Improved SDA-ss algorithm for nonsymmetric algebraic Riccati equations

Runsheng Wu$^{1,2}$ and Bo Tang$^{1,2,3}$

$^1$College of Mathematics and Computing Science, Guilin University of Electronic Technology, Guilin, P. R. China
$^2$Guangxi Key Laboratory of Cryptography and Information Security, P. R. China
$^3$Email: apriltb@163.com

Abstract. It is well known that nonsymmetric algebraic Riccati equation can be efficiently solved by using the structure-preserving doubling algorithm (SDA) with the shift-and-shrink transformation and the corresponding doubling algorithm is called the SDA-ss. In this paper, we propose an improved SDA-ss algorithm. Preliminary numerical experiments show that the algorithm is efficient to derive the minimal nonnegative solution of nonsymmetric algebraic Riccati equation with M-matrix.

1. Introduction
Consider solving the following nonsymmetric algebraic Riccati equation

$$XCX - AX - XD + B = 0,$$

(1)

and its dual form

$$YBY - DY - YA + C = 0,$$

(2)

where $A$, $B$, $C$, $D$ are real matrices of sizes, $m \times m$, $m \times n$, $n \times m$, $n \times n$.

In many real-life applications such as in transport theory [1] and the Markov process [2], the coefficient matrices in (1) and (2) constitute a block structure of $M$-matrix, which is

$$K = \begin{bmatrix} D & -C \\ -B & A \end{bmatrix}. $$

(3)

If $K$ is a nonsingular $M$-matrix or an irreducible singular $M$-matrix, we referred such equations as $M$-matrix algebraic Riccati equations (MAREs) [3]. A large number of numerical iterative methods, including Newton method and fixed point method, have been extensively studied to find the minimum non-negative solution of MARE.

The solution of the algebraic Riccati equation can be expressed by the invariant of the matrix subspace or the compressed subspace of the matrix bundle. This property is very important both in theory and in algorithm design and analysis. The doubling algorithm is mainly an algorithm that utilizes the specific structure of the matrix.

We will introduce the theory of doubling algorithm and some common doubling algorithms. Due to different matrix transformation, the doubling algorithm has different initial processes, such as ADDA algorithm [3], SDA algorithm [4], SDA-SS algorithm [5], and DAGT algorithm [6].
After analyzing and comparing the performance of several doubling algorithms, we propose an improved SDA-ss algorithm. Preliminary numerical experiments show that the algorithm is efficient to derive the minimal nonnegative solution of nonsymmetric algebraic Riccati equation with M-matrix. The new algorithm has fewer iteration steps and smaller residuals than the original algorithm.

The paper is organized as follows. We will first introduce several commonly used doubling algorithms in Section 2. Section 3 is about our new algorithm. Experiments show that the new improved algorithm is very effective. The last section is the conclusion.

Notation. Symbol $\mathbb{R}^{n \times n}$ in this paper stands for the set of $n \times n$ real matrices. For matrices $A, B \in \mathbb{R}^{n \times n}$, we write $A \succeq B (A > B)$ if $a_{ij} \geq b_{ij} (a_{ij} > b_{ij})$ for all $i, j$. A real square matrix $A$ is called a $Z$-matrix if all its off-diagonal elements are nonpositive. It is clear that any $Z$-matrix $A$ can be written as $sI - B$ with $B \succeq 0$. A $Z$-matrix $A = sI - B$ with $B \succeq 0$ is called an $M$-matrix if $s \geq \rho(B)$, where $\rho(\cdot)$ denotes the spectral radius. It is called a singular $M$-matrix if $s = \rho(B)$ and a nonsingular $M$-matrix if $s > \rho(B)$.

2. Doubling algorithm

A lot of transformations are used in the doubling algorithm. Specifically, for matrices $A$, the Structure-preserving doubling algorithm(SDA) employs Cayley transformation

$$C(A, \alpha) = (A - \alpha I) (A + \alpha I)^{-1}$$

with $\alpha > 0$.

The SDA-ss employs the shift-and-shrink transformation

$$S(A, \gamma) = I - A/\gamma$$

with $\gamma > 0$.

2.1. SDA algorithm

$$\alpha = \max_i \{A_{ii}\}, \beta = \max_i \{D_{ii}\},$$

(Step 1) Calculate $\alpha$ and $\beta$ in (6), let $\gamma \geq \max\{\alpha, \beta\}$;

(Step 2) Calculate $A_\gamma = A + \gamma I$, $D_\gamma = D + \gamma I$, $W_\gamma = A_\gamma - BD_\gamma^{-1}C$, $V_\gamma = D_\gamma - CA_\gamma^{-1}B$;

(Step 3) $E_{\gamma} = I - 2\gamma W_\gamma^{-1}$, $F_{\gamma} = I - 2\gamma W_\gamma^{-1}$,

$G_\gamma = 2\gamma D_\gamma^{-1}CW_\gamma^{-1}$,

$H_\gamma = 2\gamma W_\gamma^{-1}BD_\gamma^{-1}$;

(Step 4) $E_0 = E_{\gamma}$, $F_0 = F_{\gamma}$, $G_0 = G_{\gamma}$, $H_0 = H_{\gamma}$, do following iterations until the convergence.

$F_{k+1} = F_k (I - H_k G_k)^{-1} F_k$,

$E_{k+1} = E_k (I - G_k H_k)^{-1} E_k$,

$H_{k+1} = H_k + F_k (I - H_k G_k)^{-1} H_k E_k$,

$G_{k+1} = G_k + E_k (I - G_k H_k)^{-1} G_k F_k$.

2.2. SDA-SS algorithm

(Step 1) Calculate $\beta$ in (6), let $\gamma \geq \beta$

(Step 2) Calculate $A_\gamma = I + \gamma^{-1} A$, $D_\gamma = I - \gamma^{-1} D$,

(Step 3) $E_{\gamma} = D_\gamma + \gamma^{-2} CA_\gamma^{-1} B$, $F_{\gamma} = A_\gamma^{-1}$,

$G_{\gamma} = \gamma^{-1} CA_\gamma^{-1}$,
$H_\gamma = \gamma^{-1} A_\gamma^{-1} B_\gamma$.

(Step 4) $E_o = E_\gamma$, $F_o = F_\gamma$, $G_o = G_\gamma$, $H_o = H_\gamma$, do following iterations until the convergence.

$F_{k+1} = F_k (I - H_k G_k)^{-1} F_k$,

$E_{k+1} = E_k (I - G_k H_k)^{-1} E_k$,

$H_{k+1} = H_k + F_k (I - H_k G_k)^{-1} H_k E_k$,

$G_{k+1} = G_k + E_k (I - G_k H_k)^{-1} G_k F_k$.

3. Improved SDA-ss algorithm and numerical experiment

We found the following phenomena. When the parameters of the two transformations are very close, that is, $\alpha$ and $\gamma$ can take roughly the same value, the number of iterations of SDA algorithm will be one less than that of SDA-ss. In this case, it is obviously not recommended to use SDA-ss because the computational cost of one more iteration is greater than the cost of calculating the initial value.

But on the other hand, if the value of the parameter $\gamma$ is much smaller than the best value of the parameter $\alpha$, the SDA-ss based on the shift-contraction transformation is faster than the SDA based on the Cayley transformation. Because the convergence ratio is better, the initial value is easier.

We can use the difference between the diagonal elements of matrix $A$ and matrix $D$ to improve the SDA-ss algorithm. When an element on the diagonal of $A$ is significantly larger than that on the diagonal of $D$, the original SDA-ss can be used directly. When an element on the diagonal of $D$ is obviously larger than that on the diagonal of $A$, transpose form (1) can be solved to achieve the purpose of exchanging $A$ and $D$ positions.

$$ZC^T Z - DTZ - ZA^T + B^T = 0$$

3.1. Improved SDA-ss algorithm

(Step 1) Get $\alpha$, $\beta$ in (6);

(Step 2) If $\alpha \geq \beta$, let $\gamma = \beta$, $X$ was calculated using the original SDA-ss(algorithm 2.2).

(Step 3) If $\alpha < \beta$, let $\gamma = \alpha$.

Let $A = D^T$, $B = B^T$, $C = C^T$, $D = A^T$.

$Z = X^T$ was calculated using the original SDA-ss(algorithm 2.2).

3.2. Numerical experiment

We compare performance of the new algorithm with SDA-ss and SDA. It is clear that the new method shares the same complexity of SDA-ss at each iterative step. Thus we only need to compare the decrease of the equation residual as the iterations increase. Our computations is implemented in MATLAB with a machine error $error \approx 1.11 \times 10^{-16}$. We take the stop criterion is $NRes < tol$ with

$$NRes = \frac{||\Phi^T C \Phi - A \Phi - D \Phi + B||_1}{||\Phi||_1 (||\Phi||_1 ||C||_1 + ||A||_1 + ||D||_1 ) + ||B||_1}$$

and $tol = 10^{-14}$, where $\Phi$ is derived by the new method.

"It" represents the number of iterations of different iteration methods, "CPU" represents the CPU running time of several algorithms, and "RES" represents the normalized residuals.

We will compare the SDA, SDA-ss and improved SDA-ss algorithm from the above three indicators.

Example 1. Our first example is taken from Examples 2 in [7].

Here coefficient matrices are
\[ D = \begin{pmatrix} 102 & -100 \\ -100 & 102 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \quad A = \begin{pmatrix} 3 & -1 \\ -1 & 3 \end{pmatrix}, \quad B = \xi C. \]

with \( \xi \) a positive constant.

**Table 1.** Comparison of results of Example 1.

|        | SDA-ss  | Improved SDA-ss | SDA      |
|--------|---------|----------------|----------|
| CPU    | 0.0018401 | 0.00081787 | 0.0030833 |
| IT     | 58      | 28            | 32       |
| RES    | 3.46813e-16 | 9.51620e-17 | 1.39042e-16 |

When \( \alpha \geq \beta \), the improved SDA-ss is the original SDA-ss with the same convergence rate. If \( \alpha < \beta \), the improved SDA-ss algorithm obviously has better convergence rate and performance than the original SDA-ss algorithm. Therefore, the improved SDA-ss algorithm is obviously better than the original SDA-ss algorithm. We can also see from example 1 that the improved SDA-ss is better than SDA-ss in Table 1. In general the SDA is better than SDA-ss when the two transform is very close in calculation cost. We can see that the improved SDA-ss is better than SDA in example 1. Compared with SDA-ss, the new algorithm effectively reduces the number of iteration of solving process and improves computational accuracy.

Example 2. Here coefficient matrices are

\[ D = \begin{pmatrix} 100002 & -100000 \\ -100000 & 100002 \end{pmatrix}, \quad C = \begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}, \]

\[ A = \begin{pmatrix} 3 & -1 \\ -1 & 3 \end{pmatrix}, \quad B = \begin{pmatrix} -1.5 & -1.5 \\ -1.5 & -1.5 \end{pmatrix}. \]

**Table 2.** Comparison of results of Example 2.

|        | SDA-ss  | Improved SDA-ss | SDA      |
|--------|---------|----------------|----------|
| CPU    | 0.0015496 | 0.00037843 | 0.01326  |
| IT     | 21      | 5             | 20       |
| RES    | 3.48716e-14 | 3.88620e-18 | 7.43148e-17 |

The results of example 2 again show that the improved SDA-ss is better and faster than SDA-ss in Table 2. The new algorithm has fewer iteration steps and smaller residuals than the original algorithm.

### 4. Conclusions

We propose an improved SDA-ss algorithm. Preliminary numerical experiments show that the algorithm is efficient to derive the minimal nonnegative solution of nonsymmetric algebraic Riccati equation with M-matrix. Compared with SDA-ss, the new algorithm effectively reduces the number of iteration of solving process and improves computational accuracy.

### Acknowledgments

The work of B. Tang was supported partly by Hunan Provincial Natural Science Foundation of China (2018JJ3019) and General Foundation of Hunan Educational Department (18C0868). The work was supported by Foundation fund for improving the basic ability of young and middle-aged teachers in colleges and universities in Guangxi (NO.2020KY05016). Supported by Guangxi Key Laboratory of Cryptography and Information Security (No. GCIS201928).
References

[1] Juang J and Lin W W 1999 Nonsymmetric algebraic Riccati equations and Hamiltonian-like matrices, *SIAM J. Matrix Anal. Appl.* 20 pp. 228-243

[2] Bean N G, O’Reilly M M and Taylor P G 2005 Algorithms for return probabilities for stochastic fluid flows, *Stoch. Models* 21 pp. 149-184

[3] Wang W G, Wang W C and Li R C 2012 Alternating-directional doubling algorithm for-Matrix algebraic Riccati equations, *SIAM J. Matrix Anal. Appl.* 33 pp. 170-194

[4] Bai Z Z, Guo X X and Xu S F 2006 Alternatelylinearized implicit iteration methods for the minimal nonnegative solutions of the nonsymmetric algebraic Riccati equations, *Numer. Linear Algebra Appl.* 13 pp. 655-674

[5] Bini D A, Meini B and Poloni F 2010 Transforming algebraic Riccati equations into unilateral quadratic matrix equations, *Numer. Math.* 116 pp. 553-578

[6] Tang B, Huang Y Q and Dong N 2018 Doubling Algorithm for Nonsymmetric Algebraic Riccati Equations Based on a Generalized Transformation, *Advances in Applied Mathematics and Mechanics* 10(6) pp.1327-1343

[7] Bean N G, O'Reilly M M and Taylor P G 2008 Algorithms for the Laplace-Stieltjes transforms of first return times for stochastic fluid flows, *Methodology and Computing in Applied Probability* 10 pp. 381-408