Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 11(7): 21235–21250.
DOI: 10.3934/math.2026862
Received: 10 May 2026
Revised: 08 July 2026
Accepted: 13 July 2026
Published: 17 July 2026

Research article

Accurate computations of singular values for fg-Vandermonde-type matrices

Jiaxian Huang* **and Jiaqi Shu**

School of Mathematics and Statistics, Hunan Normal University, Changsha 410081, China

* **Correspondence:** Email: 202410100222@hunnu.edu.cn.

Abstract: This paper focused on accurate singular value computation for fg-Vandermonde-type (fg-VT) matrices, which are widely used in interpolation and other applications. We first derived explicit expressions for the initial minors of fg-VT matrices, then presented parametrization formulas for their corresponding matrices. Based on these results, we proposed an algorithm with high relative accuracy for computing all singular values. Additionally, we developed perturbation theory and error analysis to further assess the accuracy of our approach. Finally, numerical experiments were conducted to verify high accuracy of the algorithm.

Keywords: singular values; parametrization matrices; fg-Vandermonde-type matrices; high relative accuracy; bidiagonal decompositions
Mathematics Subject Classification: 65F15, 15A18

1. Introduction

High-accuracy computations for structured matrices are an active research topic in numerical linear algebra and have received growing attention in recent years. Examples include totally nonnegative matrices [1–3], totally nonpositive matrices [4, 5], sign regular matrices [6–8], and rank-structured matrices [9, 10]. For parameterized negative matrices, recent work [11] presents an algorithm for accurately computing singular values. Moreover, high-accuracy algorithms for singular value computation have been developed for Bernstein-Vandermonde matrices [12] and fg-Vandermonde matrices [13]. In this paper, we introduce a subclass of parameterized totally negative matrices, termed fg-Vandermonde-type (fg-VT) matrices, for which all singular values are computed with high-relative-accuracy (HRA).

In computer-aided geometric design, polynomial curves are usually represented using the Bernstein basis of the space $P_n(I)$ of polynomials of degree less than or equal to n, defined on a compact interval

$I=[a,b]$ on the real number line, given by the basis $\left(B_0^n,\ldots,B_n^n\right)$, where

$$B_k^n(t):=\left(\begin{array}{c}n\\k\end{array}\right)\left(\frac{t-a}{b-a}\right)^k\left(\frac{b-t}{b-a}\right)^{n-k},\ t\in[a,b],\ k=0,\ldots,n. \tag{1.1}$$

In order to accommodate more general curve modeling requirements, this basis can be generalized by considering two nonnegative continuous functions $f,g: I\to\mathbb{R}$ such that $f(t)\neq 0$ and $g(t)\neq 0$ for all $t\in(a,b)$, and such that f/g is strictly increasing on I. This leads to the system $\left(u_0^n,\ldots,u_n^n\right)$, where

$$u_k^n(t):=\left(\begin{array}{c}n\\k\end{array}\right)f^k(t)\,g^{n-k}(t),\ t\in[a,b],\ k=0,\ldots,n. \tag{1.2}$$

We refer to $\left(u_0^n,\ldots,u_n^n\right)$ given by (1.2) as the fg-Bernstein basis.

In some interpolation problems, constraints need to be imposed at endpoints, and local control is required for curve design. Accordingly, correction terms l_k and S_j are introduced to construct a new set of basis functions. To simplify the interpolation problem and enforce endpoint constraints at specified nodes, we apply a linear modification to the aforementioned basis functions to obtain the fg-Bernstein class basis as follows:

$$\left(\tilde{u}_0^n,\ldots,\tilde{u}_n^n\right),\ \tilde{u}_k^n(t):=\left(\begin{array}{c}n\\k\end{array}\right)f^k(t)\,g^{n-k}(t)-l_kS_j,\ t\in[a,b],\ k,j=0,\ldots,n, \tag{1.3}$$

where

$$S_j=g^n\left(t_j\right)+nf\left(t_j\right)g^{n-1}\left(t_j\right),$$

and l_k is the correction function.

For a given sequence of nodes $a<t_1<\cdots<t_{n+1}<b$, we define its collocation matrix as the fg-Vandermonde-type matrix (fg-VT matrix) of order $n+1$ with nodes $t_1,\ldots,t_{n+1}$:

$$A_{n+1,t_1,\cdots,t_{n+1}}:=\left(\left(\begin{array}{c}n\\k\end{array}\right)f^k\left(t_j\right)g^{n-k}\left(t_j\right)-l_kS_j\right)_{1\leq i,j\leq n+1},k=0\ldots,n. \tag{1.4}$$

We assume that $f(t_i)>0$ and $g(t_i)>0$ for all nodes t_i. The correction coefficients l_k are defined as

$$l_k=\begin{cases}1, & 0\leq k\leq 1,\\ \dfrac{\binom{n}{k}f^{k-1}(t_1)}{ng^{k-1}(t_1)}, & 2\leq k\leq n,\end{cases}$$

and the condition

$$f^n(t_{n+1})-\frac{f^{n-1}(t_1)}{ng^{n-1}(t_1)}\left[g^n(t_{n+1})+nf(t_{n+1})\,g^{n-1}(t_{n+1})\right]<0.$$

Mainar and Peña [13] established that the fg-Vandermonde matrix M, constructed from the standard fg-Bernstein basis, is strictly totally positive, allowing its singular values to be computed with HRA. In this paper, we introduce a rank-one modification to M: $A=M-lS^T$, where the vectors $l,S\in\mathbb{R}^n$

are determined by linear combinations of the columns of M to satisfy specific endpoint constraints. The key distinction lies in this structural transformation: By introducing modified basis functions via rank-one updates, we shift the matrix class from totally positive to totally negative, thereby extending the applicability of HRA algorithms to this new domain. Consequently, this construction ensures that all minors of A possess consistent signs, preserving an explicit parametrization $\mathbb{PM}(A)$.

Matrices of this type frequently arise in interpolation and curve design [14, 15], where the stability of the solution relies heavily on the accuracy of the matrix singular values. However, these fg-VT matrices are typically severely ill-conditioned. While investigating special structural features that have enabled high-precision computation for other structured matrices [16, 17], standard Singular Value Decomposition (SVD) algorithms fail to accurately compute the small singular values of this new class.

To address this challenge, we propose a novel algorithm to compute the parameterized form of the fg-VT matrix. Based on this parametrization, we prove that fg-VT matrices are totally negative and demonstrate that their singular values can be calculated with HRA. The main contributions of this paper are the introduction of this new class of matrices, the derivation of their explicit parametrization, and the development of a robust algorithm with rigorous perturbation error bounds, extending the framework of accurate computation to totally negative matrices.

The rest of the paper is organized as follows. In Section 2, we derive an explicit expression for the initial minor of the fg-VT matrix A and obtain its parameterized form. In Section 3, we develop algorithms of complexity of $O\left(n^3\right)$ to implicitly compute the parameter matrix of fg-VT matrices, and then compute all singular values of the fg-VT matrix with HRA. In Section 4, we provide perturbation theory and error analysis for the proposed algorithms. In Section 5, we present numerical experiments to verify the HRA of our method. Finally, Section 6 concludes the paper and summarizes our key findings.

2. Parametrization computation

In this section, we first derive an explicit expression for the initial minor of the fg-VT matrix A based on the initial minor of the fg-V matrix presented in [13]. Subsequently, the parameterized matrix $\mathbb{PM}(A)$ of the fg-VT matrix is obtained by applying formula (1.8) from [18].

Lemma 2.1. *Let $A \in \mathbb{R}^{(n+1)\times(n+1)}$ be an fg-VT matrix as in* (1.4). *Then,*

- *For all $i \geq j$,*

$$\begin{aligned}&\det A[i-j+1 : i \mid 1 : j]\\ &= -\frac{g(t_1)}{nf(t_1)} \prod_{m=i-j}^{i-1} \binom{n}{m} \prod_{k=1}^{j} f^{i-j}(t_k)\, g^{n-i+1}(t_k) \prod_{1\leq h<l\leq j} \left(f(t_l)\, g(t_h) - f(t_h)\, g(t_l)\right).\end{aligned} \tag{2.1}$$

- *For all $i < j$,*

$$\begin{aligned}&\det A[1 : i \mid j-i+1 : j]\\ &= \prod_{m=0}^{i-1} \binom{n}{m} \prod_{k=j-i+1}^{j} g^{n-i+1}(t_k) \prod_{j-i+1\leq h<l\leq j} \left(f(t_l)\, g(t_h) - f(t_h)\, g(t_l)\right).\end{aligned} \tag{2.2}$$

Proof. We first derive formula (2.1) by exploiting the determinant properties of A, with the specific procedure outlined in the following steps.

S_1: Starting from the second-to-last row and moving upward sequentially, multiply each row by the scalar $c_t = \frac{l_{t+1}}{l_t}$ (where $t = i-j, \ldots, i-1$) and subtract it from the next row. This step aims to eliminate all elements below the pivot in the first column.

S_2: Multiply the first column by $\left(-\frac{g^n(t_s)+nf(t_s)g^{n-1}(t_s)}{g^n(t_1)}\right)$ and add the result to each subsequent column. The purpose of this step is to simplify the form of the elements after the pivot in the first row.

S_3: Proceeding row by row starting from the first row, each row is multiplied by the scalar $c_t = \frac{l_{t+1}}{l_t}$ (for $t = i-j, \ldots, i-1$), and then added to the row immediately below it. This row operation simplifies the determinant to a single term and eases subsequent determinant evaluation.

Let

$$G_{k,l} = \begin{pmatrix} n \\ k \end{pmatrix} f^k(t_l)\, g^{n-k}(t_l); \quad \det M[j-i+1 : j|1 : i]$$

is given by [18, Eq (13)], $\det M^T[i-j+1 : i|1 : j]$ is given by [18, Eq (17)], then

$$\begin{aligned}
&\det[i-j+1 : i \mid 1 : j] \\
&= \det\begin{bmatrix}
-l_{i-j}g^n(t_1) & G_{i-j,2} - l_{i-j}S_2 & \cdots & G_{i-j,j} - l_{i-j}S_j \\
-l_{i-j+1}g^n(t_1) & G_{i-j+1,2} - l_{i-j+1}S_2 & \cdots & G_{i-j+1,j} - l_{i-j+1}S_j \\
\vdots & \vdots & \ddots & \vdots \\
-l_{i-2}g^n(t_1) & G_{i-2,2} - l_{i-2}S_2 & \cdots & G_{i-2,j} - l_{i-2}S_j \\
-l_{i-1}g^n(t_1) & G_{i-1,2} - l_{i-1}S_2 & \cdots & G_{i-1,j} - l_{i-1}S_j
\end{bmatrix} \\
&\overset{S_1}{=} \det\begin{bmatrix}
-l_{i-j}g^n(t_1) & G_{i-j,2} - l_{i-j}S_2 & \cdots & G_{i-j,j} - l_{i-j}S_j \\
0 & G_{i-j+1,2} - G_{i-j,2}c_{i-j} & \cdots & G_{i-j+1,j} - G_{i-j,j}c_{i-j} \\
\vdots & \vdots & \ddots & \vdots \\
0 & G_{i-2,2} - G_{i-3,2}c_{i-2} & \cdots & G_{i-2,j} - G_{i-3,j}c_{i-2} \\
0 & G_{i-1,2} - G_{i-2,2}c_{i-1} & \cdots & G_{i-1,j} - G_{i-2,j}c_{i-1}
\end{bmatrix} \\
&\overset{S_2}{=} \det\begin{bmatrix}
-l_{i-j}g^n(t_1) & G_{i-j,2} & \cdots & G_{i-j,j} \\
0 & G_{i-j+1,2} - G_{i-j,2}c_{i-j} & \cdots & G_{i-j+1,j} - G_{i-j,j}c_{i-j} \\
\vdots & \vdots & \ddots & \vdots \\
0 & G_{i-2,2} - G_{i-3,2}c_{i-2} & \cdots & G_{i-2,j} - G_{i-3,j}c_{i-2} \\
0 & G_{i-1,2} - G_{i-2,2}c_{i-1} & \cdots & G_{i-1,j} - G_{i-2,j}c_{i-1}
\end{bmatrix} \\
&\overset{S_3}{=} \det\begin{bmatrix}
-l_{i-j}g^n(t_1) & G_{i-j,2} & \cdots & G_{i-j,j} \\
-l_{i-j+1}g^n(t_1) & G_{i-j+1,2} & \cdots & G_{i-j+1,j} \\
\vdots & \vdots & \ddots & \vdots \\
-l_{i-2}g^n(t_1) & G_{i-2,2} & \cdots & G_{i-2,j} \\
-l_{i-1}g^n(t_1) & G_{i-1,2} & \cdots & G_{i-1,j}
\end{bmatrix} \\
&= -\frac{g(t_1)}{nf(t_1)} \det\begin{bmatrix}
G_{i-j,1} & G_{i-j,2} & \cdots & G_{i-j,j} \\
G_{i-j+1,1} & G_{i-j+1,2} & \cdots & G_{i-j+1,j} \\
\vdots & \vdots & \ddots & \vdots \\
G_{i-2,1} & G_{i-2,2} & \cdots & G_{i-2,j} \\
G_{i-1,1} & G_{i-1,2} & \cdots & G_{i-1,j}
\end{bmatrix}
\end{aligned}$$

$$= -\frac{g(t_1)}{nf(t_1)} \det M^T [i-j+1 : i \mid 1 : j].$$

Thus, (2.1) is proved.

Similarly, we now proceed to prove formula (2.2) via the following steps.

$\tilde{S}_1$: Multiply the first and second rows by (-1), then exchange the first and second rows. Using the definitions of S_j in formula (1.3), l_k in formula (1.4), and $G_{k,l}$ mentioned above, this step simplifies the first and second rows into monomial entries.

$\tilde{S}_2$: Add the first row to the second row element-wise to form a sum row. Then, for $t = 3, \ldots, i$, multiply this sum row by $(-l_{t-1})$ and add it to row t. This step converts every entry of the determinant into a monomial to ease computation.

$$\begin{aligned}
&\det[1 : i \mid j-i+1 : j] \\
&= \det\begin{bmatrix} G_{0,j-i+1} - l_0 S_{j-i+1} & G_{0,j-i+2} - l_0 S_{j-i+2} & \cdots & G_{0,j} - l_0 S_j \\ G_{1,j-i+1} - l_1 S_{j-i+1} & G_{1,j-i+2} - l_1 S_{j-i+2} & \cdots & G_{1,j} - l_1 S_j \\ G_{2,j-i+1} - l_2 S_{j-i+1} & G_{2,j-i+2} - l_2 S_{j-i+2} & \cdots & G_{2,j} - l_2 S_j \\ \vdots & \vdots & \ddots & \vdots \\ G_{i-2,j-i+1} - l_{i-2} S_{j-i+1} & G_{i-2,j-i+2} - l_{i-2} S_{j-i+2} & \cdots & G_{i-2,j} - l_{i-2} S_j \\ G_{i-1,j-i+1} - l_{i-1} S_{j-i+1} & G_{i-1,j-i+2} - l_{i-1} S_{j-i+2} & \cdots & G_{i-1,j} - l_{i-1} S_j \end{bmatrix} \\
&\overset{\tilde{S}_1}{=} \det\begin{bmatrix} l_1 S_{j-i+1} - G_{1,i-j+1} & l_1 S_{j-i+2} - G_{1,i-j+2} & \cdots & l_1 S_j - G_{1,j} \\ l_0 S_{j-i+1} - G_{0,i-j+1} & l_0 S_{j-i+2} - G_{0,i-j+2} & \cdots & l_0 S_j - G_{0,j} \\ G_{2,j-i+1} - l_2 S_{j-i+1} & G_{2,j-i+2} - l_2 S_{j-i+2} & \cdots & G_{2,j} - l_2 S_j \\ \vdots & \vdots & \ddots & \vdots \\ G_{i-2,j-i+1} - l_{i-2} S_{j-i+1} & G_{i-2,j-i+2} - l_{i-2} S_{j-i+2} & \cdots & G_{i-2,j} - l_{i-2} S_j \\ G_{i-1,j-i+1} - l_{i-1} S_{j-i+1} & G_{i-1,j-i+2} - l_{i-1} S_{j-i+2} & \cdots & G_{i-1,j} - l_{i-1} S_j \end{bmatrix} \\
&= \det\begin{bmatrix} G_{0,j-i+1} & G_{0,j-i+2} & \cdots & G_{0,j} \\ G_{1,j-i+1} & G_{1,j-i+2} & \cdots & G_{1,j} \\ G_{2,j-i+1} - l_2 S_{j-i+1} & G_{2,j-i+2} - l_2 S_{j-i+2} & \cdots & G_{2,j} - l_2 S_j \\ \vdots & \vdots & \ddots & \vdots \\ G_{i-2,j-i+1} - l_{i-2} S_{j-i+1} & G_{i-2,j-i+2} - l_{i-2} S_{j-i+2} & \cdots & G_{i-2,j} - l_{i-2} S_j \\ G_{i-1,j-i+1} - l_{i-1} S_{j-i+1} & G_{i-1,j-i+2} - l_{i-1} S_{j-i+2} & \cdots & G_{i-1,j} - l_{i-1} S_j \end{bmatrix} \\
&\overset{\tilde{S}_2}{=} \det\begin{bmatrix} G_{0,j-i+1} & G_{0,j-i+2} & \cdots & G_{0,j} \\ G_{1,j-i+1} & G_{1,j-i+2} & \cdots & G_{1,j} \\ G_{2,j-i+1} & G_{2,j-i+2} & \cdots & G_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ G_{i-2,j-i+1} & G_{i-2,j-i+2} & \cdots & G_{i-2,j} \\ G_{i-1,j-i+1} & G_{i-1,j-i+2} & \cdots & G_{i-1,j} \end{bmatrix} \\
&= -\det M[j-i+1 : j \mid 1 : i].
\end{aligned}$$

Thus, (2.2) is proved. □

The following result presents the parametrization formulas for fg-VT matrices, which are crucial for constructing the algorithm to compute $\mathbb{PM}(A)$.

Theorem 2.1. *Let $A \in \mathbb{R}^{(n+1)\times(n+1)}$ be an fg-VT matrix as in* (1.4)*. Then, A is a totally negative matrix, and its parametrization matrix $\mathbb{PM}(A) \in \mathbb{R}^{(n+1)\times(n+1)}$ is given as follows:*

- *For all $1 \le i \le n+1$,*

$$\mathbb{PM}(A)_{ii} = \begin{cases} d_1 = g^n(t_1), \\ d_i, \quad \forall\, 2 \le i \le n+1, \end{cases} \tag{2.3}$$

  where

$$d_i = \binom{n}{i-1} g^{n-i+1}(t_i) \prod_{k=1}^{i-1} \frac{f(t_i)g(t_k) - f(t_k)g(t_i)}{g(t_k)}.$$

- *For all $1 \le i < j \le n+1$,*

$$\mathbb{PM}(A)_{ij} = \begin{cases} \alpha_{1j} = \dfrac{g^n(t_j)}{g^n(t_{j-1})}, \quad \forall\, 2 \le j \le n+1, \\ \alpha_{ij} = \dfrac{g^{n-i+1}(t_j)\, g^n(t_{j-2})\, \prod_{k=j-i+1}^{j-1}\left(f(t_j)g(t_k) - f(t_k)g(t_j)\right)}{g^{n-i+2}(t_{j-1})\, \prod_{k=j-i}^{j-2}\left(f(t_{j-1})g(t_k) - f(t_k)g(t_{j-1})\right)}, \quad \forall\, 2 \le i < j \le n+1, \end{cases} \tag{2.4}$$

- *For all $1 \le j < i \le n+1$,*

$$\mathbb{PM}(A)_{ij} = \begin{cases} \beta_{ij} = \dfrac{\binom{n}{i-1} f(t_j)}{\binom{n}{i-2} g(t_j)}, \quad (i,j) \neq (3,1), \\ \beta_{31}, \end{cases} \tag{2.5}$$

  where

$$\beta_{31} = -\frac{a_{n+1,n+1}}{d_1} \prod_{k=2}^{n+1} \frac{g^n(t_k)}{g^n(t_{k-1})} \prod_{k=4}^{n+1} \frac{\binom{n}{k-1} f(t_1)}{\binom{n}{k-2} g(t_1)}.$$

Proof. The parametrization matrix $\mathbb{PM}(A)$ can be computed as follows:

- For the case $1 \le i = j \le n+1$, $d_1 = -\det A\,[2|1] = g^n(t_1) > 0$, and when $2 \le i \le n+1$, we have

$$\begin{aligned} d_i &= \frac{\det A\,[1:i]}{\det A\,[1:i-1]} \\ &= \frac{\prod_{k=0}^{i-1} \binom{n}{k} \prod_{k=1}^{i} g^{n-i+1}(t_k) \prod_{1\le h<l\le i} \left(f(t_l)\, g(t_h) - f(t_h)\, g(t_l)\right)}{\prod_{k=0}^{i-2} \binom{n}{k} \prod_{k=1}^{i-1} g^{n-i+2}(t_k) \prod_{1\le h<l\le i-1} \left(f(t_l)\, g(t_h) - f(t_h)\, g(t_l)\right)} \\ &= \binom{n}{i-1} g^{n-i+1}(t_i) \prod_{k=1}^{i-1} \frac{\left(f(t_i)\, g(t_k) - f(t_k)\, g(t_i)\right)}{g(t_k)} > 0. \end{aligned}$$

- For the case $1 \le i < j \le n+1$, when $i = 1$ and $2 \le j \le n+1$, we have

$$\alpha_{1j} = \frac{\det A[2|j]}{\det A[2|j-1]} = \frac{a_{2j}}{a_{2,j-1}} = \frac{g^n(t_j)}{g^n(t_{j-1})} > 0;$$

when $2 \le i < j \le n+1$, we have

$$\begin{aligned}\alpha_{ij} &= \frac{\det A\,[1:i|j-i+1:j]}{\det A\,[1:i-1|j-i+1:j-1]} \cdot \frac{\det A\,[1:i-1|j-i:j-2]}{\det A\,[1:i|j-i:j-1]} \\ &= \frac{g^{n-i+1}\left(t_j\right) g\left(t_{j-1}\right) \prod_{k=j-i+1}^{j-1} \left(f\left(t_j\right) g\left(t_k\right) - f\left(t_k\right) g\left(t_j\right)\right)}{g^{n-i+2}\left(t_{j-1}\right) \prod_{k=j-i}^{j-2} \left(f\left(t_{j-1}\right) g\left(t_k\right) - f\left(t_k\right) g\left(t_{j-1}\right)\right)} > 0.\end{aligned}$$

- For the case $1 \le j < i \le n+1$, when $i \ne 3$, $j \ne 1$, we have

$$\begin{aligned}\beta_{ij} &= \frac{\det A\,[i-j+1:i \mid 1:j]}{\det A\,[i-j+1:i-1 \mid 1:j-1]} \cdot \frac{\det A\,[i-j:i-2 \mid 1:j-1]}{\det A\,[i-j:i-1 \mid 1:j]} \\ &= \frac{\binom{n}{i-1} f(t_j)}{\binom{n}{i-2} g(t_j)} > 0;\end{aligned}$$

when $i = 3$, $j = 1$, we have

$$\begin{aligned}\beta_{31} &= \frac{-A\,[n+1, n+1]}{d_1 \prod_{k=2}^{n+1} \alpha_{1k} \prod_{k=4}^{n+1} \beta_{k1}} \\ &= \frac{-A\,[n+1, n+1]}{d_1} \cdot \prod_{k=2}^{n+1} \frac{g^n(t_k)}{g^n(t_{k-1})} \prod_{k=4}^{n+1} \frac{\binom{n}{k-1} f(t_1)}{\binom{n}{k-2} g(t_1)} > 0.\end{aligned}$$

Remark 1. In the parameterization of negative matrices in [11], a key observation is that the matrix entry a_{nn} is strictly decreasing with respect to β_{31}. Given the $n^2 - 1$ nontrivial elements other than β_{31} in the bidiagonal factorization, there exists a value θ, determined by these $n^2 - 1$ elements, such that $a_{nn} = 0$ if $\beta_{31} = \theta$. Consequently, to ensure $a_{nn} < 0, \beta_{31}$ must be no less than θ. We set $\beta_{31} = \theta + \delta$ with $\delta > 0$. According to [11, Theorem 1], we use the free parameter δ instead of the entry β_{31}, and set

$$\delta = \frac{-A\,[n, n]}{d_1 \prod_{k=2}^{n+1} \alpha_{1k} \prod_{k=4}^{n+1} \beta_{k1}}.$$

Since we have constructed $\mathbb{PM}(A)$ with all positive parameters, it follows from [11, Theorem 1] that A is a totally negative matrix. Thus, Theorem 2.1 is proved. □

3. The high relative accuracy algorithm

In this section, we propose an algorithm to accurately compute the parametrization matrix $\mathbb{PM}(A)$ of the fg-VT matrix A. Based on the accurate parametrization matrix $\mathbb{PM}(A)$, all singular values of fg-VT matrices are computed with HRA using Algorithm 4 in [11].

Algorithm 1 Computation of the parametrization matrix $\mathbb{PM}(A)$.

Input: Nodes$\{t_i\}_{i=1}^{n+1}$ and the function $f(t), g(t)$ satisfying (1.4).
Output: The parametrization matrix $\mathbb{PM}(A) \in \mathbb{R}^{(n+1)\times(n+1)}$.

$\mathbb{PM}(A) = \text{ones}(n+1, n+1)$
  % Computation of the parameters $\mathbb{PM}(A)_{ii}$ $(1 \le i \le n+1)$ **by** (2.3)
$\mathbb{PM}(A)_{11} = g^n(t_1)$;
for $i = 2 : n+1$ **do**
  $P_1 = 1$;
  for $k = 1 : i-1$ **do**
    $P_1 = P_1 \cdot \frac{f(t_i)g(t_k)-f(t_k)g(t_i)}{g(t_k)}$;
  end for
  $\mathbb{PM}(A)_{ii} = \binom{n}{i-1} g^{n-i+1}(t_i) \cdot P_1$;
end for
  % Computation of the parameters $\mathbb{PM}(A)_{ij}$ $(1 \le i < j \le n+1)$ **by** (2.4)
for $j = 2 : n+1$ **do**
  $\mathbb{PM}(A)_{1j} = \frac{g^n(t_j)}{g^n(t_{j-1})}$;
end for
for $i = 2 : n$ **do**
  for $j = i+1 : n+1$ **do**
    $P_2 = 1$;
    for $k = j-i+1 : j-1$ **do**
      $P_2 = \left(f(t_j)\, g(t_k) - f(t_k)\, g(t_j)\right) \cdot P_2$;
    end for
    $P_3 = 1$
    for $k = j-i : j-2$ **do**
      $P_3 = \left(f(t_{j-1})\, g(t_k) - f(t_k)\, g(t_{j-1})\right) \cdot P_3$;
    end for
    $\mathbb{PM}(A)_{ij} = \frac{g^{n-i+1}(t_j)g^n(t_{j-2})P_2}{g^{n-i+2}(t_{j-1})P_3}$;
  end for
end for
  % Computation of the parameters $\mathbb{PM}(A)_{ij}$ $(1 \le j < i \le n+1)$ **by** (2.5)
for $j = 1 : n$ **do**
  for $i = j+1 : n+1$ **do**
    $\mathbb{PM}(A)_{ij} = \binom{n}{i-1} f(t_j) / \binom{n}{i-2} g(t_j)$;
  end for
  $M_1 = 1$;
  for $k = 2 : n+1$ **do**
    $M_1 = g^n(t_k) / g^n(t_{k-1}) \cdot M_1$;
  end for
  $M_2 = 1$;
  for $k = 4 : n+1$ **do**
    $M_2 = \binom{n}{k-1} f(t_1) / \binom{n}{k-2} g(t_1) \cdot M_2$;
  end for
  $\mathbb{PM}(A)_{31} = -\frac{A[n+1,n+1]}{\mathbb{PM}(A)_{11}} \cdot M_1 \cdot M_2$;
end for

Example 3.1. *Let $A \in \mathbb{R}^{10\times10}$ be an fg-VT matrix as in* (1.4)*, where $f(t_i) = t_i$, $g(t_i) = 1$. Let the nodes be $t_1 = 1$, $t_i = t_1 + (i-1)\cdot 10^{-2}$ for $i = 2, \ldots, 9$, and $t_{10} = 14/10$. For testing the high relative accuracy of Algorithms 1, we compute all the entries $\mathbb{PM}(A)_{ij}$ by Mathematica with a 64-decimal digits arithmetic and rounded back to 16 decimal digits, and they are taken as "exact" in our experiment. The absolute values and relative errors of all the computed $\hat{\mathbb{PM}}(A)_{ij}$ $(1 \le i, j \le 10)$ by Algorithm 1 are plotted in Figure 1. As observed, all the entries of $\mathbb{PM}(A)$ are computed to high relative accuracy with a pleasant* *relative error as*

$$\textit{Algorithm 1:} \quad \max_{1\le i,j\le 10} \frac{|\hat{\mathbb{PM}}(A)_{ij} - \mathbb{PM}(A)_{ij}|}{|\mathbb{PM}(A)_{ij}|} = 1.0172e-14.$$

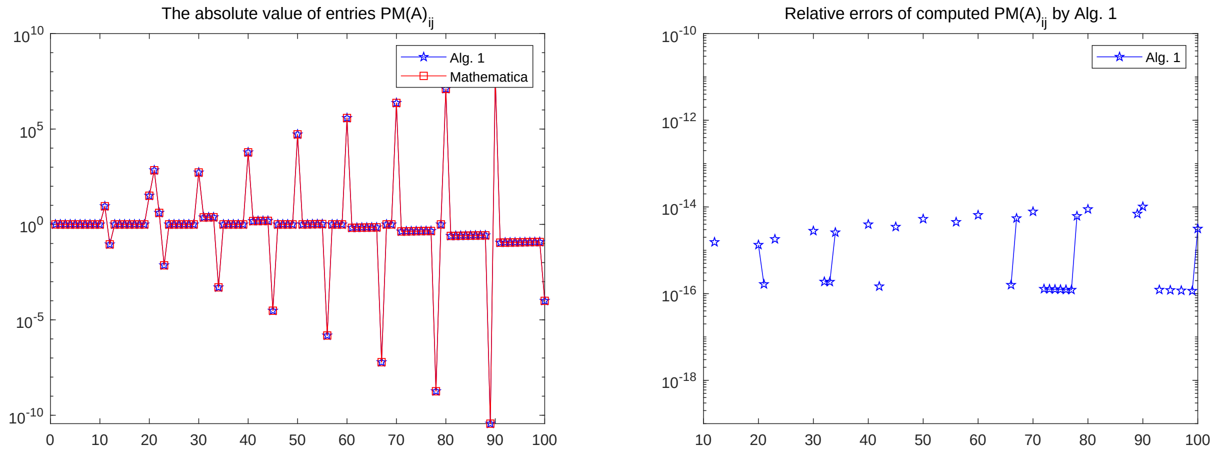

Figure 1. The absolute values and relative errors for the computed $\hat{\mathbb{PM}}(A)_{ij}$.

Remark 3.2. *The computations in* (2.3)–(2.5) *only involve additions of like-signed numbers, multiplications, divisions, and subtractions of the initial inputs* x_i, $f(t_i)$, *and* $g(t_i)$. *Thus, we conclude that the parametrization matrix* $\mathbb{PM}(A) \in \mathbb{R}^{(n+1)\times(n+1)}$ *of an fg-VT matrix A is computed with HRA by Algorithm 1.*

By combining Algorithm 1 proposed in this paper with [11, Algorithm 4], we present Algorithm 2 for computing the singular values of fg-VT matrices as follows.

Algorithm 2 The algorithm computes all the singular values of an fg-VT matrix A satisfying (1.4).

Input: Nodes $\{t_i\}_{i=1}^{n+1}$ and the functions $f(t)$, $g(t)$ satisfying (1.4).
Output: All the singular values of the fg-VT matrix A.
Step 1: The parametrization matrix $\mathbb{PM}(A) \in \mathbb{R}^{(n+1)\times(n+1)}$ is computed by Algorithm 1;
Step 2: All the singular values of A are computed by applying [11, Algorithm 4] to $\mathbb{PM}(A)$ obtained in Step 1.

4. Perturbation theory and error analysis

In this section, we first provide a relative perturbation result to show that the parametrization matrix of fg-VT matrices are relatively robust with respect to their singular values. Then, an error analysis is provided to illustrate the HRA of Algorithm 2.

Theorem 4.1. *Let* $A \in \mathbb{R}^{(n+1)\times(n+1)}$ *be an fg-VT matrix satisfying the conditions in Section 1, with positive parameterization* $\mathbb{PM}(A) = \left[p_{ij}\right]\left(p_{ij} > 0\right)$. *Suppose that* $\widehat{\mathbb{PM}}(A) = \left[\hat{p}_{ij}\right]$ *is derived from* $\mathbb{PM}(A)$ *by replacing each entry* p_{ij} *in* $\mathbb{PM}(A)$ *with* $\hat{p}_{ij} = p_{ij}\left(1+\delta_{ij}\right)$, *with* $\left(\delta_{ij}\right) \le \delta \ll 1$. *Let*

$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{n+1}$ be the singular values of A and $\hat{\sigma}_1 \geq \hat{\sigma}_2 \geq \cdots \geq \hat{\sigma}_{n+1}$ be the singular values of $\hat{A}$ corresponding to $\mathbb{PM}(A)$. Then, for all $i = 1, 2, ..., n+1$

$$|\hat{\sigma}_i - \sigma_i| \leq \frac{2(n+1)^2 \delta}{1 - 2(n+1)^2 \delta} \sigma_i.$$

Proof. Let $A^{(k)}$ denote the k-th compound matrix of A, then the singular value of $A^{(k)}$ is $\Pi_k = \prod_{i=1}^{k} \sigma_i$, and for $\hat{A}^{(k)}$, the singular value of $\hat{A}^{(k)}$ is $\widehat{\Pi}_k = \prod_{i=1}^{k} \hat{\sigma}_i$ by [1, Lemma 7.1]. For positive matrix $A^{(k)}$, $\widehat{\Pi}_k = \Pi_k (1 + \delta_k) \sigma_k$ with $|\delta_k| \leq \delta$ by [1, Theorem 7.2]. Since $\sigma_i = \frac{\Pi_i}{\Pi_{i-1}}$, $\widehat{\sigma}_i = \frac{\widehat{\Pi}_i}{\widehat{\Pi}_{i-1}}$, $\widehat{\Pi}_i = \Pi_i (1 + \delta_i)$, and $\widehat{\Pi}_{i-1} = \Pi_{i-1} (1 + \delta_{i-1})$, then we obtain $\hat{\sigma}_i = \sigma_i \cdot \frac{1+\delta_i}{1+\delta_{i-1}}$. By triangle inequality and $|\delta_i|$, $|\delta_{i-1}| \leq \delta \ll 1$, we have

$$|\hat{\sigma}_i - \sigma_i| = \sigma_i \left| \frac{1+\delta_i}{1+\delta_{i-1}} - 1 \right| = \sigma_i \left| \frac{\delta_i - \delta_{i-1}}{1+\delta_{i-1}} \right| \leq \sigma_i \cdot \frac{2\delta}{1-2\delta}, \quad i = 1, \cdots, n+1.$$

Scaling this bound by the $(n+1)^2$ nontrivial entries of $\mathbb{PM}(A)$ yields the final result. □

Theorem 4.2. *Let σ_i and $\hat{\sigma}_i$ denote the exact singular values of the nonsingular fg-VT matrix $A \in \mathbb{R}^{(n+1)\times(n+1)}$ and the approximate singular values of A obtained by Algorithm 2, respectively. Let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{n+1}$, $\hat{\sigma}_1 \geq \hat{\sigma}_2 \geq \cdots \geq \hat{\sigma}_{n+1}$, in floating-point arithmetic with machine precision ϵ. Then*

$$|\hat{\sigma}_i - \sigma_i| \leq \frac{O\left(n^3\right)\epsilon}{1 - O(n^3)\epsilon} \sigma_i, \quad i = 1, \cdots, n+1.$$

Proof. In step 1 of Algorithm 2, the total number of floating-point operations performed on Algorithm 1 is at most $\frac{9}{2}n^3 + \frac{15}{2}n^2 + O(n)$. Employing [1, Theorem 4.1], we determine that the relative perturbations of the singular values of A, arising from rounding errors in the computation of $\mathbb{PM}(A)$, are bounded by

$$\frac{\left(9n^3 + 15n^2 + O(n)\right)\epsilon}{1 - (9n^3 + 15n^2 + O(n))\epsilon}.$$

In step 2 of Algorithm 2, according to [11, Algorithm 4], the relative error in the singular values of A is bounded by $O(n^3)$. Thus, for all the descending-ordered singular values σ_i and $\hat{\sigma}_i$, it holds that

$$|\hat{\sigma}_i - \sigma_i| \leq \frac{O\left(n^3\right)\epsilon}{1 - O(n^3)\epsilon} \sigma_i, \quad i = 1, \cdots, n+1.$$

This completes the proof. □

5. Experimental results

Highly ill-conditioned collocation matrices are common in polynomial interpolation and approximation. When interpolation nodes become clustered, Vandermonde-type matrices exhibit exponentially growing condition numbers. In this section, several numerical examples are provided to illustrate the HRA of Algorithm 2 for computing singular values for fg-VT matrices. All experiments are conducted using MATLAB (R2015b). Algorithm 2 is compared with the results obtained using MATLAB's SVD function. For reference, we compute the exact singular values σ_i using Mathematica and measure the relative error using $\frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i}$.

Example 5.1. *Let* $A \in \mathbb{R}^{(n+1)\times(n+1)}$ *be an fg-VT matrix as in* (1.4)*, where* $f(t_i) = t_i$*,* $g(t_i) = 1$*. Let the nodes be* $t_1 = 1$*,* $t_i = t_1 + (i-1) \cdot 10^{-5}$ *for* $i = 2, \ldots, n$*, and* $t_{n+1} = 10004/10000$*. Set* $n = 40$*. The singular values of A range from 3.7834e-165 to 4.4747e+10. Then, the spectral condition number is* $\kappa_1(A) = 1.1827e + 175$ *as computed by Mathematica. We plot the relative errors and absolute values of all computed singular values in Figure 1, where the maximum relative errors are*

$$\textbf{\textit{Algorithm 2}} : \max_i \left| \frac{\hat{\sigma}_i - \sigma_i}{\sigma_i} \right| = 8.8697e{-}12,$$

$$\textbf{\textit{svd}} : \max_i \left| \frac{\hat{\sigma}_i - \sigma_i}{\sigma_i} \right| = 1.6174e{+}147.$$

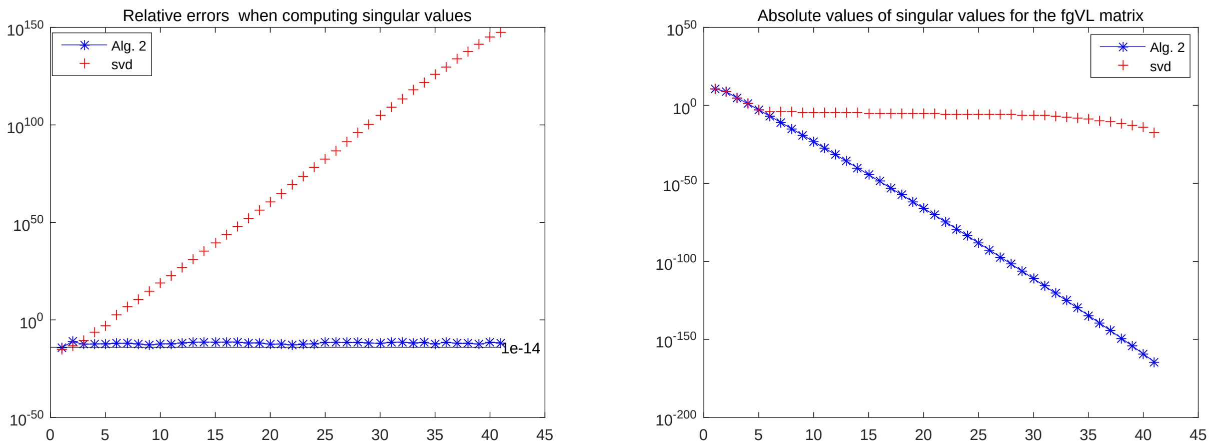

Figure 2. The relative errors and absolute values computed by Algorithm 2 and SVD in Example 5.1.

Example 5.2. *Let* $A \in \mathbb{R}^{(n+1)\times(n+1)}$ *be an fg-VT matrix as in* (1.4)*, where* $f(t_i) = t_i^2$*,* $g(t_i) = t$*. Let the nodes be* $t_1 = 12/10$*,* $t_i = t_1 + (i-1) \cdot 10^{-5}$ *for* $i = 2, \ldots, n$*, and* $t_{n+1} = 12006/10000$*. Set* $n = 40$*. The singular values of A range from 2.2603e-164 to 2.4517e+15. Then, the spectral condition number is* $\kappa_2(A) = 1.0847e + 179$ *as computed by Mathematica. We plot the relative errors and absolute values of all computed singular values in Figure 3, where the maximum relative errors are*

$$\textbf{\textit{Algorithm 2}} : \max_i \frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} = 2.1501e - 12,$$

$$\textbf{\textit{svd}} : \max_i \frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} = 2.7677e + 153.$$

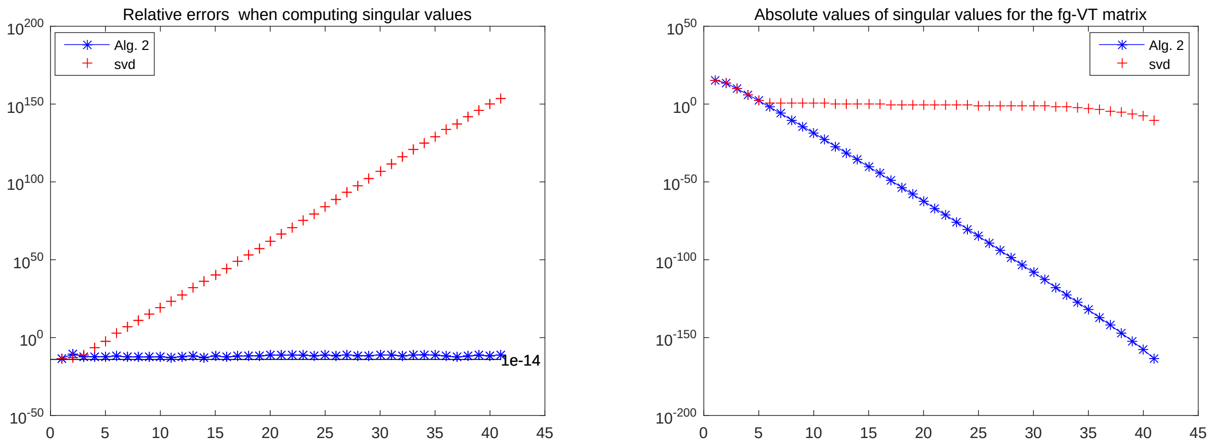

Figure 3. The relative errors and absolute values computed by Algorithm 2 and SVD in Example 5.2.

Example 5.3. *Let $A \in \mathbb{R}^{(n+1)\times(n+1)}$ be an fg-VT matrix as in* (1.4)*, where $f(t_i) = t_i^2$, $g(t_i) = 1 - t_i^2$. Let the nodes be $t_1 = 1/2$, $t_i = t_1 + (i-1)\cdot 10^{-5}$ for $i = 2, \ldots, n$, and $t_{n+1} = 5005/10000$. Set $n = 40$. Then, the spectral condition number is $\kappa_3(A) = 3.7077e + 157$ as computed by Mathematica. The relative errors of the computed singular values by Algorithm 2 and command SVD are reported in Table 1, where*

$$\textbf{\textit{Algorithm 2}} : \max_i \frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} = 5.6534e - 12,$$

$$\textbf{\textit{svd}} : \max_i \frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} = 3.8576e + 118.$$

The matrix A is badly ill-conditioned in the conventional sense. As shown in Table 1 and Figures 1 and 2, the SVD command fails to produce any correct significant digits for some extremely small singular values, and the standard SVD method suffers from a severe loss of relative accuracy for large values of i, while Algorithm 2 achieves nearly full relative accuracy for all singular values, regardless of their magnitude.

According to the error analysis in Theorem 4.2, the complexity bound for Algorithm 2 is given by $\frac{9}{2}n^3 + \frac{15}{2}n^2 + O(n)$, with machine precision $\epsilon = 2.2e - 16$. Substituting $n = 40$ into the complexity bound provided in the paper yields a theoretical upper bound of approximately $6.6573e - 11$. In our experiments, the maximum relative error of the singular values computed by Algorithm 2 is far smaller than this theoretical upper bound, which further validates the effectiveness of the algorithm and its consistency with the theoretical error analysis.

Table 1. The relative errors of computed singular values in Example 5.3.

i	σ_i by Mathematica	$\frac{\lvert\hat{\sigma}_i - \sigma_i\rvert}{\lvert\sigma_i\rvert}$ by Algorithm 2	$\frac{\lvert\hat{\sigma}_i - \sigma_i\rvert}{\lvert\sigma_i\rvert}$ by svd
1	1.335377581753002e-01	5.1962e-15	2.4942e-15
2	3.037036055291295e-03	5.6534e-12	2.6275e-14
3	2.560498597344490e-06	1.0139e-13	3.6910e-12
4	2.329048851981754e-09	4.3063e-13	1.9565e-09
5	1.562934973887177e-12	3.2535e-13	7.2774e-07
6	8.654994729349095e-16	1.4857e-13	3.5948e-03
7	4.133111835610522e-19	1.1498e-13	2.6281e+02
8	1.616755041860834e-22	2.6160e-13	6.2789e+05
9	5.442383261286435e-26	7.1146e-13	1.2112e+09
10	1.666165588853171e-29	3.5996e-13	3.0522e+12
11	4.756959944648180e-33	5.1033e-13	9.5268e+15
12	1.279710167013937e-36	3.3510e-13	2.6979e+19
13	3.260424411616585e-40	2.1827e-13	7.6924e+22
14	7.891470215011068e-44	6.3465e-14	3.1313e+26
15	1.818501587515975e-47	1.0906e-12	8.9180e+29
16	3.997552716642408e-51	2.1242e-12	3.1438e+33
17	8.406382449136807e-55	1.7472e-12	1.3035e+37
18	1.700003417007562e-58	1.1621e-12	2.5486e+40
19	3.332856653165891e-62	1.0277e-12	9.9277e+43
20	6.351736517939367e-66	1.7865e-12	2.8841e+47
21	1.153610695808425e-69	1.9022e-12	5.3565e+50
22	1.920652838493014e-73	1.0891e-12	9.2775e+53
23	2.853560847577732e-77	7.8652e-13	1.8101e+57
24	3.757330811655736e-81	1.6684e-12	3.4661e+60
25	4.380120242953018e-85	8.8423e-13	6.9118e+63
26	4.509955171002306e-89	1.9354e-12	1.0984e+67
27	4.084774436922697e-93	1.4293e-12	2.5545e+70
28	3.237314297592493e-97	1.7822e-14	4.4957e+73
29	2.230683271136465e-101	3.1715e-14	1.2021e+77
30	1.326067856539710e-105	6.1050e-13	3.5533e+80
31	6.737470886590664e-110	1.6248e-12	5.2085e+83
32	2.892416140209919e-114	1.5410e-12	1.2610e+87
33	1.034466299012726e-118	6.6563e-13	2.0200e+90
34	3.027963482510394e-123	2.1292e-12	9.2353e+93
35	7.090223235856113e-128	5.7454e-13	1.9344e+97
36	1.288680716442370e-132	7.1186e-13	6.8954e+100
37	1.744088661057028e-137	1.6940e-12	1.0655e+104
38	1.654365356429124e-142	3.4574e-12	4.3612e+107
39	9.990080485197631e-148	2.1593e-12	2.8937e+111
40	3.220439222219721e-153	5.2864e-12	7.8359e+114
41	3.601721014229221e-159	4.8969e-12	3.8576e+118

Remark 5.4. *In double-precision arithmetic, machine epsilon is* $\varepsilon \approx 2.2e-16$*. Theorem 4.2 shows that even when the matrix A is extremely ill-conditioned (e.g., with a spectral condition number as large as* 10^{175}*), Algorithm 2 achieves a relative error of order* 10^{-12} *in the computed singular values, satisfying the definition of HRA:*

$$\frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} \leq \frac{O\left(n^3\right)\epsilon}{1 - O\left(n^3\right)\epsilon}.$$

This is far smaller than the error produced by the MATLAB's command SVD, thereby establishing the HRA of the proposed method.

6. Conclusions

This paper introduces a class of fg-Vandermonde-type matrices, which belongs to the class of negative matrices first proposed in [11]. The parametrization matrix $\mathbb{PM}(A)$ is accurately computed with HRA by Algorithm 1. Consequently, all singular values of such matrices are computed with HRA by Algorithm 2. Additionally, perturbation theory and error analysis are developed to further characterize the accuracy of the proposed approach. Numerical results confirm the HRA of the proposed algorithms.

Author contributions

Jiaxian Huang: wrote the initial draft of the paper; Jiaqi Shu: provided the research ideas of the paper and further improved the paper quality. Both authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The authors would like to express their gratitude to the editor and the anonymous referees for the carefully reading of the original manuscript and giving many valuable suggestions for improvement. This work was supported by National Natural Science Foundation of China (Grant 12271153).

Conflict of interest

The authors declare that they have no conflicts of interest.

References

1. P. Koev, Accurate eigenvalues and exact zero Jordan blocks of totally nonnegative matrices, *Numer. Math.*, **141** (2019), 693–713. https://doi.org/10.1007/s00211-019-01022-0

2. X. X. Ma, Y. Q. Xiao, Z. Yang, Computing eigenvalues for products of two classes of sign regular matrices to high relative accuracy, *Linear Algebra Appl.*, **707** (2025), 80–106. https://doi.org/10.1016/j.laa.2024.11.006

3. A. Marco, J. J. Martínez, J. M. Peña, Accurate bidiagonal decomposition of totally positive Cauchy–Vandermonde matrices and applications, *Linear Algebra Appl.*, **517** (2017), 63–84. https://doi.org/10.1016/j.laa.2016.12.003

4. R. Huang, D. Chu, Relative perturbation analysis for eigenvalues and singular values of totally nonpositive matrices, *SIAM J. Matrix Anal. Appl.*, **36** (2015), 476–495. https://doi.org/10.1137/140995702

5. R. Huang, D. Chu, Computing singular value decompositions of parameterized matrices with total nonpositivity to high relative accuracy, *J. Sci. Comput.*, **71** (2017), 682–711. https://doi.org/10.1007/s10915-016-0315-5

6. R. Huang, A qd-type method for computing generalized singular values of BF matrix pairs with sign regularity to high relative accuracy, *Math. Comp.*, **89** (2020), 229–252. https://doi.org/10.1090/mcom/3444

7. P. Koev, F. Dopico, Accurate eigenvalues of certain sign regular matrices, *Linear Algebra Appl.*, **424** (2007), 435–447. https://doi.org/10.1016/j.laa.2007.02.012

8. Z. Yang, T. Chen, The accurate and efficient solutions of linear systems for generalized sign regular matrices with certain signature, *J. Comput. Appl. Math.*, **431** (2023), 115280. https://doi.org/10.1016/j.cam.2023.115280

9. R. Huang, Accurate solutions of weighted least squares problems associated with rank-structured matrices, *Appl. Numer. Math.*, **146** (2019), 416–435. https://doi.org/10.1016/j.apnum.2019.08.001

10. Z. Yang, R. Huang, W. Zhu, Accurate computations for eigenvalues of products of Cauchy-polynomial-Vandermonde matrices, *Numer. Algor.*, **85** (2020), 329–351. https://doi.org/10.1007/s11075-019-00816-5

11. R. Huang, J. Xue, Accurate singular values of a class of parameterized negative matrices, *Adv. Comput. Math.*, **47** (2021), 73. https://doi.org/10.1007/s10444-021-09898-z

12. A. Marco, J. J. Martínez, Accurate computations with totally positive Bernstein-Vandermonde matrices, *Electron. J. Linear Algebra*, **26** (2013), 357–380. https://doi.org/10.13001/1081-3810.1658

13. E. Mainar, J. M. Peña, Accurate computations with collocation matrices of a general class of bases, *Numer. Linear Algebra Appl.*, **25** (2018), e2184. https://doi.org/10.1002/nla.2184

14. V. Y. Pan, *Structured matrices and polynomials: unified superfast algorithms*, Birkhäuser Boston, MA, 2012. https://doi.org/10.1007/978-1-4612-0129-8

15. G. M. Phillips, *Interpolation and approximation by polynomials*, Springer New York, NY, 2003. https://doi.org/10.1007/b97417

16. M. Dailey, F. M. Dopico, Q. Ye, Relative perturbation theory for diagonally dominant matrices, *SIAM J. Matrix Anal. Appl.*, **35** (2014), 1303–1328. https://doi.org/10.1137/130943613

17. M. S. Arif, K. Abodayeh, Y. Nawaz, Numerical schemes for fractional energy balance model of climate change with diffusion effects, *Emerging Sci. J.*, **7** (2023), 808–820. https://doi.org/10.28991/ESJ-2023-07-03-011

18. Z. Yang, S. Liu, C. Cao, Accurate computations of singular values and linear systems for Polynomial-Vandermonde-type matrices, *Calcolo*, **61** (2024), 62. https://doi.org/10.1007/s10092-024-00615-y