SOLVING LARGE-SCALE NONSYMMETRIC ALGEBRAIC RICCATI EQUATIONS BY DOUBLING

TIEXIANG LI∗, ERIC KING-WAH CHU†, YUEH-CHENG KUO‡, AND WEN-WEI LIN§

Abstract. We consider the solution of the large-scale nonsymmetric algebraic Riccati equation $XCX - XD - AX + B = 0$, with $M = [D, -C; -B, A] \in \mathbb{R}^{(n_1 + n_2) \times (n_1 + n_2)}$ being a nonsingular M-matrix. In addition, $A$ and $D$ are sparse-like, with the products $A^{-1}u$, $A^{-\top}u$, $D^{-1}v$ and $D^{-\top}v$ computable in $O(n)$ complexity (with $n = \max\{n_1, n_2\}$), for some vectors $u$ and $v$, and $B, C$ are low-ranked. The structure-preserving doubling algorithm by Guo, Lin and Xu (2006) is adapted, with the appropriate applications of the Sherman-Morrison-Woodbury formula and the sparse-plus-low-rank representations of various iterates. The resulting large-scale doubling algorithm has an $O(n)$ computational complexity and memory requirement per iteration and converges essentially quadratically. A detailed error analysis, on the effects of truncation of iterates with an explicit forward error bound for the approximate solution from the SDA, and some numerical results will be presented.

Keywords. doubling algorithm, M-matrix, nonsymmetric algebraic Riccati equation, numerically low-ranked solution

AMS subject classifications. 15A24, 65F50

1. Introduction. Consider the nonsymmetric algebraic Riccati equation (NARE)

$$R(X) \equiv XCX - XD - AX + B = 0,$$

(1.1)

where $A$, $B$, $C$ and $D$ are real $n_1 \times n_1$, $n_1 \times n_2$, $n_2 \times n_1$ and $n_2 \times n_2$ matrices, respectively. From the solvability conditions in [12, 13], we assume the matrix

$$M = \begin{bmatrix} D & -C \\ -B & A \end{bmatrix} \in \mathbb{R}^{(n_1 + n_2) \times (n_1 + n_2)}$$

(1.2)

is a nonsingular M-matrix, i.e., $M$ has nonpositive off-diagonal entries and all elements of $M^{-1}$ are nonnegative. In this paper, we are interested in developing an efficient algorithm for solving the minimal nonnegative solution $X$ of NAREs in (1.1).

The structure-preserving doubling algorithm (SDA) in [17] is first proposed for solving the NARE (1.1) with quadratical convergence. Then in [5], more general convergence results were given, especially for the critical case. Later in [4], Bini, Meini, and Poloni developed a doubling algorithm called SDA$_{ss}$, which has shown efficient improvements over SDA in some of numerical tests, but it can happen that sometimes SDA$_{ss}$ runs slower than SDA. Recently in [31], the alternating-directional doubling algorithm (ADDA) has been developed by Wang, Wang and Li to improve the convergence of the SDA dramatically. In practice, ADDA is always faster than SDA and SDA$_{ss}$, however, it may encounter overflow in $F_k$ and $E_k$ before $H_k$ and $G_k$ converge with a desired accuracy, and the scaling technique in [31] is not suitable for the large scale case which we will study.

We state the SDA for solving (1.1) as follows. Choose suitable parameter $\gamma$ such that

$$\gamma \geq \gamma_0 = \max\left\{ \max_{1 \leq i \leq n_1} a_{ii}, \max_{1 \leq i \leq n_2} d_{ii} \right\},$$

(1.3)

∗Department of Mathematics, Southeast University, Nanjing 211189, People's Republic of China; txli@seu.edu.cn (Corresponding Author)
†School of Mathematical Sciences, Building 28, Monash University 3800, Australia; eric.chu@monash.edu
‡Department of Applied Mathematics, National University of Kaohsiung, Kaohsiung 811, Taiwan; yckuo@nuk.edu.tw
§Department of Applied Mathematics, National Chiao Tung University, Hsinchu 300, Taiwan; wulin@math.nctu.edu.tw

1
where $a_{ii}$ and $d_{ii}$ are the diagonal entries of $A$ and $D$, respectively. Compute

$$
F_0 = I_{n_1} - 2\gamma W^{-1}, \quad E_0 = I_{n_2} - 2\gamma V^{-1},
$$

$$
H_0 = 2\gamma W^{-1}BD_\gamma^{-1}, \quad G_0 = 2\gamma D_\gamma^{-1}CW_\gamma^{-1}
$$

(1.4)

with $A_\gamma \equiv A + \gamma I_{n_1}$, $D_\gamma \equiv D + \gamma I_{n_2}$, $W_\gamma \equiv A_\gamma - BD_\gamma^{-1}C$, $V_\gamma \equiv D_\gamma - CA_\gamma^{-1}B$. The SDA [17] has the form (for $k \geq 0$)

$$
F_{k+1} = F_k(I_{n_1} - H_k G_k)^{-1} F_k, \quad E_{k+1} = E_k(I_{n_2} - G_k H_k)^{-1} E_k,
$$

(1.5)

$$
H_{k+1} = H_k + F_k(I_{n_1} - H_k G_k)^{-1} H_k E_k, \quad G_{k+1} = G_k + E_k(I_{n_2} - G_k H_k)^{-1} G_k F_k,
$$

where $F_k \in \mathbb{R}^{n_1 \times n_1}$, $E_k \in \mathbb{R}^{n_2 \times n_2}$, $H_k \in \mathbb{R}^{n_1 \times n_2}$ and $G_k \in \mathbb{R}^{n_2 \times n_1}$.

The total least squares problems with or without symmetric constraints [9], the spectral factors of rational matrix functions [10], the linear and nonlinear optimal controls [2], the conicolic rational matrix functions [20], the structured complex stability radius [18], transport theory [19, 22, 23], the Wiener-Hopf factorization of Markov chains [32], and the optimal solutions of linear differential systems [21]. Symmetric algebraic Riccati equations have been the topic of extensive research, and the theory, applications and numerical solutions of these equations are the subject of [5]–[8] as well as the monographs [21, 29]. The minimal positive solution to the NARE (1.1), for medium size problems without the sparseness and low-ranked assumptions, has

be the dual equation of NARE (1.1). The following convergence theory for (1.5) is originally given in [17] and improved in [5].

**Theorem 1.1.** Let $M$ in (1.2) be a nonsingular M-matrix. Then the NARE (1.1) and its dual equation (1.6) have minimal nonnegative solutions $X \geq 0$ and $Y \geq 0$, respectively. Moreover, $S = A - BY$ and $R = D - CX$ are nonsingular M-matrices. Let $S_\gamma = (S + \gamma I_{n_1})^{-1}(S - \gamma I_{n_1})$ and $R_\gamma = (R + \gamma I_{n_2})^{-1}(R - \gamma I_{n_2})$. Then the sequences $\{F_k\}, \{E_k\}, \{H_k\}$ and $\{G_k\}$ generated by the SDA (1.5) are well-defined, and for all $k \geq 0$, we have

(a) $E_0, F_0 \leq 0$ and $F_k = (I_{n_1} - H_k Y)S_\gamma^{2k} \geq 0$, $E_k = (I_{n_2} - G_k X)R_\gamma^{2k} \geq 0$;

(b) $I - G_k H_k$ and $I - H_k G_k$ are nonsingular M-matrices;

(c) $0 \leq H_k \leq H_{k+1} \leq X$ and $0 \leq X - H_k = (I_{n_1} - H_k Y)S_\gamma^{2k} X R_\gamma^{2k} \leq S_\gamma^{2k} X R_\gamma^{2k}$;

(d) $0 \leq G_k \leq G_{k+1} \leq Y$ and $0 \leq Y - G_k = (I_{n_2} - G_k X)R_\gamma^{2k} Y S_\gamma^{2k} \leq R_\gamma^{2k} Y S_\gamma^{2k}$;

(e) $S_\gamma, R_\gamma < 0$, the spectral radii $\rho(S_\gamma), \rho(R_\gamma) < 1$, and $S_\gamma^{2k}, R_\gamma^{2k} \to 0$ as $k \to \infty$;

(f) $I_{n_1} - XY$ and $I_{n_2} - YX$ are nonsingular M-matrices.

Motivated by the low-ranked cases from the applications in transport theory [19, 22, 23], in this paper we further assume that $n_1$ and $n_2$ are large, $A$ and $D$ are sparse-like (with the products $A^{-1} u, A^{-1} v, D^{-1} v$ and $D^{-1} v$ computable in $O(n)$ complexity, where $n = \max\{n_1, n_2\}$, for some vectors $u$ and $v$), and $B$ and $C$ are of low-rank ($m$ and $l$, respectively, with $m, l \ll n_1, n_2$). In [19, 22, 23], $A$ and $D$ are low-ranked updates of nonsingular diagonal matrices, which are nonsingular but not sparse. We shall adapt the SDA [17] to solve the NARE (1.1), resulting in a large-scale doubling algorithm (SDA$_{ls,c}$) with an $O(n^3)$ computational complexity and memory requirement per iteration. Note that the orthodox SDA in [17] has a computational complexity of $O(n^5)$.

More generally, algebraic Riccati equations arise in many important applications, including the total least squares problems with or without symmetric constraints [9], the spectral factorizations of rational matrix functions [10], the linear and nonlinear optimal controls [2], the conicolic rational matrix functions [20], the structured complex stability radius [18], transport theory [19, 22, 23], the Wiener-Hopf factorization of Markov chains [32], and the optimal solutions of linear differential systems [21]. Symmetric algebraic Riccati equations have been the topic of extensive research, and the theory, applications and numerical solutions of these equations are the subject of [5]–[8] as well as the monographs [21, 29]. The minimal positive solution to the NARE (1.1), for medium size problems without the sparseness and low-ranked assumptions, has
been studied recently by several authors, employing functional iterations, Newton’s method and
the structure-preserving algorithm; see [1, 3, 4], [12]–[17], [22, 23, 26, 27, 30, 31] and the refer-
eses therein. Evidently, the applications associated with and the numerical solution to NAREs
have attracted much attention in the past decade but this paper is the first on general large-scale
NAREs.

Main Contributions. Apart from being the first paper on the numerical solution to gen-
eral large-scale NAREs, we shall formalize the discussion on the numerical rank of the solution X,
showing constructively when X is numerically low-ranked. We adapt the well-known structure-

preserving doubling method efficiently for large-scale NAREs. Then we show how the exponential
growth in the rank of the approximate solution is controlled by compression and truncation. A
first order error estimate will show that the difference between the approximate solution by the
SDA with small truncation and the exact solution has the same order as the truncation without
affecting the convergence of the SDA.

2. Large-Scale Doubling Algorithm. Borrowing from [24], we shall apply the Sherman-
Morrison-Woodbury formula (SMWF) in order to avoid the inversion of large or unstructured
matrices, and use sparse-plus-low-ranked matrices to represent iterates when appropriate. Also,
some matrix operators are computed recursively, to preserve the corresponding sparsity or low-
ranked structures, instead of forming them explicitly. If necessary, we compress and truncate
fast growing components in the iterates, to trade off the negligible amount of accuracy for better
efficiency. Together with the careful organization of convergence control in the algorithm, we
obtain an $O(n)$ computational complexity and memory requirement per iteration.

2.1. Large-Scale SDA. We assume $B \in \mathbb{R}^{n_1 \times n_2}$ and $C \in \mathbb{R}^{n_2 \times n_1}$ in (1.1) have, respectively,
the full low-ranked decompositions

$$B = B_1 B_2^\top, \quad C = C_1 C_2^\top,$$

(2.1)

where $B_1 \in \mathbb{R}^{n_1 \times m}, B_2 \in \mathbb{R}^{n_2 \times m}, C_1 \in \mathbb{R}^{n_2 \times l}, C_2 \in \mathbb{R}^{n_1 \times l}$ with $m, l \ll n \equiv \max\{n_1, n_2\}$. We
first state a basic large-scale SDA, and then propose a practical large-scale SDA later in
Section 2.2.

For the initial matrices in (1.4), we have $F_0 = I_{n_1} - 2\gamma W_{\gamma}^{-1}, E_0 = I_{n_2} - 2\gamma V_{\gamma}^{-1}, H_0 = Q_{10}\Sigma_0 Q_{20}^\top,$
and $G_0 = P_{10}\Gamma_0 P_{20}^\top,$ where

$$Q_{10} = 2\gamma W_{\gamma}^{-1} B_1, \quad Q_{20} = D_{\gamma}^{-1} B_2, \quad \Sigma_0 = \Gamma I_m;$$
$$P_{10} = 2\gamma D_{\gamma}^{-1} C_1, \quad P_{20} = W_{\gamma}^{-1} C_2, \quad \Gamma_0 = \Gamma I_l.$$

(2.2)

Note that efficient linear solvers for the large-scale $A$ and $D$, and thus for $A_{\gamma}$ and $D_{\gamma}$, are available.
Applying the SMWF, $W_{\gamma}^{-1} w$ and $V_{\gamma}^{-1} v$ can be computed economically by

$$W_{\gamma}^{-1} w = \left\{I_{n_1} + A_{\gamma}^{-1} B_1 \left[I_m - (B_2^\top D_{\gamma}^{-1} C_1) (C_2^\top A_{\gamma}^{-1} B_1)\right]^{-1} (B_2^\top D_{\gamma}^{-1} C_1) C_2^\top\right\} A_{\gamma}^{-1} w,$$

(2.3)

$$V_{\gamma}^{-1} v = \left\{I_{n_2} + D_{\gamma}^{-1} C_1 \left[I_l - (C_2^\top A_{\gamma}^{-1} B_1)(B_2^\top D_{\gamma}^{-1} C_1)\right]^{-1} (C_2^\top A_{\gamma}^{-1} B_1) B_2^\top\right\} D_{\gamma}^{-1} v.$$

(2.4)

For $k = 1, 2, \ldots$, we shall organize the SDA so that the iterates have the recursive forms

$$H_k = Q_{1k} \Sigma_k Q_{2k}^\top, \quad G_k = P_{1k} \Gamma_k P_{2k}^\top,$$
$$F_k = F_{k-1}^2 + F_{k-1} F_{2k}^\top, \quad E_k = E_{k-1}^2 + E_{1k} E_{2k}^\top,$$

(2.5)

where $F_{ik} \in \mathbb{R}^{n_1 \times l_{k-1}}, E_{ik} \in \mathbb{R}^{n_2 \times m_{k-1}}$ ($i = 1, 2$), and the kernels $\Sigma_k \in \mathbb{R}^{m_k \times m_k}$ and $\Gamma_k \in \mathbb{R}^{l_k \times l_k}$. 
We should compute products like \( E_k u, E_k^T u, F_k v, F_k^T v \), for some vectors \( u \) and \( v \), by applying (2.6) recursively. Without actually forming \( E_k \) and \( F_k \), we avoid any possible deterioration of their sparse-like properties or other structures as \( k \) grows, and preserve the \( O(n) \) computational complexity of the algorithm. As a trade-off, we need to store all the \( Q_{ik}, \Sigma_k, P_{ik}, \Gamma_k, F_{ik} \) and \( E_{ik} \) for all previous \( k \), as we shall see below.

Applying the SMWF again, we obtain

\[
(I_{n_2} - G_k H_k)^{-1} = I_{n_2} + G_k Q_{1k} \Sigma_k (I_{m_k} - Q_{2k}^+ G_k Q_{1k} \Sigma_k)^{-1} Q_{2k}^+ \\
= I_{n_2} + P_{1k} (I_{i_k} - \Gamma_k P_{2k}^+ H_k P_{1k})^{-1} \Gamma_k P_{2k}^+ H_k, \tag{2.7a}
\]

\[
(I_{n_1} - H_k G_k)^{-1} = I_{n_1} + Q_{1k} (I_{m_k} - \Sigma_k Q_{2k}^+ G_k Q_{1k})^{-1} \Sigma_k Q_{2k}^+ G_k \\
= I_{n_1} + H_k P_{1k} \Gamma_k (I_{i_k} - P_{2k}^+ H_k P_{1k} \Gamma_k)^{-1} P_{2k}^+. \tag{2.7b}
\]

Denote the direct sum of square matrices by \( \oplus \). From (1.5) and (2.7), we can choose the matrices in (2.5) and (2.6) recursively as

\[
Q_{1k+1} = [Q_{1k}, F_k Q_{1k}], \quad Q_{2k+1} = [Q_{2k}, E_k^T Q_{2k}], \quad P_{1k+1} = [P_{1k}, E_k P_{1k}], \quad P_{2k+1} = [P_{2k}, F_k^T P_{2k}]; \tag{2.8}
\]

\[
\Sigma_{k+1} = \Sigma_k \oplus \left[ \Sigma_k + (I_{m_k} - \Sigma_k Q_{2k}^+ G_k Q_{1k})^{-1} \Sigma_k Q_{2k}^+ G_k Q_{1k} \Sigma_k \right] \\
= \Sigma_k \oplus \left[ \Sigma_k + \Sigma_k Q_{2k}^+ P_{1k} \Gamma_k (I_{i_k} - P_{2k}^+ H_k P_{1k} \Gamma_k)^{-1} P_{2k}^+ Q_{1k} \Sigma_k \right] \\
\equiv \Sigma_k \oplus \tilde{\Sigma}_k, \tag{2.9}
\]

\[
\Gamma_{k+1} = \Gamma_k \oplus \left[ \Gamma_k + \Gamma_k P_{2k}^+ Q_{1k} \Sigma_k (I_{m_k} - Q_{2k}^+ G_k Q_{1k} \Sigma_k)^{-1} Q_{2k}^+ P_{1k} \Gamma_k \right] \\
\equiv \Gamma_k \oplus \tilde{\Gamma}_k, \tag{2.10}
\]

\[
F_{1k+1} = F_k H_k P_{1k} \Gamma_k (I_{i_k} - P_{2k}^+ H_k P_{1k} \Gamma_k)^{-1} \\
= F_k Q_{1k} (I_{m_k} - \Sigma_k Q_{2k}^+ G_k Q_{1k})^{-1} \Sigma_k Q_{2k}^+ P_{1k} \Gamma_k, \tag{2.11}
\]

\[
F_{2k+1} = F_k^T P_{2k}; \tag{2.12}
\]

and

\[
E_{1k+1} = E_k G_k Q_{1k} \Sigma_k (I_{m_k} - Q_{2k}^+ G_k Q_{1k} \Sigma_k)^{-1} \\
= E_k P_{1k} (I_{i_k} - \Gamma_k P_{2k}^+ H_k P_{1k})^{-1} \Gamma_k P_{2k}^+ Q_{1k} \Sigma_k, \tag{2.13}
\]

\[
E_{2k+1} = E_k^T Q_{2k}. \tag{2.14}
\]

Ultimately from (2.6), (2.8) and (2.9), we see that the SDA is dominated by the computation of products like \( E_k u, E_k^T u, F_k v, F_k^T v \), for arbitrary vectors \( u \) and \( v \). By applying (2.6) recursively, these products can be computed using (2.3) and (2.4) in \( O(n) \) complexity and memory requirement, because of our assumptions on \( A, B, C \) and \( D \).

The SDA for large-scale NAREs is a competition between the convergence of the doubling iteration and the exponential growth in the dimensions of \( Q_{ik} \) and \( P_{ik} \). From (2.5), (2.8) and (2.9), we have

\[
\text{rank}(H_k) \leq \text{rank}(Q_{ik}) \leq 2^k m, \quad \text{rank}(G_k) \leq \text{rank}(P_{ik}) \leq 2^k l. \tag{2.15}
\]
Note that $2^km$ and $2^kl$ are the numbers of columns in $Q_{ik}$ and $P_{ik}$, respectively. The success of the SDA depends on the trade-off between the accuracy of the approximate solution and its CPU-time and memory requirements, controlled by the compression and truncation of $Q_{ik}$ and $P_{ik}$ in Section 2.2. With the truncation and compression process, rank($Q_{ik}$) and rank($P_{ik}$) will be much reduced even with high accuracy for the approximate solutions $H_k$ and $G_k$.

From the convergence results in Theorem 1.1, as well as (2.8), (2.9) and (2.16), the fact that $X$ and $Y$ are numerically low-ranked can be considered constructively. We next define what we mean by being numerically low-ranked of $X$ (and similarly for $Y$):

**Definition 2.1.** Let $X \in \mathbb{C}^{n \times n}$.

(i) For a given numerical rank tolerance $\tau > 0$, the numerical rank of $X$ with respect to $\tau$, denoted by rank$_\tau(X)$, is defined as the lowest rank of $\tilde{X} \in \mathbb{C}^{n \times n}$ such that $\| \tilde{X} - X \| \leq \tau$.

(ii) $X$ is said to be numerically low-ranked with respect to the numerical rank tolerance $\tau > 0$ if rank$_\tau(X) \ll n$.

We first give a useful lemma which is simple but has not appeared in the literature.

**Lemma 2.1.** For any $A, B \in \mathbb{R}^{n \times r}$, if $0 \leq A \leq B$, then $\| A \| \leq \| B \|$.

**Proof.** Since $0 \leq A \leq B$, we have $0 \leq A^T \leq B^T$ and then $0 \leq A^T A \leq B^T B$. From the Perron-Frobenius Theorem, we have $\| A \| = \sqrt{\rho(A^T A)} \leq \sqrt{\rho(B^T B)} = \| B \|$. \qed

**2.2. Truncation and Compression of $Q_{ik}$ and $P_{ik}$.** The truncation and compression process described in this section is necessary when the convergence of the SDA is slow in comparison with the exponential growth in the dimensions of the iterates $G_k$ and $H_k$. In this situation, the numerical rank of $X$ will be high and we obviously cannot achieve high accuracy in the approximation $H_k$ of $X$ by any method. We then have to compromise its accuracy for the sake of less memory and CPU-time consumption. We should then either choose larger tolerances for the truncation and compression process ($\varepsilon_k$ below), to control the growth in the iterates and adjust them until the accuracy of the approximate solution is acceptable, or simply abandon the truncation and compression process and accept whatever approximate solution obtained within reasonable computing constraints.

We now propose a large-scale SDA with truncation error $\varepsilon$ (SDA$_{ls,\varepsilon}$). For a given sequence of tolerances $\varepsilon = \{\varepsilon_k\}_{k=0}^l$ we first compute truncated initial matrices for the SDA$_{ls,\varepsilon}$. From (2.2) we compute the SVD decompositions

$$Q_{10} = \tilde{Q}_{10} R_{1q}, \quad Q_{20} = \tilde{Q}_{20} R_{2q}, \quad P_{10} = \tilde{P}_{10} R_{1p}, \quad P_{20} = \tilde{P}_{20} R_{2p},$$

where $\tilde{Q}_{10}$, $\tilde{Q}_{20}$, $\tilde{P}_{10}$ and $\tilde{P}_{20}$ are orthogonal and $R_{1q}$, $R_{2q}$, $R_{1p}$ and $R_{2p}$ are upper triangular. Then we compute the SVD decompositions

$$R_{1q} S_0 R_{2q}^T = [U_{10}^T, U_{20}^T](\Sigma_0^T \oplus \Sigma_0^T)[U_{20}^T, U_{10}^T]^T, \quad \| \Sigma_0^T \| < \varepsilon_0;$$

$$R_{1p} \Gamma_0 R_{2p}^T = [V_{10}^T, V_{20}^T](\Gamma_0^T \oplus \Gamma_0^T)[V_{20}^T, V_{10}^T]^T, \quad \| \Gamma_0^T \| < \varepsilon_0;$$

where $[U_{10}^T, U_{20}^T]$ and $[V_{10}^T, V_{20}^T]$ ($i = 1, 2$) are orthogonal, $\Sigma_0^T \oplus \Sigma_0^T$ and $\Gamma_0^T \oplus \Gamma_0^T$ are nonnegative diagonal with $\Sigma_0^T \in \mathbb{R}^{m_0 \times m_0}$ and $\Gamma_0^T \in \mathbb{R}^{l_0 \times l_0}$. By setting

$$Q_{10}^T = \tilde{Q}_{10} U_{10}^T, \quad Q_{20}^T = \tilde{Q}_{20} U_{20}^T, \quad P_{10}^T = \tilde{P}_{10} V_{10}^T, \quad P_{20}^T = \tilde{P}_{20} V_{20}^T,$$  \hspace{1cm} (2.17)

we have the (truncated) initial matrices

$$F_0 = F_0, \quad E_0^T = E_0, \quad H_0^T = Q_{10}^T \Sigma_0^T Q_{20}^T, \quad G_0^T = P_{10}^T \Gamma_0^T P_{20}^T$$  \hspace{1cm} (2.18)

for the SDA$_{ls,\varepsilon}$. Let

$$\Delta H_0 \equiv H_0 - H_0^T = \tilde{Q}_{10} U_{10}^T \Sigma_0^T U_{20}^T \tilde{Q}_{20}^T, \quad \Delta G_0 \equiv G_0 - G_0^T = \tilde{P}_{10} V_{10}^T \Gamma_0^T V_{20}^T \tilde{P}_{20}^T.$$  \hspace{1cm} (2.19)
The truncation errors of the initial matrices can be estimated by

\[ \|\Delta H_0\| = \|\Sigma_0\| < \varepsilon_0, \quad \|\Delta G_0\| = \|\Gamma_0\| < \varepsilon_0. \quad (2.20) \]

We repeat this process and suppose it holds at the \( k \) step that

\[
\begin{align*}
H_k & = Q_{ik}^r \Sigma_k^r Q_{2k}^r, \\
G_k & = P_{ik}^r \Gamma_k^r P_{2k}^r; \\
E_k^r & = E_{k-1}^r + E_{1k}^r E_{2k}^r, \\
F_k^r & = F_{k-1}^r + F_{1k}^r F_{2k}^r;
\end{align*}
\]

(2.21)

where \( Q_{ik}^r \) and \( P_{ik}^r \) are orthogonal with widths being \( m_k \) and \( l_k \) (\( i = 1, 2 \), respectively).

To estimate the \((k + 1)\)th truncation error, from (2.10)–(2.15) as well as (2.21), we compute

\[
\begin{align*}
\hat{\Sigma}_{k+1}^r & = \Sigma_k^r + \Sigma_k^r Q_{2k}^r P_{1k}^r \Gamma_k^r (I_k - P_{2k}^r H_k^r P_{1k}^r)^{-1} P_{2k}^r Q_{1k}^r \Sigma_k^r, \\
\hat{\Gamma}_{k+1}^r & = \Gamma_k^r + (I_k - \Gamma_k^r P_{2k}^r H_k^r P_{1k}^r)^{-1} \Gamma_k^r P_{2k}^r H_k^r P_{1k}^r \Gamma_k^r;
\end{align*}
\]

(2.22)

and

\[
\begin{align*}
F_{1,k+1}^r & = F_k^r H_k^r P_{1k}^r \Gamma_k^r (I_k - P_{2k}^r H_k^r P_{1k}^r)^{-1}, \\
F_{2,k+1}^r & = F_k^r P_{2k}^r; \\
E_{1,k+1}^r & = E_k^r P_{1k}^r (I_k - \Gamma_k^r P_{2k}^r H_k^r P_{1k}^r)^{-1} \Gamma_k^r P_{2k}^r Q_{1k}^r \Sigma_k^r, \\
E_{2,k+1}^r & = E_k^r Q_{2k}^r.
\end{align*}
\]

(2.23)

From the QR decompositions

\[
\begin{align*}
[Q_{1k}^r, F_{1k}^r Q_{1k}^r] & = [Q_{1k}^r, \hat{Q}_{1k}^r] \begin{bmatrix} I & S_{1q} \\ 0 & R_{1q} \end{bmatrix}_{k}, \\
[Q_{2k}^r, E_{1k}^r Q_{2k}^r] & = [Q_{2k}^r, \hat{Q}_{2k}^r] \begin{bmatrix} I & S_{2q} \\ 0 & R_{2q} \end{bmatrix}_{k},
\end{align*}
\]

\[
\begin{align*}
[P_{1k}^r, E_{1k}^r P_{1k}^r] & = [P_{1k}^r, \hat{P}_{1k}^r] \begin{bmatrix} I & S_{1p} \\ 0 & R_{1p} \end{bmatrix}_{k}, \\
[P_{2k}^r, F_{1k}^r P_{2k}^r] & = [P_{2k}^r, \hat{P}_{2k}^r] \begin{bmatrix} I & S_{2p} \\ 0 & R_{2p} \end{bmatrix}_{k},
\end{align*}
\]

(2.24)

we set

\[
\hat{\Sigma}_{k+1}^r = \begin{bmatrix} I & S_{1q} \\ 0 & R_{1q} \end{bmatrix}_{k}, \quad \hat{\Gamma}_{k+1}^r = \begin{bmatrix} I & S_{1p} \\ 0 & R_{1p} \end{bmatrix}_{k}. \quad (2.25)
\]

We next compute the SVDs

\[
\hat{\Sigma}_{k+1}^r = \begin{bmatrix} U_{1,k+1}^r & U_{2,k+1}^r \end{bmatrix} \begin{bmatrix} \Sigma_{k+1}^r & 0 \\ 0 & \Sigma_{k+1}^r \end{bmatrix} \begin{bmatrix} U_{2,k+1}^r & U_{2,k+1}^r \end{bmatrix}^\top,
\]

(2.26)

\[
\hat{\Gamma}_{k+1}^r = \begin{bmatrix} V_{1,k+1}^r & V_{2,k+1}^r \end{bmatrix} \begin{bmatrix} \Gamma_{k+1}^r & 0 \\ 0 & \Gamma_{k+1}^r \end{bmatrix} \begin{bmatrix} V_{2,k+1}^r & V_{2,k+1}^r \end{bmatrix}^\top
\]

with \( \|\Sigma_{k+1}^r\| < \varepsilon_{k+1} \) and \( \|\Gamma_{k+1}^r\| < \varepsilon_{k+1} \). To truncate, we compute

\[
\begin{align*}
Q_{1,k+1}^r & = [Q_{1k}^r, \hat{Q}_{1k}^r] U_{1,k+1}^r \in \mathbb{R}^{n_k \times m_{k+1}}, \\
Q_{2,k+1}^r & = [Q_{2k}^r, \hat{Q}_{2k}^r] U_{2,k+1}^r \in \mathbb{R}^{n_k \times m_{k+1}}, \\
P_{1,k+1}^r & = [P_{1k}^r, \hat{P}_{1k}^r] V_{1,k+1}^r \in \mathbb{R}^{n_k \times l_{k+1}}, \\
P_{2,k+1}^r & = [P_{2k}^r, \hat{P}_{2k}^r] V_{2,k+1}^r \in \mathbb{R}^{n_k \times l_{k+1}}.
\end{align*}
\]

(2.27)

Let

\[
\hat{H}_{k+1}^r = H_k^r + F_k^r (I - H_k^r G_k^r)^{-1} H_k^r E_k^r, \quad \hat{G}_{k+1}^r = G_k^r + E_k^r (I - G_k^r H_k^r)^{-1} G_k^r F_k^r.
\]

(2.28)

We define the local truncation errors of \( H_{k+1}^r \) and \( G_{k+1}^r \) as

\[
\delta H_{k+1}^r \equiv \hat{H}_{k+1}^r - H_{k+1}^r, \quad \delta G_{k+1}^r \equiv \hat{G}_{k+1}^r - G_{k+1}^r.
\]

(2.29)
From (2.26), we see that
\[ \| \delta H_{k+1} \| = \left\| Q_{1k}^\tau \sqrt[2]{\tau} L_{k+1}^\tau \Sigma_k^\tau U_{2,k+1}^\tau \left[ Q_{2k}^\tau \sqrt[2]{\tau} \bar{Q}_{2k} \right]^\tau \right\| = \| \Sigma_k^\tau \| < \varepsilon_{k+1}, \] (2.30)
\[ \| \delta G_{k+1} \| = \left\| P_{1k}^\tau \left[ V_{1,k+1}^\tau \Gamma_{1,k+1}^\tau V_{2,k+1}^\tau \left[ P_{2k}^\tau \sqrt[2]{\tau} \bar{P}_{2k} \right]^\tau \right] \right\| = \| \Gamma_k^\tau \| < \varepsilon_{k+1}. \] (2.31)
Moreover, we define the global truncation errors of \( H_{k+1} \) and \( G_{k+1} \) by
\[ \Delta H_{k+1} \equiv H_{k+1} - H_{k+1}^\tau, \quad \Delta G_{k+1} \equiv G_{k+1} - G_{k+1}^\tau, \] (2.32)
which will be estimated in Section 3.

The SDA,ls,\( \varepsilon \) for solving large-scale NAREs realizes the iterations in (1.5) with initial matrices in (2.18), and the help of (2.3), (2.4), (2.21)–(2.27).

**Algorithm 1 (SDA,ls,\( \varepsilon \))**

**Input:**
- \( A \in \mathbb{R}^{n_1 \times n_1} \), \( B \in \mathbb{R}^{n_1 \times n_2} \), \( C \in \mathbb{R}^{n_2 \times n_1} \), \( D \in \mathbb{R}^{n_2 \times n_2} \) with \( B = B_1 B_2^\tau \) and \( C = C_1 C_2^\tau \) being full low-ranked as in (2.1); suitable shift \( \gamma \) as in (1.3);
- truncation tolerances \( \varepsilon = \{ \varepsilon_k \}_{k=0}^{\infty} \) and convergence tolerance \( \varepsilon_c > 0 \).

**Output:**
- \( H_k = Q_{1k}^\tau \Gamma_k^\tau Q_{2k}^\tau \) and \( G_k = P_{1k}^\tau \Gamma_k^\tau P_{2k}^\tau \) with \( Q_{1k} \in \mathbb{R}^{n_1 \times m_k} \), \( P_{1k} \in \mathbb{R}^{n_1 \times l_k} \) orthogonal (\( i = 1, 2; j = 1, 2, j_2 = 1 \)), approximating the solutions \( X \) and \( Y \) to the large-scale NARE (1.1) and its dual equation (1.6), respectively.

**Initial matrices:**
- Set \( k = 0 \);
- Compute \( Q_{10}, P_{10} \) (\( i = 1, 2 \)) in (2.2);
- Compute \( Q_{10}^\tau P_{10}^\tau \) (\( i = 1, 2 \)) in (2.17) with truncation tolerance \( \varepsilon_0 \);
- Do until convergence:
  - Compute \( \Sigma_k^\tau, \Gamma_k^\tau \) as in (2.22) and \( F_{i,k+1}^\tau, E_{i,k+1}^\tau \) (\( i = 1, 2 \)) as in (2.23);
  - Orthogonalize \( F_{i,k}^\tau Q_{1k}^\tau, E_{i,k}^\tau Q_{2k}^\tau, E_{i,k}^\tau P_{1k}^\tau \) and \( F_{i,k}^\tau P_{2k}^\tau \) as in (2.24);
  - Compute \( Q_{i,k+1}, P_{i,k+1} \) (\( i = 1, 2 \)) using the tolerance \( \varepsilon_{k+1} \) as in (2.27);
  - Compute \( k \leftarrow k + 1, d_k = \max \{ \| dH_k^\tau \|, \| dG_k^\tau \| \} \) and \( r_k = \| R(H_k^\tau) \| \);
  - (An economic way for computing \( \| dH_k^\tau \|, \| dG_k^\tau \| \) and \( \| R(H_k^\tau) \| \) can be found in (4.1), (4.2) and (4.3) in Section 4.1.)
- If \( d_k < \varepsilon_c \), Set \( k = k \); Stop; End If;

2.3. SDA and Krylov Subspaces. There is an interesting relationship between the SDA and Krylov subspaces. Define the Krylov subspaces
\[ K_k(A, V) \equiv \begin{cases} \text{span}\{V\} & (k = 0), \\ \text{span}\{V, AV, A^2V, \ldots, A^{k-1}V\} & (k > 0). \end{cases} \]
From (1.4), (2.2)–(2.4) and (2.8), we see that
\[ Q_{10} = 2 \gamma W_{-1}^\tau B_1 \subseteq K_0(A_{-1}, A_{-1}^\tau B_1), \quad Q_{20} = D_{-1}^\tau B_2 \subseteq K_0(D_{-1}^\tau, D_{-1}^\tau B_2); \]
\[ Q_{11} = [Q_{10}, B_0 Q_{10}] \subseteq K_1(A_{-1}, A_{-1}^\tau B_1), \quad Q_{21} = [Q_{20}, E_0^\tau Q_{20}] \subseteq K_1(D_{-1}^\tau, D_{-1}^\tau B_2). \]
(We have abused notations, with \( V \subseteq K_k(A, B) \) meaning \( \text{span}\{V\} \subseteq K_k(A, B) \).) Similarly, it is easy to show that
\[ Q_{1k} \subseteq K_k(A_{-1}, A_{-1}^\tau B_1), \quad Q_{2k} \subseteq K_k(D_{-1}^\tau, D_{-1}^\tau B_2); \]
\[ P_{1k} \subseteq K_k(D_{-1}^\tau, D_{-1}^\tau C_1), \quad P_{2k} \subseteq K_k(A_{-1}, A_{-1}^\tau C_2). \]
In other words, the general SDA is closely related to approximating the solutions $X$ and $Y$ using Krylov subspaces, with additional components diminishing quadratically. However, for problems of moderate size $n$, $Q_{ik}$ and $P_{ik}$ become full-ranked after a few iterations.

The link between the SDA and the Krylov subspaces defined above is important in explaining the fast convergence of the SDA. We used to believe the convergence of the SDA came from the link between the SDA and the Krylov subspaces, with additional components diminishing quadratically. However, for problems of moderate size $n$, $Q_{ik}$ and $P_{ik}$ become full-ranked after a few iterations.

From (2.19) and (2.20), we have

$$H_k - H_{k-1} \leq F_{k-1} \|I_{n_1} - H_{k-1}G_{k-1}\|^{-1}H_{k-1} \|E_{k-1}\|,$$

$$G_k - G_{k-1} \leq F_{k-1} \|I_{n_2} - G_{k-1}H_{k-1}\|^{-1}G_{k-1} \|F_{k-1}\|,$$

and the fact that $\|E_{k-1}\|, \|F_{k-1}\| \rightarrow 0$ quadratically, as $k \rightarrow \infty$. This is consistent with numerical results from examples associated with $M$ in (1.2) which is barely a nonsingular M-matrix, where the corresponding $E_k, F_k \rightarrow 0$ slowly but the overall convergence for $H_k$ and $G_k$ are much faster.

3. Truncation Error Estimates. In this section, we shall estimate the global truncation errors defined in (2.32). For simplicity, we derive only the first order error bounds.

From Theorem 1.1, we have $0 \leq H_k \leq X$, $0 \leq G_k \leq Y$, and

$$0 \leq (I - G_kH_k)^{-1} = I + G_kH_k + (G_kH_k)^2 + \cdots \leq I + YX + (YX)^2 + \cdots = (I - YX)^{-1}$$

and $0 \leq F_k = (I_{n_1} - H_kY)S^2_{\gamma} \leq S^2_{\gamma}$. By Lemma 2.1, we have

$$\|H_k\| \leq \|X\|, \|G_k\| \leq \|Y\|, \quad (3.1)$$

and

$$\|I - G_kH_k\|^{-1} \leq \|I - YX\|^{-1} \equiv \beta_1, \quad \|F_k\| \leq \|S^2_{\gamma}\| \rightarrow 0. \quad (3.2)$$

Similarly, from Theorem 1.1 and Lemma 2.1 we also have

$$\|I - H_kG_k\|^{-1} \leq \|I - XY\|^{-1} \equiv \beta_2, \quad \|E_k\| \leq \|R^2_{\gamma}\| \rightarrow 0. \quad (3.3)$$

Denote

$$\rho_k = \max\{\|R^2_{\gamma}\|, \|S^2_{\gamma}\|\}, \quad \alpha = \max\{\|X\|, \|Y\|\}, \quad \beta = \max\{\beta_1, \beta_2\}. \quad (3.4)$$

In the following we abuse the notation ”=“ and ”\equiv “, ignoring the higher order terms. Suppose that $\rho(H_k^*G_k^*) < 1, \|\Delta H_k\|$ and $\|\Delta G_k\|$ are sufficiently small. From (2.32) we have the first order approximation of $(I - H_k^*G_k^*)^{-1}$:

$$(I - H_k^*G_k^*)^{-1} = [I - (H_k - \Delta H_k)(G_k - \Delta G_k)]^{-1}$$

$$= [I - H_kG_k + \Delta H_kG_k + H_k\Delta G_k - \Delta H_k\Delta G_k]^{-1}$$

$$= (I - H_kG_k)^{-1} - (I - H_kG_k)^{-1}(\Delta H_kG_k + H_k\Delta G_k)(I - H_kG_k)^{-1}. \quad (3.5)$$

From (2.19) and (2.20), we have $H_0^* = H_0 - \Delta H_0$, $G_0^* = G_0 - \Delta G_0$ with $\|\Delta H_0\|, \|\Delta G_0\| < \varepsilon_0$. Since $E_0^* = E_0$ and $F_0^* = F_0$, (2.28) implies

$$\hat{H}_1 = H_0^* + F_0^*(I - H_0^*G_0^*)^{-1}H_0^*E_0^*$$

$$= H_0 - \Delta H_0$$

$$+ F_0 \left[ (I - H_0G_0)^{-1} - (I - H_0G_0)^{-1}(\Delta H_0G_0 + H_0\Delta G_0)(I - H_0G_0)^{-1} \right] (H_0 - \Delta H_0)E_0$$

$$= H_0 + F_0(I - H_0G_0)^{-1}H_0E_0 - \Delta H_0 - F_0(I - H_0G_0)^{-1}\Delta H_0E_0$$

$$- F_0(I - H_0G_0)^{-1}(\Delta H_0G_0 + H_0\Delta G_0)(I - H_0G_0)^{-1}H_0E_0$$

$$\equiv H_1 - \delta H_1, \quad (3.5)$$
where \( \hat{\delta}H_1 \) is the first order truncation error given by

\[
\hat{\delta}H_1 = \Delta H_0 + F_0(I - H_0G_0)^{-1}\Delta H_0E_0 + F_0(I - H_0G_0)^{-1}(\Delta H_0G_0 + H_0\Delta G_0)(I - H_0G_0)^{-1}H_0E_0.
\]

From (2.29), (2.32) and (3.5), it follows that \( \Delta H_1 = H_1 - H_1^* = \hat{\delta}H_1 + \delta H_1 \). By (2.30) and (3.1)–(3.4), we have

\[
\|\Delta H_1\| \leq \|\delta H_1\| + \|\hat{\delta}H_1\| \\
\leq \varepsilon_1 + \|\Delta H_0\| + \|F_0\|\|E_0\|\|G_0\|\|\Delta H_0\| \\
+ \|F_0\|\|E_0\|\|H_0\|\|\Delta H_0\|\|\Delta G_0\| + \|H_0\|\|\Delta G_0\| \\
\leq \varepsilon_1 + (1 + \rho_0^2\beta + \rho_0^2\alpha\beta^2)\|\Delta H_0\| + \rho_0\alpha^2\beta^2\|\Delta G_0\|.
\]

Similarly, we have

\[
\|\Delta G_1\| \leq \varepsilon_1 + (1 + \rho_0^2\beta + \rho_0^2\alpha^2\beta^2)\|\Delta G_0\| + \rho_0\alpha^2\beta^2\|\Delta H_0\|.
\]

From (2.21), we have

\[
F_1^* = F_0^*(I - H_0^*G_0^*)^{-1}F_0^* = F_0(I - H_0^*G_0^*)^{-1}F_0 \\
= F_0(I - H_0G_0)^{-1}F_0 - F_0(I - H_0G_0)^{-1}(\Delta H_0G_0 + H_0\Delta G_0)(I - H_0G_0)^{-1}F_0 \\
\equiv F_1 - \Delta F_1,
\]

where \( \Delta F_1 = F_0(I - H_0G_0)^{-1}(\Delta H_0G_0 + H_0\Delta G_0)(I - H_0G_0)^{-1}F_0 \) is the first order truncation error and satisfies

\[
\|\Delta F_1\| \leq \|F_0\|^2\|(I - H_0G_0)^{-1}\|\|\Delta H_0\|\|G_0\| + \|H_0\|\|\Delta G_0\| \\
\leq \rho_0\alpha\beta^2(\|\Delta H_0\| + \|\Delta G_0\|).
\]

Similarly, we also have

\[
\|\Delta E_1\| \leq \rho_0\alpha\beta^2(\|\Delta H_0\| + \|\Delta G_0\|).
\]

Performing the \((k+1)\)th step in the SDA-Ls-\(\varepsilon\) algorithm, we obtain

\[
\tilde{H}_{k+1} = H_1^* + F_k^*(I - H_k^*G_k^*)^{-1}H_k^*E_k^* \\
= H_k - \Delta H_k + (F_k - \Delta F_k)(I - H_kG_k)^{-1}(H_k - \Delta H_k)(E_k - \Delta E_k) \\
- (F_k - \Delta F_k)(I - H_kG_k)^{-1}(\Delta H_kG_k + H_k\Delta G_k)(I - H_kG_k)^{-1}(H_k - \Delta H_k)(E_k - \Delta E_k) \\
= H_k + F_k(I - H_kG_k)^{-1}H_kE_k - \Delta H_k - F_k(I - H_kG_k)^{-1}(H_k\Delta E_k + \Delta H_kE_k) \\
- \Delta F_k(I - H_kG_k)^{-1}H_kE_k - F_k(I - H_kG_k)^{-1}(\Delta H_kG_k + H_k\Delta G_k)(I - H_kG_k)^{-1}H_kE_k \\
\equiv H_{k+1} - \hat{\delta}H_{k+1},
\]

where

\[
\hat{\delta}H_{k+1} = \Delta H_k + F_k(I - H_kG_k)^{-1}(H_k\Delta E_k + \Delta H_kE_k) + \Delta F_k(I - H_kG_k)^{-1}H_kE_k \\
+ F_k(I - H_kG_k)^{-1}(\Delta H_kG_k + H_k\Delta G_k)(I - H_kG_k)^{-1}H_kE_k
\]

is the first order truncation error. Then (2.30)–(2.32) and (3.1)–(3.4) imply

\[
\|\Delta H_{k+1}\| \leq \|\delta H_{k+1}\| + \|\hat{\delta}H_{k+1}\| \\
\leq \|\delta H_{k+1}\| + \|\Delta H_{k+1}\| + \|\Delta F_k\| + \|\Delta E_k\| + \|\Delta G_k\| \\
+ \|\Delta H_{k+1}\| + \|\Delta F_k\| + \|\Delta G_k\| \\
\leq \varepsilon_{k+1} + (1 + \rho_k^2\beta + \rho_k^2\alpha^2\beta^2)\|\Delta H_k\| \\
+ \rho_k\alpha\beta(\|\Delta E_k\| + \|\Delta F_k\|).
\]
Similarly, we also have
\[ \| \Delta G_{k+1} \| \leq \varepsilon_{k+1} + (1 + \rho^2_2 \beta + \rho^2_k \alpha^2 \beta^2) \| \Delta G_k \| + \rho^2_k \alpha \beta \| \Delta H_k \| + \rho_k \alpha \beta (\| \Delta F_k \| + \| \Delta E_k \|) \]  
(3.7)

From (2.21), it holds that
\[ F^*_k = (I - H_k G_k)^{-1} F_k = (F_k - \Delta F_k)(I - H_k G_k)^{-1}(F_k - \Delta F_k) 
= F_k(I - H_k G_k)^{-1} F_k - F_k(I - H_k G_k)^{-1} \Delta F_k - \Delta F_k(I - H_k G_k)^{-1} F_k 
- \Delta F_k(I - H_k G_k)^{-1}(\Delta H_k G_k + H_k \Delta G_k)(I - H_k G_k)^{-1} F_k 
= F_{k+1} - \Delta F_{k+1}, \]

where
\[ \Delta F_{k+1} = F_k(I - H_k G_k)^{-1}(\Delta F_k + \Delta F_k)(I - H_k G_k)^{-1} F_k 
+ F_k(I - H_k G_k)^{-1}(\Delta H_k G_k + H_k \Delta G_k)(I - H_k G_k)^{-1} F_k \]
is the first order truncation error and satisfies
\[ \| \Delta F_{k+1} \| \leq 2\| F_k \|\| (I - H_k G_k)^{-1} \| \| \Delta F_k \| + \| F_k \|^2\| (I - H_k G_k)^{-1} \|^2 (\| \Delta H_k \| \| G_k \| + H_k \| \Delta G_k \|) \]
\[ \leq 2\rho_k \| \Delta F_k \| + \rho^2_k \alpha \beta \| \| \Delta H_k \| + \| \Delta G_k \| \]  
(3.8)

Similarly, we also have
\[ \| \Delta E_{k+1} \| \leq 2\rho_k \| \Delta E_k \| + \rho^2_k \alpha \beta \| \| \Delta H_k \| + \| \Delta G_k \| \]  
(3.9)

Assemble (3.6)–(3.9) in matrix form, we have
\[ \begin{bmatrix} \| \Delta H_{k+1} \| \\
\| \Delta G_{k+1} \| \\
\| \Delta F_{k+1} \| \\
\| \Delta E_{k+1} \| 
\end{bmatrix} \leq \begin{bmatrix} 1 + \rho^2_2 \beta + \rho^2_k \alpha^2 \beta^2 & \rho^2_k \alpha \beta & \rho_k \alpha \beta & \rho_k \alpha \beta \\
\rho^2_k \alpha^2 \beta^2 & 1 + \rho^2_2 \beta + \rho^2_k \alpha^2 \beta^2 & \rho_k \alpha \beta & \rho_k \alpha \beta \\
\rho^2_2 \alpha \beta^2 & \rho^2_k \alpha \beta^2 & 2\rho_k \beta & 0 \\
\rho^2_2 \alpha \beta^2 & \rho^2_k \alpha \beta^2 & 0 & 2\rho_k \beta 
\end{bmatrix} \begin{bmatrix} \| \Delta H_k \| \\
\| \Delta G_k \| \\
\| \Delta F_k \| \\
\| \Delta E_k \| 
\end{bmatrix} + \begin{bmatrix} \varepsilon_{k+1} \\
\varepsilon_{k+1} \\
0 \\
0 
\end{bmatrix} \]
\[ = \Psi_k \begin{bmatrix} \| \Delta H_k \|, \| \Delta G_k \|, \| \Delta F_k \|, \| \Delta E_k \| \end{bmatrix}^T + \begin{bmatrix} \varepsilon_{k+1}, \varepsilon_{k+1}, 0, 0 \end{bmatrix}^T. \]
(3.10)

Substituting $\Psi_k$ in (3.10) recursively, the error bound can be estimated by
\[ \begin{bmatrix} \| \Delta H_{k+1} \| \\
\| \Delta G_{k+1} \| \\
\| \Delta F_{k+1} \| \\
\| \Delta E_{k+1} \| 
\end{bmatrix} \leq \sum_{i=1}^{k+1} \left( \prod_{j=0}^{i-1} \Psi_{k-j} \begin{bmatrix} \varepsilon_{k-i+1} \\
\varepsilon_{k-i+1} \\
0 \\
0 
\end{bmatrix} \right) + \begin{bmatrix} \varepsilon_{k+1} \\
\varepsilon_{k+1} \\
0 \\
0 
\end{bmatrix}. \]
(3.11)

In the following theorem we claim that the first order forward error bounds of $H_k^T$, $G_k^T$, and the first order truncation errors of $F_k^*$ and $E_k^*$, which only depend on $\rho_k$ and the tolerance $\varepsilon = \{ \varepsilon_k \}_{k=0}^k$.

**Theorem 3.1.** Let $X$ and $Y$ be the minimal nonnegative solutions of NARE (1.1) and its dual equation (1.6), respectively. For given tolerances $\varepsilon = \{ \varepsilon_k \}_{k=0}^k$, suppose $\{ H_k^T, G_k^T, F_k^*, E_k^* \}_{k=0}^k$ is the sequence generated by the SDA$_{ls,\varepsilon}$ satisfying $\rho(H_k^T G_k^T) < 1$ for all $k$. Then we have
\[ \| H_k^T - X \|, \| G_k^T - Y \| \leq \varepsilon_k + \frac{1}{2} \sum_{i=1}^{k} \left[ 1 + \prod_{j=1}^{i} (1 + \eta_{k-j}) \right] \varepsilon_{k-i} + \rho^2_k \alpha, \]
(3.12)
and
\[ \|F_k^T - F_k\|, \|E_k^T - E_k\| \leq \frac{1}{2} \sum_{i=1}^{k} \left[ 1 + \prod_{j=1}^{i} (1 + \eta_{k-j}) \right] \varepsilon_{k-i} \tag{3.13} \]
for \( k = 0, 1, \cdots, k \), where \( \rho_k \) is given by (3.4) and \( \eta_k \) is defined by
\[ \eta_k = 4 \max \{ \rho_k^2 \beta + \rho_k^2 \alpha^2 \beta^2, \rho_k \alpha \beta, \rho_k^2 \alpha^2 \beta^2, 2 \rho_k \beta \}. \tag{3.14} \]

**Proof.** For convenience, we let
\[ \hat{\Psi}_k = \text{diag}(1, 1, 0, 0) + \frac{1}{4} \eta_k e e^\top \equiv J_0 + \eta_k J_1, \]
where \( e = [1, 1, 1, 1]^\top \) and \( \eta_k \) is given in (3.14). Then (3.11) can be simplified to
\[ \begin{bmatrix} \|\Delta H_{k+1}\| \\ \|\Delta G_{k+1}\| \\ \|\Delta F_{k+1}\| \\ \|\Delta E_{k+1}\| \end{bmatrix} \leq \sum_{i=1}^{k+1} \prod_{j=0}^{i-1} \hat{\Psi}_{k-j} \begin{bmatrix} \varepsilon_{k-i+1} \\ \varepsilon_{k-i+1} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} \varepsilon_{k+1} \\ \varepsilon_{k+1} \\ 0 \\ 0 \end{bmatrix}. \tag{3.15} \]

It is easily seen that
\[ J_0^2 = J_0, \quad J_1^2 = J_1, \quad J_0 J_1 J_0 \leq J_1, \quad J_1 J_0 J_1 \leq J_1. \tag{3.16} \]
Let \( C(J_0^s, J_1^t) \) denote the product of \( s \)'s \( J_0 \) and \( t \)'s \( J_1 \) in any order. From (3.16), it follows that
\[ C(J_0^s, J_1^t) \leq \begin{cases} J_1, & \text{if } t \geq 1, \\ J_0, & \text{if } t = 0. \end{cases} \tag{3.17} \]

By (3.17), the products in (3.15) can be bounded by
\[ \prod_{j=0}^{i-1} \hat{\Psi}_{k-j} = \hat{\Psi}_k \cdots \hat{\Psi}_{k-i+1} \leq J_0 + \sum_{i=1}^{k+1} \prod_{j=0}^{i-1} \left( \sum_{k_{r_1} > \cdots > k_{r_i} \geq k-i+1} \eta_{r_1} \cdots \eta_{r_i} \right) J_1. \tag{3.18} \]
Post-multiplying (3.18) by \( [\varepsilon_{k-i+1}, \varepsilon_{k-i+1}, 0, 0]^\top \) and substituting the result into (3.15), we obtain the first order upper bounds
\[ \|\Delta H_{k+1}\| \leq \varepsilon_{k+1} + \sum_{i=1}^{k+1} \left[ 1 + \frac{1}{2} \sum_{t=1}^{i} \prod_{j=0}^{t-1} (1 + \eta_{k-j}) \right] \varepsilon_{k-i+1}, \tag{3.19} \]
\[ \|\Delta G_{k+1}\| \leq \frac{1}{2} \sum_{i=1}^{k+1} \left[ 1 + \prod_{j=0}^{i-1} (1 + \eta_{k-j}) \right] \varepsilon_{k-i+1}, \tag{3.20} \]
which also hold for \( \|\Delta G_{k+1}\| \) and \( \|\Delta E_{k+1}\| \), respectively. By Theorem 1.1 and (3.4), we have
\[ \|H_k - X\| \leq \rho_k^2 \alpha \quad \text{and} \quad \|G_k - Y\| \leq \rho_k^2 \alpha. \]
Since \( \|H_k^T - X\| \leq \|H_k^T - H_k\| + \|H_k - X\| \) and \( \|G_k^T - Y\| \leq \|G_k^T - G_k\| + \|G_k - Y\| \), it follows from (3.19) and (3.20), by setting \( k \leftarrow k + 1 \), we prove the assertions in (3.12) and (3.13). \( \square \)

**Remark 3.1.**
(a) To obtain an approximate solution $H_k^T$ from \{\(Q_{1k}^T, \Sigma_k^T, Q_{2k}^T\)\} generated by the SDA, ls-\(k\) algorithm will be the most expensive step. Specifically, we need a post process for the computation of $H_k^T = Q_{1k}^T \Sigma_k^T Q_{2k}^T$ which require $O(n^2)$ flops and $n^2$ memory. Furthermore, the computed $H_k^T$ is no longer nonnegative. If a nonnegative solution is required, it is suggested to set $H_k^T := (H_k^T + |H_k^T|)/2$. Since $X > 0$, it is easily seen that the forward error of $H_k^T$, $\|H_k^T - X\|$, can be estimated by the upper bound of (3.12) in Theorem 3.1.

(b) For $k = \check{k}$ in (3.12), we see that the coefficients $c_i \equiv 1 + \prod_{j=1}^{k-i}(1 + \eta_{k-j})$ of $\varepsilon_i$ are decreasing, for $i = 1, \ldots, \check{k}$. It is reasonable to choose the tolerance sequence $\{\varepsilon_i\}_{i=1}^{\check{k}}$ as an increasing sequence. However, in general, it is hard to estimate those coefficients of $\varepsilon_i$ beforehand. Therefore, in practice, we suggest to choose a constant sequence of tolerances $\{\varepsilon_i\}_{i=1}^{\check{k}}$ (e.g., $\varepsilon_i = 10^{-3}, 10^{-4}, \ldots, 10^{-15}$). From our numerical experiments, the forward errors of $H_k^T$ and $G_k^T$ almost have the same order of the given truncation tolerance $\varepsilon_r$.


4.1. Residual and Convergence Control. In the SDA, ls-\(k\), we should compute residuals and differences of iterates carefully in $O(n)$ complexity.

From (3.12), consider the difference of successive iterates

$$dH_{k+1}^T = H_{k+1}^T - H_k^T = Q_{1,k+1}^T \Sigma_{k+1}^T Q_{2,k+1}^T - Q_{1k}^T \Sigma_k^T Q_{2k}^T$$

$$= \begin{bmatrix} Q_{1,k+1}^T & (\Sigma_{k+1}^T \oplus \Sigma_k^T) \left[ Q_{2,k+1}^T - Q_{2k}^T \right] \\ \Sigma_k^T & 0 \end{bmatrix} = Y_{1k}(\Sigma_{k+1}^T \oplus \Sigma_k^T) Y_{2k}^T,$$

where $Q_{1k}$, $Q_{1,k+1}$ ($i = 1, 2$) are orthogonal. We compute $\|dH_{k+1}^T\|$ efficiently as follows.

$$\|dH_{k+1}^T\| = \|R_{1k}^T(\Sigma_{k+1}^T \oplus \Sigma_k^T) R_{2k}^T\|,$$  \hspace{1cm} (4.1)

where $Y_{1k} = W_{1k}^R R_{1k}^h$ and $Y_{2k} = W_{2k}^R R_{2k}^h$ are the QR decompositions of $Y_{1k}$ and $Y_{2k}$, respectively.

Similarly, we have

$$\|dG_{k+1}^T\| = \|R_{1k}^T(\Gamma_{k+1}^T \oplus \Gamma_k^T) R_{2k}^T\|,$$  \hspace{1cm} (4.2)

with the QR decompositions $[P_{1,k+1}^T, P_{1k}^T] = W_{1k}^g R_{1k}^g$ and $[P_{2,k+1}^T, -P_{2k}^T] = W_{2k}^g R_{2k}^g$.

From the NARE (1.1) we have the

$$\mathcal{R}(H_k^T) = Q_{1k}^T \Sigma_k^T \left( Q_{2k}^T C Q_{1k}^T \right) \Sigma_k^T Q_{2k}^T - AQ_{1k}^T \Sigma_k^T Q_{2k}^T - Q_{1k}^T \Sigma_k^T Q_{2k}^T D + B_1 B_2^T$$

$$= \begin{bmatrix} Q_{1k}^T & A Q_{1k} & B_1 \end{bmatrix} \begin{bmatrix} \Sigma_k^T (Q_{2k}^T C Q_{1k}^T) \Sigma_k^T & -\Sigma_k & 0 \\ -\Sigma_k & 0 & 0 \\ 0 & 0 & I_m \end{bmatrix} \begin{bmatrix} Q_{2k}^T \\ -Q_{2k}^T \Sigma_k^T \Sigma_k^T Q_{2k}^T + D + B_1 B_2^T \end{bmatrix}$$

$$= Z_{1k} \Phi_k Z_{1k}^T.$$  \hspace{1cm} (4.3)

Then the residual and the relative residual,

$$r_k \equiv \|\mathcal{R}(H_k^T)\|, \quad \tilde{r}_k = \frac{r_k}{\|H_k^T C H_k^T\| + \|H_k^T D\| + \|A H_k^T\| + \|B\|},$$

can be efficiently calculated by

$$\|\mathcal{R}(H_k^T)\| = \|R_{1k}^T \Phi_k R_{2k}^T\|, \quad \|H_k^T C H_k^T\| = \|\Sigma_k^T (Q_{2k}^T C Q_{1k}^T) \Sigma_k^T\|,$$

$$\|A H_k^T\| = \|R_{1k}^T \Sigma_k^T\|, \quad \|H_k^T D\| = \|\Sigma_k^T R_{2k}^T\|, \quad \|B\| = \|R_{01} R_{02}\|,$$

with the QR decompositions $Z_{ik} = W_{ik}^R R_{ik}^h$, $AQ_{1k} = W_{ik}^g R_{ik}^g$, $D^T Q_{2k} = W_{ik}^d R_{ik}^d$ and $B_i = W_{ik}^b R_{b,i}$, for $i = 1, 2$. 

4.2. Operation and Memory Counts. We shall assume that \( c, n \) flops \((n = \max\{n_1, n_2\})\) are required in the solution of \( \tilde{M}z = b \) or \( \tilde{M}'z = b \) (with \( \tilde{M} = A', b \in \mathbb{R}^{n_1} \) or \( \tilde{M} = D', b \in \mathbb{R}^{n_2} \)). The operation count for the QR decomposition of an \( n \times r \) matrix is \( 4r^2n \) flops [11, p. 250]. A start up cost of \((c_1 + c_2 + c_3)n\) flops is made up of the following:

1. set up \( A' = A + \gamma I_{n_1} \) and \( D' = D + \gamma I_{n_2} \), requiring \( n_1 + n_2 \leq 2n \) flops; we shall denote this part of the count by \( c_1 \) flops, with \( c_1 = 2 \);

2. set up \( Q_{10} \) and \( P_{10} \) \((i = 1, 2)\) as in (2.2) with the help of (2.3), requiring \( c_2n \) flops with \( c_2 = 2[(c_1 + 1)(m + l) + 2lm + m^2 + l^2] \); and

3. set up \( Q_{10}^* \) and \( P_{10}^* \) \((i = 1, 2)\) as in (2.17), requiring \( c_3n \) flops with \( c_3 = 12(l^2 + m^2) \).

The operation and memory counts of Algorithm 1 (SDA_{ls,\varepsilon}) for the \( k \)th iteration are summarized in Table 4.1 below. In the third column, the number of variables is recorded. Only \( O(n) \) operations or memory requirement are included. Note that most of the work is done in the computation of \( F_k^T Q_{1k}^r, E_k^TP_{2k}^r, E_k^T P_{1k}^r \) and \( E_k^T Q_{2k}^r \) in (2.23) have to be calculated recursively, as \( E_k^T \) and \( F_k^T \) in (2.21) are not available explicitly.

<table>
<thead>
<tr>
<th>Computation</th>
<th>Flops</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Sigma_k, \Gamma_k^r )</td>
<td>( 4l_km_kn )</td>
<td>( - )</td>
</tr>
<tr>
<td>( F_k^r Q_{1k}^r, F_k^T P_{2k}^r )</td>
<td>( 2^k(c_\gamma + 4 \sum_{j=1}^k 2^{-j}l_j)(l_k + m_k)n )</td>
<td>( 2n \sum_{j=1}^{k-1} l_j )</td>
</tr>
<tr>
<td>( E_k^T P_{1k}^r, E_k^T Q_{2k}^r )</td>
<td>( 2^k(c_\gamma + 4 \sum_{j=1}^k 2^{-j}m_j)(l_k + m_k)n )</td>
<td>( 2n \sum_{j=1}^{k-1} m_j )</td>
</tr>
<tr>
<td>( F_{1,k+1}^r, E_{1,k+1}^r ) Orthogonalize ( F_k^T Q_{1k}^r, ) ( F_k^T P_{2k}^r, E_k^T P_{1k}^r ) and ( E_{1,k+1}^T Q_{2k}^r )</td>
<td>( 4l_km_kn )</td>
<td>( 2(nm_k + l_k) )</td>
</tr>
<tr>
<td>( \Sigma_{k+1}, \tilde{\Sigma}<em>{k+1}, \Gamma</em>{k+1}^r ) ( Q_{1k+1}^r, P_{1k+1}^r ) ((i = 1, 2))</td>
<td>( O(l_k^3 + m_k^3) )</td>
<td>( \frac{l_k+1 + m_k+1}{8(l_kl_{k+1} + m_km_{k+1})n} )</td>
</tr>
<tr>
<td>Total flops &amp; memory</td>
<td>( 2^{k+1} \left(c_\gamma + 2 \sum_{j=1}^k 2^{-j}(l_j + m_j)\right)(l_k + m_k) + 8(l_km_k + l_kl_{k+1} + m_km_{k+1}) + 12(m_k^2 + l_k^2) + 2(m_k + l_k) )</td>
<td>( 2n \sum_{j=1}^{k+1} (l_j + m_j) )</td>
</tr>
</tbody>
</table>

With \( l_k \) and \( m_k \) controlled by the compression and truncation in Section 2.2, the operation count will be dominated by the calculation of \( F_k^r Q_{1k}^r, F_k^T P_{2k}^r, E_k^T P_{1k}^r \) and \( E_k^T Q_{2k}^r \). In our numerical examples in Section 5, the flop count near the end of Algorithm 1 dominates, with the work involved in one iteration approximately doubled that of the previous one. This corresponds to the \( 2^{k+1} \) factor in the total flop count.

5. Numerical Examples. We constructed the examples as in [23], \( A \) and \( D \) are rank one updates of nonsingular diagonal matrices and \( B \) and \( C \) are rank one, generated randomly. Three examples of sizes \( n = n_1 = n_2 = 1000, 10000, 100000 \) were generated, all satisfying the corresponding solvability conditions. The numerical results in Examples 5.1–5.3 \((n = 1000, 10000, 100000)\) were computed using MATLAB [28] Version R2012b, on an iMac with a 2.97GHz Intel Core i7 processor and 8GB RAM, with machine accuracy \( \varepsilon_p = 2.22 \times 10^{-16} \).

In Algorithm 1, the stopping criterion is \( d_k \equiv \max\{\|dH_k^i\|, \|dG_k^i\|\} < \varepsilon_c \) where \( \|dH_k^i\| = \|H_k^i - H_k^{i-1}\| \) and \( \|dG_k^i\| = \|G_k^i - G_k^{i-1}\| \) and convergence tolerance \( \varepsilon_c; \) please also consult the convergence results in Theorem 3.1. All numerical experiments were considered with a constant truncation tolerance \( \varepsilon_r \) in each iteration, i.e., \( \varepsilon_i = \varepsilon_r \) for \( i = 0, 1, \ldots, k \).
In Example 5.1 with the smallest $n = 1000$, we apply the SDA (1.5) to compute the near-exact solutions $X$ and $Y$ of NARE (1.1) and its dual equation (1.6). These were then used to illustrate the results for rank$_r(X)$, rank$_r(Y)$ in Table 5.1 and the forward errors in Tables 5.2–5.3. Effects of different tolerances $\varepsilon_r$ (or $\varepsilon_c$) are also presented in Table 5.2 (or in Table 5.3).

In Examples 5.2–5.3, the iterations in the SDA, $\varepsilon_r$ are reported for a corresponding set of tolerances $\varepsilon_c$ and $\varepsilon_r$. In Tables 5.4–5.5 below, $\|H_k\|$, $\|G_k\|$, $d_k$, $r_k$, $\bar{r}_k$, $m_k$, $l_k$, $\delta t_k$ and $t_k$ are displayed. Note that $\delta t_i$ is the execution time for the $i$th iteration and the sub-total execution time $t_k = \sum_{i=1}^{k} \delta t_i$.

**EXAMPLE 5.1. ($n = 1000$)** In this example, we have performed three tests.

**Test 1:** We first apply the SDA (1.5) with initial (1.4) to compute the near-exact solutions $X$ and $Y$ of NARE (1.1) and its dual equation (1.6). The SDA converges after 12 iterations and the norms ($\|X\|$, $\|Y\|$), residuals ($\|\mathcal{R}(X)\|$, $\|\mathcal{D}(Y)\|$) are estimated, respectively, as

\[
\begin{align*}
\|X\| &= 2.5748 \times 10^{-1}, \quad \|\mathcal{R}(X)\| = 5.9875 \times 10^{-17}, \\
\|Y\| &= 2.6545 \times 10^{-1}, \quad \|\mathcal{D}(Y)\| = 5.6928 \times 10^{-17}.
\end{align*}
\]

Table 5.1 shows the rank$_r(X)$ and rank$_r(Y)$ with $\tau = 10^{-3}, 10^{-5}, 10^{-7}, 10^{-9}, 10^{-11}, 10^{-13}$ and $10^{-15}$. Note that rank$_r(X)$ and rank$_r(Y)$ are much smaller than the matrix size $n = 1000$.

<table>
<thead>
<tr>
<th>$\tau$</th>
<th>$10^{-3}$</th>
<th>$10^{-5}$</th>
<th>$10^{-7}$</th>
<th>$10^{-9}$</th>
<th>$10^{-11}$</th>
<th>$10^{-13}$</th>
<th>$10^{-15}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>rank$_r(X)$</td>
<td>4</td>
<td>7</td>
<td>11</td>
<td>14</td>
<td>17</td>
<td>21</td>
<td>24</td>
</tr>
<tr>
<td>rank$_r(Y)$</td>
<td>4</td>
<td>8</td>
<td>11</td>
<td>14</td>
<td>18</td>
<td>21</td>
<td>24</td>
</tr>
</tbody>
</table>

**Test 2:** In the test, we set convergence tolerance $\varepsilon_c = 10^{-8}$ and employ Algorithm 1 with various truncation tolerances $\varepsilon_r$. Suppose that Algorithm 1 converges after $k$ iterations, i.e., $d_k < \varepsilon_c = 10^{-8}$. To determinate whether the computed solutions, $H_k^T$ and $G_k^T$, are nonnegative, we denote

\[
H_k^T = (H_k^T - |H_k^T|)/2, \quad G_k^T = (G_k^T - |G_k^T|)/2.
\]

From the results of **Test 1**, we have near-exact solutions $X$ and $Y$. Hence, we can compute the forward errors, $\|H_k^T - X\|$ and $\|G_k^T - Y\|$ in the example. The numerical results are shown in Table 5.2.

Table 5.2 shows that Algorithm 1 converges within 12 iterations (in $2.1 \sim 8.4$ seconds) for various tolerances $\varepsilon_r$ and the residual $r_k$ and forward errors, $\|H_k^T - X\|$ and $\|G_k^T - Y\|$, are heavily dependent on the chosen truncation tolerances $\varepsilon_r$. Furthermore, the computed solutions, $H_k^T$ and $G_k^T$, are nonnegative matrices when $\varepsilon_r = 10^{-7}, 10^{-11}, 10^{-15}$.

**Test 3:** In the test, we set truncation tolerance $\varepsilon_r = 10^{-12}$ and employ Algorithm 1 with various convergence tolerances $\varepsilon_c$. The numerical results are shown in Table 5.3.

Table 5.3 shows that Algorithm 1 converges within $8 \sim 13$ iterations for various tolerances $\varepsilon_c$ and the residual and forward errors achieve the accuracy of $O(\varepsilon_r)$ for $\varepsilon_c = 10^{-7}, 10^{-11}, 10^{-15}$.

**EXAMPLE 5.2. ($n = 10000$)** In this example, we set the truncation tolerance $\varepsilon_r = 10^{-12}$ and convergence tolerance $\varepsilon_c = 10^{-8}$. In Table 5.4, the residual (or relative residual) achieves the accuracy of $O(\varepsilon_r)$ within 12 iterations, in 140 seconds (execution time).

**EXAMPLE 5.3. ($n = 100000$)** In this example, we set the truncation tolerance $\varepsilon_r = 10^{-12}$ and convergence tolerance $\varepsilon_c = 10^{-8}$. In Table 5.5, the residual (or relative residual) achieves the accuracy of $O(\varepsilon_r)$ within 13 iterations, in 4000 seconds (execution time).
Table 5.2
Numerical results with various truncation tolerances $\varepsilon$.  

<table>
<thead>
<tr>
<th>$\varepsilon$</th>
<th>$10^{-3}$</th>
<th>$10^{-7}$</th>
<th>$10^{-11}$</th>
<th>$10^{-15}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>$\tilde{k}$</td>
<td>3</td>
<td>10</td>
<td>17</td>
<td>24</td>
</tr>
<tr>
<td>$\bar{k}$</td>
<td>3</td>
<td>11</td>
<td>17</td>
<td>24</td>
</tr>
<tr>
<td>$|H^{\tau}<em>{k} - H^{\tau}</em>{k-1}|$</td>
<td>1.208e-13</td>
<td>6.706e-13</td>
<td>6.844e-13</td>
<td>6.844e-13</td>
</tr>
<tr>
<td>$|G^{\tau}<em>{k} - G^{\tau}</em>{k-1}|$</td>
<td>1.767e-12</td>
<td>9.638e-12</td>
<td>9.640e-12</td>
<td>9.640e-12</td>
</tr>
<tr>
<td>$|H^{\tau}_{k} - X|$</td>
<td>1.494e-03</td>
<td>1.473e-07</td>
<td>1.843e-11</td>
<td>7.091e-15</td>
</tr>
<tr>
<td>$|G^{\tau}_{k} - Y|$</td>
<td>2.174e-03</td>
<td>8.537e-08</td>
<td>1.456e-11</td>
<td>7.691e-15</td>
</tr>
</tbody>
</table>

Table 5.3
Numerical results with various truncation tolerances $\varepsilon_c$.  

<table>
<thead>
<tr>
<th>$\varepsilon_c$</th>
<th>$10^{-3}$</th>
<th>$10^{-7}$</th>
<th>$10^{-11}$</th>
<th>$10^{-15}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>8</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>$\tilde{k}$</td>
<td>17</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>$\bar{k}$</td>
<td>17</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>$|H^{\tau}<em>{k} - H^{\tau}</em>{k-1}|$</td>
<td>1.251e-04</td>
<td>7.131e-09</td>
<td>6.844e-13</td>
<td>1.918e-16</td>
</tr>
<tr>
<td>$|G^{\tau}<em>{k} - G^{\tau}</em>{k-1}|$</td>
<td>4.982e-04</td>
<td>9.899e-08</td>
<td>9.640e-12</td>
<td>1.311e-16</td>
</tr>
<tr>
<td>$|H^{\tau}_{k} - X|$</td>
<td>1.494e-05</td>
<td>1.094e-12</td>
<td>1.077e-12</td>
<td>1.077e-12</td>
</tr>
<tr>
<td>$|G^{\tau}_{k} - Y|$</td>
<td>2.174e-04</td>
<td>9.765e-12</td>
<td>1.316e-12</td>
<td>1.316e-12</td>
</tr>
</tbody>
</table>

6. Conclusions. We have proposed a structure-preserving doubling algorithm for the large-scale nonsymmetric algebraic Riccati equation (1.1), the SDA$_{ls}$,$\varepsilon$, with $A$ and $D$ being large and sparse(-like), and $B$ and $C$ being low-ranked. We apply the Sherman-Morrison-Woodbury formula when appropriate and do not form $E_k$ and $F_k$ (the iterates for $E$ and $F$) explicitly. For well-behaved NAREs, low-ranked approximations to the solutions $X$ and $Y$ can be obtained efficiently. The convergence of the SDA$_{ls}$,$\varepsilon$ is quadratic, ignoring the compression and truncation of $Q_{i,k}$ and $P_{i,k}$, as shown in [17, 25]. The computational complexity and memory requirement are both $O(n)$, provided that the growth in the dimensions of $Q_{i,k}$ and $P_{i,k}$ is controlled. In the error analysis part, we gave a first order forward error bound for the computed approximate solution in Theorem 3.1. Notice that large-scale NAREs, arisen naturally from transport theory [22, 23], have not been investigated before. Our technique can be applied when $A$ and $D$ are large and sparse(-like), or are products (inverses) of such matrices. The feasibility of the SDA$_{ls}$,$\varepsilon$ depends on whether $A^{-1}u$, $A^{-\top}u$, $D^{-1}v$ and $D^{-\top}v$ can be formed efficiently, for arbitrary vectors $u$ and $v$.

Acknowledgements. The first author was supported by the NSFC (No.11101080) and the SRFDP (No.20110092120023), China. Parts of this project were completed while the first author...
![Markdown rendering of the image content](image-url)