The key ingredient of the proposed method is using the conjugate gradient method to solve a linear system with G/D. A major advantage of this method is that it is matrix-free: the

[13] In this hypothesis test, a random observation refers to each of the k obtained solutions with the different starting vectors.

matrix G/D does not need to be stored explicitly. The CG method only requires an efficient implementation of a matrix-vector product with G/D. This section is describes a structure-exploiting implementation for this operation without computing G/D explicitly. An analysis of the memory cost of the proposed algorithm is presented and compared with the classic algorithm. Throughout this section, exact arithmetic is assumed; the behavior of the algorithm in floating point arithmetic is investigated in the next section.

We detail the three steps in the proposed algorithm that may benefit from structure-exploiting operations in the next subsections. First, we investigate a procedure to generate points on the Segre variety. Next, the rank computation of D is explained. Finally, a significant portion of this section is devoted to the matrix-vector product with G/D.

4.1 Generating points

We simply generate the matrices $A^{(k)}$, $1 \leq k \leq d$, such that $a_{ij}^{(k)} \sim N(0,1)$ is standard normally distributed. The permutation matrices P_i, $1 \leq i \leq d$, are constructed as explained in section 2; they may be represented compactly as a vector $\mathbf{l}^i$ where $l^i_j = k$ iff $(P_i)_{kj} = 1$. These permutation vectors $\mathbf{l}^i$, $1 \leq i \leq r$ are then be combined into a large permutation vector $\mathbf{l}$, such that $l_j = k$ iff $p_{kj} = 1$, with P as in (6).

4.2 Rank computations

The implementation of Algorithm 2 in inexact arithmetic does not verify whether D_1 and D/D_1 are of full rank; this is motivated in section 5.

4.3 Conjugate gradient algorithm

The key operation of the proposed algorithm is solving the system $(G/D)\mathbf{x} = 0$ using the conjugate gradient method; its dominant cost is applying G/D to a vector. Here, we detail an efficient algorithm to perform this operation without forming the matrix G/D explicitly, thus vastly restraining the memory demands. First, the structure of $\mathscr{T}^T\mathscr{T}$ is presented; it admits a natural block structure inherited from (3) and (4):

$$\mathscr{T}^T\mathscr{T} = \begin{bmatrix} D_1 & L_1^{\star T} \\ L_1^{\star} & S^{\star} \end{bmatrix} = \begin{bmatrix} {T^1}^T T^1 & {T^1}^T T^2 & \cdots & {T^1}^T T^d \\ {T^2}^T T^1 & {T^2}^T T^2 & \cdots & {T^2}^T T^d \\ \vdots & \vdots & \ddots & \vdots \\ {T^d}^T T^1 & {T^d}^T T^2 & \cdots & {T^d}^T T^d \end{bmatrix}, \tag{13}$$

where, in addition, the blocks ${T^i}^T T^j$ admit a block structure:

$${T^i}^T T^j = \begin{bmatrix} {T_1^i}^T T_1^j & {T_1^i}^T T_2^j & \cdots & {T_1^i}^T T_r^j \\ {T_2^i}^T T_1^j & {T_2^i}^T T_2^j & \cdots & {T_2^i}^T T_r^j \\ \vdots & \vdots & \ddots & \vdots \\ {T_r^i}^T T_1^j & {T_r^i}^T T_2^j & \cdots & {T_r^i}^T T_r^j \end{bmatrix}. \tag{14}$$

From (3) it is straightforward to derive that if $i \neq j$, the (g,h) block of ${T^i}^T T^i$ is a scaled rank-1 matrix:

$$T_g^{iT} T_h^j = \prod_{\substack{k=1 \\ k\neq i,j}}^{d} \left\langle \mathbf{a}_g^k, \mathbf{a}_h^k \right\rangle \ \mathbf{a}_h^i {\mathbf{a}_g^j}^T; \tag{15}$$

on the other hand, if $i = j$, the (g,h) block of ${T^i}^T T^i$ is a scaled diagonal matrix:

$$T_g^{iT} T_h^i = \prod_{\substack{k=1 \\ k\neq i}}^{d} \langle \mathbf{a}_g^k, \mathbf{a}_h^k \rangle \ I_{n_i}. \tag{16}$$

The structure of $T_{(2)}^T T_{(2)}$ with $d = 3$, for example, may be visualized as

$$T_{(2)}^T T_{(2)} = \begin{bmatrix} \diagdown & \diagdown & \sqcap & \sqcap & \sqcap & \sqcap \\ \diagdown & \diagdown & \sqcap & \sqcap & \sqcap & \sqcap \\ \sqcap & \sqcap & \diagdown & \diagdown & \sqcap & \sqcap \\ \sqcap & \sqcap & \diagdown & \diagdown & \sqcap & \sqcap \\ \sqcap & \sqcap & \sqcap & \sqcap & \diagdown & \diagdown \\ \sqcap & \sqcap & \sqcap & \sqcap & \diagdown & \diagdown \end{bmatrix},$$

where $\diagdown$ denotes a diagonal matrix and $\sqcap$ a rank-1 matrix. From (14) and (16) it follows that the diagonal blocks of (13) are given by

$${T^i}^T T^i = \left(\bigodot_{\substack{k=1 \\ k\neq i}}^{d} {A^{(k)}}^T A^{(k)} \right) \otimes I_{n_i} := H^{(i)} \otimes I_{n_i}, \tag{17}$$

where $A \odot B$ denotes the Hadamard, or elementwise, product of the matrices A and B. Because the inner product matrices

$$J^{(k)} := {A^{(k)}}^T A^{(k)}$$

are often required in the algorithm implementing the matrix-vector product with G/D, i.e., Algorithm 3, we propose to compute them beforehand and necessitate them as input to the algorithm.

Consider next a matrix-vector product with one of the blocks in the right hand side of (13). For future reference, define the following block partitioning of the vectors $\mathbf{y}$ and $\mathbf{z}$ that is compatible with the two-level block structure (13)–(14) for the operation $\mathbf{y} = \mathscr{T}^T \mathscr{T} \mathbf{z}$:

$$\mathbf{y}^T := \begin{bmatrix} \mathbf{y}_1^T & \mathbf{y}_\star^T \end{bmatrix} = \begin{bmatrix} \mathbf{y}_1^T & \mathbf{y}_2^T & \cdots & \mathbf{y}_d^T \end{bmatrix}, \quad \text{and} \tag{18}$$

$$\mathbf{z}^T := \begin{bmatrix} \mathbf{z}_1^T & \mathbf{z}_\star^T \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1^T & \mathbf{z}_2^T & \cdots & \mathbf{z}_d^T \end{bmatrix}, \tag{19}$$

with $\mathbf{y}_i, \mathbf{z}_i \in \mathbb{R}^{n_i r}$, and let every $\mathbf{y}_i$ and $\mathbf{z}_i$ be partitioned as

$$\mathbf{y}_i^T := \begin{bmatrix} \mathbf{y}_{i,1}^T & \mathbf{y}_{i,2}^T & \cdots & \mathbf{y}_{i,r}^T \end{bmatrix} \quad \text{and} \quad \mathbf{z}_i^T := \begin{bmatrix} \mathbf{z}_{i,1}^T & \mathbf{z}_{i,2}^T & \cdots & \mathbf{z}_{i,r}^T \end{bmatrix}, \tag{20}$$

where $\mathbf{y}_{i,j}, \mathbf{z}_{i,j} \in \mathbb{R}^{n_i}$. Let us define furthermore the matrices $Y^{(i)}, Z^{(i)} \in \mathbb{R}^{n_i \times r}$:

$$Y^{(i)} := \begin{bmatrix} \mathbf{y}_{i,1} & \mathbf{y}_{i,2} & \cdots & \mathbf{y}_{i,r} \end{bmatrix} \quad \text{and} \quad Z^{(i)} := \begin{bmatrix} \mathbf{z}_{i,1} & \mathbf{z}_{i,2} & \cdots & \mathbf{z}_{i,r} \end{bmatrix}, \tag{21}$$

i.e., $\mathbf{y}_i$ is the vector obtained by stacking the columns of $Y^{(i)}$, and similarly for $\mathbf{z}_i$:

$$\operatorname{vec}\left(Y^{(i)}\right) = \mathbf{y}_i \quad \text{and} \quad \operatorname{vec}\left(Z^{(i)}\right) = \mathbf{z}_i. \tag{22}$$

Multiplying (17) with a vector on the right can be performed efficiently without computing the Kronecker product, due to a basic property of the latter; it is known that [53]

$$\mathbf{y}_i = {T^i}^T T^i \mathbf{z}_i = (H^{(i)} \otimes I_{n_i})\mathbf{z}_i \quad \text{is equivalent with} \quad Y^{(i)} = Z^{(i)} {H^{(i)}}^T. \tag{23}$$

Similarly, the off-diagonal blocks of (13) may also be applied to a vector without explicitly computing them; we have

$$\mathbf{y}_{i,g} := \sum_{h=1}^{r} {T_g^i}^T T_h^j \mathbf{z}_{j,h} = \sum_{h=1}^{r} \left(\prod_{\substack{k=1 \\ k\neq i,j}}^{d} \left\langle \mathbf{a}_g^k, \mathbf{a}_h^k \right\rangle \right) \mathbf{a}_h^i ({\mathbf{a}_g^j}^T \mathbf{z}_{j,h}) := \sum_{h=1}^{r} h_{gh}^{(i,j)} \mathbf{a}_h^i,$$

where

$$h_{gh}^{(i,j)} := \left({\mathbf{a}_g^j}^T \mathbf{z}_{j,h} \right) \cdot \prod_{\substack{k=1 \\ k\neq i,j}}^{d} \left\langle \mathbf{a}_g^k, \mathbf{a}_h^k \right\rangle.$$

These equations can be rewritten tersely in matrix form:

$$\mathbf{y}_i := {T^i}^T T^j \mathbf{z}_j \quad \text{is equivalent to} \quad Y^{(i)} = A^{(i)} {H^{(i,j)}_{Z^{(j)}}}^T, \tag{24}$$

and where we define

$$H_X^{(i,j)} = {A^{(j)}}^T X \odot \bigodot_{\substack{k=1 \\ k\neq i,j}}^{d} {A^{(k)}}^T A^{(k)}, \tag{25}$$

assuming that $X \in \mathbb{R}^{n_j \times r}$. Note that $H^{(i)}$ is a shorthand for $H^{(i,i)}_{A^{(i)}}$.

We consider now the problem of computing a matrix-vector product with G/D without computing the latter matrix, in order to limit the memory cost of the proposed algorithm. We write

$$\widehat{\mathbf{y}} := (G/D)\widehat{\mathbf{z}} := \widehat{\mathbf{y}}^1 - \widehat{\mathbf{y}}^2 := P_\star^T (\mathbf{y}^1 - \mathbf{y}^2),$$

wherein, deriving from (10),

$$\widehat{\mathbf{y}}^1 = (S - L_1 D_1^{-1} L_1^T)\widehat{\mathbf{z}} \tag{26}$$

$$\widehat{\mathbf{y}}^2 = (L_2 - L_1 D_1^{-1} L_{1,2}^T)(D/D_1)^{-1}(L_2 - L_1 D_1^{-1} L_{1,2}^T)^T \widehat{\mathbf{z}}. \tag{27}$$

Consider first (26):

$$\widehat{\mathbf{y}}^1 = P_\star^T \left(S^\star - L_1^\star D_1^{-1} L_1^{\star T} \right) P_\star \widehat{\mathbf{z}}. \tag{28}$$

Computing $\mathbf{z}_\star = P_\star \widehat{\mathbf{z}}$, with $\mathbf{z}_\star$ as in (19), is straightforward: if the ith row of $P_\star$ has a one in position l_i, i.e., $(p_\star)_{i,l_i} = 1$, then $(z_\star)_i = \widehat{z}_{l_i}$, otherwise $(z_\star)_i = 0$. A similar observation holds

for the product with $P_\star^T$. Consider now $\dot{\mathbf{y}} := S^\star \mathbf{z}_\star$ where $\dot{\mathbf{y}}$ is partitioned as $\mathbf{y}_\star$ in (18). Then, we obtain for $2 \leq i \leq d$ that

$$\dot{\mathbf{y}}_i = \sum_{j=2}^{d} {T^i}^T T^j \mathbf{z}_j = {T^i}^T T^i \mathbf{z}_i + \sum_{\substack{j=2 \\ j\neq i}}^{d} {T^i}^T T^j \mathbf{z}_j,$$

by definition. Applying (23) and (24), one finds

$$\dot{Y}^{(i)} = Z^{(i)} H^{(i)} + A^{(i)} \sum_{\substack{j=2 \\ j\neq i}}^{d} {H_{Z^{(j)}}^{(i,j)}}^T,$$

where we used that $H^{(i)}$ is symmetric. Next, the computation of $\ddot{\mathbf{y}} := L_1^\star D_1^{-1} {L_1^\star}^T \mathbf{z}_\star$ is investigated. Letting $\dot{\mathbf{w}} := {L_1^\star}^T \mathbf{z}_\star$, we have, by definition,

$$\dot{\mathbf{w}} := \sum_{j=2}^{d} {T^1}^T T^j \mathbf{z}_j; \quad \text{applying (24) results in} \quad \dot{W} = A^{(1)} \sum_{j=2}^{d} {H_{Z^{(j)}}^{(1,j)}}^T,$$

where $\dot{\mathbf{w}} = \operatorname{vec}\left(\dot{W}\right)$ in the usual manner. Then,

$$D_1 = H^{(1)} \otimes I_{n_1} \quad \text{and, thus,} \quad D_1^{-1} = {H^{(1)}}^{-1} \otimes I_{n_1}.$$

Consequently, $\ddot{\mathbf{w}} := D_1^{-1} \dot{\mathbf{w}}$ may be written as

$$\ddot{W} = \dot{W} {H^{(1)}}^{-1}, \quad \text{where} \quad \ddot{\mathbf{w}} = \operatorname{vec}\left(\ddot{W}\right).$$

This is a consequence of the Kronecker structure of D_1^{-1} [53] and the symmetry of $H^{(1)}$. Assume that $\ddot{\mathbf{y}}$ is partitioned as $\mathbf{y}_\star$ in (18), i.e., $\ddot{\mathbf{y}}^T = \left[\ddot{\mathbf{y}}_2^T \cdots \ddot{\mathbf{y}}_d^T\right]$. Then, $\ddot{\mathbf{y}} = L_1^\star \ddot{\mathbf{w}}$ is given as follows:

$$\ddot{\mathbf{y}}_i = (T^i)^T T^1 \ddot{\mathbf{w}}, \quad \text{which is equivalent to} \quad \ddot{Y}^{(i)} = A^{(i)} {H_{\dot{W}}^{(i,1)}}^T,$$

by (24). Finally, $\mathbf{y}^1 := \dot{\mathbf{y}} - \ddot{\mathbf{y}}$, so that applying $P_\star^T$ completes the computation of (26). The second update term for the Schur complement, presented in (27), will be applied through straightforward matrix-vector products because these matrices are sufficiently small to represent explicitly, and for reasons related to accuracy that will not become evident until Theorem 5. We explicitly compute

$$L_2' := (L_2^\star - L_1^\star D_1^{-1} (L_{1,2}^\star)^T) P_R C_R, \quad \text{and} \tag{29}$$

$$D_2' := C_R^T P_R^T (T_R^T T_R - L_{1,2}^\star D_1^{-1} (L_{1,2}^\star)^T) P_R C_R; \tag{30}$$

the correctness of these equations is obvious from (9) and its succeeding definitions. Then,

$$\mathbf{y}^2 = L_2'((D_2')^{-1}({L_2'}^T \mathbf{z}_\star))$$

is readily computable using unstructured matrix-vector products. The computation of the Schur complement (26)–(27) is summarized in Algorithm 3; its numerical properties are investigated in section 5.

We continue with the practical computation of (29)–(30). Consider the computation of L_2'. $L_2^\star$ and $L_{1,2}^\star$ are constructed simply by applying the definitions (15) and (16). Consider then $\dot{L} = D_1^{-1} {L_{1,2}^\star}^T$, and let

$${L_{1,2}^\star}^T := \left[\mathbf{l}_1^\star \; \mathbf{l}_2^\star \cdots \mathbf{l}_{\Sigma-d+1}^\star\right] \in \mathbb{R}^{n_1 r \times \Sigma - d + 1}.$$

Algorithm 3: Computing a matrix-vector product with the Schur complement G/D.

input : $\{A^{(k)}\}_k$, $\{J^{(k)}\}_k$, L'_2, and D'_2.
input : The vector $\mathbf{z}_\star$ as in (19), satisfying (21) and (22).
output: The vector $\mathbf{y}_\star$ as in (18), satisfying (21) and (22).
output: $\mathbf{y}_\star = (S^\star - L_1^\star D_1^{-1} L_1^{\star T} - L'_2 (D'_2)^{-1} L_2'^T)\mathbf{z}_\star$.

$W^{(1)} \leftarrow 0$
for $j = 2,3,\ldots,d$ **do**
  $K^{(j)} \leftarrow A^{(j)^T} Z^{(j)}$
  $W^{(1)} \leftarrow W^{(1)} + K^{(j)} \odot \bigodot_{k\neq 1,j} J^{(k)}$
end
$\dot{W} \leftarrow A^{(1)} W^{(1)^T}$
$H^{(1)} \leftarrow \bigodot_{k\neq 1} J^{(k)}$
$\ddot{W} \leftarrow \dot{W} H^{(1)^{-1}}$
$C \leftarrow A^{(1)^T} \ddot{W}$
for $i = 2,3,\ldots,d$ **do**
  $W^{(i)} \leftarrow 0$
  for $j = 2,3,\ldots,d$ **do**
    if $i \neq j$ **then**
      $W^{(i)} \leftarrow W^{(i)} + K^{(j)} \odot \bigodot_{k\neq i,j} J^{(k)}$
    end
  end
  $H^{(i,1)} \leftarrow \bigodot_{k\neq 1,i} J^{(k)}$
  $H^{(i,1)}_{\dot{W}} \leftarrow C \odot H^{(i,1)}$
  $\ddot{W}^{(i)} \leftarrow W^{(i)} - H^{(i,1)}_{\dot{W}}$
  $\dot{Y}^{(i)} \leftarrow A^{(i)} \ddot{W}^{(i)^T}$
  $H^{(i)} \leftarrow J^{(1)} \odot H^{(i,1)}$
  $Y_1^{(i)} \leftarrow \dot{Y}^{(i)} + Z^{(i)} H^{(i)}$
end
$\mathbf{y}^1 \leftarrow \left[\operatorname{vec}\left(Y_1^{(2)}\right)^T \operatorname{vec}\left(Y_1^{(3)}\right)^T \cdots \operatorname{vec}\left(Y_1^{(d)}\right)^T\right]^T$
$\mathbf{y}_\star \leftarrow \mathbf{y}^1 - L'_2((D'_2)^{-1}(L_2'^T \mathbf{z}_\star))$

Each of the columns of $L_{1,2}^{\star\,T}$ may be interpreted as a vectorization of some matrix:

$$\operatorname{vec}\left(L^\star_{\{i\}}\right) := \mathbf{l}_i^\star \quad \text{where} \quad L^\star_{\{i\}} \in \mathbb{R}^{n_1 \times r}.$$

We may then write the product

$$\dot{\mathbf{l}}_i := D_1^{-1} \mathbf{l}_i^\star \quad \text{equivalently as} \quad \dot{L}_{\{i\}} := L^\star_{\{i\}} H^{(1)^{-1}},$$

where $\dot{\mathbf{l}}_i = \operatorname{vec}\left(\dot{L}_{\{i\}}\right)$ in the usual manner. Continuing with $\ddot{L} := L_1^\star \dot{L}$, or, column-wise, $\ddot{\mathbf{l}}_i := L_1^\star \dot{\mathbf{l}}_i$, and partitioning $\ddot{\mathbf{l}}_i$ like $\mathbf{y}_\star$ in (18), we may write

$$(\ddot{\mathbf{l}}_i)_j := T^{j^T} T^1 \dot{\mathbf{l}}_i, \quad \text{for } 2 \leq j \leq d,$$

by definition (24); thus it may be computed as

$$(\ddot{\mathbf{l}}_i)_j = \operatorname{vec}\left(A^{(j)} H^{(j,1)}_{\dot{L}_{\{i\}}}\right).$$

Finally, having computed $L_2^\star$ and $\ddot{L}$ explicitly, we can compute

$$L_2' := (L_2^\star - \ddot{L})P_R C_R$$

using unstructured operations. Computing D_2' is accomplished as follows: $T_R^T T_R$ is constructed explicitly by applying (23) and (24). The product $\dot{L} = D_1^{-1} {L_{1,2}^\star}^T$ was already detailed, and as $L_{1,2}^\star$ is also available explicitly, we can simply compute

$$D_2' = C_R^T P_R^T ((T_R^T T_R - L_{1,2}^\star \dot{L}) P_R C_R)$$

using unstructured operations. For the sake of completeness, we summarize the above discussion in Algorithm 4; the numerical properties of this algorithm are investigated in section 5.

Algorithm 4: Computing $(D/D_1)^{-1}$ and L_2'

input : $\{A^{(k)}\}_k$, $\{J^{(k)}\}_k$.
output: L_2' as in (29) and D_2' as in (30).

$L_{1,2}^\star \leftarrow T_R^T T^1$
$L_2^\star \leftarrow \left[T_R^T T^2 \;\; T_R^T T^3 \;\; \cdots \;\; T_R^T T^d\right]^T$
$D_2 \leftarrow T_R^T T_R$
$H^{(1)} \leftarrow \bigodot_{k=2}^{d} J^{(k)}$
for $i = 1, 2, \ldots, \Sigma - d + 1$ **do**
    $\dot{L}_{\{i\}} \leftarrow L_{\{i\}}^\star {H^{(1)}}^{-1}$
end
for $j = 2, 3, \ldots, d$ **do**
    $H \leftarrow \bigodot_{k=2, k\neq j} J^{(k)}$
    for $i = 1, 2, \ldots, \Pi - r(\Sigma - d + 1)$ **do**
        $H_{\dot{L}_{\{i\}}}^{(j,1)} \leftarrow \left({A^{(1)}}^T \dot{L}_{\{i\}}\right) \odot H$
        $(\ddot{\mathbf{I}}_i)_j \leftarrow \mathrm{vec}\left(A^{(j)} H_{\dot{L}_{\{i\}}}^{(j,1)}\right)$
    end
end
$L_2' \leftarrow (L_2^\star - \ddot{L}) P_R C_R$
$D_2' \leftarrow C_R^T P_R^T ((D_2 - L_{1,2}^\star \dot{L}) P_R C_R)$

4.4 Memory cost

A beneficial property of the proposed structure-exploiting operations is its diminished memory cost with respect to the classic algorithm. The latter stores $\mathscr{O}(\Pi^2)$ values to represent the tangent space matrix, of which the rank is computed. The memory cost of the proposed algorithm is established as follows. We need to store dr^2 values to represent $\{J^{(k)}\}_k$, Σr values to represent $\{A^{(k)}\}_k$, $c^2 = \mathscr{O}(\Sigma^2)$ values for D_2', and $\mathscr{O}(rc(\Sigma - d + 1 - n_1))$ values for L_2'. We also store three vectors of length $r(\Sigma - n_1)$ during the execution of the conjugate gradient algorithm. It is straightforward to verify that none of the operations in Algorithm 3 requires more memory than the aforementioned storage costs; the asymptotic memory consumption thus remains unaffected. An analogous observation holds for Algorithm 4, so that we may conclude that the structure exploiting probabilistic algorithm stores

$$\mathscr{O}((r+1)\Sigma^2) = \mathscr{O}(\Pi\Sigma) \ll \mathscr{O}(\Pi^2)$$

values, which is asymptotically less than the classic algorithm. In section 6.3 we present a comparison between the classic and proposed algorithm in terms of memory cost; the results clearly indicate the advantage of the proposed algorithm.

5 Numerical analysis

In the previous sections, we have either assumed exact arithmetic or an unspecified form of inexact arithmetic. In this section, we study the behavior of the proposed algorithm in fixed precision floating point arithmetic. We will assume the standard model of floating point arithmetic [36, 56] in the remainder of this paper:

$$\mathrm{fl}\,(x \operatorname{op} y) = (1+\delta)(x \operatorname{op} y), \quad \text{with } |\delta| \leq \varepsilon, \quad \text{and where } \operatorname{op} = +, -, \cdot, /; \tag{31}$$

herein, ε is the unit roundoff. The IEEE floating point arithmetic standard satisfies this model and is implemented by most modern desktop computer systems, including the test setup employed in our numerical experiments in section 6. It may be noted that on many modern computer system a fused multiply-add, i.e., $c \leftarrow a + \gamma b$, also satisfies $\mathrm{fl}\,(a+\gamma b) = (1+\delta)(a+\gamma b)$; however, we have chosen the more conservative path in our analysis, as it is difficult to guarantee that the compiler generates the appropriate machine code. The presented analysis will provide only a rough upper bound on the actual computation errors.

In this section, we shall abide by the convention that $\underline{a} := \mathrm{fl}\,(a)$ denotes the actual number stored by the computer system, and that $|\mathbf{a}|$ denotes the vector of absolute values of $\mathbf{a}$. Whenever a matrix or vector appears in an inequality with a scalar, the inequality should be interpreted componentwise.

The focus of this section is extending Lemma 2 with a practically computable upper bound on the total computation error of the conjugate gradient algorithm *in the null space* when applied within the standard model of floating point arithmetic. Many works have analyzed the behavior of the conjugate gradient method in a numerical setting (see [41] for a good overview) from the viewpoint of convergence [27, 28, 43, 48], error estimation and reliable stopping criteria [25, 49], and loss of orthogonality [45]; the results we present below concern the perturbation of the components of the obtained approximation in the null space of a singular but consistent linear system using a running forward error analysis [56], which are original to the best of our knowledge.

Theorem 3 *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive semidefinite matrix, and let $\mathbf{x}^{(0)} \in \mathbb{R}^n$ be a vector whose components were drawn i.i.d. from a standard normal distribution. Let $\underline{\mathbf{x}}^{(m)}$ be the approximate solution to $A\mathbf{x} = 0$ obtained by applying the numerical conjugate gradient method with starting vector $\mathbf{x}^{(0)}$, and let $\mathbf{x}$ be the true solution obtained by applying the CG method in exact arithmetic to the starting vector $\mathbf{x}^{(0)}$. Let $A = VDV^T$ be the eigenvalue decomposition of A where the diagonal elements of D are sorted by decreasing magnitude, so that*

$$\underline{\mathbf{x}}^{(m)} = \sum_{i=1}^{n} \left\langle \underline{\mathbf{x}}^{(m)}, \mathbf{v}_i \right\rangle \mathbf{v}_i \quad \textit{and} \quad \mathbf{x} = \sum_{i=s+1}^{n} \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle \mathbf{v}_i,$$

where s is the rank of A. Let

$$\eta^{(m)} := \|\underline{\mathbf{x}}^{(m)}\| \quad \textit{and} \quad \chi^{(m)} := \sqrt{\sum_{i=s+1}^{n} \left(\left\langle \underline{\mathbf{x}}^{(m)}, \mathbf{v}_i \right\rangle - \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle\right)^2},$$

and let $\underline{\gamma}_k$, $\underline{\beta}_k$, $\underline{\mathbf{p}}^{(k)}$, $\underline{\mathbf{r}}^{(k)}$ *be the floating point approximations of the corresponding quantities in Algorithm 1, and* $\underline{\mathbf{q}}^{(k-1)} := \mathrm{fl}\left(A\underline{\mathbf{p}}^{(k-1)}\right)$. *Then,*

$$\left|\left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle\right| \leq \eta^{(m)} + \chi^{(m)}, \quad \textit{for } i \geq s+1,$$

where $\chi^{(m)}$ *satisfies the three-term recurrence:*

$$\chi^{(k)} \leq \chi^{(k-1)} + |\underline{\gamma}_{k-1}|\phi^{(k-1)} + \varepsilon \left\| (2+\varepsilon)\,|\underline{\gamma}_{k-1}|\,|\underline{\mathbf{p}}^{(k-1)}| + |\underline{\mathbf{x}}^{(k-1)}| \right\| \tag{32}$$

$$\phi^{(k)} \leq \rho^{(k)} + |\underline{\beta}_k|\,|\phi^{(k-1)}| + \varepsilon \left\| (2+\varepsilon)\,|\underline{\beta}_k|\,|\underline{\mathbf{p}}^{(k-1)}| + |\underline{\mathbf{r}}^{(k)}| \right\| \tag{33}$$

$$\rho^{(k)} \leq \rho^{(k-1)} + \varepsilon \left\| (2+\varepsilon)\,|\underline{\gamma}_{k-1}|\,|\underline{\mathbf{q}}^{(k-1)}| + |\underline{\mathbf{r}}^{(k-1)}| + |\underline{\gamma}_{k-1}|\,\delta(A, \underline{\mathbf{p}}^{(k-1)})\mathbf{1} \right\| \tag{34}$$

where $\mathbf{1}$ *is a vector of all ones, and* $\delta(\cdot,\cdot)$ *is a scalar upper bound on the forward error of the matrix-vector product* $A\underline{\mathbf{p}}^{(k-1)}$*:*

$$\left| A\underline{\mathbf{p}}^{(k-1)} - \mathrm{fl}\left(A\underline{\mathbf{p}}^{(k-1)}\right) \right| \leq \varepsilon \cdot \delta(A, \underline{\mathbf{p}}^{(k-1)}).$$

The initial values of the recurrence are:

$$\chi^{(0)} = 0 \quad \textit{and} \quad \phi^{(0)} = \rho^{(0)} = \varepsilon \cdot \sqrt{r(\Sigma - d + 1 - n_1)} \cdot \delta(A, \mathbf{x}^{(0)}).$$

Proof Let

$$\underline{\mathbf{x}}^{(k)} := \sum_{i=1}^{n} \underline{\chi}_i^{(k)} \mathbf{v}_i \qquad \text{and} \qquad \chi^{(k)} := \sqrt{\sum_{i=s+1}^{n} (\underline{\chi}_i^{(k)} - \left\langle \mathbf{x}^{(0)}, \mathbf{v}_i \right\rangle)^2}, \tag{35}$$

$$\underline{\mathbf{r}}^{(k)} := \sum_{i=1}^{n} \underline{\rho}_i^{(k)} \mathbf{v}_i \qquad \text{and} \qquad \rho^{(k)} := \sqrt{\sum_{i=s+1}^{n} (\underline{\rho}_i^{(k)} - \left\langle \mathbf{r}^{(0)}, \mathbf{v}_i \right\rangle)^2}, \tag{36}$$

$$\underline{\mathbf{p}}^{(k)} := \sum_{i=1}^{n} \underline{\phi}_i^{(k)} \mathbf{v}_i \qquad \text{and} \qquad \phi^{(k)} := \sqrt{\sum_{i=s+1}^{n} (\underline{\phi}_i^{(k)} - \left\langle \mathbf{p}^{(0)}, \mathbf{v}_i \right\rangle)^2}, \tag{37}$$

where $\mathbf{p}^{(0)} = \mathbf{r}^{(0)} = -A\mathbf{x}^{(0)}$ exactly. Note that $\left\langle \mathbf{r}^{(0)}, \mathbf{v}_i \right\rangle = \left\langle \mathbf{p}^{(0)}, \mathbf{v}_i \right\rangle = 0$ for all $i \geq s+1$. In words, $\chi^{(k)}$ measures the norm of the projection of $\underline{\mathbf{x}}^{(k)} - \mathbf{x}$ onto the span of the null space of A, i.e., the norm of the components *generated in the null space due to the numerical execution of the conjugate gradient algorithm*, and similarly for $\rho^{(k)}$ and $\phi^{(k)}$. In exact arithmetic, no components are generated in the null space by the conjugate gradient algorithm: $\chi^{(k)} = \rho^{(k)} = \phi^{(k)} = 0$; however, in floating point arithmetic these norms can become nonzero. We continue by establishing bounds on their magnitude.

The *a posteriori* bound on the magnitude of the components of the starting vector in the null space was already proved as Lemma 2. It thus remains to prove the bound on $\chi^{(m)}$. Consider the computation of the approximation $\underline{\mathbf{x}}^{(k)}$ in the conjugate gradient algorithm. In floating point arithmetic we compute

$$\begin{aligned} \underline{\mathbf{x}}^{(k)} &= \mathrm{fl}\left(\underline{\mathbf{x}}^{(k-1)} + \mathrm{fl}\left(\underline{\gamma}_{k-1}\underline{\mathbf{p}}^{(k-1)}\right)\right) \\ &= \mathrm{fl}\left(\underline{\mathbf{x}}^{(k-1)} + \underline{\gamma}_{k-1}\underline{\mathbf{p}}^{(k-1)} + \mathbf{e}^1\right) \\ &= \underline{\mathbf{x}}^{(k-1)} + \underline{\gamma}_{k-1}\underline{\mathbf{p}}^{(k-1)} + \mathbf{e}^1 + \mathbf{e}^2, \end{aligned} \tag{38}$$

where

$$|\mathbf{e}^1| \leq |\underline{\gamma}_{k-1}||\underline{\mathbf{p}}^{(k-1)}|\varepsilon \quad \text{and} \quad |\mathbf{e}^2| \leq (|\underline{\mathbf{x}}^{(k-1)}| + |\underline{\gamma}_{k-1}||\underline{\mathbf{p}}^{(k-1)}| + |\mathbf{e}^1|)\varepsilon.$$

Letting $\mathbf{e}^X := \mathbf{e}^1 + \mathbf{e}^2$, we find that the forward error is

$$\|\mathbf{e}^X\| \leq \||\mathbf{e}^1| + |\mathbf{e}^2|\| \leq \varepsilon\||\underline{\mathbf{x}}^{(k-1)}| + (2+\varepsilon)|\underline{\gamma}_{k-1}||\underline{\mathbf{p}}^{(k-1)}|\|.$$

Continuing from (38) and letting $V_\perp = \begin{bmatrix}\mathbf{v}_{s+1} & \mathbf{v}_{s+2} & \dots & \mathbf{v}_n\end{bmatrix}$, we may consider[14]

$$V_\perp^T(\underline{\mathbf{x}}^{(k)} - \mathbf{x}) = V_\perp^T(\underline{\mathbf{x}}^{(k-1)} - \mathbf{x}) + \underline{\gamma}_{k-1}V_\perp^T\underline{\mathbf{p}}^{(k-1)} + V_\perp^T\mathbf{e}^X; \tag{39}$$

from (35) it follows that

$$\left\|V_\perp^T(\underline{\mathbf{x}}^{(k)} - \mathbf{x})\right\| = \left\|\begin{bmatrix} \chi_{s+1}^{(k)} - \left\langle \mathbf{x}^{(0)}, \mathbf{v}_{s+1}\right\rangle \\ \chi_{s+2}^{(k)} - \left\langle \mathbf{x}^{(0)}, \mathbf{v}_{s+2}\right\rangle \\ \vdots \\ \chi_n^{(k)} - \left\langle \mathbf{x}^{(0)}, \mathbf{v}_n\right\rangle \end{bmatrix}\right\| = \chi^{(k)},$$

and similarly, $\|V_\perp^T\underline{\mathbf{p}}^{(k-1)}\| = \phi^{(k-1)}$ so that taking norms and applying the triangle inequality to (39) yields the first recursion

$$\chi^{(k)} \leq \chi^{(k-1)} + |\underline{\gamma}_{k-1}|\phi^{(k-1)} + \|\mathbf{e}^X\|,$$

using $\|V_\perp^T\mathbf{e}^X\| \leq \|\mathbf{e}^X\|$ in the last term. We can similarly derive the recurrence for ϕ. We initially compute

$$\underline{\mathbf{p}}^{(k)} := \mathrm{fl}\left(\underline{\mathbf{r}}^{(k)} + \mathrm{fl}\left(\underline{\beta}_k\underline{\mathbf{p}}^{(k-1)}\right)\right) = \underline{\mathbf{r}}^{(k)} + \underline{\beta}_k\underline{\mathbf{p}}^{(k-1)} + \mathbf{e}^1 + \mathbf{e}^2, \tag{40}$$

where

$$|\mathbf{e}^1| \leq \varepsilon|\underline{\beta}_k||\underline{\mathbf{p}}^{(k-1)}|, \quad \text{and} \quad |\mathbf{e}^2| \leq \varepsilon(|\underline{\mathbf{r}}^{(k)}| + |\underline{\beta}_k||\underline{\mathbf{p}}^{(k-1)}| + |\mathbf{e}^1|).$$

Letting $\mathbf{e}^P := \mathbf{e}^1 + \mathbf{e}^2$, we find that the forward error is

$$\|\mathbf{e}^P\| \leq \varepsilon\||\underline{\mathbf{r}}^{(k)}| + (2+\varepsilon)|\underline{\beta}_k||\underline{\mathbf{p}}^{(k-1)}|\|.$$

Applying $V_\perp^T$ on the left and right hand sides of (40), taking norms, applying the triangle inequality, bounding $\|V_\perp^T\mathbf{e}^P\| \leq \|\mathbf{e}^P\|$ and applying definitions (36) and (37), we find the second recurrence

$$\phi^{(k)} \leq \rho^{(k)} + |\underline{\beta}_k|\phi^{(k-1)} + \|\mathbf{e}^P\|.$$

The final recurrence is obtained as follows. Let $\underline{\mathbf{q}}^{(k-1)} := \mathrm{fl}\left(A\underline{\mathbf{p}}^{(k-1)}\right)$. Then,

$$\begin{aligned}\underline{\mathbf{r}}^{(k)} &= \mathrm{fl}\left(\underline{\mathbf{r}}^{(k-1)} - \mathrm{fl}\left(\underline{\gamma}_{k-1}\,\mathrm{fl}\left(A\underline{\mathbf{p}}^{(k-1)}\right)\right)\right) \\ &= \mathrm{fl}\left(\underline{\mathbf{r}}^{(k-1)} - \mathrm{fl}\left(\underline{\gamma}_{k-1}(A\underline{\mathbf{p}}^{(k-1)} + \mathbf{e}^1)\right)\right)\end{aligned}$$

[14] We are conveniently assuming the product of a zero-sized matrix with any other matrix or vector is zero.

$$
\begin{aligned}
&= \mathrm{fl}\left(\underline{\mathbf{r}}^{(k-1)} - \underline{\gamma}_{k-1}(A\underline{\mathbf{p}}^{(k-1)} + \mathbf{e}^1) + \mathbf{e}^2\right) \\
&= \underline{\mathbf{r}}^{(k-1)} - \underline{\gamma}_{k-1}(A\underline{\mathbf{p}}^{(k-1)} + \mathbf{e}^1) + \mathbf{e}^2 + \mathbf{e}^3,
\end{aligned}
$$

wherein the error terms can be bounded as follows

$$
\begin{aligned}
|\mathbf{e}^1| &\leq \varepsilon \cdot \delta(A, \underline{\mathbf{p}}^{(k-1)}) \\
|\mathbf{e}^2| &\leq \varepsilon |\underline{\gamma}_{k-1}||\underline{\mathbf{q}}^{(k-1)}| \\
|\mathbf{e}^3| &\leq \varepsilon |\mathbf{r}^{(k-1)}| + \varepsilon |\underline{\gamma}_{k-1}||\underline{\mathbf{q}}^{(k-1)}| + \varepsilon |\mathbf{e}^2|.
\end{aligned}
$$

The forward error of the computation of $\mathbf{r}^{(k)}$ is then

$$
\begin{aligned}
\|\mathbf{e}^2 + \mathbf{e}^3 - \underline{\gamma}_{k-1}\mathbf{e}^1\| &\leq \||\mathbf{e}^2| + |\mathbf{e}^3| + |\underline{\gamma}_{k-1}||\mathbf{e}^1|\| \\
&\leq \varepsilon \|(2+\varepsilon)|\underline{\gamma}_{k-1}||\underline{\mathbf{q}}^{(k-1)}| + |\underline{\mathbf{r}}^{(k-1)}| + |\underline{\gamma}_{k-1}|\delta(A, \underline{\mathbf{p}}^{(k-1)})\mathbf{1}\|,
\end{aligned}
$$

where $\mathbf{1}$ denotes the vector of all ones. The recurrence thus reads

$$
\rho^{(k)} \leq \rho^{(k-1)} + \varepsilon \|(2+\varepsilon)|\underline{\gamma}_{k-1}||\mathbf{q}^{(k-1)}| + |\mathbf{r}^{(k-1)}| + |\underline{\gamma}_{k-1}|\delta(A, \underline{\mathbf{p}}^{(k-1)})\mathbf{1}\|.
$$

Finally, the initial values of the recurrence are similarly found, concluding the proof. □

The gist of this theorem is its computable upper bound on $\chi^{(k)}$, which we may then employ in the hypothesis test in section 3, yielding our proposed numerical algorithm. However, in Theorem 3, we still need a bound for the forward error $\varepsilon \cdot \delta(G/D, \mathbf{x})$ of a matrix-vector product with G/D. This is handled in the remainder of this section, and consists of a straightforward but technical numerical analysis. Before proceeding, we recall some useful identities from [36, 56] which will be used in further derivations. First,

$$
\prod_{i=1}^{k}(1+\delta_i) = 1 + \theta_k, \quad \text{where} \quad |\theta_k| \leq \varepsilon_k := \frac{k\varepsilon}{1-k\varepsilon},
$$

provided that $|\delta_i| \leq \varepsilon$ and $1 - k\varepsilon > 0$. From this, one can deduce that

$$
\left| \mathrm{fl}\left(\sum_{i=1}^{k} a_i\right) - \sum_{i=1}^{k} a_i \right| \leq \varepsilon_{k-1} \sum_{i=1}^{k} |a_i|; \tag{41}
$$

see, e.g., [56]. Analogously, we have [56]:

$$
\left| \mathrm{fl}\left(\prod_{i=1}^{k} a_i\right) - \prod_{i=1}^{k} a_i \right| \leq \varepsilon_{k-1} \prod_{i=1}^{k} |a_i|. \tag{42}
$$

The forward error in matrix multiplication can also be bounded easily [36]: if $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times p}$, then

$$
\mathrm{fl}\,(AB) = AB + E \quad \text{with} \quad |E| \leq \varepsilon_n |A||B| \leq \varepsilon_n \omega(A,B) \tag{43}
$$

where the absolute value should be taken componentwise, and the function ω is given by

$$
\omega(A,B) = \min\{\|A\|_{\max}\|B\|_1, \|A\|_\infty \|B\|_{\max}\}.
$$

Assuming that $a, b \geq 1$, a useful property is [36, Lemma 3.3]:

$$
\varepsilon_a + \varepsilon_b + \varepsilon_a\varepsilon_b \leq \varepsilon_{a+b} \quad \text{and, thus,} \quad \varepsilon_a + \varepsilon_b \leq \varepsilon_{a+b} \quad \text{and} \quad \varepsilon_a \leq \varepsilon_{a+1}, \tag{44}
$$

assuming all involved terms are smaller than one. It is also elementary to prove that

$$b\varepsilon_a \leq \varepsilon_{ab}.$$

The following theorem proves to be useful in further derivations.

Theorem 4 *Assume that* $\underline{X} = X + E_X$ *with* $|E_X| \leq \varepsilon_X$ *and let* $\alpha_j := \|A^{(j)}\|_1$. *Then, computing* $H^{(i,j)}_{\underline{X}}$ *in finite precision floating-point arithmetic results in*

$$\mathrm{fl}\left(H^{(i,j)}_{\underline{X}}\right) = H^{(i,j)}_X + E^{(i,j)}_{H_X}$$

where

$$|E^{(i,j)}_{H_X}| \leq \varepsilon_{(d-1)(n_1+1)}\alpha_j\|X\|_{\max} + (1+\varepsilon_{(d-1)(n_1+1)})\alpha_j\varepsilon_X.$$

Proof The inner product matrices satisfy

$$\underline{J}^{(k)} = {A^{(k)}}^T A^{(k)} + E^{(k)}_J \quad \text{with} \quad |E^{(k)}_J| \leq \varepsilon_{n_k},$$

where we observed that every entry of $|A^{(k)}|^T|A^{(k)}|$ is the (positive) inner product of two unit norm vectors, and thus the Cauchy–Schwarz inequality bounds it by the norm of these vectors, i.e., one. From the previous equation we find

$$\mathrm{fl}\left(\bigodot_{\substack{k=1\\k\neq i,j}}^{d} \underline{J}^{(k)}\right) = \bigodot_{\substack{k=1\\k\neq i,j}}^{d} \left(J^{(k)} + E^{(k)}_J\right) + E^{(i,j)}_{\odot J'},$$

with

$$\left|E^{(i,j)}_{\odot J'}\right| \leq \varepsilon_{d-3} \bigodot_{\substack{k=1\\k\neq i,j}}^{d} |\underline{J}^{(k)}| \leq \varepsilon_{d-3}, \tag{45}$$

where we observe that even in floating-point arithmetic the product of values strictly smaller than 1 remains smaller than 1. Recalling that $n_1 = \max_k\{n_k\}$, we can bound

$$\begin{aligned}\left|\bigodot_{\substack{k=1\\k\neq i,j}}^{d} \left(J^{(k)} + E^{(k)}_J\right) - \bigodot_{\substack{k=1\\k\neq i,j}}^{d} J^{(k)}\right| &\leq \sum_{i=1}^{d-2} \binom{d-2}{i} (\varepsilon_{n_1})^i \\ &= (1+\varepsilon_{n_1})^{d-2} - 1 \leq \varepsilon_{(d-2)n_1},\end{aligned} \tag{46}$$

where the first inequality is obtained from

$$\left|\prod_{i=1}^{k}(a_i+b_i) - \prod_{i=1}^{k} a_i\right| \leq \sum_{i=1}^{k} \binom{k}{i} a^{k-i}b^i,$$

where $|a_i| \leq a$ and $|b_i| \leq b$ for all i, and using that $|j^{(k)}_{rs}| \leq 1$ and $|(E^{(k)}_J)_{rs}| \leq \varepsilon_{n_1}$. Using (44), we find that (45) plus (46) is bounded by $\varepsilon_{(d-2)(n_1+1)}$, and consequently,

$$\underline{H} := \mathrm{fl}\left(\bigodot_{\substack{k=1\\k\neq i,j}}^{d} \underline{J}^{(k)}\right) = \bigodot_{\substack{k=1\\k\neq i,j}}^{d} J^{(k)} + E^{(i,j)}_{\odot J}, \quad \text{where} \quad |E^{(i,j)}_{\odot J}| \leq \varepsilon_{(d-2)(n_1+1)}. \tag{47}$$

Let $K := {A^{(j)}}^T X$, then we obtain

$$\underline{K} := \mathrm{fl}\left({A^{(j)}}^T \underline{X}\right) = {A^{(j)}}^T X + {A^{(j)}}^T E_X + E_{AX'} := K + E_{AX}, \tag{48}$$

where, by the product rule (43),

$$|E_{AX'}| \leq \varepsilon_{n_j} |A^{(j)}|^T |\underline{X}| \leq \varepsilon_{n_j} \|{A^{(j)}}^T\|_\infty \|\underline{X}\|_{\max} = \varepsilon_{n_j} \|A^{(j)}\|_1 \|\underline{X}\|_{\max},$$

and, thus,

$$\begin{aligned}
|E_{AX}| &\leq |E_{AX'}| + |A^{(j)}|^T |E_X| \leq \varepsilon_{n_j} \|A^{(j)}\|_1 \|\underline{X}\|_{\max} + \|A^{(j)}\|_1 \|E_X\|_{\max} \\
&\leq \varepsilon_{n_j} \alpha_j \|\underline{X}\|_{\max} + \varepsilon_X \alpha_j \\
&\leq \varepsilon_{n_j} \alpha_j \|X\|_{\max} + \alpha_j (1 + \varepsilon_{n_j}) \varepsilon_X.
\end{aligned}$$

Then, combining (47) and (48),

$$\begin{aligned}
\underline{H_X^{(i,j)}} &= \mathrm{fl}\left((K + E_{AX}) \odot \left(E_{\odot J}^{(i,j)} + \bigodot_{\substack{k=1 \\ k \neq i,j}}^{d} J^{(k)} \right) \right) \\
&= H_X^{(i,j)} + E_{AX} \odot \bigodot_{\substack{k=1 \\ k \neq i,j}}^{d} J^{(k)} + E_{\odot J}^{(i,j)} \odot K + E_{AX} \odot E_{\odot J}^{(i,j)} + E_{K \odot J}^{(i,j)} \\
&:= H_X^{(i,j)} + E_{H_X}^{(i,j)}.
\end{aligned} \tag{49}$$

We can then bound

$$|E_{H_X}^{(i,j)}| \leq \left(|E_{AX}| + |K| \odot |E_{\odot J}^{(i,j)}| + |E_{AX}| \odot |E_{\odot J}^{(i,j)}| \right) + |E_{K \odot J}^{(i,j)}|.$$

The last term in the above expression is bounded by ε times the bracketed expression plus $\varepsilon |H_X^{(i,j)}|$, which can be verified through straightforward computations. Furthermore,

$$|H_X^{(i,j)}| = |{A^{(j)}}^T X| \odot \left| \bigodot_{k \neq i,j} J^{(k)} \right| \leq |A^{(j)}|^T |X| \leq \|A^{(j)}\|_1 \|X\|_{\max}. \tag{50}$$

We thus obtain

$$\begin{aligned}
|E_{H_X}^{(i,j)}| &\leq (1+\varepsilon) \left(|E_{AX}| + |K| \odot |E_{\odot J}^{(i,j)}| + |E_{AX}| \odot |E_{\odot J}^{(i,j)}| \right) + \varepsilon |H_X^{(i,j)}| \\
&\leq (1+\varepsilon) \left((1 + \varepsilon_{(d-2)(n_1+1)}) |E_{AX}| + \varepsilon_{(d-2)(n_1+1)} \alpha_j \|X\|_{\max} \right) + \varepsilon \alpha_j \|X\|_{\max} \\
&\leq (1+\varepsilon) \left(\varepsilon_{(d-2)(n_1+1)+n_j} \alpha_j \|X\|_{\max} + (1 + \varepsilon_{(d-2)(n_1+1)})(1 + \varepsilon_{n_j}) \alpha_j \varepsilon_X \right) \\
&\quad + \varepsilon \alpha_j \|X\|_{\max} \\
&\leq \varepsilon_{(d-1)(n_1+1)} \alpha_j \|X\|_{\max} + (1 + \varepsilon_{(d-1)(n_1+1)}) \alpha_j \varepsilon_X,
\end{aligned}$$

noting that we bounded ε_{n_j} by ε_{n_1} in the penultimate step, concluding the proof. □

We shall also need a computable forward error bound for computing the inverse of a matrix. This is handled by the following lemma.

Lemma 4 *Let $A \in \mathbb{R}^{n\times n}$ and $\underline{A} = A + E$. Then*

$$\|A^{-1}\|_1 \leq \frac{\sqrt{n}}{1-\psi} \left\| \mathrm{fl}\left(\underline{A}^{-1}\right)\right\|_F := \delta$$

with

$$\psi := \left\|I - \mathrm{fl}\left(\underline{A}^{-1}\right)\underline{A}\right\|_F + \|\,\mathrm{fl}\left(\underline{A}^{-1}\right)\|_F \|E\|_F,$$

provided that $\psi < 1$.

Proof The lemma follows immediately from [46, Lemma 2.1]:

$$\frac{1}{\sqrt{n}}\|A^{-1}\|_1 \leq \|A^{-1}\|_2 \leq \frac{\left\|\mathrm{fl}\left(\underline{A}^{-1}\right)\right\|_2}{1-\alpha} \quad \text{where} \quad \alpha = \left\|I - \mathrm{fl}\left(\underline{A}^{-1}\right)A\right\|_2 \leq \psi.$$

□

To compute matrix vector products with G/D, Algorithm 3 demands L_2' and D_2' as input. The forward error of their computation via Algorithm 4 is provided by the following theorem.

Theorem 5 *Computing D_2' and L_2' by Algorithm 4 in finite precision floating-point arithmetic results in*

$$\mathrm{fl}\left(D_2'\right) = D_2' + E_{D_2'} \quad \textit{and} \quad \mathrm{fl}\left(L_2'\right) = L_2' + E_{L_2'} \tag{51}$$

where the forward errors are bounded by

$$|E_{L_2'}| \leq (1 + \alpha^{\infty}\alpha_1\delta_0)\varepsilon_{c'+r+1+(\eta+1)(d-1)(n_1+1)}\|C_R\|_1 \tag{52}$$

and

$$|E_{D_2'}| \leq \left(\varepsilon_{2c'+1+(d-1)(n_1+1)} + n_1 r\delta_0\varepsilon_{2c'+n_1r+1+(\eta+1)(d-1)(n_1+1)}\right)\|C_R\|_1^2.$$

Herein, $c' := \Sigma - d + 1$,

$$\eta := 1 + \max_i \|E_{B_{\{i\}}}\|_{\max}\varepsilon_{(d-1)(n_1+1)}^{-1} + \max_i \left\|\underline{\dot{L}}_{\{i\}}\right\|_{\infty},$$

with $E_{B_{\{i\}}} := \underline{\dot{L}}_{\{i\}}\underline{H}^{(1)} - \underline{L}^{\star}_{\{i\}}$,

$$\alpha^{\infty} := \max_{2\leq j\leq d}\|A^{(j)}\|_{\infty}, \quad \textit{and} \quad \alpha_1 := \|A^{(1)}\|_1,$$

and, further,

$$\delta_0 := \frac{\sqrt{r}}{1-\psi_0}\left\|\mathrm{fl}\left((\underline{H}^{(1)})^{-1}\right)\right\|_F$$

with

$$\psi_0 := \left\|I - \mathrm{fl}\left(\underline{H}^{(1)^{-1}}\right)\underline{H}^{(1)}\right\|_F + \left\|\mathrm{fl}\left(\underline{H}^{(1)^{-1}}\right)\right\|_F \|E_H^{(1)}\|_F,$$

where $|E_H^{(1)}| \leq \varepsilon_{(d-1)(n_1+1)}$.

Proof The forward error on the computations in lines 1 through 3 of Algorithm 4 can be handled simultaneously; we have

$$\mathrm{fl}\left((T_R^i)^T T_k^j\right) = (T_R^i)^T T_k^j + E_{T^{(i,j)}}$$

where

$$|E_{T^{(i,i)}}| \leq \varepsilon_{(d-2)(n_1+1)} \quad \text{and} \quad |E_{T^{(i,j)}}| \leq \varepsilon_{(d-1)(n_1+1)};$$

the first bound follows readily from (47), and the second bound exploits the observation that the roundoff error for computing $\mathbf{a}_k^i(\mathbf{a}_R^j)^T$ is bounded by ε, because the vectors $\mathbf{a}$ are normalized. We can thus conclude:

$$\begin{aligned}
\underline{L}_{1,2}^\star &= L_{1,2}^\star + E_{L_{1,2}^\star} && \text{with} \quad |E_{L_{1,2}^\star}| \leq \varepsilon_{(d-1)(n_1+1)} \\
\underline{L}_2^\star &= L_2^\star + E_{L_2^\star} && \text{with} \quad |E_{L_2^\star}| \leq \varepsilon_{(d-1)(n_1+1)} \\
\underline{D}_2 &= D_2 + E_{D_2} && \text{with} \quad |E_{D_2}| \leq \varepsilon_{(d-1)(n_1+1)}.
\end{aligned}$$

The forward error in line 4 follows from applying a reasoning similar to the derivation up to (47) in the proof of Theorem 4:

$$\underline{H}^{(1)} = H^{(1)} + E_H^{(1)} \quad \text{with} \quad |E_H^{(1)}| \leq \varepsilon_{(d-1)(n_1+1)}.$$

Considering line 6, we can write

$$\underline{\dot{L}}_{\{i\}}(H^{(1)} + E_H^{(1)}) = L_{\{i\}}^\star + E_{L_{\{i\}}^\star} + E_{B_{\{i\}}}.$$

Subtracting the true system $\dot{L}_{\{i\}} H^{(1)} = L_{\{i\}}^\star$ yields

$$E_{\dot{L}_{\{i\}}} := \underline{\dot{L}}_{\{i\}} - \dot{L}_{\{i\}} = \left(E_{L_{\{i\}}^\star} + E_{B_{\{i\}}} - \underline{\dot{L}}_{\{i\}} E_H^{(1)}\right) {H^{(1)}}^{-1}.$$

This forward error can be bounded by

$$|E_{\dot{L}_{\{i\}}}| \leq \|{H^{(1)}}^{-1}\|_1 \left(\|E_{B_{\{i\}}}\|_{\max} + \varepsilon_{(d-1)(n_1+1)} + \|\underline{\dot{L}}_{\{i\}} E_H^{(1)}\|_{\max}\right).$$

Applying Lemma 4, we have $\|{H^{(1)}}^{-1}\|_1 \leq \delta_0$. Using $\|AB\|_{\max} \leq \|A\|_\infty \|B\|_{\max}$ on the last term in the brackets, we find

$$\begin{aligned}
|E_{\dot{L}_{\{i\}}}| &\leq \delta_0 \varepsilon_{(d-1)(n_1+1)} \left(\varepsilon_{(d-1)(n_1+1)}^{-1} \|E_{B_{\{i\}}}\|_{\max} + 1 + \|\underline{\dot{L}}_{\{i\}}\|_\infty\right) \\
&\leq \delta_0 \eta \varepsilon_{(d-1)(n_1+1)} \leq \delta_0 \varepsilon_{\eta(d-1)(n_1+1)}.
\end{aligned}$$

Lines 9 and 11 are covered by Theorem 4; applying it yields:

$$\begin{aligned}
|E_{H_{\dot{L}_{\{i\}}}}^{(j,1)}| &\leq \varepsilon_{(d-1)(n_1+1)} \alpha_1 \|\dot{L}_{\{i\}}\|_{\max} + (1 + \varepsilon_{(d-1)(n_1+1)}) \alpha_1 \delta_0 \varepsilon_{\eta(d-1)(n_1+1)} \\
&\leq \alpha_1 \delta_0 \varepsilon_{(\eta+1)(d-1)(n_1+1)},
\end{aligned}$$

where we noted that

$$\|\dot{L}_{\{i\}}\|_{\max} \leq \|L_{\{i\}}^\star\|_{\max} \|{H^{(1)}}^{-1}\|_1 \leq \delta_0.$$

Line 12 computes

$$\underline{\ddot{L}}^{(j)}_{\{i\}} := \mathrm{fl}\left(A^{(j)}\underline{H}^{(j,1)}_{\dot{L}_{\{i\}}}\right) = A^{(j)}H^{(j,1)}_{\dot{L}_{\{i\}}} + A^{(j)}E^{(j,1)}_{H_{\dot{L}_{\{i\}}}} + E'^{(j)}_{\ddot{L}_{\{i\}}} := \ddot{L}_{\{i\}} + E^{(j)}_{\ddot{L}_{\{i\}}},$$

where the error term

$$\begin{aligned}|E'^{(j)}_{\ddot{L}_{\{i\}}}| &\leq \varepsilon_r\|A^{(j)}\|_\infty\left(\left\|H^{(j,1)}_{\dot{L}_{\{i\}}}\right\|_{\max} + \left\|E^{(j,1)}_{H_{\dot{L}_{\{i\}}}}\right\|_{\max}\right)\\ &\leq \varepsilon_r\|A^{(j)}\|_\infty\left(\left\|{A^{(1)}}^T\dot{L}_{\{i\}}\right\|_{\max} + \alpha_1\delta_0\varepsilon_{(\eta+1)(d-1)(n_1+1)}\right),\end{aligned}$$

so that the forward error is bounded by

$$\begin{aligned}|E^{(j)}_{\ddot{L}_{\{i\}}}| &\leq \|A^{(j)}\|_\infty\alpha_1\delta_0(\varepsilon_{(\eta+1)(d-1)(n_1+1)} + \varepsilon_r(1+\varepsilon_{(\eta+1)(d-1)(n_1+1)}))\\ &\leq \|A^{(j)}\|_\infty\alpha_1\delta_0\varepsilon_{r+(\eta+1)(d-1)(n_1+1)}.\end{aligned}$$

Taking the maximum over $2 \leq j \leq d$ and $1 \leq i \leq \Sigma - d + 1$ allows us to write $\underline{\ddot{L}} = \ddot{L} + E_{\ddot{L}}$, where

$$|E_{\ddot{L}}| \leq \delta_0\alpha_1\varepsilon_{r+(\eta+1)(d-1)(n_1+1)}\max_{2\leq j\leq d}\|A^{(j)}\|_\infty.$$

Let $\alpha^\infty := \max_{2\leq j\leq d}\|A^{(j)}\|_\infty$ in the remainder. Note that

$$|\ddot{L}| \leq \|A^{(j)}\|_\infty\|{A^{(1)}}^T\dot{L}_{\{i\}}\|_{\max} \leq \alpha^\infty\alpha_1\delta_0$$

We can now bound the error in line 15 of Algorithm 4, as follows:

$$\underline{L}''_2 := \mathrm{fl}\left(\underline{L}^\star_2 - \underline{\ddot{L}}\right) = L^\star_2 - \ddot{L} + E_{L^\star_2} - E_{\ddot{L}} + E'_{L''_2} := L'_2 + E_{L''_2},$$

where a short calculation reveals

$$|E'_{L''_2}| \leq \varepsilon\left(|\underline{L}^\star_2| + |\underline{\ddot{L}}|\right) \leq \varepsilon(|L^\star_2| + |\ddot{L}| + |E_{L^\star_2}| + |E_{\ddot{L}}|)$$

Because $|L^\star_2| \leq 1$ and $|\ddot{L}| \leq \alpha^\infty\alpha_1\delta_0$, we obtain

$$\begin{aligned}|E_{L''_2}| &\leq \varepsilon(1+\alpha^\infty\alpha_1\delta_0) + (1+\varepsilon)(\varepsilon_{(d-1)(n_1+1)} + \alpha^\infty\alpha_1\delta_0\varepsilon_{r+(\eta+1)(d-1)(n_1+1)})\\ &\leq \varepsilon_{r+1+(\eta+1)(d-1)(n_1+1)}\alpha^\infty\alpha_1\delta_0 + \varepsilon_{1+(d-1)(n_1+1)}\\ &\leq \varepsilon_{r+1+(\eta+1)(d-1)(n_1+1)}(1+\alpha^\infty\alpha_1\delta_0),\end{aligned}$$

and, furthermore, $|L''_2| \leq |L^\star_2| + |\ddot{L}| \leq 1 + \alpha^\infty\alpha_1\delta_0$.

We note that applying P_R introduces no roundoff error, as we simply select some columns of $\underline{L}''_2$. Then, the forward error of the computation of L'_2 is obtained as follows:

$$\underline{L}'_2 := \mathrm{fl}\left(\underline{L}''_2P_RC_R\right) = \underline{L}''_2P_RC_R + E'_{L'_2} = L'_2 + E_{L''_2}P_RC_R + E'_{L'_2} = L'_2 + E_{L'_2},$$

which we can bound, after some computations, by

$$\begin{aligned}|E_{L'_2}| &\leq (1+\alpha^\infty\alpha_1\delta_0)\|P_RC_R\|_1\varepsilon_{r+1+(\eta+1)(d-1)(n_1+1)} + \varepsilon_{c'}\left(\|L''_2\|_{\max} + \|E_{L''_2}\|_{\max}\right)\|P_RC_R\|_1\\ &\leq (1+\alpha^\infty\alpha_1\delta_0)\|C_R\|_1\left((1+\varepsilon_{c'})\varepsilon_{r+1+(\eta+1)(d-1)(n_1+1)} + \varepsilon_{c'}\right)\\ &\leq (1+\alpha^\infty\alpha_1\delta_0)\|C_R\|_1\varepsilon_{c'+r+1+(\eta+1)(d-1)(n_1+1)},\end{aligned}$$

where we observed $\|P_R C_R\|_1 \leq \|P_R\|_{\max}\|C_R\|_1$ and P_R is a binary matrix. This provides a bound on the forward error of the computation of L_2' by Algorithm 4.

Finally, the forward error for the computation of D_2' in line 16 can be derived. Consider

$$\underline{\dot{D}}_2 := \mathrm{fl}\left(\underline{L}_{1,2}^{\star}\underline{\dot{L}}\right) = L_{1,2}^{\star}\dot{L} + E_{\dot{D}_2},$$

where, after some computations, the error term may be bounded as

$$\begin{aligned}
|E_{\dot{D}_2}| &\leq |\underline{L}_{1,2}^{\star}E_{\dot{L}}| + |E_{L_{1,2}^{\star}}\dot{L}| + \varepsilon_{n_1 r}|\underline{L}_{1,2}^{\star}||\underline{\dot{L}}| \\
&\leq \|\underline{L}_{1,2}^{\star}\|_{\infty}\left(\delta_0\varepsilon_{\eta(d-1)(n_1+1)} + \varepsilon_{n_1 r}(\|\dot{L}\|_{\max} + \|E_{\dot{L}}\|_{\max}\right) + \|E_{L_{1,2}^{\star}}\|_{\infty}\|\dot{L}\|_{\max} \\
&\leq \|\underline{L}_{1,2}^{\star}\|_{\infty}\delta_0\left(\varepsilon_{\eta(d-1)(n_1+1)} + \varepsilon_{n_1 r}(1+\varepsilon_{\eta(d-1)(n_1+1)})\right) + n_1 r\varepsilon_{(d-1)(n_1+1)}\delta_0 \\
&\leq n_1 r\delta_0\varepsilon_{n_1 r+(\eta+1)(d-1)(n_1+1)},
\end{aligned}$$

where in the last step we observed $\|\underline{L}_{1,2}^{\star}\|_{\infty} \leq n_1 r\|\underline{L}_{1,2}^{\star}\|_{\max}$ and that even in floating-point arithmetic the product of numbers smaller than one remains bounded by one. Then,

$$\underline{\ddot{D}}_2 := \mathrm{fl}\left(\underline{D}_2 - \underline{\dot{D}}_2\right) = D_2 - \dot{D}_2 + E_{\ddot{D}_2} := \ddot{D}_2 + E_{\ddot{D}_2},$$

where the forward error is bounded by

$$\begin{aligned}
|E_{\ddot{D}_2}| &\leq (1+\varepsilon)\left(|E_{D_2}| + |E_{\dot{D}_2}|\right) + \varepsilon(|D_2| + |\dot{D}_2|) \\
&\leq (1+\varepsilon)\left(\varepsilon_{(d-1)(n_1+1)} + |E_{\dot{D}_2}|\right) + \varepsilon(1+n_1 r\delta_0) \\
&\leq \varepsilon_{1+(d-1)(n_1+1)} + n_1 r\delta_0\varepsilon_{n_1 r+1+(\eta+1)(d-1)(n_1+1)},
\end{aligned}$$

where in the first inequality we observed that

$$|\dot{D}_2| \leq \|L_{1,2}^{\star}\|_{\infty}\|\dot{L}\|_{\max} \leq n_1 r\|L_{1,2}^{\star}\|_{\max}\|\dot{L}\|_{\max} \leq n_1 r\delta_0.$$

Consequently, $|\ddot{D}| \leq 1 + n_1 r\delta_0$. Proceeding with the matrix multiplication, we write

$$\underline{\dddot{D}}_2 := \mathrm{fl}\left(\underline{\ddot{D}}_2 P_R C_R\right) = \ddot{D}_2 P_R C_R + E_{\dddot{D}_2},$$

where the forward error obeys

$$\begin{aligned}
|E_{\dddot{D}_2}| &\leq |E_{\ddot{D}_2}||P_R C_R| + \varepsilon_{c'}|\underline{\ddot{D}}_2||P_R C_R| \\
&\leq \left((1+\varepsilon_{c'})\|E_{\ddot{D}_2}\|_{\max} + \varepsilon_{c'}(\|D_2\|_{\max} + \|\dot{D}_2\|_{\max})\right)\|C_R\|_1 \\
&\leq \left((1+\varepsilon_{c'})\|E_{\ddot{D}_2}\|_{\max} + \varepsilon_{c'}(1+n_1 r\delta_0)\right)\|C_R\|_1 \\
&\leq \left(\varepsilon_{c'+1+(d-1)(n_1+1)} + n_1 r\delta_0\varepsilon_{c'+n_1 r+1+(\eta+1)(d-1)(n_1+1)}\right)\|C_R\|_1.
\end{aligned}$$

Finally, the last matrix product reads

$$\underline{D}_2' = \mathrm{fl}\left(C_R^T P_R^T \underline{\dddot{D}}_2\right) = C_R^T P_R^T \dddot{D}_2 + E_{D_2'},$$

in which the error term is similarly bounded:

$$\begin{aligned}
|E_{D_2'}| &\leq |P_R C_R|^T |E_{\dddot{D}_2}| + \varepsilon_{c'}|P_R C_R|^T |\underline{\dddot{D}}_2| \\
&\leq \left((1+\varepsilon_{c'})\|E_{\dddot{D}_2}\|_{\max} + \varepsilon_{c'}\|\ddot{D}_2 P_R C_R\|_{\max}\right)\|C_R\|_1 \\
&\leq \left((1+\varepsilon_{c'})\|E_{\dddot{D}_2}\|_{\max} + \varepsilon_{c'}(1+n_1 r\delta_0)\|C_R\|_1\right)\|C_R\|_1 \\
&\leq \left(\varepsilon_{2c'+1+(d-1)(n_1+1)} + n_1 r\delta_0\varepsilon_{2c'+n_1 r+1+(\eta+1)(d-1)(n_1+1)}\right)\|C_R\|_1^2.
\end{aligned}$$

This concludes the derivation. □

The previous theorem illustrates that the forward error on the computation of L_2' and D_2' may be high. Fortunately, these matrices should be computed only once; it is feasible to compute them in higher precision, i.e., with a roundoff error $\varepsilon' \ll \varepsilon$, and then round the result to the normal precision. As a hypothetical situation, we could use single precision floating point calculations throughout, i.e., $\varepsilon = 2^{-24}$, but compute L_2' and D_2' using double precision floating point arithmetic, i.e., $\varepsilon' = 2^{-53} \ll 2^{-24}$. Provided that the forward error of computing L_2' and D_2' is strictly smaller than ε, rounding the double precision result to the nearest single precision floating point number would result in a relative forward error smaller than ε for the calculation of L_2' and D_2'. We therefore adhere to

Assumption 1 *Computing D_2' and L_2' by Algorithm 4 in finite precision floating-point arithmetic results in*

$$\mathrm{fl}\left(D_2'\right) = D_2' + E_{D_2'} \quad \textit{and} \quad \mathrm{fl}\left(L_2'\right) = L_2' + E_{L_2'} \tag{53}$$

where

$$|E_{D_2'}| \leq \varepsilon \quad \textit{and} \quad |E_{L_2'}| \leq \varepsilon. \tag{54}$$

In our numerical experiments, all calculations are by default performed in a fixed precision floating point arithmetic with $\varepsilon = 2^{-106}$. The calculation of L_2' and D_2' via Algorithm 4 is performed in fixed precision floating point arithmetic with $\varepsilon' = 2^{-212}$. Using Theorem 5, we can verify whether the obtained error bound is strictly smaller than ε, i.e., whether assumption 1 is valid.

Finally, we can bound the forward error of a matrix-vector product with G/D, as implemented in Algorithm 3:

Proof (Proof of Theorem 6)

The forward error for the computation in line 17 of Algorithm 3 is given by (47) in the proof of Theorem 4. Analogously, we find

$$\underline{H}^{(i)} = H^{(i)} + E_H^{(i)} \quad \text{where} \quad |E_H^{(i)}| \leq \varepsilon_{(d-1)(n_1+1)}.$$

providing the forward error for lines 7 and 20. Note that $|\underline{H}^{(i)}| \leq 1$ because even in floating-point arithmetic, the product of numbers smaller than one will remain smaller than one.

Bounding the error of the computation in lines 4 and 14 of the algorithm is more involved. First, we can apply Theorem 4 with $\underline{X} = \underline{Z}^{(j)} = Z^{(j)}$, so that we may write

$$\underline{H}^{(i,j)}_{Z^{(j)}} = H^{(i,j)}_{Z^{(j)}} + E^{(i,j)}_{H_{Z^{(j)}}}$$

with

$$|E^{(i,j)}_{H_{Z^{(j)}}}| \leq \varepsilon_{(d-1)(n_1+1)} \alpha_j \|Z^{(j)}\|_{\max}.$$

The sum is handled as follows: using (41), we find

$$\begin{aligned}\left|\mathrm{fl}\left(\sum_{j\neq i} \underline{H}^{(i,j)}_{Z^{(j)}}\right) - \sum_{j\neq i} \underline{H}^{(i,j)}_{Z^{(j)}}\right| &\leq \varepsilon_{d-2} \sum_{j\neq i} \left|\underline{H}^{(i,j)}_{Z^{(j)}}\right| \\ &\leq \varepsilon_{d-2} \sum_{j\neq i} \left(|H^{(i,j)}_{Z^{(j)}}| + |E^{(i,j)}_{H_{Z^{(j)}}}|\right).\end{aligned}$$

Then, the forward error can be bounded as follows:

$$\begin{aligned}\left|\underline{W}^{(i)} - W^{(i)}\right| &= \left|\underline{W}^{(i)} - \sum_{j\neq i} H^{(i,j)}_{Z^{(j)}} - \sum_{j\neq i} E^{(i,j)}_{H_{Z^{(j)}}} + \sum_{j\neq i} E^{(i,j)}_{H_{Z^{(j)}}}\right| \\ &\leq \varepsilon_{d-2} \sum_{j\neq i} \left(\left|H^{(i,j)}_{Z^{(j)}}\right| + \left|E^{(i,j)}_{H_{Z^{(j)}}}\right|\right) + \sum_{j\neq i}\left|E^{(i,j)}_{H_{Z^{(j)}}}\right| \\ &\leq \varepsilon_{d-2} \sum_{j\neq i} \alpha_j \|Z^{(j)}\|_{\max} + (1+\varepsilon_{d-2})\varepsilon_{(d-1)(n_1+1)} \sum_{j\neq i} \alpha_j \|Z^{(j)}\|_{\max} \\ &\leq \varepsilon_{(d-2)+(d-1)(n_1+1)} \sum_{j\neq i} \alpha_j \|Z^{(j)}\|_{\max},\end{aligned}$$

where (50) was used in the next to last step. Setting $\zeta := \sum_j \alpha_j \|Z^{(j)}\|_{\max}$, we may conclude that

$$\underline{W}^{(i)} = W^{(i)} + E^{(i)}_W \quad \text{with} \quad |E^{(i)}_W| \leq \varepsilon_{(d-1)(n_1+2)}\zeta,$$

completing the analysis for lines 4 and 14 in Algorithm 3.

The computation in line 6 is

$$\underline{\dot{W}} := \mathrm{fl}\left(A^{(1)}(\underline{W}^{(1)})^T\right) = A^{(1)}{W^{(1)}}^T + A^{(1)}{E^{(1)}_W}^T + E^{(1)}_{\dot{W}'} := A^{(1)}{W^{(1)}}^T + E^{(1)}_{\dot{W}},$$

where

$$\begin{aligned}|E^{(1)}_{\dot{W}}| &\leq \left|A^{(1)}\right|\left|E^{(1)}_W\right|^T + \varepsilon_r \left|A^{(1)}\right|\left|W^{(1)} + E^{(1)}_W\right|^T \\ &\leq \|A^{(1)}\|_\infty((1+\varepsilon_r)\|E^{(1)}_W\|_{\max} + \varepsilon_r\|W^{(1)}\|_{\max}) \\ &\leq (1+\varepsilon_r)\varepsilon_{(d-1)(n_1+2)}\zeta\|A^{(1)}\|_\infty + \varepsilon_r\zeta\|A^{(1)}\|_\infty \\ &\leq \varepsilon_{r+(d-1)(n_1+2)}\zeta\|A^{(1)}\|_\infty,\end{aligned}$$

where in the next to last inequality we noted that

$$\|W^{(1)}\|_{\max} = \left\|\sum_{j\neq i} H^{(i,j)}_{Z^{(j)}}\right\|_{\max} \leq \sum_{j\neq i}\left\|{A^{(j)}}^T Z^{(j)}\right\|_{\max} \leq \zeta,$$

using (50) again.

To bound the error in line 8, we note that numerically we have computed

$$\underline{\ddot{W}}\left(H^{(1)} + E^{(1)}_H\right) = A^{(1)}{W^{(1)}}^T + E^{(1)}_{\dot{W}} + E_B,$$

where $\underline{\ddot{W}}$ is the obtained numerical solution and $E_B := \underline{\ddot{W}}\underline{H}^{(1)} - \underline{\dot{W}}$ is the backward error. Subtracting the true system $\ddot{W}H^{(1)} = \dot{W}$ from the above system yields, after some regrouping,

$$E_{\ddot{W}} := \underline{\ddot{W}} - \ddot{W} = \left(E^{(1)}_{\dot{W}} + E_B - \underline{\ddot{W}}E^{(1)}_H\right){H^{(1)}}^{-1},$$

which is the forward error. It can be bounded by

$$|E_{\ddot{W}}| \leq \left\|{H^{(1)}}^{-1}\right\|_1 (\|E_B\|_{\max} + \varepsilon_{r+(d-1)(n_1+2)}\zeta\|A^{(1)}\|_\infty + \varepsilon_{(d-1)(n_1+1)}\|\underline{\ddot{W}}\|_\infty),$$

having used $\|AB\|_{\max} \leq \|A\|_\infty \|B\|_{\max}$. Then, we apply Lemma 4:

$$\left\| H^{(1)^{-1}} \right\|_1 \leq \frac{\sqrt{r}}{1-\psi_1} \left\| \mathrm{fl}\left((\underline{H}^{(1)})^{-1} \right) \right\|_F := \delta_1,$$

where ψ_1 is the readily computable quantity

$$\psi_1 := \left\| I - \mathrm{fl}\left((\underline{H}^{(1)})^{-1} \right) \underline{H}^{(1)} \right\|_F + \left\| \mathrm{fl}\left((\underline{H}^{(1)})^{-1} \right) \right\|_F \|E_H^{(1)}\|_F;$$

for simplicity of future reference, we define

$$\varepsilon_{\ddot{W}} := \|E_B\|_{\max} + \varepsilon_{r+(d-1)(n_1+2)} \zeta \|A^{(1)}\|_\infty + \varepsilon_{(d-1)(n_1+1)} \|\underline{\ddot{W}}\|_\infty,$$

so that $|E_{\ddot{W}}| \leq \delta_1 \varepsilon_{\ddot{W}}$.

Handling line 18, we first note that

$$\underline{C} = {A^{(1)}}^T \ddot{W} + E_C \quad \text{where} \quad |E_C| \leq \varepsilon_{n_1} \alpha_1 \|\underline{\ddot{W}}\|_{\max} + \alpha_1 \|E_{\ddot{W}}\|_{\max}.$$

Then, we essentially apply Theorem 4 with a few changes:

$$\underline{H}^{(1,i)}_{\underline{\ddot{W}}} = H^{(1,i)}_{\ddot{W}} + E^{(1,i)}_{H_{\ddot{W}}},$$

where, continuing from (49),

$$\begin{aligned}
|E^{(1,i)}_{H_{\ddot{W}}}| &\leq |\underline{C}| \odot \left| E^{(1,i)}_{\odot J} + \bigodot_{\substack{k=2 \\ k\neq i}}^{d} J^{(k)} \right| - H^{(1,i)}_{\ddot{W}} \\
&\leq |\underline{C}| \odot |E^{(1,i)}_{\odot J}| + |E_C| \odot \left| \bigodot_{\substack{k=2 \\ k\neq i}}^{d} J^{(k)} \right| + |E^{(1,i)}_{K \odot J}| \\
&\leq (\varepsilon + \varepsilon_{(d-2)(n_1+1)}) |\underline{C}| + \varepsilon_{n_1} \alpha_1 \|\underline{\ddot{W}}\|_{\max} + \alpha_1 \|E_{\ddot{W}}\|_{\max} \\
&\leq \varepsilon_{(d-2)(n_1+2)} \|\underline{C}\|_{\max} + \varepsilon_{n_1} \alpha_1 \|\underline{\ddot{W}}\|_{\max} + \alpha_1 \delta_1 \varepsilon_{\ddot{W}},
\end{aligned}$$

where we noted in the third step that $|E^{(1,i)}_{K\odot J}| \leq \varepsilon |\underline{C}| |\underline{H}^{(1,i)}| \leq \varepsilon |\underline{C}|$, because the product of floating-point numbers smaller than 1 in absolute value remains smaller than 1.

The forward error in line 19 satisfies

$$\underline{\dddot{W}}^{(i)} = \dddot{W}^{(i)} + E^{(i)}_{\dddot{W}},$$

with

$$\begin{aligned}
|E^{(i)}_{\dddot{W}}| &\leq (1+\varepsilon) \left(|E^{(i)}_W| + |E^{(1,i)}_{H_{\ddot{W}}}| \right) + \varepsilon |W^{(i)}| + \varepsilon |H^{(1,i)}_{\ddot{W}}| \\
&\leq (1+\varepsilon) \left(\varepsilon_{(d-1)(n_1+2)} \zeta + \varepsilon_{(d-2)(n_1+2)} \|\underline{C}\|_{\max} + \varepsilon_{n_1} \alpha_1 \|\underline{\ddot{W}}\|_{\max} + \alpha_1 \delta_1 \varepsilon_{\ddot{W}} \right) \\
&\qquad + \varepsilon \zeta + \varepsilon \|\underline{C}\|_{\max} \\
&\leq \varepsilon_{(d-1)(n_1+3)} (\zeta + \|\underline{C}\|_{\max}) + \varepsilon_{n_1+1} \alpha_1 \|\underline{\ddot{W}}\|_{\max} + \alpha_1 \delta_1 (1+\varepsilon) \varepsilon_{\ddot{W}}.
\end{aligned}$$

Next, line 20 computes

$$\underline{\dot{Y}}^{(i)} := \mathrm{fl}\left(A^{(i)} \underline{\ddot{W}}^{(i)} \right) = A^{(i)} \ddot{W}^{(i)} + A^{(i)} E^{(i)}_{\ddot{W}} + E^{(i)}_{\dot{Y}'} := A^{(i)} \ddot{W}^{(i)} + E^{(i)}_{\dot{Y}}, \tag{55}$$

where the error is readily bounded by

$$|E_{\dot{Y}}^{(i)}| \leq \|A^{(i)}\|_\infty \|E_{\ddot{W}}^{(i)}\|_{\max} + \varepsilon_r \|A^{(i)}\|_\infty \|\underline{\ddot{W}}\|_{\max}.$$

Finally, line 22 can be handled. First, we have

$$\mathrm{fl}\left(Z^{(i)}\underline{H}^{(i)}\right) = Z^{(i)}H^{(i)} + Z^{(i)}E_H^{(i)} + E_{ZH'}^{(i)} := Z^{(i)}H^{(i)} + E_{ZH}^{(i)}$$

where the error term satisfies the bound

$$|E_{ZH}^{(i)}| \leq |Z^{(i)}||E_H^{(i)}| + \varepsilon_r |Z^{(i)}||\underline{H}^{(i)}| \leq \varepsilon_{r+(d-1)(n_1+1)} \|Z^{(i)}\|_\infty.$$

Then, the forward error of line 22 can be determined:

$$\underline{Y}_1^{(i)} := \mathrm{fl}\left(\underline{\dot{Y}}^{(i)} + \mathrm{fl}\left(Z^{(i)}\underline{H}^{(i)}\right)\right) = \dot{Y}^{(i)} + E_{\dot{Y}}^{(i)} + Z^{(i)}H^{(i)} + E_{ZH}^{(i)} + E_{Y_1'}^{(i)} := Y_1^{(i)} + E_{Y_1}^{(i)}$$

Herein,

$$|E_{Y_1'}^{(i)}| \leq \varepsilon \left|\underline{\dot{Y}}^{(i)}\right| + \varepsilon \left|\mathrm{fl}\left(Z^{(i)}\underline{H}^{(i)}\right)\right|.$$

Concluding,

$$\begin{aligned}|E_{Y_1}^{(i)}| \leq{}& \|A^{(i)}\|_\infty(\varepsilon_{(d-1)(n_1+3)}(\zeta + \|\underline{C}\|_{\max}) + \varepsilon_{n_1+1}\alpha_1\|\underline{\ddot{W}}\|_{\max} + \alpha_1\delta_1(1+\varepsilon)\varepsilon_{\dot{W}}) \\ &+ \varepsilon_r\|A^{(i)}\|_\infty\|\underline{\ddot{W}}\|_{\max} + \varepsilon_{r+(d-1)(n_1+1)}\|Z^{(i)}\|_\infty \\ &+ \varepsilon\left\|\underline{\dot{Y}}^{(i)}\right\|_{\max} + \varepsilon\left\|\mathrm{fl}\left(Z^{(i)}\underline{H}^{(i)}\right)\right\|_{\max}.\end{aligned}$$

The total forward error after the execution of the loop in lines 10 to 23 of Algorithm 3 is then

$$\mathrm{fl}\left((S^\star - L_1^\star D_1^{-1} L_1^{\star T})\mathbf{z}_\star\right) = (S^\star - L_1^\star D_1^{-1} L_1^{\star T})\mathbf{z}_\star + \mathbf{e}, \quad \text{where} \quad |\mathbf{e}| \leq \max_{2\leq i\leq d} E_{Y_1}^{(i)}.$$

We conclude the analysis with line 25. First, we compute

$$\underline{\mathbf{f}}_1 := \mathrm{fl}\left((\underline{L}_2')^T\mathbf{z}_\star\right) = L_2'^T\mathbf{z}_\star + E_{L_2'}^T\mathbf{z}_\star + E_{\mathbf{f}_1'} := \mathbf{f}_1 + E_{\mathbf{f}_1},$$

where the error term is bounded by

$$|E_{\mathbf{f}_1}| \leq \varepsilon_{\Pi-n_1 r}\|\underline{L}_2'\|_1\|\mathbf{z}_\star\|_{\max} + \|E_{L_2'}\|_{\max}\|\mathbf{z}_\star\|_1$$

using Lemma 5. By the same lemma, $(\underline{D}_2')^{-1}$ is computed beforehand with forward error ε, so that

$$\underline{\mathbf{f}}_2 := \mathrm{fl}\left((\underline{D}_2')^{-1}\underline{\mathbf{f}}_1\right) = D_2'^{-1}\mathbf{f}_1 + D_2'^{-1}E_{\mathbf{f}_1} + E_{(D_2')^{-1}}\underline{\mathbf{f}}_1 + E_{\mathbf{f}_2'} := \mathbf{f}_2 + E_{\mathbf{f}_2},$$

and the error may be bounded by

$$\begin{aligned}|E_{\mathbf{f}_2}| &\leq \left(\|E_{(D_2')^{-1}}\|_{\max} + \varepsilon_c\|(\underline{D}_2')^{-1}\|_{\max}\right)\|\underline{\mathbf{f}}_1\|_1 \\ &\quad + (\|(\underline{D}_2')^{-1}\|_\infty + \|E_{(D_2')^{-1}}\|_\infty)\|E_{\mathbf{f}_1}\|_{\max} \\ &\leq \left(\|E_{(D_2')^{-1}}\|_{\max} + \varepsilon_c\|(\underline{D}_2')^{-1}\|_{\max}\right)\|\underline{\mathbf{f}}_1\|_1 \\ &\quad + (\|(\underline{D}_2')^{-1}\|_\infty + c\|E_{(D_2')^{-1}}\|_{\max})\|E_{\mathbf{f}_1}\|_{\max}.\end{aligned}$$

The final matrix product is

$$\underline{\mathbf{f}_3} := \mathrm{fl}\left(\underline{L_2'}\underline{\mathbf{f}_2}\right) = L_2'\mathbf{f}_2 + E_{L_2'}\underline{\mathbf{f}_2} + L_2'E_{\mathbf{f}_2} + E_{\mathbf{f}_3'} := \mathbf{f}_3 + E_{\mathbf{f}_3};$$

the error term is bounded as follows:

$$\begin{aligned}|E_{\mathbf{f}_3}| &\leq \|E_{L_2'}\|_{\max}\|\underline{\mathbf{f}_2}\|_1 + (\|\underline{L_2'}\|_\infty + \|E_{L_2'}\|_\infty)\|E_{\mathbf{f}_2}\|_{\max} + \varepsilon_c\|\underline{L_2'}\|_\infty\|\underline{\mathbf{f}_2}\|_{\max} \\ &\leq \|E_{L_2'}\|_{\max}\|\underline{\mathbf{f}_2}\|_1 + (\|\underline{L_2'}\|_\infty + c\|E_{L_2'}\|_{\max})\|E_{\mathbf{f}_2}\|_{\max} + \varepsilon_c\|\underline{L_2'}\|_\infty\|\underline{\mathbf{f}_2}\|_{\max},\end{aligned}$$

where c is as in Theorem 2. Note that $\|E_{L_2'}\|_{\max}$ and $\|E_{(D_2')^{-1}}\|_{\max}$ can be bounded either by Theorem 5 or Assumption 1, assuming in the latter case that L_2' and $(D_2')^{-1}$ were precomputed in higher precision.

Finally, the forward error algorithm of Algorithm 3 is:

$$|E| \leq \left(|E_{\mathbf{f}_3}| + \max_{2\leq j\leq d}\|E_{Y_1}^{(j)}\|_{\max}\right) + \varepsilon\left(\|\underline{\mathbf{f}_3}\|_{\max} + \max_{2\leq j\leq d}\|\underline{Y_1}^{(j)}\|_{\max}\right). \tag{56}$$

□

Theorem 6 *Computing $G/D\mathbf{z}_\star$ by Algorithm 3 in finite precision floating-point arithmetic results in*

$$\mathrm{fl}\,(G/D\mathbf{z}) = G/D\mathbf{z} + \mathbf{e}.$$

where the components of $\mathbf{e}$ *are bounded by (56).*

In section 4.2 we mentioned that, in the presented numerical setting, it is not required to compute the rank of D. With the above derivations in mind, we may now understand this. In a numerical setting, it is extremely unlikely that the computation of the inverse of D_1 and D_2' fails, precisely due to roundoff errors. Should the computation of these inverses fail because the Gaussian elimination procedure encounters a pivot which is exactly zero, we terminate the proposed algorithm and let it report UNKNOWN. Usually, neither of the aforementioned matrices is exactly singular, but could be considered numerically singular; i.e., one or more singular values are very small. The key observation is that in the above derivations it was only assumed that D_1 and D_2' are not *exactly* singular; all derivations are still valid for numerically singular matrices. In particular, the forward error bounds of on the computations in the conjugate gradient algorithm applied to G/D will still hold. When performing the hypothesis test as usual, the probability of incorrectly rejecting the null hypothesis is thus still bounded by α^k. In the next section, we present an example of the behavior of the proposed algorithm when D_2' can be considered numerically singular.

6 Experimental results

In this section, we first discuss the details of our computer implementation of the proposed numerical algorithm. Next, the typical behavior of the proposed algorithm, in particular w.r.t. Theorem 3, is investigated in greater detail. Thereafter, a comparison is made with the classic algorithm in terms of execution time and memory consumption; we investigate both a numerical as well as a symbolical implementation of that algorithm. Finally, we present results pertaining to the nondefectivity of Segre varieties obtained with the proposed algorithm.

6.1 Implementation details

The proposed algorithm was implemented in C++ in fixed precision floating point arithmetic. Our code employs the Eigen [29] library, which implements basic data structures such as vectors and matrices and key algorithms such as matrix multiplication. Eigen allows the user to extend the library with other numerical data types; we extended it with fixed precision floating point data types provided by the QD library [34,35]: the `dd_real` class offers double–double floating point arithmetic satisfying the standard model of floating point arithmetic in (31) with $\varepsilon = 2^{-106} \approx 1.23 \cdot 10^{-32}$ and `qd_real` offers quadruple–double floating point arithmetic with $\varepsilon = 2^{-212} \approx 1.52 \cdot 10^{-64}$. Most of our experiments were performed with `dd_real` because its arithmetic is much faster and consumes only half of the memory required by `qd_real`; i.e., 16 versus 32 bytes of memory. Only Algorithm 4 is performed using quadruple-double floating point arithmetic, in order to satisfy Assumption 1.

The proposed algorithm is compared with two implementations of the classic algorithm; we implemented the classic algorithm in Maple 16 using *exact arithmetic* by generating matrices whose integer components are drawn from a uniform distribution between -100 and 100, forming the tangent space matrix exactly, and then computing its exact rank with the `Rank` function.[15] We refer to this approach as the symbolical classic algorithm. Additionally, a numerical implementation of this algorithm was developed in C++ using Eigen and QD: we compute the rank revealing QR decomposition using the Eigen method `fullPivHouseholderQr` in double–double precision and count the number of diagonal elements of the upper triangular factor that are larger in magnitude than 10^{-20}.[16] We stress that, to the best of our knowledge, the probability that this algorithm fails is unknown: the results produced by the numerical classic algorithm are strong suggestions.

The experiments were performed on a computer system comprising two hexa-core processors and 48GB of main memory. Our code, including the QD library, was compiled with the GNU project C and C++ compiler version 4.4.3 with the flags `-O2 -fwhole-program` and `-frounding-math -mfpmath=sse -msse2`. The latter set of flags enforces the compiler to generate machine code such that all floating point operations satisfy the correct rounding for single and double precision floating point values, as specified in the IEEE 754 standard [1], by forcing the compiler to use the available SSE2 registers for performing floating point arithmetic.[17] We additionally compiled the C version of W. Kahan's Paranoia program [51], which attempts to verify the correct implementation of the IEEE 754 standard, with the same compiler and flags; running the program suggests that the target computer system indeed satisfies the IEEE 754 standard.

6.2 A posteriori bound

We consider here the numerical execution of the proposed algorithm. In Figure 1, we visualize a typical progression of the three-term recurrence presented in Theorem 3 for a nondefective Segre variety. $\|A\mathbf{x}\|$ is the backward error, or residual, of the current approximation.

[15] Comparisons with this algorithm are only indicative of true performance, as the computational kernels of Maple might be implemented more efficiently.

[16] We recognize that a more traditional implementation of the classic algorithm in a numerical setting computes the singular value decomposition and counts the number of sufficiently large singular values, e.g., as in [19]; however, as there are no guarantees with respect to the correctness of the outcome, we have preferred, for the sake of this comparison, the fastest algorithm. We additionally performed experiments with the accurate `jacobiSvd` method provided by Eigen, but found that it was very slow compared to the approach based on the QR algorithm; it would sometimes even be outperformed by the symbolic classic algorithm that was implemented in Maple.

[17] See http://gcc.gnu.org/wiki/x87note and http://gcc.gnu.org/wiki/FloatingPointMath.

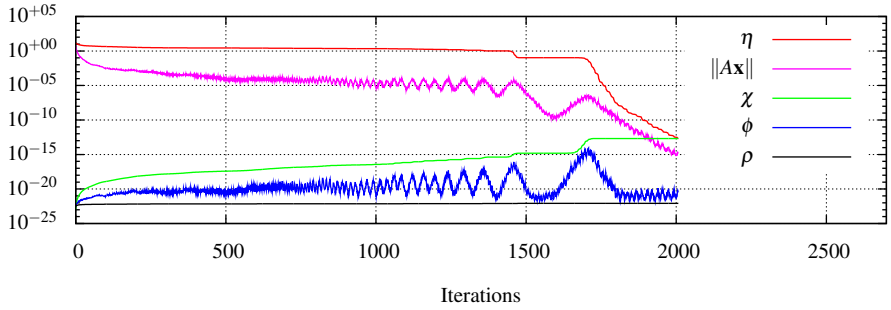


Fig. 1: Typical evolution of the error estimates in the conjugate gradient algorithm for a nondefective Segre variety. Results displayed for the Segre variety $\mathbb{PC}^{10} \times \mathbb{PC}^{9} \times \mathbb{PC}^{8}$.

The computation error related to applying the matrix-vector product, i.e., $\rho \approx 10^{-22}$, can be seen to be large with respect to the roundoff error, $\varepsilon \approx 10^{-32}$, but remains nearly constant. This term in the recurrence dictates the absolute minimum precision which is required to be able to reject the null hypothesis. In practice, the attainable precision will be much less than ρ, as can be seen from χ in the figure. We note that χ increases most strongly whenever there is a peak for ϕ. These peaks are, in turn, caused by values of β strictly larger than 1 in subsequent iterations of the CG algorithm.

It can be observed in Figure 1 that the norm of the approximation, as well as the residual, hardly decreases in the first 1650 iterations, and then suddenly decays. This is to be expected, because the starting vector has standard normally distributed coefficients with respect to any basis for the matrix; the support of the starting vector is noncompact. Therefore, in exact arithmetic, applying the conjugate gradient method to G/D would only be able to reduce the residual to zero after $r(\Sigma - d + 1 - n_1)$ iterations. It is, however, well-known that floating-point arithmetic causes a delay of convergence in the conjugate method, because of the loss of orthogonality [41]. This delayed convergence, which cannot be avoided as the starting vector has a noncompact support in the eigenbasis of G/D by construction, makes the choice of α crucial to attain good performance; it will at least take $r(\Sigma - d + 1 - n_1)$ iterations before η will decrease significantly in magnitude. On the other hand, when convergence sets in, we observed that it progresses rapidly until the attainable accuracy is reached. Therefore, we suggest to choose α as small as feasible.

In Figure 2, the typical behavior of the three-term recurrence is displayed for a defective Segre variety. It shows that the residual can be decreased to approximately the roundoff error, while the norm of the solution remains of order one; for a nondefective Segre variety, in contrast, we observed that from some iteration onward, the norm of the approximation decays approximately as fast as the residual norm, as in Figure 1. Note that the CG algorithm eventually terminates because the upper bound on the computation errors increases to the same order of magnitude as η, even though η remains practically unchanged.

As we mentioned before, Theorem 3 is valid even when D_2' is numerically singular. As an illustration, consider the Segre variety $\mathbb{PC}^{20} \times \mathbb{PC}^{5} \times \mathbb{PC}^{2}$, which is unbalanced and thus defective. We find that D_2' can be considered numerically singular, as its condition number is approximately $6.72 \cdot 10^{34}$. Applying the proposed algorithm results in $\rho^{(0)} = \phi^{(0)} \approx 1.28 \cdot 10^{38}$ and $\chi^{(1)} \approx 5.63 \cdot 10^{6}$. As $\eta^{(1)} \approx 7.32$, the algorithm immediately terminates, and retains the null hypothesis; i.e., there is insufficient evidence that the variety is nondefective.

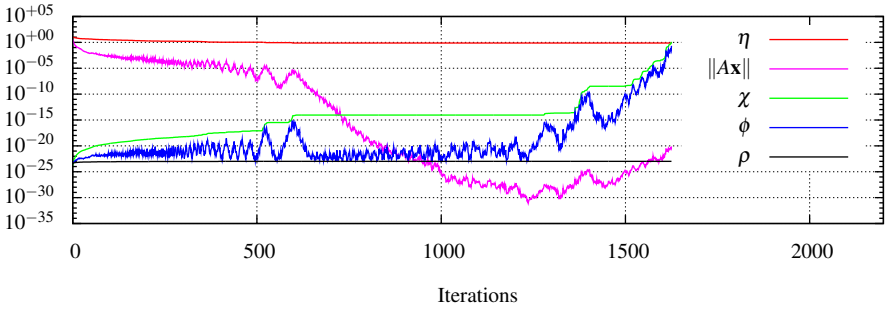


Fig. 2: Typical evolution of the error estimates in the conjugate gradient algorithm for a defective Segre variety. Results displayed for the Segre variety $\mathbb{PC}^{11} \times \mathbb{PC}^{11} \times \mathbb{PC}^{3}$.

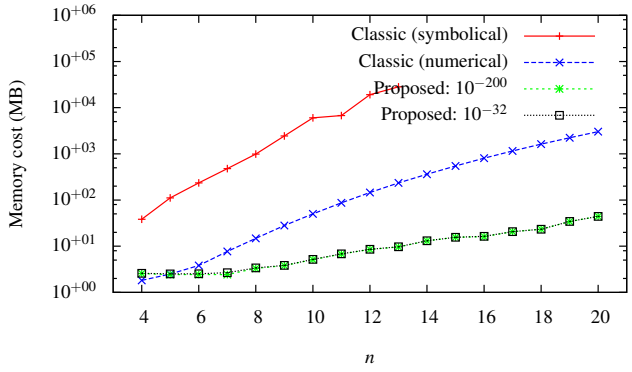
6.3 Comparison with the classic algorithm

In this subsection, the proposed algorithm is run with $\alpha = 10^{-08}$. We tested both $k = 25$ and $k = 4$, so that the type I error of the proposed algorithm was 10^{-200} and 10^{-32}, respectively. The memory cost and execution time of the classic and proposed algorithm in verifying the nondefectivity of the Segre varieties $\mathbb{PC}^n \times \mathbb{PC}^n \times \mathbb{PC}^n$, $4 \leq n \leq 20$ is investigated. The obtained results are summarized in Figure 3.

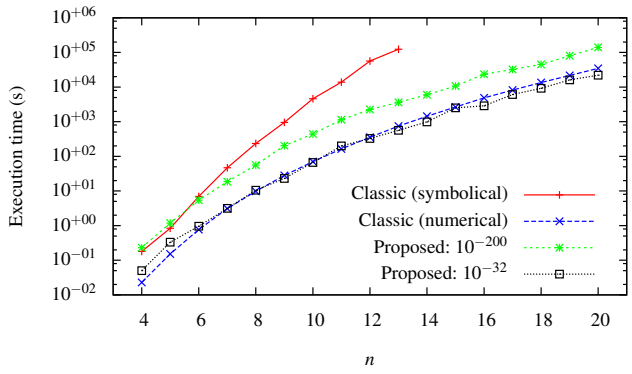
We start by verifying whether the proposed algorithm improves upon the memory cost of the classic algorithm by employing the proposed structure exploiting operations. For the numerical classic algorithm and the proposed algorithm, we monitored the maximum memory consumption as reported by the operating system. The memory consumption of the symbolical classic algorithm was reported by Maple. The measured memory costs are displayed in Figure 3(a). The Maple implementation of the symbolical classic algorithm could not be applied to Segre varieties with $n \geq 14$ because it required more main memory than the 48GB that was available on the computer system. Compared to the numerical classic algorithm, we find, for instance, that it requires about 4.5 times more memory than the proposed method for $n = 8$, and approximately 68 times more memory than the proposed method for $n = 20$. The data illustrate one of the main features of the proposed algorithm: the type I error rate can be reduced *without* increasing the precision of the calculations. This is clear in Figure 3(a), where the memory consumption of the proposed method is independent of the desired type I error rate. On the other hand, for the numerical classic method, it is usually suggested to increase the precision of the calculations to improve the "certainty" of the obtained result; this would entail that the line corresponding to the classic numerical algorithm in Figure 3(a) be shifted upwards.

A comparison of the execution times of the algorithms is presented in Figure 3(b). We note that the proposed algorithm with the type I error of 10^{-32} is consistently able to decide whether the variety is defective within approximately the same time as the numerical classic algorithm, which, we recall, is unable to provide a measure of likeliness that its answer is correct. Increasing the number of tests to decrease the type I error causes the execution time of the proposed algorithm to increase accordingly.

Finally, we mention that all algorithms correctly reported that the considered Segre varieties are nondefective, which was already proved in [39].



(a) Memory cost



(b) Execution time

Fig. 3: A comparison of the execution time and memory cost for verifying the nondefectivity of the Segre variety $\mathbb{PC}^n \times \mathbb{PC}^n \times \mathbb{PC}^n$ with the proposed algorithm and the classic algorithm.

6.4 Results

Because of the relaxed memory requirements imposed by the proposed algorithm, we were able to concurrently verify up to 12 Segre varieties on the target computer system. To verify the nondefectivity of Segre varieties, the proposed algorithm was initially run with $\alpha = 10^{-05}$ and $k = 11$, so that the type I error rate is less than 10^{-55}. With these choices, we verified that nearly all balanced Segre varieties whose expected generic rank is less than or equal to 55 satisfy Conjecture 1: the proposed method does not suggest previously unknown defective Segre varieties. Unfortunately, double–double arithmetic was not suffciently accurate to prove nondefectivity of some shapes conjectured to be nondefective. In particular, the following nondefective Segre varieties could not be proved with double–double arithmetic:

$$n \times n - 1 \times 2 \qquad \text{with } 37 \leq n \leq 55 \quad \text{and} \quad n \neq 38, 39, 41, 42, 43, 45, 46, 49, 51,$$
$$53 \times 5 \times 3 \times 2 \times 2,$$

$54 \times 7 \times 3 \times 3$,
$54 \times 18 \times 2 \times 2$,
$55 \times 8 \times 2 \times 2 \times 2$,

as well as the perfect shapes:

$n \times n \times 2$ with $34 \leq n \leq 55$ and $n \neq 36, 38, 40, 53$,
$34 \times 21 \times 2 \times 2$,
$51 \times 26 \times 3$,
$53 \times 14 \times 5$,
$53 \times 18 \times 2 \times 2$,
$53 \times 27 \times 3$,
$55 \times 28 \times 3$.

Note that all of these Segre varieties, except for $34 \times 21 \times 2 \times 2$, are nearly unbalanced defective. Running the proposed algorithm with quadruple–double arithmetic with $\alpha = 10^{-11}$ and $k = 5$ allowed us to immediately verify the nondefectivity of aforementioned Segre varieties; the type I error is 10^{-55}.

The nondefectivity results pertaining to Segre varieties with more than 4 factors are new, excluding the known case of $(\mathbb{PC}^2)^d$, $d \geq 5$ [17]. With the exception of the nondefectivity of $(\mathbb{PC}^n)^4$, $3 \leq n \leq 8$, and $n = 10$, recently proved by Gesmundo [24], the nondefectivity results for the 4-factor Segre variety are also new. In addition, we verified that both $(\mathbb{PC}^9)^4$ and $(\mathbb{PC}^{11})^4$ are nondefective (type I error 10^{-57}); two cases which could not be handled completely by the approach in [24].

7 Conclusions

We described an alternative probabilistic algorithm based on Terracini's lemma to check whether a given Segre variety is defective. The algorithm exploits the observation that the conjugate gradient method ignores the null space of the matrix in computing a solution to a linear system. We proved that, in a numerical setting, the components initially in the null space of the matrix can be bounded *a posteriori* by the norm of the obtained approximation plus the computation errors. Through a numerical analysis, a computable upper bound on these computation errors was established, allowing us to compute an upper bound on the probability of observing the computed approximations assuming that the Segre variety is defective. If the obtained probability is below a user-defined threshold, the null hypothesis is rejected, and the algorithm concludes that the Segre variety is nondefective. While the proposed algorithm cannot provide the certainty of a fully symbolical calculation, it offers flexibility in terms of execution time and has a significantly lower memory consumption, allowing it to tackle much larger problems than symbolical algorithms. Over a numerical implementation of the classic algorithm, the proposed algorithm has the distinct advantage that it provides the user with a bound on the probability that a defective Segre variety is identified as nondefective. Using the proposed method, it was verified that all Segre varieties whose generic rank is smaller than 55 satisfy the conjecture proposed in [5], with type I error 10^{-55}.

The major challenge of the proposed method is the selection of the parameters α and k. Ideally, α is chosen as small as possible; however, it appears to be difficult to assess what that constitutes. In our experiments, we found that $\alpha = 10^{-5}$ was suitable, although some

Segre varieties could be verified with $\alpha = 10^{-8}$ or even 10^{-12}. In general, it appears that a higher rank entails higher computation errors, thus constraining the choice of α.

There are several extensions of the proposed method that warrant further research. One promising direction consists of employing a block conjugate gradient method with block size k. In this manner, the execution time of the algorithm may be reduced by a factor k, because the Krylov space should be constructed only once for all k vectors, whose support is noncompact, instead of the current approach in which a full Krylov space is constructed k times. Our preliminary experiments suggest that using a block Krylov method in the proposed method may be a viable approach to reduce the execution time, while moderately increasing the memory requirements. A detailed numerical analysis more involved than the one presented in this paper would be required, however, to establish a bound on the type I error similar to this paper. Another extension consists of applying the techniques proposed in this paper to secant varieties of Segre–Veronese varieties.

Acknowledgements The authors were supported by PFV/10/002 Optimization in Engineering (OPTEC), and the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, Belgian Network DYSCO (Dynamical Systems, Control, and Optimization). The first author's research was additionally supported by a Ph.D. fellowship of the Research Foundation—Flanders (FWO). The second author also acknowledges support by the Research Council KU Leuven project OT/11/055 (Spectral Properties of Perturbed Normal Matrices and their Applications), and by the Fund for Scientific Research–Flanders (Belgium) project G034212N (Reestablishing smoothness for matrix manifold optimization via resolution of singularities). The third author was, in addition, supported by the Research Council KU Leuven project OT/10/038 (Multi-parameter model order reduction and its applications).

We are grateful to D. Abbeloos, M. Ferranti and B. Jeuris for the stimulating and encouraging discussions we shared.

References

1. IEEE 754–2008 standard for floating-point arithmetic, 2008.
2. H. Abo. On three conjectures about the secant defectivity of classically studied varieties. In *Proceedings of the Algebraic Geometry Symposium*, Waseda University, Japan, 2010.
3. H. Abo and MC. Brambilla. On the dimensions of secant varieties of Segre-Veronese varieties. *Ann. Mat. Pura Appl.*, pages 1–32, 2011.
4. H. Abo and MC. Brambilla. New examples of defective secant varieties of Segre–Veronese varieties. *Collect. Math.*, 63(3):287–297, 2012.
5. H. Abo, G. Ottaviani, and C. Peterson. Induction for secant varieties of Segre varieties. *Trans. Amer. Math. Soc.*, 361:767–792, 2009.
6. H. Abo, G. Ottaviani, and C. Peterson. Non-defectivity of grassmannians of planes. *J. Algebraic Geom.*, 21:1–20, 2012.
7. J. Alexander and A. Hirschowitz. Polynomial interpolation in several variables. *J. Algebraic Geom.*, 4(2):201–222, 1995.
8. D. J. Bates, C. Peterson, and A. J. Sommese. Application of a numerical version of Terracinis Lemma for secants and joins. In A. Dickenstein, F.-O. Schreyer, and A. J. Sommese, editors, *Algorithms in Algebraic Geometry*, volume 146 of *The IMA Volumes in Mathematics and its Applications*, pages 1–14. Springer New York, 2008.
9. MC. Brambilla and G. Ottaviani. On the Alexander–Hirschowitz theorem. *J. Pure Appl. Algebra*, 212(5):1229–1251, 2008.
10. J. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart–Young" decomposition. *Psychometrika*, 35:283–319, 1970.
11. MV. Catalisano, A.V. Geramita, and A. Gimigliano. Ranks of tensors, secant varieties of Segre varieties and fat points. *Linear Algebra Appl.*, 355(1–3):263–285, 2002.
12. MV. Catalisano, A.V. Geramita, and A. Gimigliano. Higher secant varieties of Segre–Veronese varieties. In C. Ciliberto, A.V. Geramita, B. Harbourne, R.M. Miró-Roig, and K. Ranestad, editors, *Projective varieties with unexpected properties: A volume in memory of Giuseppe Veronese: proceedings of the international conference "Varieties with Unexpected Properties," Siena, Italy, June 8–13, 2004*, pages 81–107, Berlin, Germany, 2005. Walter de Gruyter GmbH & Co. KG.
13. MV. Catalisano, A.V. Geramita, and A. Gimigliano. Higher secant varieties of the Segre varieties $\mathbb{P}^1 \times \ldots \times \mathbb{P}^1$. *J. Pure Appl. Algebra*, 201(1–3):367–380, 2005.

14. MV. Catalisano, A.V. Geramita, and A. Gimigliano. Secant varieties of Grassmann varieties. *Proc. Amer. Math. Soc.*, 133(3):633–642, 2005.
15. MV. Catalisano, A.V. Geramita, and A. Gimigliano. Segre–Veronese embeddings of $\mathbb{P}^1 \times \mathbb{P}^1 \times \mathbb{P}^1$ and their secant varieties. *Collect. Math.*, 1:1–24, 2007.
16. MV. Catalisano, A.V. Geramita, and A. Gimigliano. On the ideals of secant varieties to certain rational varieties. *J. Algebra*, 319(5):1913–1931, 2008.
17. MV. Catalisano, A.V. Geramita, and A. Gimigliano. Secant varieties of $\mathbb{P}^1 \times \cdots \times \mathbb{P}^1$ (n-times) are not defective for $n \geq 5$. *J. Algebraic Geom.*, 20:295–327, 2011.
18. C. Ciliberto and F. Cools. On Grassmann secant extremal varieties. *Adv. Geom.*, 8:377–386, 2008.
19. P. Comon, J.M.F. ten Berge, L. De Lathauwer, and J. Castaing. Generic and typical ranks of multi-way arrays. *Linear Algebra Appl.*, 430(11–12):2997–3007, 2009.
20. L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.
21. V. de Silva and L.-H. Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM J. Matrix Anal. Appl.*, 30(3):1084–1127, 2008.
22. J.W. Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997.
23. A. Edelman and N.R. Rao. Random matrix theory. *Acta Numer.*, 14:1–65, 2005.
24. F. Gesmundo. An asymptotic bound for secant varieties of segre varieties. *arXiv:1209.1732*, 2012.
25. G.H. Golub and Z. Strakoš. Estimates in quadratic formulas. *Numer. Algorithms*, 8(2):241–268, 1994.
26. L. Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. Appl.*, 31(4):2029–2054, 2010.
27. A. Greenbaum. Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences. *Linear Algebra Appl.*, 113:7–63, 1989.
28. A. Greenbaum and Z. Strakoš. Predicting the behavior of finite precision Lanczos and conjugate gradient computations. *SIAM J. Matrix Anal. Appl.*, 13(1):121–137, 1992.
29. G. Guennebaud, B Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.
30. L. Guttman. Enlargement methods for computing the inverse matrix. *The annals of mathematical statistics*, 17(3):336–343, 1946.
31. R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.
32. E.V. Haynsworth. Determination of the inertia of a partitioned Hermitian matrix. *Linear Algebra Appl.*, 1(1):73–81, 1968.
33. M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436, 1952.
34. Y. Hida, X.S. Li, and D.H. Bailey. Algorithms for quad-double precision floating point arithmetic. In *Proceedings of the 15th IEEE Symposium on Computer Arithmetic*, 2001.
35. Y. Hida, X.S. Li, and D.H. Bailey. Library for double–double and quad–double arithmetic. 2008.
36. N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, PA, 1996.
37. T.G. Kolda and B.W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.
38. J.M. Landsberg. *Tensors: Geometry and Applications*, volume 128 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, 2012.
39. T. Lickteig. Typical tensorial rank. *Linear Algebra Appl.*, 69:95–120, 1985.
40. B. McGillivray. A probabilistic algorithm for the secant defect of Grassmann varieties. *Linear Algebra Appl.*, 418:708–718, 2006.
41. G. Meurant and Z. Strakoš. The Lanczos and conjugate gradient algorithms in finite precision arithmetic. *Acta Numer.*, 15:471–542, 2006.
42. M. Mørup. Applications of tensor (multiway array) factorizations and decompositions in data mining. *WIRE: Data Mining Knowl. Disc.*, 1(1):24–40, 2011.
43. Y. Notay. On the convergence of the conjugate gradients in presence of rounding errors. *Numer. Math.*, 65:301–317, 1993.
44. I. V. Oseledets. Tensor-train decomposition. *SIAM J. Sci. Comput.*, 33(5):2295–2317, 2011.
45. C.C. Paige. Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem. *Linear Algebra Appl.*, 34:235–258, 1980.
46. S.M. Rump. Verified bounds for singular values, in particular for the spectral norm of a matrix and its inverse. *BIT*, 51(2):367–384, 2011.
47. Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, PA, USA, 2nd edition, 2003.
48. Z. Strakoš. On the real convergence rate of the conjugate gradient method. *Linear Algebra Appl.*, 154–156:535–549, 1991.
49. Z. Strakoš and P. Tichý. On error estimation in the conjugate gradient method and why it works in finite precision computations. *Electron. Trans. Numer. Anal.*, 13:56–80, 2002.
50. V. Strassen. Rank and optimal computation of generic tensors. *Linear Algebra Appl.*, 52–53:645–685, 1983.
51. T. Sumner and D. Gay. A C version of Kahan's floating point test "paranoia", 1992.
52. A. Terracini. Sulla V_k per cui la varietà degli S_h $h+1$-secanti ha dimensione minore dell'ordinario. *Rend. Circ. Mat. Palermo*, 31:392–396, 1911.

53. C.F. Van Loan. The ubiquitous Kronecker product. *J. Comput. Appl. Math.*, 123(1–2):85–100, 2000.
54. N. Vannieuwenhoven, J. Nicaise, R. Vandebril, and K. Meerbergen. On the nonexistence of the Schmidt–Eckart–Young decomposition for tensors. Tech. rep., KU Leuven, Leuven, Belgium, 2013.
55. N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen. A new truncation strategy for the higher-order singular value decomposition. *SIAM J. Sci. Comput.*, 34(2):A1027–A1052, 2012.
56. J. H. Wilkinson. *The Algebraic Eigenvalue Problem*. Oxford University Press, Amen House, London E.C. 4, United Kingdom, 1965.
57. F. L. Zak. Tangents and secants of algebraic varieties. In *Translations of mathematical monographs*, volume 127. AMS, Providence, Rhode Island, 1993.