An iterative method for solving the stable subspace of a matrix pencil and its application.

Matthew M. Lin [*] Chun-Yueh Chiang [†]

November 21, 2016

Abstract

This work is to propose an iterative method of choice to compute a stable subspace of a regular matrix pencil. This approach is to define a sequence of matrix pencils via particular left null spaces. We show that this iteration preserves a discrete-type flow depending only on the initial matrix pencil. Via this recursion relationship, we propose an accelerated iterative method to compute the stable subspace and use it to provide a theoretical result to solve the principal square root of a given matrix, both nonsingular and singular. We show that this method can not only find out the matrix square root, but also construct an iterative approach which converges to the square root with any desired order.

Keywords: Stable subspace, Sherman Morrison Woodbury formula, Matrix square root, Accelerated iterative method, Q-superlinear convergence

1 Introduction

Throughout this paper we shall use the following notation to facilitate our discussions. $\lambda(A)$ and $\lambda(A,B)$ denote the sets of eigenvalues of the matrix A and the matrix pencil $A-\lambda B$, respectively, and let $\rho(A)$ be the spectral radius of the square matrix A. $\mathbb{C}^+$ and $\mathbb{C}^-$ represent the open right and left half complex planes.

Given a regular $n\times n$ matrix pencil $A-\lambda B$ (i.e., $\det(A-\lambda B)$ is not identically zero for all λ) and an integer $m\leq n$, we want to find in this work a full rank matrix $U\in\mathbb{C}^{n\times m}$ such that

$$AU = BU\Lambda, \tag{1}$$

where $\Lambda\in\mathbb{C}^{m\times m}$ and $\rho(\Lambda)<1$.

[*]Department of Mathematics, National Cheng Kung University, Tainan 701, Taiwan. The first author was supported by the Ministry of Science and Technology of Taiwan under grant 104-2115-M-006-017-MY3. (mhlin@mail.ncku.edu.tw)

[†]Corresponding Author, Center for General Education, National Formosa University, Huwei 632, Taiwan. The second author was supported by the Ministry of Science and Technology of Taiwan under grant 105-2115-M-150-001. (chiang@nfu.edu.tw)

Note that the column space $\mathcal{U} = \operatorname{Span}\{U\}$ is called the stable deflating subspace of $A - \lambda B$. Specially, $\mathcal{U}$ is called the stable invariant space if B is the identity matrix. Over the past few decades, considerable attention has been paid to study the property of the invariant and deflating subspace [7]. In application, one can obtain the solutions of algebraic Riccati-type matrix equations by computing its corresponding stable deflating subspaces or stable invariant subspaces, e.g., [18, 5]. Particularly, this problem is related to the so-called *generalized spectral divide and conquer* (SDC) problem [2, 5], which is to find a pair of left and right deflating subspaces $\mathcal{L}$ and $\mathcal{R}$ such that

$$A\mathcal{R} \subset \mathcal{L}, \quad B\mathcal{R} \subset \mathcal{L},$$

corresponding to eigenvalues of the pair $A - \lambda B$ in a specified region $\mathcal{D} \subset \mathbb{C}$. That is, find two nonsingular partitioned matrices $U_L = \begin{bmatrix} U_{L_1}, U_{L_2} \end{bmatrix}$ and $U_R = \begin{bmatrix} U_{R_1}, U_{R_2} \end{bmatrix}$ with $\mathcal{L} = \operatorname{span}(U_{L_1})$ and $\mathcal{R} = \operatorname{span}(U_{R_1})$ so that

$$AU_R = U_L \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad BU_R = U_L \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix},$$

and the eigenvalues of $A_{11} - \lambda B_{11}$ are the eigenvalues of $A - \lambda B$ in the region $\mathcal{D}$. We notice that if $A_{11} - \lambda B_{11}$ has no infinite eigenvalues, then B_{11} is invertible and

$$AU_{R_1} = BU_{R_1}(B_{11}^{-1}A_{11});$$

if $A_{11} - \lambda B_{11}$ has no zero eigenvalues, then A_{11} is invertible and

$$AU_{R_1}(A_{11}^{-1}B_{11}) = BU_{R_1}.$$

Note that the region $\mathcal{D}$ in the SDC problem is generally assumed in the interior (or exterior) of the unit disk. Otherwise, the Möbius transformations $(\alpha A + \beta B)(\gamma A + \delta B)^{-1}$ can be applied to transform original region as a rather general region [2].

One direct method to solve (1) (not requires $\rho(\Lambda) < 1$), is to apply the so-called QZ algorithm. That is, through the QZ algorithm, the matrix A is reduced to triangular or upper quasi-triangular form and B to upper triangular form. One is then able to compute eigenvectors through the reduced form (see [1, 8, 16] for the details). Unlike the direct method, we propose in this work an iterative method, AB-algorithm, to solve (1). This method is done by defining a sequence of matrix pencils $\{A_k - \lambda B_k\}$ with $(A_1, B_1) = (A, B)$ and $(A_k, B_k) = (\mathcal{M}_{k-1}A_{k-1}, \mathcal{N}_{k-1}B_1)$ for any integer $k > 1$. Here, $(\mathcal{N}_{k-1}, \mathcal{M}_{k-1})$ is a solution belonging to the left null space of $\begin{bmatrix} A_1 \\ -B_k \end{bmatrix}$, that is,

$$\mathcal{N}_k A_1 = \mathcal{M}_k B_k. \tag{2}$$

Due to the specific structure embedded in the matrix pencil $A_1 - \lambda B_k$, some $(\mathcal{N}_k, \mathcal{M}_k)$ can be designed such that $\{A_k - \lambda B_k\}$ have the same structure. We refer the reader to [21, 5] and to the references therein. In these works, iterative algorithms for computing invariant subspace of structured matrix pencil $A_1 - \lambda B_k$ are provided with a quadratic convergence for solving algebraic Riccati and related matrix equations.

Observe that $A_1U = B_1U\Lambda^1$. Suppose this process can be continually iterated to obtain the new pencil $A_k - \lambda B_k$ such that $A_kU = B_kU\Lambda^k$. It must be that

$$\begin{aligned} A_{k+1}U &= \mathcal{M}_kA_kU = \mathcal{M}_kB_kU\Lambda^k \\ &= \mathcal{N}_kA_1U\Lambda^k = \mathcal{N}_kB_1U\Lambda^{k+1} = B_{k+1}U\Lambda^{k+1}. \end{aligned} \tag{3}$$

This implies that if $\rho(\Lambda) < 1$, and if the sequence $\{B_k\}$ is uniformly bounded, then $\lim_{k\to\infty} A_kU = 0$. Once the sequence $\{A_k\}$ also converges, say $A_\infty := \lim_{k\to\infty} A_k$, we are able to solve the solution U by computing the the right null space of A_∞. We notice that this AB-algorithm theoretically not only preserves a discrete-type flow property (see Theorem 2.3) but can be accelerated with the rate of convergence of any desired order. To demonstrate the capability of this algorithm, we use it to provide a theoretical result to compute the matrix square root as an example. It is known that matrix square root is not unique (even up to sign) or even exists, for example,

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ \sin(\theta) & -\cos(\theta) \end{bmatrix}^2$$

for any $\theta \in \mathbb{R}$ and $\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ does not have a square root. Indeed, let $S \in \mathbb{C}^{n\times n}$ be a matrix having no nonpositive real eigenvalues. Then the quadratic matrix equation

$$X^2 - S = 0 \tag{4}$$

has a unique solution X such that $\lambda(X) \subset \mathbb{C}^+$. This is called the principal square root of S and denote it by $\sqrt{S}$ [6, 10]. Numerical methods for computing the matrix principal square root, including the (modified) Schur method, Newton's method, and its variants, have been widely discussed in the numerical linear algebra community. See [17, 6, 9, 10, 11, 22, 12, 23] and the references therein. Unlike conventional methods, this AB-algorithm can be modified to obtain the square root efficiently, that is, the rate of convergence of this algorithm can be of any desired order r. Specifically, our method is equivalent to the so-called Newton's method when $r = 2$. More precisely, though preserving a similar convergence property like the Newton's method, our algorithm can be shown that, under mild adjustments, the speed of convergence can be q-superlinearly with any order [15].

This work is organized as follows. In section 2 we provide properties of the AB-Algorithm. In section 3 we modified this AB-Algorithm so that its convergence can be of any order. In section 4 we report a numerical application to solve the matrix square root, and the concluding remarks are given in section 5.

2 The AB-Algorithm and Its Corresponding Properties

Recall that the idea of the AB-Algorithm depends heavily on the determination of the left null space of $\begin{bmatrix} A_1 \\ -B_k \end{bmatrix}$. Observe that $B_1(A_1+B_1)^{-1}A_1 - A_1(A_1+B_1)^{-1}B_1 = 0$, if $-1 \notin \lambda(A_1, B_1)$ and (A_1, B_1) are two $n \times n$ matrices. That is, $(N_1, M_1) := (B_1(A_1+B_1)^{-1}, A_1(A_1+B_1)^{-1})$ is the left null space of $\begin{bmatrix} A_1 \\ -B_1 \end{bmatrix}$. Using the same procedure, we would like to generate the matrix sequences $\{A_k\}$ and $\{B_k\}$ by defining

$$A_k = A_1(A_1 + B_{k-1})^{-1}A_{k-1}, \tag{5a}$$

$$B_k = B_{k-1}(A_1 + B_{k-1})^{-1}B_1, \tag{5b}$$

once the process can be iterated.

It should be noted that if $\mathcal{M}_{k-1} = B_{k-1}(A_1+B_{k-1})^{-1}$ and $\mathcal{N}_{k-1} = A_1(A_1+B_{k-1})^{-1}$ for any integer $k > 1$, it can be seen that $B_{k-1}(A_1+B_{k-1})^{-1}A_1 = A_1(A_1+B_{k-1})^{-1}B_{k-1}$, which satisfies the assumption (2). For simplicity, we let $\Delta_{i,j} := (A_i + B_j)^{-1}$ so that the sequences $\{A_k\}$ and $\{B_k\}$ in (5) can be rewritten as

$$A_k = A_1\Delta_{1,k-1}A_{k-1} = A_{k-1} - B_{k-1}\Delta_{1,k-1}A_{k-1}, \tag{6a}$$

$$B_k = B_{k-1}\Delta_{1,k-1}B_1 = B_1 - A_1\Delta_{1,k-1}B_1. \tag{6b}$$

Based on (6), we propose the following AB-algorithm for computing the stable subspace of the matrix pencil $A_1 - \lambda B_1$:

Algorithm 2.1. (AB-Algorithm)

```
1. Given a pencil $A_1 - \lambda B_1$, initialize a tolerance $\tau > 0$ and a positive integer kmax.
2. For $k = 2...$, iterate until $dist(Null(A_{k-1}), Null(A_k)) < \tau$ or $k > kmax$.
    (a) $A_k = A_1\Delta_{1,k-1}A_{k-1}$,
    (b) $B_k = B_{k-1}\Delta_{1,k-1}B_1$,
```

Here, "Null($\cdot$)" denotes the null space of the given matrix and "dist($\cdot,\cdot$)"denotes the distance between two subspaces [8].

Note that on the one hand, Algorithm 2.1 provides an alternative approach for finding the stable invariant subspace U (i.e., $A_1U = U\Lambda$ and $\rho(\Lambda) < 1$) of the matrix A_1 by constructing A_∞ (once it exists) directly as follows:

Remark 2.1. *If no breakdown occurs in Algorithm 2.1 and $B_1 = I_n$, for any integer $k > 1$ we have*

$$A_k = A_1^k(\sum_{j=0}^{k-1} A_1^j)^{-1}, \tag{7a}$$

$$B_k = (\sum_{j=0}^{k-1} A_1^j)^{-1}. \tag{7b}$$

In other words, to obtain the stable subspace of the matrix A_1, we only need to focus on the iterations generated by (7a)*.*

On the other hand, once the iteration is available, we are interested in characterizing the transformation of eigenvalues of the matrix pencil $A_1 - \lambda B_1$ after each iteration. First, we give an observation about the relationship between the eigenvalues of $A_k - \lambda B_k$ and the eigenvalues of $A_1 - \lambda B_1$. Since the proof can be read off from (3), we omit our proof here.

Lemma 2.1. *Let $A_1 - \lambda B_1$ be a regular matrix pencil, and let $\{A_k - \lambda B_k\}$ be the sequence of matrix pencils generated by Algorithm 2.1, if no breakdown occurs. If $\lambda \in \lambda(A_1, B_1)$ with $\lambda \in \mathbb{C} \cup \{\infty\}$, then $\lambda^k \in \lambda(A_k, B_k)$. (Here, $\infty^k := \infty$)*

Subsequently, we have the following theorem which gives rise to the appearance of new eigenvalues induced by the AB-algorithm.

Theorem 2.1. *Let $A_1 - \lambda B_1$ be a regular matrix pencil, and let $\{A_k - \lambda B_k\}$ be the sequence of matrix pencils generated by Algorithm 2.1, if no breakdown occurs. Let $\{\lambda_1^{(i,k)}, \dots, \lambda_n^{(i,k)}\}$ be the set of eigenvalues of the matrix pencils $A_i - \lambda B_k$ for any two positive integers i and k. Then, for $1 \leq j \leq n$, the set of eigenvalues has the following properties:*

1. $$\lambda_j^{(1,k)} = \begin{cases} \sum_{s=1}^{k} (\lambda_j^{(1,1)})^s, & \lambda_j^{(1,1)} \in \mathbb{C}, \\ \infty, & \lambda_j^{(1,1)} = \infty. \end{cases}$$

2. $$\lambda_j^{(i,1)} = \begin{cases} \dfrac{(\lambda_j^{(1,1)})^i}{\sum_{s=0}^{i-1} (\lambda_j^{(1,1)})^s}, & \lambda_j^{(1,1)} \in \mathbb{C}, \\ \infty, & \lambda_j^{(1,1)} = \infty. \end{cases}$$

3. $$\lambda_j^{(i,k)} = \begin{cases} (\lambda_j^{(1,1)})^i \dfrac{\sum_{s=0}^{k-1} (\lambda_j^{(1,1)})^s}{\sum_{s=0}^{i-1} (\lambda_j^{(1,1)})^s}, & \lambda_j^{(1,1)} \in \mathbb{C}, \\ \infty, & \lambda_j^{(1,1)} = \infty. \end{cases}$$

Proof. Assume without loss of generality that A_1 and B_1 are upper triangular matrices. Otherwise, let U and V be two unitary matrices such that $U^H A_1 V$ and $U^H B_1 V$ both are upper triangular matrices. Upon using (5), it can be seen that A_k and B_k are also upper triangular, and

$$\begin{aligned}
\lambda_j^{(1,k)} &= \begin{cases} (1+\lambda_j^{(1,k-1)})\lambda_j^{(1,1)}, & \lambda_j^{(1,1)} \in \mathbb{C}, \\ \infty, & \lambda_j^{(1,1)} = \infty, \end{cases} \\
\lambda_j^{(i,1)} &= \begin{cases} \dfrac{\lambda_j^{(i,i)}}{1+\lambda_j^{(1,i-1)}}, & \lambda_j^{(1,1)} \in \mathbb{C}, \\ \infty, & \lambda_j^{(1,1)} = \infty. \end{cases}
\end{aligned}$$

Moreover,

$$\lambda_j^{(i,k)} = \begin{cases} (1+\lambda_j^{(1,k-1)})\lambda_j^{(i,1)}, & \lambda_j^{(1,1)} \in \mathbb{C}, \\ \infty, & \lambda_j^{(1,1)} = \infty, \end{cases}$$

for $i, k \geq 2$. We remark that $\lambda_j^{(1,i-1)} \neq -1$ since $A_i - \lambda B_i$ is well-defined, and from Lemma 2.1, we have $\lambda_j^{(i,i)} = (\lambda_j^{(1,1)})^i$, which completes the proof of the theorem.

□

We notice that Algorithm 2.1 is workable if and only if the sum of matrices A_1 and B_{k-1}, for any integer $k > 1$, is invertible, that is, $-1 \notin \lambda(A_1, B_{k-1})$, for any integer $k > 1$. This capacity can be completely characterized by the pth roots of unity, except itself.

Theorem 2.2. *Let $A_1 - \lambda B_1$ be a regular matrix pencil, and let*

$$S_k = \bigcup_{2 \leq p \leq k+1} \{e^{\frac{2q\pi i}{p}} : 1 \leq q \leq p-1\}.$$

If

$$S_k \cap \lambda(A_1, B_1) = \phi,$$

then the sequence of matrix pencils $A_k - \lambda B_k$, for any integer $k \geq 1$, can be generated using Algorithm 2.1, or, generally, all sequences of matrices $\{A_k - \lambda B_k\}$ generated by iterations (5) *with the initial matrix pencil $A_1 - \lambda B_1$ are no breakdown, if*

$$S_\infty \cap \lambda(A_1, B_1) = \phi. \qquad (8)$$

Corollary 2.1. *For any positive integers i, j and k, we have $A_k - B_k = A_1 - B_1$, that is, $A_i - A_j = B_i - B_j$, provided that $S_{\max\{i,j,k\}} \cap \lambda(A_1, B_1) = \phi$.*

Proof. The proof is by induction on k. When $k = 1$, the result is evident. Suppose we have proved this corollary for $k = \ell$. Then, by the induction hypothesis

$$\begin{aligned} A_{\ell+1} - B_{\ell+1} &= A_\ell - B_\ell \Delta_{1,\ell} A_\ell - B_\ell \Delta_{1,\ell} B_1 \\ &= A_\ell - B_\ell \Delta_{1,\ell}(A_\ell + B_1) = A_\ell - B_\ell = A_1 - B_1. \end{aligned}$$

This completes the proof. □

From Corollary 2.1, each step of B_k can be obtained by $B_k = A_k + B_1 - A_1$. We conclude that the counts of Algorithm 2.1 for one iteration is $\frac{14}{3}n^3$ flops. This is because the computation is preliminary determined by the product of two $n \times n$ matrices, the calculation of the Gaussian elimination with partial pivoting, and the performance of solving n lower triangular systems and n upper triangular systems. Hence, the calculation of the counts contains a PLU factorization (cost : $\frac{2}{3}n^3$ flops) and two multiplication (cost : $4n^3$ flops). Here, we ignore any $O(n^2)$ operation counts and the memory counts. We notice that the computational cost of QZ algorithm is about $46n^3$ flops (the right eigenvectors are desired). On the other hand, it follows from Theorem 2.2 that Algorithm 2.1 is well-defined, once (8) is satisfied. Here, we use Gaussian elimination with partial pivoting, which is known to perform well and usually eliminate the numerical instability in practice [13], to compute the matrix inverse so that the iteration will not terminate prematurely. To perform the error analysis and decide the numerical stability of Algorithm 2.1, the reader is referred to [14] for a similar discussion.

We remark that Corollary 2.1 also implies that $\lim\limits_{k\to\infty} A_k$ exists if and only if $\lim\limits_{k\to\infty} B_k$ exists. Note that in (5), the iterations of the matrix pencils $A_k - \lambda B_k$, for $k \geq 1$, are relative to the initial pencil $A_1 - \lambda B_1$. We would like to derive a more general iterative method, which are easily accessible through any initial pencil $A_i - \lambda B_i$. To this purpose, we shall first introduce the well-known Sherman Morrison Woodbury formula (SMWF).

Lemma 2.2. *[4] Let A and B be two arbitrary matrices of size n, and let X and Y be two $n \times n$ nonsingular matrices. Assume that $Y^{-1} \pm BX^{-1}A$ is nonsingular. Then, $X \pm AYB$ is invertible and*

$$(X \pm AYB)^{-1} = X^{-1} \mp X^{-1}A(Y^{-1} \pm BX^{-1}A)^{-1}BX^{-1}.$$

This lemma gives a useful method to prove the following result.

Theorem 2.3. *Let the assumption (8) holds and $\{A_k - \lambda B_k\}$ be the sequence of matrix pencils obtained by (5) with initial $A_1 - \lambda B_1$. Then,*

$$A_{i+j} = A_i(A_i + B_j)^{-1}A_j, \tag{9a}$$

$$B_{i+j} = B_j(A_i + B_j)^{-1}B_i, \tag{9b}$$

where i and j are any two positive integers.

Proof. This proof is divided into two parts. We first fix $j = 1$ and show that the statement (9) is true for any positive integer i. We prove by induction on i. When $i = 1$, the statement (9) is definitely true from the definition of A_2 and B_2. Suppose (9) is true for $i = s$. It follows from Lemma 2.2 that

$$\begin{aligned}
\Delta_{1,s+1} &= (A_1 + B_s - A_s\Delta_{s,1}B_s)^{-1} \\
&= \Delta_{1,s} + \Delta_{1,s}A_s(A_s + B_1 - B_s\Delta_{1,s}A_s)^{-1}B_s\Delta_{1,s} \\
&= \Delta_{1,s} + \Delta_{1,s}A_s\Delta_{1+s,1}B_s\Delta_{1,s}, \\
\Delta_{1+s,1} &= (A_s - B_s\Delta_{1,s}A_s + B_1)^{-1} \\
&= \Delta_{s,1} + \Delta_{s,1}B_s(A_1 + B_s - A_s\Delta_{s,1}B_s)^{-1}A_s\Delta_{s,1} \\
&= \Delta_{s,1} + \Delta_{s,1}B_s\Delta_{1,s+1}A_s\Delta_{s,1}.
\end{aligned}$$

Thus, we have

$$\begin{aligned}
A_{(s+1)+1} &= A_{1+(s+1)} = A_{s+1} - B_{s+1}\Delta_{1,s+1}A_{s+1} \\
&= A_1 - B_1\left[\Delta_{s,1} + \Delta_{s,1}B_s\Delta_{1,s+1}A_s\Delta_{s,1}\right]A_1 \\
&= A_1 - B_1\Delta_{s+1,1}A_1 = A_{s+1}\Delta_{s+1,1}A_1, \\
B_{(s+1)+1} &= B_{1+(s+1)} = B_1 - A_1\Delta_{1,s+1}B_1 \\
&= B_1 - A_1\left[\Delta_{1,s} + \Delta_{1,s}A_s\Delta_{s+1,1}B_s\Delta_{1,s}\right]B_1 \\
&= B_{s+1} - A_{s+1}\Delta_{s+1,1}B_{s+1} = B_1\Delta_{s+1,1}B_{s+1},
\end{aligned}$$

which completes the proof of the first part.

Now suppose that (9) is true for $j = s$ and any i. In particular,

$$\begin{aligned}
\Delta_{i,s+1} &= (A_i + B_s - A_s\Delta_{s,1}B_s)^{-1} \\
&= \Delta_{i,s} + \Delta_{i,s}A_s(A_s + B_1 - B_s\Delta_{i,s}A_s)^{-1}B_s\Delta_{i,s} \\
&= \Delta_{i,s} + \Delta_{i,s}A_s\Delta_{i+s,1}B_s\Delta_{i,s}, \\
\Delta_{i+s,1} &= (A_s - B_s\Delta_{i,s}A_s + B_1)^{-1} \\
&= \Delta_{s,1} + \Delta_{s,1}B_s(A_i + B_s - A_s\Delta_{s,1}B_s)^{-1}A_s\Delta_{s,1} \\
&= \Delta_{s,1} + \Delta_{s,1}B_s\Delta_{i,s+1}A_s\Delta_{s,1}.
\end{aligned}$$

This implies

$$\begin{aligned}
A_{i+(s+1)} &= A_{(i+s)+1} = A_1 - B_1\Delta_{i+s,1}A_1 \\
&= A_1 - B_1\left[\Delta_{s,1} + \Delta_{s,1}B_s\Delta_{i,s+1}A_s\Delta_{s,1}\right]A_1 \\
&= A_{s+1} - B_{s+1}\Delta_{i,s+1}A_{s+1} = A_i\Delta_{i,s+1}A_{s+1}, \\
B_{i+(s+1)} &= B_{(i+s)+1} = B_{i+s} - A_{i+s}\Delta_{i+s,1}B_{i+s} \\
&= B_i - A_i\left[\Delta_{i,s} + \Delta_{i,s}A_s\Delta_{i+s,1}B_s\Delta_{i,s}\right]B_i \\
&= B_i - A_i\Delta_{i,s+1}B_i = B_{s+1}\Delta_{i,s+1}B_i,
\end{aligned}$$

which completes the proof of the theorem. □

Two things are required to be noted. First, Theorem 2.3 implies that the iterative sequence $\{A_k - \lambda B_k\}$ can be formulated explicitly from any two matrix pencils $A_i - \lambda B_i$ and $A_j - \lambda B_j$, where $i + j = k$. The formula also gives rise to a discrete-type flow and can be used to accelerate the iterations given in Algorithm 2.1. Second, it follows from Corollary 2.1 and Theorem 2.3 that $A_k = A_{k-1}\Delta_{1,k-1}A_1 = A_1\Delta_{1,k-1}A_{k-1} = A_{k-1}\Delta_{k-1,1}A_1 = A_1\Delta_{k-1,1}A_{k-1}$. It shows that the iterations A_k and B_k, regardless of the assumptions (5), have the following four equivalent forms by using the same initial matrix pencil:

1.	$A_k^{(1)} = A_1^{(1)}(A_1^{(1)} + B_{k-1}^{(1)})^{-1}A_{k-1}^{(1)}$, $B_k^{(1)} = B_{k-1}^{(1)}(A_1^{(1)} + B_{k-1}^{(1)})^{-1}B_1^{(1)}$;
2.	$A_k^{(2)} = A_1^{(2)}(B_1^{(2)} + A_{k-1}^{(2)})^{-1}A_{k-1}^{(2)}$, $B_k^{(2)} = B_{k-1}^{(2)}(A_1^{(2)} + B_{k-1}^{(2)})^{-1}B_1^{(2)}$;
3.	$A_k^{(3)} = A_1^{(3)}(A_1^{(3)} + B_{k-1}^{(3)})^{-1}A_{k-1}^{(3)}$, $B_k^{(3)} = B_{k-1}^{(3)}(B_1^{(3)} + A_{k-1}^{(3)})^{-1}B_1^{(3)}$;
4.	$A_k^{(4)} = A_1^{(4)}(B_1^{(4)} + A_{k-1}^{(4)})^{-1}A_{k-1}^{(4)}$, $B_k^{(4)} = B_{k-1}^{(4)}(B_1^{(4)} + A_{k-1}^{(4)})^{-1}B_1^{(4)}$.

The next theorem is to know how the eigeninformation is transferred during the iterative process.

Theorem 2.4. *Let $A_1 - \lambda B_1$ be a regular matrix pencil, and let $\{A_k - \lambda B_k\}$ be the sequence of matrices generated by Algorithm 2.1. Suppose that the condition* (8) *holds and $A_1 U = B_1 U \Lambda$. Then,*

(a) $A_1 U = B_k U \sum_{j=1}^{k} \Lambda^j$.

(b) $A_k U = B_k U \Lambda^k$. *In particular, if $1 \notin \lambda(\Lambda)$, then*

$$A_k U = (B_1 - A_1)U\Lambda^k(I_n - \Lambda^k)^{-1}. \tag{10}$$

(c) $A_i U \sum_{j=1}^{i} \Lambda^j = B_k U \Lambda^i \sum_{j=1}^{k} \Lambda^j$, *for any two positive integers i and k.*

Proof. Clearly, (a) is true for $k = 1$. Suppose that the statement is true for a positive integer $k = s$; that is,

$$A_1 U = B_s U \sum_{j=1}^{s} \Lambda^j.$$

We notice that

$$\begin{aligned} A_1 U - B_s\Delta_{1,s}A_1 U &= (A_1 + B_s)\Delta_{1,s}A_1 U - B_s\Delta_{1,s}A_1 U \\ &= B_s\Delta_{1,s}A_1 U \sum_{j=1}^{s} \Lambda^j = B_s\Delta_{1,s}B_1 U \sum_{j=2}^{s+1} \Lambda^j, \end{aligned}$$

so that

$$A_1U = B_s\Delta_{1,s}B_1U\Lambda + B_s\Delta_{1,s}B_1U\sum_{j=2}^{s+1}\Lambda^j = B_{s+1}U\sum_{j=1}^{s+1}\Lambda^j.$$

The result of the first part of (b) has been given in our introduction. We thus omit the proof here. Since

$$A_kU = B_kU\Lambda^k = (A_k + B_1 - A_1)U\Lambda^k = A_kU\Lambda^k + (B_1 - A_1)U\Lambda^k,$$

we see that (10) holds, while $1 \notin \lambda(\Lambda)$. Here, the second equality follows from Corollary 2.1.

To prove (c), we first show that for any positive integer i,

$$A_1U\Lambda^i = A_iU\sum_{j=1}^{i}\Lambda^j.$$

By Theorem 2.3, since $A_i = A_1 - B_1\Delta_{i-1,1}A_1$ and $B_i = B_1\Delta_{i-1,1}B_{i-1}$, we have

$$(A_1 - A_i)U = (B_1\Delta_{i-1,1}A_1)U = B_1\Delta_{i-1,1}B_{i-1}U\sum_{j=1}^{i-1}\Lambda^j = B_iU\sum_{j=1}^{i-1}\Lambda^j.$$

Or, equivalently,

$$A_1U\Lambda^i = A_iU\Lambda^i + B_iU\Lambda^i\sum_{j=1}^{i-1}\Lambda^j = A_iU\sum_{j=1}^{i}\Lambda^j,$$

since $A_iU = B_iU\Lambda^i$.

Second, from (a), we have already proved (c) for $i = 1$ and a given positive integer k. Assume (c) is true for $i = s$; that is,

$$A_sU\sum_{j=1}^{s}\Lambda^j = B_kU\Lambda^s\sum_{j=1}^{k}\Lambda^j.$$

Then

$$A_{s+1}U\sum_{j=1}^{s+1}\Lambda^j = A_1U\Lambda^{s+1} = (A_sU\sum_{j=1}^{s}\Lambda^j)\Lambda = B_kU\Lambda^{s+1}\sum_{j=1}^{k}\Lambda^j.$$

□

3 Modified AB-Algorithm

Let $\{A_k - \lambda B_k\}$ be the sequence of matrices generated by Algorithm 2.1. Before we move on, we should emphasize that the structure of the matrix pencil $A_k -$

λB_k is invariant once the subscripts $i+j=k$; that is, the generation of the sequence $\{A_k - \lambda B_k\}$ is independent of the subscript in A_i, A_j, B_i and B_j. To fully take advantages of this invariance, we would like to design algorithms by applying Theorem 2.3 to generate accelerated iterations with convergence of any desired order as follows.

Algorithm 3.1. (Modified AB-Algorithm)

1. *Given a positive integer $r > 1$, a tolerance $\tau > 0$, and a positive integer kmax, let $(\widehat{A}_1, \widehat{B}_1) = (A_1, B_1)$;*
2. *For $k = 2, \dots,$ iterate until dist$(Null(\widehat{A}_{k-1}), Null(\widehat{A}_k)) < \tau$ or $k > kmax$.*

$$\begin{aligned}\widehat{A}_k &= A_{k-1}^{(r-1)}(A_{k-1}^{(r-1)} + \widehat{B}_{k-1})^{-1}\widehat{A}_{k-1},\\ \widehat{B}_k &= \widehat{B}_{k-1}(A_{k-1}^{(r-1)} + \widehat{B}_{k-1})^{-1}B_{k-1}^{(r-1)},\end{aligned}$$

   until convergence, where $(A_{k-1}^{(r-1)}, B_{k-1}^{(r-1)})$ is defined in step 3.

3. *For $\ell = 1, \dots, r-2$, iterate*

$$\begin{aligned}A_{k-1}^{(\ell+1)} &= A_{k-1}^{(\ell)}(A_{k-1}^{(\ell)} + \widehat{B}_{k-1})^{-1}\widehat{A}_{k-1},\\ B_{k-1}^{(\ell+1)} &= \widehat{B}_{k-1}(A_{k-1}^{(\ell)} + \widehat{B}_{k-1})^{-1}B_{k-1}^{(\ell)},\end{aligned}$$

   with $(A_{k-1}^{(1)}, B_{k-1}^{(1)}) = (\widehat{A}_{k-1}, \widehat{B}_{k-1})$.

For clarity, a thing should be emphasized here. The AB algorithm has been developed to obtain the stable deflating subspace of the generalized eigenvalue problem $A_1 U = B_1 U \Lambda$. However, the sequence $\{A_k U\}$ provided in Algorithm 2.1 converges only r-linearly to 0, once the spectral radius of Λ is less than 1, and the sequence $\{B_k\}$ is uniformly bounded. From Algorithm 3.1 it follows that

$$A_{k-1}^{(\ell+1)}U = B_{k-1}^{(\ell+1)}U\Lambda^{(\ell+1)r^{k-2}}, \tag{11a}$$

$$\widehat{A}_k U = \widehat{B}_k U \Lambda^{r^{k-1}}, \tag{11b}$$

for $k = 2, \ldots$, and $\ell = 1, \dots, r-2$, and $(\widehat{A}_k, \widehat{B}_k) = (A_{r^{k-1}}, B_{r^{k-1}})$. It follows from Theorem 2.4 that

$$\|\widehat{A}_k U\| \le \frac{\|(B_1 - A_1)U\|}{1 - \|\Lambda\|^{r^{k-1}}}\|\Lambda\|^{r^{k-1}},$$

where $\|.\|$ is a matrix induced norm such that $\|\Lambda\| < 1$. Thus the sequence $\{\widehat{A}_k U\}$ converges to 0 with r-order r. For a full account of the definition of the rate of convergence, the reader is referred to [15].

Note that given two initial $n \times n$ matrices A_1 and B_1, the overall cost for computing the modified AB-algorithm per iteration is $\frac{14(r-1)}{3}n^3$ flops. The computation cost of the modified AB-algorithm with positive integer r definitely increases as r increases. Theoretically, Algorithm 3.1 provide a r-order convergence sequence which approximates the solution of the stable subspace of $A - \lambda B$. Numerically, if $\rho(\Lambda)$ is not sufficiently close to 1, choosing $r = 2$ will be fast enough.

4 Application of the AB-Algorithm for Solving the Matrix Square Root

We notice that only recently, the modified AB-Algorithm with $r = 2$ have been adjusted specifically for solving a kind of Sylvester matrix equations [20] and the palindromic generalized eigenvalue problem [19]. In this section, we show that the AB-algorithm provide an alternative way to compute the matrix square root. In particular, the speed of convergence of the AB-algorithm can be of any desired order. As mentioned before, numerical methods for solving the matrix square root are numerous. Comparison of numerical performance among different methods is something worthy of our investigation and is in process. In (11a) we see that the sequence $\{\widehat{A}_i U\}$ converges with r-order r to 0. We then in this section use this accelerated techniques to solve the quadratic matrix equation defined in (4), i.e., find the principle square root $\sqrt{S}$ of the matrix S with $\lambda(S) \subset \mathbb{C}^+$. To this end, we relate (4) to the generalized eigenvalue problem

$$A \begin{bmatrix} I_n \\ \sqrt{S} \end{bmatrix} = B \begin{bmatrix} I_n \\ \sqrt{S} \end{bmatrix} \sqrt{S}, \tag{12}$$

where $A = \begin{bmatrix} 0 & I_n \\ S & 0 \end{bmatrix}$ and $B = I_{2n}$. Since $\lambda(S) \subseteq \mathbb{C}^+$, there is no guarantee that the AB-algorithm will converge. To remedy this situation, this matrix $\sqrt{S}$ in (12) must be retreated. One way is to apply the Möbius transformation

$$\mathcal{C}_{\sqrt{S}}(\gamma I_n) = (\gamma I_n - \sqrt{S})(\gamma I_n + \sqrt{S})^{-1},$$

where $\gamma > 0$ and $-1 \notin \lambda(\gamma I_n - \lambda\sqrt{S})$, i.e., -1 is not an eigenvalue of the matrix pencil $\gamma I_n - \lambda\sqrt{S}$; that is, recast (12) in the following equation

$$A_1 \begin{bmatrix} I_n \\ \sqrt{S} \end{bmatrix} = B_1 \begin{bmatrix} I_n \\ \sqrt{S} \end{bmatrix} \mathcal{C}_{\sqrt{S}}(\gamma I_n), \tag{13}$$

where $A_1 = \gamma B - A$ and $B_1 = \gamma B + A$. Observe that $\rho(\mathcal{C}_{\sqrt{S}}(\gamma I_n)) < 1$ since $\lambda(S) \subseteq \mathbb{C}^+$. Upon using the AB-algorithm, it can be easily checked that for any integer $k \geq 1$, A_k and B_k can be expressed as

$$A_k = \begin{bmatrix} Q_k & -I_n \\ -S & Q_k \end{bmatrix}, \; B_k = \begin{bmatrix} Q_k & I_n \\ S & Q_k \end{bmatrix}, \tag{14}$$

respectively, where the sequence $\{Q_k\}$ satisfies $Q_iQ_j = Q_jQ_i$, for any integers $i, j > 0$, and the following iteration

$$Q_{k+1} = (\gamma Q_k + S)(\gamma I_n + Q_k)^{-1} \tag{15}$$

with $Q_1 = \gamma I_n$. Note that once $Q_k = \sqrt{S}$ for some k, it follows that $Q_\ell = \sqrt{S}$ for all $\ell \geq k$.

Specifically, let $\mathcal{C}_\gamma(\lambda) = \frac{\gamma-\lambda}{\lambda+\gamma}$ be the Möbius transformation with a parameter $\gamma \neq 0$ and $\lambda \neq -\gamma$. Then, the inverse scalar Möbius transformation can be written as

$$\mathcal{C}_\gamma^{-1}(\lambda) = \gamma\frac{1-\lambda}{1+\lambda}, \quad \lambda \neq -1.$$

Let $\lambda = e^{\frac{2j\pi i}{n}} \in S_n\backslash\{-1\}$, where $1 \leq j < n$. It follows that the real part of the square of $a := \mathcal{C}_\gamma^{-1}(\lambda)$ is a real negative number, since

$$a^2 = \gamma^2(\frac{1-e^{\frac{2j\pi i}{n}}}{1+e^{\frac{2j\pi i}{n}}})^2 = -\gamma^2\tan^2(\frac{j\pi}{n}) < 0. \tag{16}$$

From (16) and Theorem (2.2), it follows that the AB-algorithm will terminate prematurely only if $\lambda(S) \subseteq \mathbb{C}^-$; that is, once $\lambda(S) \subseteq \mathbb{C}^+$, or even, $\lambda(S) \subseteq \mathbb{C}^+ \cup \{0\}$, the sequence of matrix pencils $\{A_k - \lambda B_k\}$, initiated by (13), is well-defined.

With an eye on the structure of the matrix pencil $A_k - \lambda B_k$, we look for an accelerated iteration induced by the assumption of $\widehat{A}_k - \lambda\widehat{B}_k$ in Algorithm 3.1.

Algorithm 4.1. (Iteration for solving the matrix square root)

1. *Given a positive integer $r > 1$, a tolerance $\tau > 0$, and a positive integer kmax, let $\widehat{Q}_1 = Q_1 = \gamma I_n$;*
2. *For $i = 2, \ldots,$ iterate until $\|\widehat{Q}_k - \sqrt{S}\| < \tau$ or $k > kmax$.*

$$\widehat{Q}_k := (S + \widehat{Q}_{k-1}Q_{k-1}^{(r-1)})(\widehat{Q}_{k-1} + Q_{k-1}^{(r-1)})^{-1},$$

until convergence, where $\widehat{Q}_{k-1}^{(r-1)}$ is defined in step 3.

3. *For $\ell = 1, \cdots, r-2$, iterate*

$$Q_{k-1}^{(\ell+1)} := (S + \widehat{Q}_{k-1}Q_{k-1}^{(\ell)})(\widehat{Q}_{k-1} + Q_{k-1}^{(\ell)})^{-1},$$

with $\widehat{Q}_{k-1}^{(1)} = \widehat{Q}_{k-1}$.

Note that $\widehat{Q}_k = Q_{r^{k-1}}$ for $k \geq 1$, and with the assumption of the existence of iterative sequences, we immediately have the following iterative formulae. We omit the proof here because the result can be straightforwardly shown by using induction.

Theorem 4.1. *Assume that the sequences generated by Algorithm 4.1 can be constructed with no break down. Then, we have the following two iterative formulae.*

1. *When r is even, let $q = \frac{r}{2}$. We have*

$$\widehat{Q}_{k+1} = (\sum_{j=0}^{q} \binom{r}{2j} \widehat{Q}_k^{r-2j} S^j)(\sum_{j=0}^{q-1} \binom{r}{2j+1} \widehat{Q}_k^{r-2j-1} S^j)^{-1}. \tag{17}$$

2. *While r is odd, let $q = \frac{r-1}{2}$. We have*

$$\widehat{Q}_{k+1} = (\sum_{j=0}^{q} \binom{r}{2j} \widehat{Q}_k^{r-2j} S^j)(\sum_{j=0}^{q} \binom{r}{2j+1} \widehat{Q}_k^{r-2j-1} S^j)^{-1}, \tag{18}$$

where the notation $\binom{n}{k}$ denotes the number of k-combinations from the set $S = \{1, 2, \cdots, n\}$ of n elements.

We notice that if S is a nonsingular matrix, then (17) and (18) can be simply expressed by the following rule:

$$\widehat{Q}_{k+1} = V_m U_m^{-1},$$

where

$$V_m = \sum_{j=0}^{[\frac{m}{2}]} \binom{m}{2j} \widehat{Q}_k^{m-2j} S^j = \frac{1}{2}((\widehat{Q}_k + \sqrt{S})^m + (\widehat{Q}_k - \sqrt{S})^m),$$

$$U_m = \sum_{j=0}^{[\frac{m-1}{2}]} \binom{m}{2j+1} \widehat{Q}_k^{m-2j-1} S^j = \frac{(\sqrt{S})^{-1}}{2}((\widehat{Q}_k + \sqrt{S})^m - (\widehat{Q}_k - \sqrt{S})^m).$$

Importantly, under nonsingularity assumption, a strong result related to the sequences $\{\mathcal{C}_{\sqrt{S}}(Q_i)\}$ and $\{\mathcal{C}_{\sqrt{S}}(\widehat{Q}_i)\}$ hold.

Lemma 4.1. *Suppose that S is nonsingular. Let i, j, and k be any positive integers, and $1 \leq i, j \leq k$. Then the following properties hold.*

1. *For the sequence $\{Q_k\}$, we have*
   a. $Q_k = \sqrt{S}(I_n + \mathcal{C}_{\sqrt{S}}(Q_1)^k)(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k)^{-1}$,
   b. $\mathcal{C}_{\sqrt{S}}(Q_i)^j = \mathcal{C}_{\sqrt{S}}(Q_j)^i$.

2. *For the sequence $\{\widehat{Q}_k\}$, we have*
   a. $\widehat{Q}_k = \sqrt{S}(I_n + \mathcal{C}_{\sqrt{S}}(\widehat{Q}_1)^{r^{k-1}})(I_n - \mathcal{C}_{\sqrt{S}}(\widehat{Q}_1)^{r^{k-1}})^{-1}$,
   b. $\mathcal{C}_{\sqrt{S}}(\widehat{Q}_i)^{r^{k-i}} = \mathcal{C}_{\sqrt{S}}(\widehat{Q}_j)^{r^{k-j}}$.

Proof. It follows from Theorem 2.4 and $Q_1 = \gamma I_n$ that

$$A_k U(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k) = (B_1 - A_1)U\mathcal{C}_{\sqrt{S}}(Q_1)^k. \tag{19}$$

Then, (14) and (19) yield

$$(Q_k - \sqrt{S})(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k) = 2\sqrt{S}\mathcal{C}_{\sqrt{S}}(Q_1)^k. \tag{20}$$

By adding $2\sqrt{S}(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k)$ to both sides of (20), we have

$$(Q_k + \sqrt{S})(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k) = 2\sqrt{S}. \tag{21}$$

From (20) and (21) together, it must be that

$$Q_k(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k) = \sqrt{S}(I_n + \mathcal{C}_{\sqrt{S}}(Q_1)^k).$$

Since S is nonsingular, it follows that $1 \notin \lambda(\mathcal{C}_{\sqrt{S}}(Q_1))$ so that

$$Q_k = \sqrt{S}(I_n + \mathcal{C}_{\sqrt{S}}(Q_1)^k)(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k)^{-1},$$

which is equivalent to

$$\mathcal{C}_{\sqrt{S}}(Q_1)^k = \mathcal{C}_{\sqrt{S}}(Q_k).$$

Since k is an arbitrary positive integer, we have

$$(\mathcal{C}_{\sqrt{S}}(Q_i))^j = \mathcal{C}_{\sqrt{S}}(Q_1)^{ij} = (\mathcal{C}_{\sqrt{S}}(Q_j))^i,$$

for $1 \leq i, j \leq k$.

Also, by Theorem 2.4, Algorithm 3.1, $\widehat{Q}_1 = \gamma I_n$, we have

$$\widehat{A}_k U = \widehat{B}_k U(\mathcal{C}_{\sqrt{S}}(\widehat{Q}_1))^{r^{k-1}},$$

which then completes the proof of part 2a. and part 2b. by applying the same strategies as above. □

Indeed, this iteration in Algorithm 4.1 converges to $\sqrt{S}$ with q-order r.

Theorem 4.2. *Suppose that S is a nonsingular matrix. Let $\|.\|$ be a matrix induced norm such that $\|\mathcal{C}_{\sqrt{S}}(\widehat{Q}_1)\| < 1$. Then,*

$$\|\widehat{Q}_{k+1} - \sqrt{S}\| \leq \mu\|\widehat{Q}_k - \sqrt{S}\|^r,$$

for some $\mu > 0$; that is, $\widehat{Q}_k \to \sqrt{S}$ with q-order r.

Proof. Using (20) and $\widehat{Q}_k = Q_{r^{k-1}}$, we see that

$$\widehat{Q}_k - \sqrt{S} = 2\sqrt{S}\mathcal{C}_{\sqrt{S}}(Q_1)^{r^{k-1}}(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^{r^{k-1}})^{-1}.$$

Without loss of generality we assume that $\widehat{Q}_k \neq \sqrt{S}$ for all k. Otherwise, $\widehat{Q}_\ell = \sqrt{S}$ for all $\ell \geq k$. It follows that

$$\begin{aligned}
\frac{\|\widehat{Q}_{k+1} - \sqrt{S}\|}{\|\widehat{Q}_k - \sqrt{S}\|^r} &\leq \frac{\|2\sqrt{S}\mathcal{C}_{\sqrt{S}}(Q_1)^{r^k}\|\|I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^{r^{k-1}}\|^r}{\|2\sqrt{S}\mathcal{C}_{\sqrt{S}}(Q_1)^{r^{k-1}}\|^r(1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)^{r^k}\|)} \\
&\leq \frac{2\|\sqrt{S}\|\|\mathcal{C}_{\sqrt{S}}(Q_1)^{r^{k-1}}\|^r\|I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^{r^{k-1}}\|^r}{2^r\frac{\|\mathcal{C}_{\sqrt{S}}(Q_1)^{r^{k-1}}\|^r}{\|(\sqrt{S})^{-1}\|^r}(1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)^{r^k}\|)} \\
&\leq 2^{1-r}\|\sqrt{S}\|\|(\sqrt{S})^{-1}\|^r \sup_{k\geq 1} \frac{(1 + \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^{r^{k-1}})^r}{1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^{r^k}} \\
&\leq \mu := \frac{2\|\sqrt{S}\|}{1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^r}\|(\sqrt{S})^{-1}\|^r < \infty.
\end{aligned}$$

□

Note that for $r = 2$ the iteration $\widehat{Q}_{k+1} = (\widehat{Q}_k^2 + S)(2\widehat{Q}_k)^{-1} = \frac{1}{2}(\widehat{Q}_k + S\widehat{Q}_k^{-1})$ with initial $\widehat{Q}_1 = \gamma I_n$, which is equivalent to the Newton's method for solving the matrix square root [12], converges to $\sqrt{S}$ with quadratic convergence. For $r = 3$ we have $\widehat{Q}_{k+1} = (\widehat{Q}_k^3 + 3\widehat{Q}_k S)(3\widehat{Q}_k^2 + S)^{-1}$, which provides that a cubically convergent iteration converges to $\sqrt{S}$ with initial $\widehat{Q}_1 = \gamma I_n$. Similarly, by Algorithm 4.1 we can make $\widehat{Q}_{k+1}$ converges to $\sqrt{S}$ q-superlinearly with any desired q-order r. However, without the accelerated technique, we can show in the following that the original sequence $\{Q_k\}$ only converges to $\sqrt{S}$ q-linearly.

Theorem 4.3. *Suppose that S is a nonsingular matrix. Let $\|.\|$ be a matrix induced norm such that $\|\mathcal{C}_{\sqrt{S}}(\widehat{Q}_1)\| < 1$. Then,*

$$\|Q_{k+1} - \sqrt{S}\| \leq \mu\|Q_k - \sqrt{S}\|,$$

for some $\mu \in (0, 1)$ and sufficient large k; that is, $Q_k \to \sqrt{S}$ q-linearly with q-factor μ.

Proof. From (20), we have

$$Q_k - \sqrt{S} = 2\sqrt{S}\mathcal{C}_{\sqrt{S}}(Q_1)^k(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k)^{-1}.$$

Thus,

$$\begin{aligned}
\|Q_{k+1} - \sqrt{S}\| &= \|(Q_k - \sqrt{S})(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^k)\mathcal{C}_{\sqrt{S}}(Q_1)(I_n - \mathcal{C}_{\sqrt{S}}(Q_1)^{k+1})^{-1}\| \\
&\leq \|\mathcal{C}_{\sqrt{S}}(Q_1)\|\frac{1 + \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^k}{1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^{k+1}}\|Q_k - \sqrt{S}\|.
\end{aligned}$$

Since $\dfrac{1 + \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^k}{1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^{k+1}} \to 1$ as $k \to \infty$, there exists a constant k_0 such that

$$\|\mathcal{C}_{\sqrt{S}}(Q_1)\|\frac{1 + \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^k}{1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^{k+1}} < 1$$

for $k \geq k_0$. Let $\mu = \dfrac{1 + \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^{k_0}}{1 - \|\mathcal{C}_{\sqrt{S}}(Q_1)\|^{k_0+1}} \|\mathcal{C}_{\sqrt{S}}(Q_1)\|$, which completes the proof. □

In the next result, we show that the AB-algorithm still converges, while solving the square root of a singular matrix, which is hard to be handled in general. See [22] for further discussion.

Corollary 4.1. *Suppose that S is a singular matrix having $\lambda(S) \subseteq \mathbb{C}^+ \cup \{0\}$ and the null eigenvalues are semisimple. Then,*

1. *$Q_k \to \sqrt{S}$ sublinearly,*
2. *$\widehat{Q}_k \to \sqrt{S}$ q-linearly with q-factor $\frac{1}{r}$.*

Proof. Let P be an invertible matrix so that $\mathrm{diag}(0_p, J_{n-p}) = P\sqrt{S}P^{-1}$ be the Jordan canonical form of $\sqrt{S}$ with $\lambda(J_{n-p}) \subset \mathbb{C}^+$. Upon the use of substitution and Lemma 4.1, we have

$$PQ_kP^{-1} = \mathrm{diag}(Q_k^{(11)}, Q_k^{(22)}),$$

where $Q_k^{(11)}$ is derived directly by (15) and $Q_k^{(22)}$ is followed from Lemma 4.1 such that

$$\begin{aligned} Q_k^{(11)} &= \frac{\gamma I_n}{k}, \\ Q_k^{(22)} &= J_{n-p}(I + \mathcal{C}_{J_{n-p}}(\gamma I_{n-p})^k)(I - \mathcal{C}_{J_{n-p}}(\gamma I_{n-p})^k)^{-1}. \end{aligned}$$

Since $\{Q_k^{(11)}\}$ converges to zero sublinearly and $\{Q_k^{(22)}\}$ converges to zero q-linearly, $\{Q_k\}$ converges to $\sqrt{S}$ sublinearly. In the similar way, we have

$$P\widehat{Q}_kP^{-1} = \mathrm{diag}(\widehat{Q}_k^{(11)}, \widehat{Q}_k^{(22)}),$$

where

$$\begin{aligned} \widehat{Q}_k^{(11)} &= \frac{\gamma I_n}{r^k}, \\ \widehat{Q}_k^{(22)} &= J(I_n + \mathcal{C}_J(\gamma I_n)^{r^{k-1}})(I_n - \mathcal{C}_J(\gamma I_n)^{r^{k-1}})^{-1}. \end{aligned}$$

Since $\{\widehat{Q}_k^{(11)}\}$ converges to zero q-linearly with q-factor r, it follows that $\{\widehat{Q}_k\}$ converges to $\sqrt{S}$ q-linearly with q-factor r. □

Remark 4.1. *Once the spectral radius of $\sqrt{S}$ in* (12) *is not less than 1, we can apply the Möbius transformations to shift eigenvalues of S such that $\rho(\mathcal{C}_{\sqrt{S}}(\gamma I_n))$ in* (13) *is less than 1. We would like our γ to have a capacity such that the optimal convergence speed in Algorithm 4.1 can be achieved. To this end, we seek γ to be equal to the optimal solution γ_0 of the following min-max problem*

$$\gamma_0 := \min_{\gamma>0} \max_{\lambda\in\lambda(S)} |\frac{\sqrt{\lambda} - \gamma}{\sqrt{\lambda} + \gamma}|. \tag{22}$$

This min-max problem is also known as the ADI min-max problem [25]. Numerical approaches for solving (22) *are numerous. Here we will not discuss it further. The reader is referred to [3, 25, 24] for example.*

5 Concluding remarks

By computing left null spaces, the contribution of this work is twofold. Theoretically, it provides an iterative method, embedded with a discrete-type flow property, to solve the stable deflating subspace of a matrix pencil $A - \lambda B$. This property then allows us to advance the iterative method. Numerically, we have discussed with the numerical behavior of the AB-algorithm, including both low computational cost and high numerical reliability. Since the solution of the matrix square root can be interpreted in terms of the stable deflating subspace of a matrix pencil, our method can be used to compute the matrix square root. We show that the speed of convergence has q-order r, and even more, for the singular case, where S is singular having no negative real eigenvalues, and the null eigenvalues are semisimple, the iteration still succeeds with a linear rate of convergence.

Particularly, since Algorithm 4.1 corresponds to Newton iteration with $r = 2$ and the initial guess γI_n, the limiting accuracy should not be worse than $\kappa(\sqrt{S})\epsilon$, where $\kappa(\sqrt{S})$ is the condition number of $\sqrt{S}$ and ϵ is machine precision [12, Table 6.2 on p.147]. Numerically, it is known that a stable variant of Newton iteration, the IN iteration [12, (6.20) on p.142], has been proposed with the limiting accuracy equal to ϵ. Whether the AB-algorithm for $r = 2$ has the desired accuracy or even more for $r > 2$ is something worthy of further investigation. Numerically, modified AB-algorithms for $r = 2$ were also developed for solving generalized continuous/discrete-time algebraic Riccati equations [19] and $\star$-Sylvester matrix equation [20]. How to apply the accelerated techniques in the work for solving other matrix equations (for example, matrix pth root) leads to the work in future.

Acknowledgment

This research work is partially supported by the Ministry of Science and Technology and the National Center for Theoretical Sciences in Taiwan.

References

[1] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, volume 11 of *Software, Environments, and Tools.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[2] Z. Bai, J. Demmel, and M. Gu. An inverse free parallel spectral divide and conquer algorithm for nonsymmetric eigenproblems. *Numer. Math.*, 76(3):279–308, 1997.

[3] P. Benner, H. Mena, and J. Saak. On the parameter selection problem in the Newton-ADI iteration for large-scale Riccati equations. *Electron. Trans. Numer. Anal.*, 29:136–149, 2007/08.

[4] D. S. Bernstein. *Matrix Mathematics: Theory, Facts, and Formulas.* Princeton University Press, Princeton, NJ, 2005.

[5] D. A. Bini, B. Iannazzo, and B. Meini. *Numerical Solution of Algebraic Riccati Equations*, volume 9 of *Fundamentals of Algorithms.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2012.

[6] C. R. DePrima and C. R. Johnson. The range of $A^{-1}A^*$ in GL(n, C). *Linear Algebra and Appl.*, 9:209–222, 1974.

[7] I. Gohberg, P. Lancaster, and L. Rodman. *Invariant Subspaces of Matrices with Applications.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2006.

[8] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[9] N. J. Higham. Newton's method for the matrix square root. *Math. Comp.*, 46(174):537–549, 1986.

[10] N. J. Higham. Computing real square roots of a real matrix. *Linear Algebra Appl.*, 88/89:405–430, 1987.

[11] N. J. Higham. Stable iterations for the matrix square root. *Numer. Algorithms*, 15(2):227–242, 1997.

[12] N. J. Higham. *Functions of Matrices: Theory and Computation.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.

[13] N. J. Higham. Gaussian elimination. *Wiley Interdisciplinary Reviews: Computational Statistics*, 3(3):230–238, 2011.

[14] T.-M. Huang and W.-W. Lin. Structured doubling algorithms for weakly stabilizing hermitian solutions of algebraic riccati equations. *Linear Algebra Appl.*, 430(5):1452 – 1478, 2009.

[15] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*, volume 16 of *Frontiers in Applied Mathematics.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995.

[16] D. Kressner. *Numerical Methods for General and Structured Eigenvalue Problems*, volume 46 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2005.

[17] P. Laasonen. On the iterative solution of the matrix equation $AX^2 - I = 0$. *Math. Tables Aids Comput.*, 12:109–116, 1958.

[18] P. Lancaster and L. Rodman. *Algebraic Riccati equations*. Oxford Science Publications. The Clarendon Press, Oxford University Press, New York, 1995.

[19] T. Li, C.-Y. Chiang, E. K. wah Chu, and W.-W. Lin. The palindromic generalized eigenvalue problem $A^*x = \lambda Ax$: Numerical solution and applications. *Linear Algebra Appl.*, 434(11):2269 – 2284, 2011.

[20] M. M. Lin and C.-Y. Chiang. A note on Sylvester-type equations. *Journal of the Franklin Institute*, 352(5):2171 – 2186, 2015.

[21] W.-W. Lin and S.-F. Xu. Convergence analysis of structure-preserving doubling algorithms for Riccati-type matrix equations. *SIAM J. Matrix Anal. Appl.*, 28(1):26–39, 2006.

[22] B. Meini. The matrix square root from a new functional perspective: theoretical results and computational issues. *SIAM J. Matrix Anal. Appl.*, 26(2):362–376 (electronic), 2004/05.

[23] S. Mizuno, Y. Moriizumi, T. S. Usuda, and T. Sogabe. An initial guess of Newton's method for the matrix square root based on a sphere constrained optimization problem. *JSIAM Lett.*, 8:17–20, 2016.

[24] T. Penzl. A cyclic low-rank Smith method for large sparse Lyapunov equations. *SIAM J. Sci. Comput.*, 21(4):1401–1418 (electronic), 1999/00.

[25] E. Wachspress. *The ADI model problem*. Springer, New York, 2013.