Linear Algebra and its Applications 439 (2013) 914–932

Contents lists available at SciVerse ScienceDirect

Linear Algebra and its Applications

journal homepage: www.elsevier.com/locate/laa

Solving large-scale nonlinear matrix equations by doubling

Peter Chang-Yi Weng [a], Eric King-Wah Chu [a,*], Yueh-Cheng Kuo [b], Wen-Wei Lin [c]

[a] School of Mathematical Sciences, Building 28, Monash University, Vic 3800, Australia
[b] Department of Applied Mathematics, National University of Kaohsiung, Kaohsiung 811, Taiwan
[c] Department of Applied Mathematics, National Chiao Tung University, Hsinchu 300, Taiwan

ARTICLE INFO

Article history:
Received 22 November 2011
Accepted 7 August 2012
Available online 19 September 2012

Submitted by P. Šemrl

AMS classification:
15A24
65F50

Keywords:
Doubling algorithm
Green's function
Krylov subspace
Leaky surface wave
Nano research
Nonlinear matrix equation
Surface acoustic wave

ABSTRACT

We consider the solution of the large-scale nonlinear matrix equation $X + BX^{-1}A - Q = 0$, with $A, B, Q, X \in \mathbb{C}^{n\times n}$, and in some applications $B = A^{\star}$ ($\star = \top$ or H). The matrix Q is assumed to be nonsingular and sparse with its structure allowing the solution of the corresponding linear system $Qv = r$ in $O(n)$ computational complexity. Furthermore, B and A are respectively of ranks $r_a, r_b \ll n$. The type 2 structure-preserving doubling algorithm by Lin and Xu (2006) [24] is adapted, with the appropriate applications of the Sherman–Morrison–Woodbury formula and the low-rank updates of various iterates. Two resulting large-scale doubling algorithms have an $O((r_a + r_b)^3)$ computational complexity per iteration, after some pre-processing of data in $O(n)$ computational complexity and memory requirement, and converge quadratically. These are illustrated by the numerical examples.

1. Introduction

Consider the nonlinear matrix equation (NME)

$$\mathcal{R}(X) \equiv X + BX^{-1}A - Q = 0 \tag{1}$$

with $A, B, Q, X \in \mathbb{C}^{n\times n}$. We assume that Q is nonsingular with structures, like being banded or sparse, allowing the solution of the corresponding linear system $Qv = r$ in $O(n)$ computational complexity.

* Corresponding author.
E-mail addresses: peter.weng@monash.edu (P.C.-Y. Weng), eric.chu@monash.edu (E.K.-w. Chu), yckuo@nuk.edu.tw (Y.-C. Kuo), wwlin@am.nctu.edu.tw (W.-W. Lin).

0024-3795/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.laa.2012.08.008

We further assume that A, B are respectively of ranks r_a, $r_b \ll n$. These NMEs arise in the solution of palindromic eigenvalue problems, with applications in the computation of Green's function in nano research [12–14,16] and surface acoustic simulations [18,19]; for the individual models, structures of the particular NMEs and their solvability conditions, please consult these references. For the surface acoustic wave application in [18,19], we have $Q = Q^\top \in \mathbb{C}^{n\times n}$ and $B = A^\top \in \mathbb{C}^{n\times n}$. In some applications as in [18,19], selected eigenvalues from the pencils $\lambda X - A$ or $\lambda B - X$ are required, after the solution X to (1) is found.

We shall adapt the structure-preserving doubling algorithm (SDA) of type 2 [13,16,24] for the NME in (1), resulting in an efficient large-scale doubling algorithm (SDA_ls). The original SDA is usually attributed to Anderson [1],[1] as an accelerated variant of the direct functional iteration method. It has recently been revitalized and further developed in [5–7], for a great variety of applications [9]. Recently, we have extended the SDA (of type 1) for large-scale algebraic Riccati equations (AREs) [21,22,28] and the associated linear equations [23], with the resulting algorithms possessing an efficient $O(n)$ computational complexity and memory requirement per iteration. We shall extend these methods to NMEs, based on similar philosophy. Notice the important difference in the large-scale NME, from large-scale AREs, that the solution X is nonsingular and not numerically low-ranked. (We shall see later that X is a numerically low-ranked update of the nonsingular Q.) Interestingly, the essential steps of compression and truncation of Krylov bases for large-scale AREs are not required for NMEs. Also, the SDA_ls for large-scale NMEs is of a more efficient $O((r_a+r_b)^3)$ computational complexity per iteration, as compared to $O(n)$ for AREs. The overall algorithm shares the $O(n)$ computational complexity and memory requirement because of the pre-processing of data.

Similar techniques in this paper are applicable to the cyclic reduction method in [25] for $X \pm A^\star X^{-1} A - Q = 0$ ($\star = \top, H$; denoting the transpose and the Hermitian).

2. Preliminaries

In this section, we shall introduce some notations, briefly describe the solvability condition for NME (1) and give some preliminary results. Throughout this paper, we denote the unit circle in the complex plane by $\mathbb{T}$. For a matrix $A \in \mathbb{C}^{n\times n}$, $\sigma(A)$ and $\rho(A)$ denote respectively the spectrum and spectral radius of A, and $\sigma_{\max}(A)$ and $\sigma_{\min}(A)$ are respectively the maximum and minimum singular values of A. The conjugate transpose and transpose of A are denoted by A^H and $A^\top$ respectively. We can write $A = A_R + iA_I$, where the Hermitian matrices

$$A_R = \frac{1}{2}(A + A^H), \quad A_I = \frac{1}{2i}(A - A^H)$$

are called the real part and the imaginary part of A, respectively. For Hermitian matrices $A_1, A_2 \in \mathbb{C}^{n\times n}$, we use $A_1 > A_2$ ($A_1 \geq A_2$) to denote the fact that $A_1 - A_2$ is positive definitive (positive semi-definite).

The NME in (1) can be reformulated as

$$\mathcal{R}(X) = X + (C^H + iD^H)X^{-1}(C + iD) - (Q_R + iQ_I) = 0,$$

where $C \equiv \frac{1}{2}(A + B^H)$, $D \equiv \frac{1}{2i}(A - B^H)$ and Q_R, Q_I are the real part and the imaginary part of Q, respectively. Let

$$\psi(z) = zD^H + Q_I + z^{-1}D \tag{2}$$

be a rational matrix-valued function. The following is a consequence of the solvability results from [13] and the proof can be found in [14], after superficial modifications.

[1] In [2, p. 149], we have the quotation "Doubling algorithms have been part of the folklore associated with Riccati equations in linear systems problems for some time. We are unable to give any original reference containing material close to that presented here." This quotation from Anderson, to whom the SDA is widely attributed, indicates the uncertain origin of the method.

Theorem 2.1. *Let $A = C + iD$, $B = C^H + iD^H$ and $Q = Q_R + iQ_I$ be $n \times n$ complex matrices. Suppose that $\psi(z)$ defined in (2) is positive definite for each $z \in \mathbb{T}$.*

(i) The matrix polynomial $P(z) = z^2B - zQ + A$ has exactly n eigenvalues each inside and outside $\mathbb{T}$.
(ii) The NME (1) has a solution $X = X_R + iX_I$ with $\rho(X^{-1}A) < 1$ and $X_I > 0$.

Note that if X is solution of NME, then $P(z) = (zBX^{-1} - I)X(zI - X^{-1}A)$. So the eigenvalues of $X^{-1}A$ are the n eigenvalues of $P(z)$, and the eigenvalues of BX^{-1} are the reciprocals of remaining n eigenvalues of $P(z)$. A solution X of NME is said to be stabilizing if $\rho(X^{-1}A) < 1$. From Theorem 2.1, if $\psi(z) > 0$ for each $z \in \mathbb{T}$, then the NME and its dual

$$\widehat{X} + A\widehat{X}^{-1}B = Q \tag{3}$$

have stabilizing solutions.

Remark 2.1. Theorem 2.1 gives a sufficient condition for the existence of stabilizing solutions of NMEs. For applications in the computation of the surface Greens function in nano research [13,16] , we have $Q_I = I$, $D = 0$ and C, Q_R being real, and it is easy to check that the sufficient condition holds. For the surface wave application in [18,19], we have C, D, Q_R and Q_I being real such that $\psi(z) > 0$ for each $z \in \mathbb{T}$. For the special case where C, $Q_R = 0$, if $\psi(z) > 0$ for each $z \in \mathbb{T}$, then

$$X + (iD^H)X^{-1}(iD) = iQ_I \tag{4}$$

has a stabilizing solution X_S with $X_{S,I} > 0$, $\rho(X_S^{-1}(iD)) < 1$ and $\rho((iD^H)X_S^{-1}) < 1$. We shall show that the real part of X_S is zero. Consider the nonlinear matrix equation

$$X + (iD^H)X^{-1}(iD) = -iQ_I. \tag{5}$$

It is obvious that $-X_S$ and X_S^H are solutions with $\rho(-X_S^{-1}(iD)) < 1$ and $\rho(X_S^{-H}(iD)) < 1$, i.e., $-X_S$ and X_S^H are stabilizing solutions. Since the stabilizing solution of (5) is unique, $X_S^H = -X_S$. Hence, X_S has only imaginary part, i.e., $X_S = iX_{S,I}$. Since $X_{S,I} > 0$, substituting $X_S = iX_{S,I}$ into (4) implies that $X + D^HX^{-1}D = Q_I$ has a Hermitian positive definite solution $X_{S,I}$. This coincides with the result in [10].

Under the assumption that $\psi(z) > 0$ for each $z \in \mathbb{T}$, we know that the NME and its dual have stabilizing solutions X and $\widehat{X}$, respectively. The corresponding SDA of type 2 [13,16,24] has the form

$$\begin{aligned}
&A_0 = A, \quad B_0 = B, \quad Q_0 = Q, \quad P_0 = 0;\\
&A_{k+1} = A_k(Q_k - P_k)^{-1}A_k, \quad B_{k+1} = B_k(Q_k - P_k)^{-1}B_k;\\
&Q_{k+1} = Q_k - B_k(Q_k - P_k)^{-1}A_k, \quad P_{k+1} = P_k + A_k(Q_k - P_k)^{-1}B_k.
\end{aligned} \tag{6}$$

From [13, Theorem 3.1], all the iterates are well-defined (i.e., $M_k \equiv Q_k - P_k$ are always invertible), $Q_k \to X$, $Q - P_k \to \widehat{X}$ and A_k, $B_k \to 0$, all quadratically as $k \to \infty$. In the following, we shall give an upper bound of $\{\|M_k^{-1}\|_2 : k = 0, 1, \ldots\}$ For the case where $B = A^H$ and $Q = Q^H$, an upper bound has been given in [3, Theorem 15].

The following lemma is useful to estimate the upper bound.

Lemma 2.2. *Let $\Phi = \Phi_R + i\Phi_I \in \mathbb{C}^{n\times n}$, where Φ_R and Φ_I are the real and imaginary parts of Φ. Suppose that Φ_I is positive (negative) definite.*

(i) Φ is invertible and

$$\Phi^{-1} = \Phi_I^{-1}\Phi_R(\Phi_I + \Phi_R\Phi_I^{-1}\Phi_R)^{-1} + i[-(\Phi_I + \Phi_R\Phi_I^{-1}\Phi_R)^{-1}] \tag{7}$$

with $\Phi_I^{-1}\Phi_R(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}$ *being Hermitian and* $-(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}$ *being Hermitian negative (positive) definite.*

(ii) If $\Phi_I \geqslant \epsilon I$ $(\Phi_I \leqslant -\epsilon I)$ *for some* $\epsilon > 0$, *then* $\sigma_{\min}(\Phi) \geqslant \epsilon$, *i.e.,* $\|\Phi^{-1}\|_2 \leqslant \epsilon^{-1}$.

Proof. (*i*) For each unit vector $x \in \mathbb{C}^n$, since Φ_R, Φ_I are Hermitian and Φ_I is positive (negative) definite, we have

$$\|\Phi x\|_2 = \|(\Phi_R + i\Phi_I)x\|_2 \geqslant \|x^H(\Phi_R + i\Phi_I)x\|_2 \geqslant \|x^H\Phi_I x\|_2 \geqslant \sigma_{\min}(\Phi_I). \tag{8}$$

Hence, Φ is invertible. From

$$\begin{aligned}&(\Phi_R + i\Phi_I)[\Phi_I^{-1}\Phi_R(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1} - i(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}]\\ &\quad= (\Phi_R + i\Phi_I)(\Phi_I^{-1}\Phi_R - iI)(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}\\ &\quad= (\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1} = I,\end{aligned}$$

we obtain (7). Since Φ_I is positive (negative) definite and $\Phi_R = \Phi_R^H$, it is easy to see that $-(\Phi_I + \Phi_R\Phi_I^{-1}\Phi_R)^{-1}$ is Hermitian negative (positive) definite. Finally, we show that $\Phi_I^{-1}\Phi_R(\Phi_I + \Phi_R\Phi_I^{-1}\Phi_R)^{-1}$ is Hermitian. From (7), we have

$$[\Phi_I^{-1}\Phi_R(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1} - i(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}](\Phi_R + i\Phi_I) = I.$$

It follows that $\Phi_I^{-1}\Phi_R(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}\Phi_I - (\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}\Phi_R = 0$. Since Φ_R and Φ_I are Hermitian, we deduce that $\Phi_I^{-1}\Phi_R(\Phi_I+\Phi_R\Phi_I^{-1}\Phi_R)^{-1}$ is Hermitian.

(*ii*) If $\Phi_I \geqslant \epsilon I$ $(\Phi_I \leqslant -\epsilon I)$ for some $\epsilon > 0$ then from (8), we have

$$\sigma_{\min}(\Phi) = \min_{\|x\|=1} \|\Phi x\|_2 \geqslant \epsilon.$$

Hence, $\|\Phi^{-1}\|_2 = \sigma_{\min}(\Phi)^{-1} \leqslant \epsilon^{-1}$. □

Suppose that $\psi(z) > 0$ for each $z \in \mathbb{T}$. Let

$$M_k = Q_k - P_k, \quad \phi_k(z) = -B_k z + M_k - A_k z^{-1}, \tag{9}$$

where A_k, B_k, Q_k, P_k are generated by SDA (6). From [13], we know that M_k is invertible. Then

$$\begin{aligned}\phi_k(z)M_k^{-1}\phi_k(-z) &= (-B_k z + M_k - A_k z^{-1})(M_k^{-1}B_k z + I + M_k^{-1}A_k z^{-1})\\ &= -B_k M_k^{-1}B_k z^2 + (M_k - B_k M_k^{-1}A_k - A_k M_k^{-1}B_k) - A_k M_k^{-1}A_k z^{-2}\\ &= -B_{k+1}z^2 + M_{k+1} - A_{k+1}z^{-2} = \phi_{k+1}(z^2).\end{aligned} \tag{10}$$

Let $\phi_{0,R}(z)$ and $\phi_{0,I}(z)$ be the real and imaginary parts of $\phi_0(z)$, respectively. Since, for $z \in \mathbb{T}$,

$$\phi_0(z) = \phi_{0,R}(z) + i\phi_{0,I}(z), \quad \phi_{0,I}(z) = \psi(-z) > 0. \tag{11}$$

Lemma 2.2 implies that $\phi_0(z)$ is invertible for $z \in \mathbb{T}$. It follows from (10) that for $k = 0, 1, \ldots$, $\phi_k(z)$ is invertible for $z \in \mathbb{T}$. Let $\varphi_k(z) = \phi_k(z)^{-1}$ for $z \in \mathbb{T}$ and

$$\varphi_k(z) = \varphi_{k,R}(z) + i\varphi_{k,I}(z), \tag{12}$$

where $\varphi_{k,R}(z)$ and $\varphi_{k,I}(z)$ are the real and imaginary parts of $\varphi_k(z)$. Taking the inverse of (10) yields

$$\varphi_{k+1}(z^2) = \phi_{k+1}(z^2)^{-1} = \phi_k(-z)^{-1} M_k \phi_k(z)^{-1} = \phi_k(-z)^{-1} \left(\frac{\phi_k(z) + \phi_k(-z)}{2} \right) \phi_k(z)^{-1}$$

$$= \frac{1}{2}[\phi_k(z)^{-1} + \phi_k(-z)^{-1}] = \frac{1}{2}[\varphi_k(z) + \varphi_k(-z)]. \tag{13}$$

From (11), (12) and Lemma 2.2, we have $\varphi_{0,I}(z) = -(\phi_{0,I}(z) + \phi_{0,R}(z)\phi_{0,I}(z)^{-1}\phi_{0,R}(z))^{-1} < 0$ for $z \in \mathbb{T}$. It follows from (13) that for each k, $\varphi_{k,I}(z) < 0$ for $z \in \mathbb{T}$ and

$$\begin{aligned}
&\max_{z\in\mathbb{T}} \sigma_{\max}(\varphi_{k,R}(z)) \leqslant \max_{z\in\mathbb{T}} \sigma_{\max}(\varphi_{0,R}(z)) \equiv \sigma_{\max,R},\\
&\max_{z\in\mathbb{T}} \sigma_{\max}(\varphi_{k,I}(z)) \leqslant \max_{z\in\mathbb{T}} \sigma_{\max}(\varphi_{0,I}(z)) \equiv \sigma_{\max,I},\\
&\min_{z\in\mathbb{T}} \sigma_{\min}(\varphi_{k,I}(z)) \geqslant \min_{z\in\mathbb{T}} \sigma_{\min}(\varphi_{0,I}(z)) \equiv \sigma_{\min,I} > 0.
\end{aligned} \tag{14}$$

For $z \in \mathbb{T}$, we then have

$$\|\varphi_{k,R}(z)\|_2 \leqslant \sigma_{\max,R}, \quad \|\varphi_{k,I}(z)\|_2 \leqslant \sigma_{\max,I}, \quad \|\varphi_{k,I}(z)^{-1}\|_2 \leqslant \sigma_{\min,I}^{-1}. \tag{15}$$

The following theorem gives upper bounds of $\|M_k\|_2$ and $\|M_k^{-1}\|_2$ for $k = 0, 1, \ldots$

Theorem 2.3. *Let $A = C + iD$, $B = C^H + iD^H$ and $Q = Q_R + iQ_I$ be given such that $\psi(z)$ is positive for each $z \in \mathbb{T}$. Let $M_k = Q_k - P_k$, where Q_k and P_k are generated by the SDA in (6). Then*

$$\|M_k\|_2 \leqslant \frac{\sqrt{\sigma_{\max,R}^2 + \sigma_{\min,I}^2}}{\sigma_{\min,I}^2}, \quad \|M_k^{-1}\|_2 \leqslant \sigma_{\max,I} + \frac{\sigma_{\max,R}^2}{\sigma_{\min,I}},$$

where $\sigma_{\max,R}$, $\sigma_{\max,I}$ and $\sigma_{\min,I}$ are given in (14), which are only dependent on $\phi_0(z)$ for $z \in \mathbb{T}$.

Proof. For each $k = 0, 1, \ldots$, from (15), we have that for each $z \in \mathbb{T}$,

$$-\left(\frac{\sigma_{\max,R}}{\sigma_{\min,I}}\right)\varphi_{k,I}(z) \leqslant \varphi_{k,R}(z) \leqslant \left(\frac{\sigma_{\max,R}}{\sigma_{\min,I}}\right)\varphi_{k,I}(z),$$

$$0 < \sigma_{\min,I} I \leqslant -\varphi_{k,I}(z) - \varphi_{k,R}(z)\varphi_{k,I}(z)^{-1}\varphi_{k,R}(z) \leqslant \left(\sigma_{\max,I} + \frac{\sigma_{\max,R}^2}{\sigma_{\min,I}}\right) I.$$

Let $\phi_{k,R}(z)$ and $\phi_{k,I}(z)$ be the real and imaginary parts of $\phi_k(z)$, respectively. From Lemma 2.2 (i) and using the fact that $\phi_k(z) = \varphi_k(z)^{-1}$, we have for $z \in \mathbb{T}$,

$$\left(\frac{\sigma_{\max,R}}{\sigma_{\min,I}^2}\right) I \geqslant \phi_{k,R}(z) \geqslant -\left(\frac{\sigma_{\max,R}}{\sigma_{\min,I}^2}\right) I,$$

$$\sigma_{\min,I}^{-1} I \geqslant \phi_{k,I}(z) \geqslant \left(\sigma_{\max,I} + \frac{\sigma_{\max,R}^2}{\sigma_{\min,I}}\right)^{-1} I.$$

Since $M_k = [\phi_k(z) + \phi_k(-z)]/2 = [\phi_{k,R}(z) + \phi_{k,R}(-z)]/2 + i[\phi_{k,I}(z) + \phi_{k,I}(-z)]/2$, we have

$$\|M_k\|_2 \leqslant \sqrt{\left(\frac{\sigma_{\max,R}}{\sigma^2_{\min,I}}\right)^2 + \sigma^{-2}_{\min,I}} = \frac{\sqrt{\sigma^2_{\max,R} + \sigma^2_{\min,I}}}{\sigma^2_{\min,I}}$$

and

$$M_{k,I} = \frac{1}{2i}(M_k - M_k^H) = \frac{\phi_{k,I}(z) + \phi_{k,I}(-z)}{2} \geqslant \left(\sigma_{\max,I} + \frac{\sigma^2_{\max,R}}{\sigma_{\min,I}}\right)^{-1} I.$$

It follows from Lemma 2.2 (*ii*) that $\|M_k^{-1}\|_2 \leqslant \sigma_{\max,I} + \sigma^2_{\max,R}/\sigma_{\min,I}$. □

For the special case $C, Q_R = 0$, it follows from (14) and Theorem 2.3 that $\|M_k\|_2 \leqslant \sigma^{-1}_{\min,I}$ and $\|M_k^{-1}\|_2 \leqslant \sigma_{\max,I}$. This coincides with the results in [3, Theorem 15].

3. Large-scale doubling algorithm

3.1. Main ideas

From [21–23,28], the main ideas behind the algorithm for large-scale problems are:

(a) The appropriate application of the Sherman–Morrison–Woodbury formula (SMWF) in order to avoid the inversion of large or unstructured matrices.
(b) The use of low-rank updates for various iterates.
(c) The computation of matrix operators (A_k) recursively, to preserve the corresponding sparsity or low-rank structures, instead of forming them explicitly.
(d) The careful organization of convergence control in the algorithm, so as to preserve the low computational complexity and memory requirement per iteration.

For the SDA for large-scale NMEs, we shall see that (c) is not relevant.

Let $A, B, Q \in \mathbb{C}^{n\times n}$ be given such that $\psi(z)$ defined in (2) is positive definite for each $z \in \mathbb{T}$ and A, B be respectively of ranks $r_a, r_b \ll n$. Assume the full-rank decompositions

$$A = F_a R_a G_a^H, \quad B = F_b R_b G_b^H \tag{16}$$

with $R_a \in \mathbb{C}^{r_a\times r_a}$ and $R_b \in \mathbb{C}^{r_b\times r_b}$. Without loss of generality, we shall assume that F_a, F_b, G_a and G_b are unitary. In this paper, we shall call some matrices "kernels", mostly denoted by R with various subscripts (Y is also used in Section 3.3). Most of our computation will be done in terms of kernels.

From Theorem 2.1, we know that NME (1) and its dual (3) have stabilizing solutions X and $\widehat{X}$, respectively. The SDA for NME and its dual has the form in (6) which requires the inverse of $M_k = Q_k - P_k$. It is shown in [13] that M_k is invertible for each $k = 0, 1, \ldots$ Furthermore, an upper bound of $\{\|M_k^{-1}\|_2 \mid k = 0, 1, \ldots\}$ is given in Theorem 2.3. Similar to the approach in [21,23], we shall apply the SMWF:

$$(\widetilde{A} \pm U\widetilde{C}V)^{-1} = \widetilde{A}^{-1} \mp \widetilde{A}^{-1}U(I \pm \widetilde{C}V\widetilde{A}^{-1}U)^{-1}\widetilde{C}V\widetilde{A}^{-1}$$

to various inverses of matrices in sparse-plus-low-rank (splr) form, enabling the computation of large inverses of size n in terms of much smaller matrices. In the following lemma, we show that the small size matrix $(I \pm \widetilde{C}V\widetilde{A}^{-1}U)$ is invertible provided that $\widetilde{A}$ and $(\widetilde{A} \pm U\widetilde{C}V)$ are invertible.

Lemma 3.1. *If $\widetilde{A}$ and $(\widetilde{A} \pm U\widetilde{C}V)$ are invertible then $(I \pm \widetilde{C}V\widetilde{A}^{-1}U)$ is invertible.*

Proof. Suppose that $\widetilde{A}$ and $(\widetilde{A} \pm U\widetilde{C}V)$ are invertible. Then $\begin{bmatrix} \widetilde{A} \pm U\widetilde{C}V & 0 \\ 0 & \mp I \end{bmatrix}$ is invertible. Since $\widetilde{A}$ is invertible and

$$\begin{aligned} &\begin{bmatrix} I & \mp U \\ 0 & I \end{bmatrix} \begin{bmatrix} \widetilde{A} \pm U\widetilde{C}V & 0 \\ 0 & \mp I \end{bmatrix} \begin{bmatrix} I & 0 \\ \mp\widetilde{C}V & I \end{bmatrix} = \begin{bmatrix} \widetilde{A} & U \\ \widetilde{C}V & \mp I \end{bmatrix} \\ &\quad = \begin{bmatrix} I & 0 \\ \widetilde{C}V\widetilde{A}^{-1} & \mp I \end{bmatrix} \begin{bmatrix} \widetilde{A} & 0 \\ 0 & I \pm \widetilde{C}V\widetilde{A}^{-1}U \end{bmatrix} \begin{bmatrix} I & \widetilde{A}^{-1}U \\ 0 & I \end{bmatrix}, \end{aligned}$$

we have shown that $(I \pm \widetilde{C}V\widetilde{A}^{-1}U)$ is invertible. □

3.2. Algorithm 1

For $k = 0, 1, \ldots$, we can organize the SDA so that the iterates have the recursive forms

$$\begin{aligned} &A_k = F_a R_{ak} G_a^H, \quad B_k = F_b R_{bk} G_b^H; \\ &Q_k = Q - F_b R_{qk} G_a^H, \quad P_k = F_a R_{pk} G_b^H; \end{aligned} \tag{17}$$

with $R_{ak} \in \mathbb{C}^{r_a \times r_a}$, $R_{bk} \in \mathbb{C}^{r_b \times r_b}$ and R_{pk}, $R_{qk}^H \in \mathbb{C}^{r_a \times r_b}$. The general forms in (17) can be verified easily from (6), when identifying the updating formulae in (23) and the initail values in (25). Note that we can equivalently formulate the SDA in terms of A_k, B_k, P_k and the new variable $\widehat{Q}_k \equiv Q - Q_k$, with the symmetry in the low-ranked P_k and $\widehat{Q}_k$. Note also that the row and column spaces of all these matrices remain constant, with only the various kernels Rs varying with k. Also, Q_k are low rank updates of Q. (For the nano research application in [13,16], this corresponds to the behaviour that only upper right corner in A_k and the lower right corner of Q_k are changing for different k.)

We require the inverse of $M_k = Q_k - P_k$ in the SDA in (6). From (17), we have

$$M_k = Q - [F_a, F_b] R_{mk} [G_a, G_b]^H, \tag{18}$$

where $R_{mk} \equiv \begin{bmatrix} 0 & R_{pk} \\ R_{qk} & 0 \end{bmatrix} \in \mathbb{C}^{(r_a+r_b)\times(r_a+r_b)}$. Applying the SMWF to M_k^{-1} yields

$$M_k^{-1} = Q^{-1} + Q^{-1}[F_a, F_b] N_k [G_a, G_b]^H Q^{-1} \tag{19}$$

with

$$N_k \equiv (I_{r_a+r_b} - R_{mk}T)^{-1} R_{mk} \tag{20}$$

and

$$T = \begin{bmatrix} T_{aa} & T_{ab} \\ T_{ba} & T_{bb} \end{bmatrix} \equiv [G_a, G_b]^H Q^{-1} [F_a, F_b]. \tag{21}$$

Note that N_k is symmetric if $B = A^\top$ and $Q^\top = Q$, as in [18,19].

Remark 3.1. Since Q and M_k are invertible, it follows from Lemma 3.1 that $I_{r_a+r_b} - R_{mk}T$ $(k = 0, 1, \ldots)$ are invertible. In the following, we shall give upper bounds of $\|I - R_{mk}T\|_2$ and $\|(I - R_{mk}T)^{-1}\|_2$ for $k = 0, 1, \ldots$ when $[G_a, G_b]$ and $[F_a, F_b]$ are of full column rank. Assume $\sigma_{\min}([G_a, G_b]) = \sigma_{\min,G} > 0$ and $\sigma_{\min}([F_a, F_b]) = \sigma_{\min,F} > 0$. Since G_a, G_b, F_a and F_b are unitary, we obtain

$$\begin{aligned} &\|[G_a, G_b]\|_2 \leqslant \sqrt{2}, && \|[F_a, F_b]\|_2 \leqslant \sqrt{2}, \\ &\|[G_a, G_b]^\dagger\|_2 = 1/\sigma_{\min,G}, && \|[F_a, F_b]^\dagger\|_2 = 1/\sigma_{\min,F}, \end{aligned} \tag{22}$$

where $(\cdot)^\dagger$ denotes the pseudoinverse of a matrix. From (18), (21) and (22), we have $\|R_{m,k}\|_2 \leqslant (\|M_k\|_2 + \|Q\|_2)/(\sigma_{\min,F}\sigma_{\min,G})$ and $\|T\|_2 \leqslant 2\|Q^{-1}\|_2$. Hence, we obtain

$$\|I + R_{mk}T\|_2 \leqslant 1 + \frac{2\|Q^{-1}\|_2}{\sigma_{\min,F}\sigma_{\min,G}}(\|M_k\|_2 + \|Q\|_2).$$

From (19) and (22), we have $\|N_k\|_2 \leqslant (\|Q\|_2\|M_k^{-1}\|_2 + 1)\|Q\|_2/(\sigma_{\min,F}\sigma_{\min,G})$. Using the fact that $(I - R_{mk}T)^{-1}(I - R_{mk}T) = I$, it follows that $(I - R_{mk}T)^{-1} = N_kT + I$ and hence

$$\|(I - R_{mk}T)^{-1}\|_2 \leqslant 1 + \frac{2\|Q\|_2\|Q^{-1}\|_2}{\sigma_{\min,F}\sigma_{\min,G}}(\|Q\|_2\|M_k^{-1}\|_2 + 1).$$

With (17) and (19), the SDA in (6) now becomes the updating formulae:

$$\begin{aligned} &R_{a,k+1} = R_{ak}W_k^{aa}R_{ak}, \quad R_{b,k+1} = R_{bk}W_k^{bb}R_{bk}; \\ &R_{q,k+1} = R_{qk} + R_{bk}W_k^{ba}R_{ak}, \quad R_{p,k+1} = R_{pk} + R_{ak}W_k^{ab}R_{bk}; \end{aligned} \tag{23}$$

where $W_k^{uv} \equiv G_u^H M_k^{-1} F_v$ $(u, v = a, b)$. From (19), we obtain

$$\begin{bmatrix} W_k^{aa} & W_k^{ab} \\ W_k^{ba} & W_k^{bb} \end{bmatrix} = T + TN_kT. \tag{24}$$

The computation requires about $\frac{26}{3}(r_a^3 + r_b^3) + 22r_ar_b(r_a + r_b)$ flops for each iteration (the detailed count is given in Table 1), with the help of (19), after the pre-processing in $O(n)$ complexity for quantities like $Q^{-1}U$, $Q^{-H}V$ $(U = F_a, F_b$ and $V = G_a, G_b)$ in (20) and T in (21).

For initial values, we have the obvious

$$R_{a0} = R_a, \quad R_{b0} = R_b; \quad R_{p0}, R_{q0} = 0. \tag{25}$$

In [21–23,28], the SDA of type 1 has been extended for large-scalar Stein/Lyapunov and AREs equations. The iterates A_k in the SDA of type 1 are computed recursively, without being forming explicitly. As the SDA of type 2 in (6) now translates to the updating formulae (23) in $\mathbb{C}^{r_u \times r_v}$ $(u, v = a, b)$, this previously important aspect (c) in Section 3.1 of the SDA_ls is now irrelevant.

The SDA_ls for an NME (and its dual), realizes the iteration in (6) with the help of (17), (19) and (23), the initial values in (25), and the convergence control in Section 4.1. We summarize the SDA_ls in Algorithm 1 below.

Algorithm 1 (SDA_ls)

Input: $A = F_a R_a G_a^H$, $B = F_b R_b G_b^H$, $Q \in \mathbb{C}^{n\times n}$, positive tolerance ϵ;

Output: $R_{p\epsilon}, R_{q\epsilon}^H \in \mathbb{C}^{r_a \times r_b}$, with $Q - F_b R_{q\epsilon} G_a^H$ and $Q - F_a R_{p\epsilon} G_b^H$, approximating, respectively, the solutions X and $\widehat{X}$ to the large-scale NME (1) and its dual (3);

Orthogonalize F_a, F_b, G_a, G_b, modify R_a, R_b; (if required)

Set $k = 0, \tilde{r}_0 = 2\epsilon$; $R_{a0} = R_a, R_{b0} = R_b$; $R_{p0}, R_{q0} = 0$;

(as in (25))

Do until convergence:

If the relative residual $\tilde{r}_k = |r_k/(q_k + m_k)| < \epsilon$,

Set $R_{q\epsilon} = R_{qk}$ and $R_{p\epsilon} = R_{pk}$;

Exit

End If

Compute $W_k^{uv} = G_u^H M_k^{-1} F_v$ $(u, v = a, b)$, (as in (20) and (24))

$R_{a,k+1} = R_{ak} W_k^{aa} R_{ak}$,

$R_{b,k+1} = R_{bk} W_k^{bb} R_{bk}$,

$R_{q,k+1} = R_{qk} + R_{bk} W_k^{ba} R_{ak}$, and

$R_{p,k+1} = R_{pk} + R_{ak} W_k^{ab} R_{bk}$; (as in (23))

Compute $k \leftarrow k + 1$, r_k, q_k and m_k; (as in (40) in Section 4.1)

End Do

3.3. A non-symmetric discrete-time algebraic Riccati equation

From Section 3.2, $Q - Q_k$ and P_k^H are low-ranked with small $r_b \times r_a$ kernels. It suggests that the solution X, or its kernel Y, can be characterized by a matrix equation of lower dimensions, as stated in the following lemma. We shall not elaborate on the similar result for the solution $\widehat{X}$ of dual equation.

Lemma 3.2. *Let X be a solution of the large-scale NME (1). Then there exists $Y \in \mathbb{C}^{r_b \times r_a}$ such that*

$$X = Q - F_b Y G_a^H. \tag{26}$$

Proof. Suppose that X is a solution of NME (1). Substituting (16) into (1) and setting $Y = R_b G_b^H X^{-1} F_a R_a$, we obtain (26). □

We shall now show that Y in (26) satisfies an uncommon non-symmetric discrete-time algebraic Riccati equation (NARE_D). Note that the standard non-symmetric algebraic Riccati equations (NARE) [28] are of continuous-time type in most literature.

Substituting (16) and (26) into (1), we have

$$-F_b Y G_a^H + F_b R_b G_b^H (Q - F_b Y G_a^H)^{-1} F_a R_a G_a^H = 0. \tag{27}$$

Multiplying F_b^H and G_a from the left and the right of (27), respectively, and applying the SMWF, we obtain

$$-Y + R_b G_b^H \left[Q^{-1} + Q^{-1} F_b Y (I - G_a^H Q^{-1} F_b Y)^{-1} G_a^H Q^{-1} \right] F_a R_a = 0. \tag{28}$$

With the notation in (21), (28) is equivalent to $-Y + R_b T_{ba} R_a + R_b T_{bb} Y (I - T_{ab} Y)^{-1} T_{aa} R_a = 0$, or, the NARE_D

$$\mathcal{D}(Y) \equiv -Y + \widetilde{T}_{bb} Y (I - \widetilde{T}_{ab} Y)^{-1} \widetilde{T}_{aa} + \widetilde{T}_{ba} = 0, \tag{29}$$

where

$$\widetilde{T}_{aa} \equiv T_{aa} R_a, \quad \widetilde{T}_{bb} \equiv R_b T_{bb}, \quad \widetilde{T}_{ba} \equiv R_b T_{ba} R_a, \quad \widetilde{T}_{ab} \equiv T_{ab}. \tag{30}$$

In [15], an NARE is first processed by Cayley transform to the form (29) and then doubling is applied. We shall refer to this approach (without the Cayley transform) as Algorithm 2.

Algorithm 2 (SDA_ls)

Input: $A = F_a R_a G_a^H$, $B = F_b R_b G_b^H$, $Q \in \mathbb{C}^{n\times n}$, positive tolerance ϵ;

Output: Y_ϵ, $\widehat{Y}_\epsilon^H \in \mathbb{C}^{r_b \times r_a}$, with $Q - F_b Y_\epsilon G_a^H$ and $Q - F_a \widehat{Y}_\epsilon G_b^H$, approximating, respectively, the solutions X and $\widehat{X}$ to the large-scale NME (1) and its dual (3);

Orthogonalize F_a, F_b, G_a, G_b, modify R_a, R_b; (if required)

`Compute` $\widetilde{T}_{aa}, \widetilde{T}_{bb}, \widetilde{T}_{ab}, \widetilde{T}_{ba}$; (as in (30)) $t_{ba} = \|\widetilde{T}_{ba}\|$;

`Set` $k = 0$, $\tilde{r}_0 = 2\epsilon$; $\widetilde{T}_{aa,0} = \widetilde{T}_{aa}$, $\widetilde{T}_{bb,0} = \widetilde{T}_{bb}$, $\widetilde{T}_{ba,0} = \widetilde{T}_{ba}$, $\widetilde{T}_{ab,0} = \widetilde{T}_{ab}$;

Do until convergence:

If the relative residual $\tilde{r}_k = |r_k/(q_k + m_k + t_{ba})| < \epsilon$,

Set $Y_\epsilon = \widetilde{T}_{ba,k}$ and $\widehat{Y}_\epsilon = \widetilde{T}_{ab,k}$;

Exit

End If

Compute

$$\widetilde{T}_{aa,k+1} = \widetilde{T}_{aa,k}(I_{r_a} - \widetilde{T}_{ab,k}\widetilde{T}_{ba,k})^{-1}\widetilde{T}_{aa,k},$$
$$\widetilde{T}_{bb,k+1} = \widetilde{T}_{bb,k}(I_{r_b} - \widetilde{T}_{ba,k}\widetilde{T}_{ab,k})^{-1}\widetilde{T}_{bb,k},$$
$$\widetilde{T}_{ab,k+1} = \widetilde{T}_{ab,k} + \widetilde{T}_{aa,k}\widetilde{T}_{ab,k}(I_{r_b} - \widetilde{T}_{ba,k}\widetilde{T}_{ab,k})^{-1}\widetilde{T}_{bb,k}, \text{ and}$$
$$\widetilde{T}_{ba,k+1} = \widetilde{T}_{ba,k} + \widetilde{T}_{bb,k}\widetilde{T}_{ba,k}(I_{r_a} - \widetilde{T}_{ab,k}\widetilde{T}_{ba,k})^{-1}\widetilde{T}_{aa,k};$$

Compute $k \leftarrow k+1$, $r_k = \|\mathcal{D}(\widetilde{T}_{ba,k})\|$, $q_k = \|\widetilde{T}_{ba,k}\|$ and $m_k = \|\widetilde{T}_{bb}\widetilde{T}_{ba,k}(I - \widetilde{T}_{ab}\widetilde{T}_{ba,k})^{-1}\widetilde{T}_{aa}\|$;

End Do

Notice that the computation of Algorithm 2 can be realized as: computing $T_{u,v}$ $(u, v = a, b)$ in (21) requires $O(n)$ computational complexity; computing $\widetilde{T}_{u,v}$ $(u, v = a, b)$ in (30) requires $2r_a^3 + 2r_b^3 + 2r_a r_b(r_a + r_b)$ flops; for each iteration, it requires $\frac{22}{3}r_a^3 + 10r_a^2 r_b + 8r_a r_b^2 + \frac{14}{3}r_b^3$ flops. The detailed count

is given in Table 2. For the case $r_a = r_b$, Algorithm 2 requires $8r_a^3 + (30r_a^3)k$ flops, where k is the number of iterations, after the pre-processing in $O(n)$ computational complexity. Algorithms 1 and 2 all need to compute $T_{u,v}$ $(u, v = a, b)$ in (21) which required $O(n)$ computational complexity. From Table 1, we obtain that Algorithm 1 requires $74r_a^3$ flops per iteration when $r_a = r_b$. When $r_a \approx n^{1/3} \ll n$, Algorithm 2 will be more efficient than Algorithm 1. However, we guarantee that all iterations in Algorithm 1 are well-defined, but the same for Algorithm 2 cannot be guaranteed. After each iterative step, R_{qk} in Algorithm 1 and $\widetilde{T}_{ba,k}$ in Algorithm 2 are different but they converge to the same Y. From our numerical experience, for a given example, the two algorithms converge in the same number of iterations, possibly the result of Theorem 3.3 below.

Suppose that $X = Q - F_b Y G_a^H$ and $\widehat{X} = Q - F_a \widehat{Y} G_b^H$ are the stabilizing solutions of NME and its dual, respectively. We shall show below, when all iterations of Algorithm 2 are well-defined then $\widetilde{T}_{ba,k} \to Y$ and $\widetilde{T}_{ab,k} \to \widehat{Y}$ as $k \to \infty$.

Let

$$M = \begin{bmatrix} T_{aa}R_a & 0 \\ -R_b T_{ba} R_a & I_{r_b} \end{bmatrix}, \quad L = \begin{bmatrix} I_{r_a} & -T_{ab} \\ 0 & R_b T_{bb} \end{bmatrix} \in \mathbb{C}^{(r_a+r_b)\times(r_a+r_b)}. \tag{31}$$

Suppose that (29) has a solution $Y \in \mathbb{C}^{r_b \times r_a}$. Then Y satisfies that

$$M \begin{bmatrix} I_{r_a} \\ Y \end{bmatrix} = L \begin{bmatrix} I_{r_a} \\ Y \end{bmatrix} S, \tag{32}$$

where $S = (I_{r_a} - T_{ab}Y)^{-1} T_{aa} R_a \in \mathbb{C}^{r_a \times r_a}$.

Theorem 3.3. *Let $X = Q - F_b Y G_a^H$ and $\widehat{X} = Q - F_a \widehat{Y} G_b^H$ be the stabilizing solutions of, respectively, NME (1) and its dual (3). Then*

(i) $X^{-1}A$ and $S = (I_{r_a} - T_{ab}Y)^{-1} T_{aa} R_a$ have the same nonzero eigenvalues.
(ii) $\widehat{X}^{-1}B$ and $\widehat{S} = (I_{r_b} - T_{ba}\widehat{Y})^{-1} T_{bb} R_b$ have the same nonzero eigenvalues.

Proof. (*i*) From (16), we have $X^{-1}A = (Q - F_b Y G_a^H)^{-1} F_a R_a G_a^H$. Applying the SMWF to $(Q - F_b Y G_a^H)^{-1}$ yields

$$X^{-1}A = (Q^{-1}F_a + Q^{-1}F_b Y (I_{r_a} - G_a^H Q^{-1} F_b Y)^{-1} G_a^H Q^{-1} F_a) R_a G_a^H. \tag{33}$$

Let $G = [G_a, \widehat{G}_a]$ be a unitary matrix. Multiply G^H and G from the left and right of (33), respectively, we obtain

$$G^H X^{-1} A G = \begin{bmatrix} Z & 0 \\ \star & 0 \end{bmatrix},$$

where

$$Z = \left[T_{aa} + T_{ab} Y (I_{r_a} - T_{ab}Y)^{-1} T_{aa} \right] R_a = (I_{r_a} - T_{ab}Y)^{-1} T_{aa} R_a = S.$$

Hence, $X^{-1}A$ and $S = (I_{r_a} - T_{ab}Y)^{-1} T_{aa} R_a$ have the same nonzero eigenvalues.

The proof of (*ii*) is similar to the proof of (*i*). □

As mentioned before, we may require the eigenvalues of $X^{-1}A$ or $\widehat{X}^{-1}B$ in some applications, and Theorem 3.3 provides an efficient route to the nonzero parts of these spectra via the much smaller matrices S and $\widehat{S}$.

Suppose that $\widehat{X} = Q - F_a\widehat{Y}G_b^H$ is a stabilizing solution of (3). Then the kernel $\widehat{Y} \in \mathbb{C}^{r_a\times r_b}$ and $\widehat{S} = (I_{r_b} - T_{ba}\widehat{Y})^{-1}T_{bb}R_b$ satisfy

$$\begin{bmatrix} T_{bb}R_b & 0 \\ -R_aT_{ab}R_b & I_{r_a} \end{bmatrix}\begin{bmatrix} I_{r_b} \\ \widehat{Y} \end{bmatrix} = \begin{bmatrix} I_{r_b} & -T_{ba} \\ 0 & R_aT_{aa} \end{bmatrix}\begin{bmatrix} I_{r_b} \\ \widehat{Y} \end{bmatrix}\widehat{S}.$$

Let

$$\widetilde{Y} = R_a^{-1}\widehat{Y}R_b^{-1}, \quad \widetilde{S} = R_b\widehat{S}R_b^{-1}. \tag{34}$$

It is easily seen that

$$M\begin{bmatrix} \widetilde{Y} \\ I_{r_b} \end{bmatrix}\widetilde{S} = L\begin{bmatrix} \widetilde{Y} \\ I_{r_b} \end{bmatrix}, \tag{35}$$

where M and L are defined in (31). It follows from (32), (34), (35) and Theorem 3.3 that the matrix pencil $\lambda L - M$ has r_a eigenvalues inside the unit circle and r_b eigenvalues outside the unit circle. Similar to the theory in [5,15], if the matrix sequences $\{\widetilde{T}_{aa,k}\}$, $\{\widetilde{T}_{bb,k}\}$, $\{\widetilde{T}_{ab,k}\}$ and $\{\widetilde{T}_{ba,k}\}$ generated by Algorithm 2 are well-defined, then we have

$$\|\widetilde{T}_{ba,k} - Y\| \simeq O(\rho(S)^{2^k}\rho(\widetilde{S})^{2^k}), \quad \|\widetilde{T}_{ab,k} - \widehat{Y}\| \simeq O(\rho(S)^{2^k}\rho(\widetilde{S})^{2^k}), \tag{36}$$

where $S = (I_{r_a} - T_{ab}Y)^{-1}T_{aa}R_a$ and $\widetilde{S}$ is defined in (34) with $\rho(S) < 1$ and $\rho(\widetilde{S}) < 1$. For Algorithm 1, it has been shown in [13] that

$$\begin{aligned} \|R_{q,k} - Y\| &= \|(Q - F_bR_{q,k}G_a^H) - (Q - F_bYG_a^H)\| \simeq O(\rho(\widehat{X}^{-1}B)^{2^k}\rho(X^{-1}A)^{2^k}), \\ \|R_{p,k} - \widehat{Y}\| &= \|(Q - F_aR_{p,k}G_b^H) - (Q - F_a\widehat{Y}G_b^H)\| \simeq O(\rho(\widehat{X}^{-1}B)^{2^k}\rho(X^{-1}A)^{2^k}), \end{aligned} \tag{37}$$

where $\{R_{q,k}\}$ and $\{R_{p,k}\}$ are generated by Algorithm 1. From (36), (37) and Theorem 3.3, it is easily seen that Algorithms 1 and 2 converge in the same number of iterations.

It is intriguing, that we started from an NME associated with the SDA of type 2 and ended up with an equivalent NARE_D associated with the SDA of type 1. Similar links between NMEs and AREs have been considered before. In [10], an NME has been transformed to a discrete-time ARE when $B = A^*$. The transformation of an NARE into a unilateral quadratic matrix polynomial (UQME), which was then solved by the SDA of type 2, has recently been studied in [4].

3.4. Errors in SDA_ls

It is easy to see from (6) that errors in the iterates will propagate through the SDA. Let δA_k, δB_k, δP_k and δQ_k be the errors in A_k, B_k, P_k and Q_k, respectively. From (6), with $\Delta_k \equiv \delta Q_k - \delta P_k$, and ignoring higher order terms, we have

$$\begin{aligned} \delta A_{k+1} &\approx \delta A_kM_k^{-1}A_k - A_kM_k^{-1}\Delta_kM_k^{-1}A_k + A_kM_k^{-1}\delta A_k, \\ \delta B_{k+1} &\approx \delta B_kM_k^{-1}B_k - B_kM_k^{-1}\Delta_kM_k^{-1}B_k + B_kM_k^{-1}\delta B_k, \\ \delta P_{k+1} &\approx \delta P_k + \delta A_kM_k^{-1}B_k - A_kM_k^{-1}\Delta_kM_k^{-1}B_k + A_kM_k^{-1}\delta B_k, \\ \delta Q_{k+1} &\approx \delta Q_k - \delta B_kM_k^{-1}A_k + B_kM_k^{-1}\Delta_kM_k^{-1}A_k - B_kM_k^{-1}\delta A_k. \end{aligned}$$

With $\delta_k \equiv \max\{\|\delta A_k\|, \|\delta B_k\|, \|\delta P_k\|, \|\delta Q_k\|\}$, $c_{ak} \equiv \|M_k^{-1}\|\|A_k\|$ and $c_{bk} \equiv \|M_k^{-1}\|\|B_k\|$, we have

$$\delta_{k+1} \leqslant \delta_k \cdot \max\{2c_{ak}(1+c_{ak}),\ 2c_{bk}(1+c_{bk}),\ 1+(c_{ak}+c_{bk})^2\} + O(\delta_k^2).$$

From Theorem 2.3, we have the upper bounds of $\{\|M_k^{-1}\|_2 \mid k = 0, 1, \ldots\}$ dependent only on $\phi_0(z)$, or from (9), only on A, B and Q. Using the fact that $\|A_k\|$ and $\|B_k\|$ are also bounded in [13], the errors δA_k, δB_k, δP_k and δQ_k then pass into δA_{k+1}, δB_{k+1}, δP_{k+1} and δQ_{k+1}, creating errors of the same order. Note that ignoring the higher terms simplifies the error equations, without altering the conclusions of the discussion. The fact that A_k, $B_k \to 0$, or c_{ak}, $c_{bk} \to 0$, will contribute towards diminishing the errors.

4. Computational issues

4.1. Residual and convergence control

For the convergence control in Algorithm 1, we should compute residuals and differences of iterates carefully. Note that it is much easier to compute the smaller analogous quantities in Algorithm 2.

Consider the differences of successive iterates:

$$dQ_k \equiv Q_{k+1} - Q_k = F_b(R_{qk} - R_{q,k+1})G_a^H,$$
$$dP_k \equiv P_{k+1} - P_k = F_a(R_{p,k+1} - R_{pk})G_b^H,$$

implying that

$$\|dQ_k\| = \|R_{qk} - R_{q,k+1}\|, \quad \|dP_k\| = \|R_{pk} - R_{p,k+1}\|$$

in 2- or F-norm, because the Fs and Gs are unitary by choice. The computations of $\|dQ_k\|$ and $\|dP_k\|$ can be achieved in about $8r_a r_b \hat{r}_{ab}$ (see [11, p. 254]) and $4r_a r_b$ flops for 2-norm and F-norm, respectively, where $\hat{r}_{ab} = \max\{r_a, r_b\}$.

Similarly, we have the residual $r_k \equiv \|\mathcal{R}(Q_k)\|$ of the NME, the corresponding relative residual equals

$$\widetilde{r}_k \equiv \frac{r_k}{q_k + m_k} \tag{38}$$

with

$$q_k \equiv \|Q - Q_k\| = \|\widehat{Q}_k\|, \quad m_k \equiv \|BQ_k^{-1}A\|. \tag{39}$$

With the low-rank forms in (17), we then have

$$r_k = \| - F_b R_{qk} G_a^H + F_b R_b G_b^H (Q - F_b R_{qk} G_a^H)^{-1} F_a R_a G_a^H \|,$$
$$q_k = \|F_b R_{qk} G_a^H\|,$$
$$m_k = \|F_b R_b G_b^H (Q - F_b R_{qk} G_a^H)^{-1} F_a R_a G_a^H\|.$$

After applying the SMWF, using the notation in (21), we have the efficient formulae

$$r_k = \left\| -R_{qk} + R_b \left[T_{ba} + T_{bb}(I_{r_b} - R_{qk}T_{ab})^{-1} R_{qk} T_{aa} \right] R_a \right\|,$$
$$q_k = \|R_{qk}\|,$$
$$m_k = \left\| R_b \left[T_{ba} + T_{bb}(I_{r_b} - R_{qk}T_{ab})^{-1} R_{qk} T_{aa} \right] R_a \right\|. \tag{40}$$

The computation of relative residual requires about $6r_a^2r_b + 4r_ar_b^2 + \frac{8}{3}r_b^3$ flops, assuming the Ts are available, and using F-norm in (40). If we use 2-norm, then an additional $12r_ar_b\hat{r}_{ab}$ flops are required. Notice that the computation of relative residual in Algorithm 2 requires $\frac{8}{3}r_a^3 + 4r_a^2r_b + 2r_ar_b^2$ flops.

On the relation between residuals and actual errors in the computed solutions, please consult [27,29].

4.2. Operation and memory counts

In Algorithms 1 and 2, the dominant calculations of $O(n)$ computational complexity and memory requirement are in the pre-processing, with help from the structure of Q. We shall assume that c_qn flops are required in the solution of $Qv = r$ or $Q^Hv = r$, with $v, r \in \mathbb{C}^n$.

For the pre-processing, the cost of $2(c_q + r_a + r_b)(r_a + r_b)n$ flops is made up of the following:

(1) compute $Q^{-1}U$ and V^HQ^{-1} ($U = F_a, F_b$ and $V = G_a, G_b$), requiring $2c_q(r_a + r_b)n$ flops; and
(2) compute $V^HQ^{-1}U$ ($U = F_a, F_b$ and $V = G_a, G_b$), requiring $2(r_a + r_b)^2n$ flops.

There is also a memory requirement for $(r_a + r_b)(r_a + r_b + 2)n$ variables. In addition, there may be up to $8(r_a^2 + r_b^2)n$ flops required for the orthogonalization of F_u, G_v ($u, v = a, b$) [11, p. 250], if required and dependent on the exact structures in A and B.

After the pre-processing of data, it is easily seen that the memory requirement is about $O((r_a+r_b)^2)$ variables, per iteration in both Algorithms 1 and 2. The memory requirement of each iteration is much smaller than $(r_a + r_b)(r_a + r_b + 2)n$ (the memory required in the pre-processing). Hence, we do not need to count the memory requirement for each iteration in Algorithms 1 and 2. In Algorithm 2, we need to compute $\widetilde{T}_{u,v}$ ($u, v = a, b$) as in (30), which requires $2r_a^3 + 2r_b^3 + 2r_ar_b(r_a + r_b)$ flops, before starting the iteration.

The detailed operation counts for the kth iteration in the SDA for large-scale NMEs of Algorithms 1 and 2 are summarized in Tables 1 and 2 below, respectively, with all the kernels formed explicitly. Only the dominant counts are recorded and the F-norm is applied. When $B = A^\star$ and Q is Hermitian, the workload and memory requirement will be halved.

Table 1
Operation counts for the kth iteration in Algorithm 1 (SDA_ls).

Computation	Flops
$R_{mk}T$	$4r_ar_b(r_a + r_b)$
$(I - R_{mk}T)^{-1}R_{mk}T$	$\frac{8}{3}(r_a + r_b)^3$
W_k^{uv} ($u, v = a, b$)	$2(r_a + r_b)^3$
$R_{a,k+1}, R_{b,k+1}$	$4(r_a^3 + r_b^3)$
$R_{q,k+1}, R_{p,k+1}$	$4r_ar_b(r_a + r_b)$
r_k, q_k, m_k	$6r_a^2r_b + 4r_ar_b^2 + \frac{8}{3}r_b^3$
Total	$\frac{26}{3}r_a^3 + 28r_a^2r_b + 26r_ar_b^2 + \frac{34}{3}r_b^3$

Table 2
Operation counts for the kth iteration in Algorithm 2 (SDA_ls).

Computation	Flops
$\widetilde{T}_{ab,k}\widetilde{T}_{ba,k}$	$2r_a^2r_b$
$\widetilde{T}_{ba,k}\widetilde{T}_{ab,k}$	$2r_ar_b^2$
$\widetilde{T}_{aa,k+1}, \widetilde{T}_{ba,k+1}$	$\frac{14}{3}r_a^3 + 2r_ar_b(r_a + r_b)$
$\widetilde{T}_{bb,k+1}, \widetilde{T}_{ab,k+1}$	$\frac{14}{3}r_b^3 + 2r_ar_b(r_a + r_b)$
r_k, q_k, m_k	$\frac{8}{3}r_a^3 + 4r_a^2r_b + 2r_ar_b^2$
Total	$\frac{22}{3}r_a^3 + 10r_a^2r_b + 8r_ar_b^2 + \frac{14}{3}r_b^3$

5. Numerical examples

All the numerical experiments were conducted using MATLAB 2010a on an iMac with a 2.97 GHz Intel i7 processor and 8 Gigabyte RAM. To measure the accuracy of a computed stabilizing solution X of NME (1), we use the relative and absolute residuals:

$$\text{RRes} \equiv \frac{\|X + BX^{-1}A - Q\|}{\|X - Q\| + \|BX^{-1}A\|}, \quad \text{ARes} \equiv \|X + BX^{-1}A - Q\|. \tag{41}$$

Suppose that $R_{q\epsilon} = R_{qk_*}$ and $Y_\epsilon = \widetilde{T}_{ba,k_*}$ are the computed kernels by Algorithms 1 and 2, respectively, and k_* is the required number of iterations for convergence. From (38)–(40), we obtain that the relative and absolute residuals of the computed stabilizing solution $X = Q - F_b R_{q,\epsilon} G_a^H$ are $\text{RRes} = r_{k_*}/(q_{k_*} + m_{k_*})$ and $\text{ARes} = r_{k_*}$, respectively, where r_{k_*}, q_{k_*} and m_{k_*} are defined in (40). From (27)–(30), it is easily seen that the relative and absolute residuals of the computed stabilizing solution $X = Q - F_b Y_\epsilon G_a^H$ are $\text{RRes} = r_{k_*}/(q_{k_*} + m_{k_*} + t_{ab})$ and $\text{ARes} = r_{k_*}$, respectively, where r_{k_*}, q_{k_*}, m_{k_*} and t_{ab} are given in Algorithm 2. Furthermore, we use "Time 1" and "Time 2" to represent the execution times for the pre-processing data and SDA, respectively.

Example 1. In order to report the actual errors in the computed solution, we consider an example with exact solution. Suppose that $A = iD$ and $B = iD^H$ where $D \in \mathbb{C}^{n\times n}$ with $r_a = r_b = 3$. We randomly generate $R_D \in \mathbb{C}^{r_a\times r_a}$, $F_a,\ G_a \in \mathbb{C}^{n\times r_a}$ and set $F_a := F_a(F_a^H F_a)^{-\frac{1}{2}}$, $G_a := G_a(G_a^H G_a)^{-\frac{1}{2}}$ and $R_D = R_D/(4\|R_D\|_2)$. Assume that $D = F_a R_D G_a^H$, then A and B have the full-rank decompositions $A = F_a R_a G_a^H$ and $B = F_b R_b G_b^H$ where

$$G_b = F_a, \quad F_b = G_a, \quad R_a = iR_D, \quad R_b = iR_D^H.$$

Let $H \in \mathbb{C}^{n\times r_a}$ and set $H = H(H^H H)^{-\frac{1}{2}}$. Assume that

$$X_e = i(I_n - 0.5HH^H) \tag{42}$$

is the exact solution of NME. Then $X_e^{-1} = -i(I_n + HH^H)$ and

$$\begin{aligned} Q =& iQ_I = i(I_n - 0.5HH^H + G_a R_D^H F_a^H (I_n + HH^H) F_a R_D G_a^H) \\ =& i(I_n - 0.5HH^H + G_a R_D^H R_D G_a^H + G_a R_D^H F_a^H HH^H F_a R_D G_a^H). \end{aligned}$$

Since $\|D\|_2 = \|R_D\|_2 = 1/4$ and $\sigma_{\min}(Q_I) \geqslant \sigma_{\min}(I_n - 0.5HH^H) = 1/2$, it is easily seen that $\psi(z) = zD^H + Q_I + z^{-1}D$ is positive definite for each $z \in \mathbb{T}$ and $\rho(X_e^{-1}A) < 1$. That is, X_e in (42) is the stabilizing solution of $X + BX^{-1}A = Q$.

We compute the stabilizing solution X with $n = 10^2$, 5×10^2, 10^3 and 5×10^3, respectively, by using Algorithm 1 with positive tolerance $\epsilon = 10^{-10}$. The numerical results are shown in Table 3.

Example 2. We first consider an example associated with the computation of Green function in nano research [13,16].

Table 3
(Example 1) Numbers of iterations, absolute residuals, relative residuals and $\|X - X_e\|_2$.

n	10^2	5×10^2	10^3	5×10^3
# iterations (k_*)	5	5	5	5
ARes	1.46×10^{-17}	1.75×10^{-17}	1.82×10^{-17}	1.39×10^{-17}
RRes	6.48×10^{-17}	7.86×10^{-17}	8.28×10^{-17}	6.35×10^{-17}
$\|X - X_e\|_2$	4.01×10^{-17}	1.11×10^{-16}	1.11×10^{-16}	1.11×10^{-16}

Table 4
(Example 2) Numbers of iterations, absolute residuals, relative residuals and execution times in seconds.

n	10^2	10^3	10^4	10^5	10^6	10^7
# iterations (k_*)	6	6	6	6	7	6
ARes	1.85×10^{-16}	2.36×10^{-16}	2.71×10^{-16}	2.07×10^{-16}	2.52×10^{-16}	1.83×10^{-16}
RRes	7.16×10^{-17}	8.14×10^{-17}	8.92×10^{-17}	7.16×10^{-17}	6.96×10^{-17}	6.60×10^{-17}
Time 1	1.65×10^{-3}	3.65×10^{-3}	1.57×10^{-2}	1.24×10^{-1}	9.94×10^{-1}	133.16
Time 2	5.06×10^{-3}	7.05×10^{-3}	7.16×10^{-3}	7.76×10^{-3}	8.62×10^{-4}	7.41×10^{-3}

Table 5
(Example 2) $O(n)$ Computational Complexity.

n	1×10^6	2×10^6	3×10^6	4×10^6	5×10^6	6×10^6
# iterations (k_*)	7	6	6	6	6	6
ARes	2.52×10^{-16}	1.98×10^{-16}	2.46×10^{-16}	1.85×10^{-16}	2.10×10^{-16}	1.89×10^{-16}
RRes	6.96×10^{-17}	7.19×10^{-17}	9.86×10^{-17}	6.76×10^{-17}	5.95×10^{-17}	7.26×10^{-17}
Time 1	0.994	1.89	2.81	3.75	4.70	5.69
Time 2	8.62×10^{-4}	5.08×10^{-3}	6.27×10^{-3}	8.17×10^{-3}	5.16×10^{-4}	8.47×10^{-4}

With $r_a = 3$, and $r_b = 5$ and we randomly generate $R_a \in \mathbb{C}^{r_a \times r_a}$, $R_b \in \mathbb{C}^{r_b \times r_b}$, $F_a,\ G_a \in \mathbb{C}^{n \times r_a}$ and $F_b,\ G_b \in \mathbb{C}^{n \times r_b}$ and set $F_u := F_u(F_u^H F_u)^{-\frac{1}{2}}$, $G_u := G_u(G_u^H G_u)^{-\frac{1}{2}}$ $(u = a, b)$. Then F_u, G_u $(u = a, b)$ are unitary. Recall that A and B have the forms

$$A = F_a R_a G_a^H, \quad B = F_b R_b G_b^H.$$

Let Q be the tridiagonal matrix of dimension n with 2 on the main diagonal and -1 on the two adjacent diagonals. Choose a suitable $\varrho \in \mathbb{R}$ and set $Q := Q + i\varrho I$ such that A, B and Q satisfy the solvability condition which is given in Theorem 2.1, i.e.,

$$\psi(\lambda) = \varrho I + \lambda D + \lambda^{-1} D^H > 0, \quad \text{for all } \lambda \in \mathbb{T}, \tag{43}$$

where $D = (A - B^H)/(2i)$. In the following numerical experiments, we choose $\varrho = 5$. We compute the stabilizing solution X with $n = 10^2, 10^3, 10^4, 10^5, 10^6$ and 10^7, by using Algorithm 1 with the positive tolerance $\epsilon = 10^{-10}$. The numerical results are shown in Table 4. Somehow, when the size of the problem jumped from $n = 10^6$ to 10^7, the memory requirement crossed some critical boundary concerning virtual memory on the computer. The amount of execution time jumped 134-folds, instead of the previous 1.9- to 8-folds when n was increased 10-folds. Anyway, Algorithm 1 was successful in solving the associated NMEs to machine accuracy (in terms of residuals), for $n = 10^2$ to 10^6, in reasonably quick execution times. This is consistent with the predicted $O(n)$ computational complexity and memory requirement for the SDA.

To see the $O(n)$ computational complexity more clearly, we construct Table 5 for $n = j \times 10^6$ $(j = 1 : 6)$, with the corresponding execution time equals approximately to j seconds.

The numerical results from Algorithm 2 is similar.

Example 3.

Next we consider a small example associated with the surface acoustic wave simulation [18,19]. Here we have

$$A_1 = GM_2^{-\top} F^\top, \quad A_0 = FM_2^{-1} F^\top + GM_2^{-1} G^\top - M_1$$

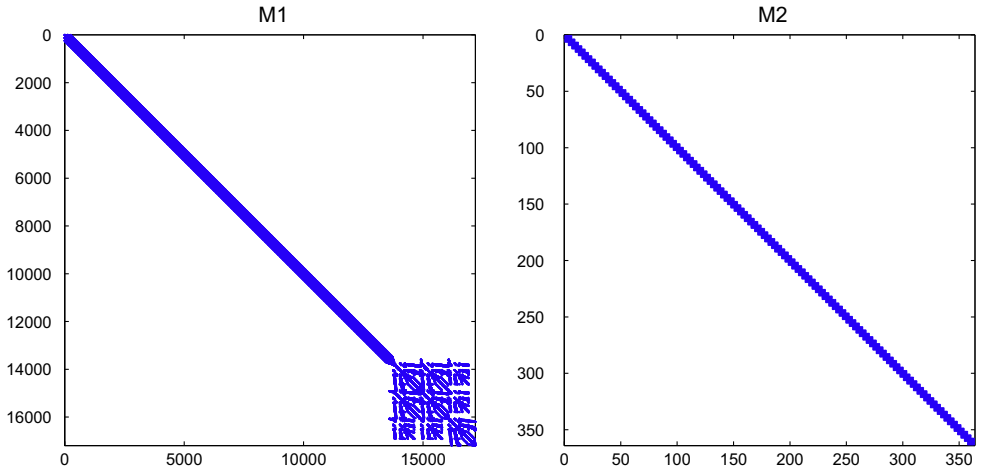

Fig. 1. (Example 3) Sparsity patterns in M_1 and M_2.

Table 6
(Example 3) Numbers of iterations, absolute/relative residuals and execution times in seconds.

Algorithm	# iterations	ARes	RRes	Time 1	Time 2
1	18	9.88×10^{-13}	4.84×10^{-14}	39.80	27.13
2	18	3.40×10^{-13}	1.32×10^{-14}	39.80	7.562

with $n = 17192$, $r = 363$, $M_1 \in \mathbb{C}^{n\times n}$, $M_2 \in \mathbb{C}^{r\times r}$, and $F, G \in \mathbb{C}^{n\times r}$. We would like to solve for selected eigenvalues of the palindromic quadratic

$$(\lambda^2 A_1^\top + \lambda A_0 + A_1)x = 0, \quad x \neq 0$$

via the NME

$$X + A_1^\top X^{-1} A_1 - A_0 = 0.$$

Note that A_0 is sparse-like, in the sense that the associated linear systems can be solved in $O(n)$ computational complexity with the help of the SMWF. The sparsity patterns in M_1 and M_2 can be found in Fig. 1.

In this case, all iterations in Algorithm 2 are well-defined. Hence, the convergence of the SDA in Algorithm 2 is guaranteed by Theorem 3.3 (see the discussion following the theorem) or the less general results in [5,8,9]. The numerical results are shown in Table 6. The results from Algorithms 1 and 2 are very similar, except for a small advantage in the relative residual and execution time for Algorithm 2. It is well-known that the execution times from MATLAB are not that reliable and should be used as a rough guide only.

6. Conclusions

For the solution of NMEs for many applications, the problems are naturally large-scale. We have shown that these problems are equivalent to, or can be solved as, much smaller nonlinear matrix equations. Apart from the pre-processing of $O(n)$ computational complexity and memory requirement, the resulting structure-preserving doubling algorithm turns out to be very efficient, of $O((r_a + r_b)^3)$ computational complexity per iteration. We have presented some numerical results illustrating the feasibility and efficiency of the algorithms.

For really large problems, the solution of various linear systems by inexact solvers changes the nature of the algorithms significantly. This raises additional challenges and research possibilities, and may be something for the future.

In order to limit the length of the paper, we have not considered the fast train problem [8,17,20,26], which is similar to the surface acoustic wave simulation in Example 3 with some differences in structures. In this application, we have $B = A^\top$ being complex with only the upper right corner of A being nonzero and Q being complex symmetric, such that $\psi(z) > 0$ for each $z \in \mathbb{T}$ (for solvability). The eigenvalues of $\lambda X - A$ are required. Our techniques here can be applied on the application but we have ignored the details here.

Acknowledgements

The first author has been supported by a Monash Graduate Scholarship and a Monash International Postgraduate Research Scholarship. Part of the work was completed when the second author visited the National Centre for Theoretical Sciences at Tainan and the National Chiao Tung University, and we would like to thank these institutions. The third and fourth authors would like to acknowledge the support from the National Science Council and the National Centre for Theoretical Sciences, Taiwan. The fourth author would also like to thank the ST Yau Centre and the Centre for Mathematical Modelling and Scientific Computing at the National Chiao Tung University for its support.

Last but not least, we would like to thank the referees for their helpful suggestions and comments.

References

[1] B. Anderson, Second-order convergent algorithm for the steady-state Riccati equation, Internat. J. Control 28 (1978) 295–306.
[2] B.D.O. Anderson, J.B. Moore, Optimal Filtering, Prentice-Hall, Englewood Cliffs, NJ, 1979.
[3] D.A. Bini, L. Gemignani, B. Meini, Computations with infinite Toeplitz matrices and polynomials, Linear Algebra Appl. 343–344 (2002) 21–61.
[4] D.A. Bini, B. Meini, F. Poloni, Transforming algebraic Riccati equations into unilateral quadratic matrix equations, Numer. Math. 116 (2010) 553–578.
[5] C.-Y. Chiang, E.H.-W. Chu, C.-H. Guo, T.-M. Huang, W.-W. Lin, S.-F. Xu, Convergence analysis of the doubling algorithm for several nonlinear matrix equations in the critical case, SIAM J. Matrix Anal. Appl. 31 (2009) 227–247.
[6] E.K.-W. Chu, H.-Y. Fan, W.-W. Lin, A structure-preserving doubling algorithm for continuous-time algebraic Riccati equations, Linear Algebra Appl. 396 (2005) 55–80.
[7] E.K.-W. Chu, H.-Y. Fan, W.-W. Lin, C.-S. Wang, A structure-preserving doubling algorithm for periodic discrete-time algebraic Riccati equations, Internat. J. Control 77 (2004) 767–788.
[8] E.K.-W. Chu, T.M. Huang, W.-W. Lin, C.-T. Wu, Vibration of fast trains, palindromic eigenvalue problems and structure-preserving doubling algorithms, J. Comput. Appl. Math. 219 (2007) 237–252.
[9] E.K.-W. Chu, T.M. Huang, W.-W. Lin, C.-T. Wu, Palindromic eigenvalue problems: a brief survey, Taiwanese J. Math. 14 (2010) 743–779.
[10] J.C. Engwerda, A.C.M. Ran, A.L. Rijkeboer, Necessary and sufficient conditions for the existence of a positive definite solution of the matrix equation $X + A^*X^{-1}A = Q$, Linear Algebra Appl. 186 (1993) 255–275.
[11] G.H. Golub, C.F. Van loan, Matrix Computations, second ed., Johns Hopkins University Press, Baltimore, MD, 1989.
[12] C.-H. Guo, Y.-C. Kuo, W.-W. Lin, On a nonlinear matrix equation arising in nano research, SIAM Matrix Anal. Appl. 33 (2012) 235–262.
[13] C.-H. Guo, Y.-C. Kuo, W.-W. Lin, Numerical solution of nonlinear matrix equations arising from Green's function calculations in nano research, J. Comput. Appl. Math. 236 (2012) 4166–4180.
[14] C.-H. Guo, Y.-C. Kuo, W.-W. Lin, Complex symmetric stabilizing solution of the matrix equation $X + A^\top X^{-1}A = Q$, Linear Algebra Appl. 435 (2011) 1187–1192.
[15] X.-X. Guo, W.-W. Lin, S.-F. Xu, A structure-preserving doubling algorithm for nonsymmetric algebraic Riccati equations, Numer. Math. 103 (2006) 393–412.
[16] C.-H. Guo, W.-W. Lin, The matrix equation $X + A^\top X^{-1}A = Q$ and its application in nano research, SIAM J. Sci. Comput. 32 (2010) 3020–3038.
[17] T.-M. Huang, W.-W. Lin, J. Qian, Structure-preserving algorithms for palindromic quadratic eigenvalue problems arising from vibration of fast trains, SIAM J. Matrix Anal. Appl. 30 (2009) 1566–1592.
[18] T.-M. Huang, W.-W. Lin, C.-T. Wu, Structure-preserving Arnoldi-type algorithms for solving palindromic quadratic eigenvalue problems in leaky surface wave propagation, Technical Report, NCTS Preprints in Mathematics, 2011-2-001, National Tsing Hua University, Hsinchu, Taiwan. Available from: <http://www.math.cts.nthu.edu.tw/publish/publish.php?class=102>.
[19] T.-M. Huang, C.-T. Wu, T. Li, Numerical studies on structure-preserving algorithm for surface acoustic wave simulations, Technical Report, NCTS Preprints in Mathematics, 2012-5-006, National Tsing Hua University, Hsinchu, Taiwan. Available from: <http://www.math.cts.nthu.edu.tw/publish/publish.php?class=102>.
[20] I.C.F. Ipsen, Accurate eigenvalues for fast trains, SIAM News 37 (2004) 1–2.

[21] T. Li, E.K.-W. Chu, W.-W. Lin, Solving large-scale discrete-time algebraic Riccati equations by doubling, Technical Report, NCTS Preprints in Mathematics, 2012-5-002, National Tsing Hua University, Hsinchu, Taiwan. Available from: <http://www.math.cts.nthu.edu.tw/publish/publish.php?class=102>.
[22] T. Li, E.K.-W. Chu, W.-W. Lin, C.-Y. Weng, Solving large-scale continuous-time algebraic Riccati equations by doubling, Technical Report, NCTS Preprints in Mathematics, 2012-5-001, National Tsing Hua University, Hsinchu, Taiwan. Available from: <http://www.math.cts.nthu.edu.tw/publish/publish.php?class=102>.
[23] T. Li, C.-Y. Weng, E.K.-W. Chu, W.-W. Lin, Solving large-scale Stein and Lyapunov equations by doubling, Technical Report, NCTS Preprints in Mathematics, 2012-5-005, National Tsing Hua University, Hsinchu, Taiwan. Available from: <http://www.math.cts.nthu.edu.tw/publish/publish.php?class=102>.
[24] W.-W. Lin, S.-F. Xu, Convergence analysis of structure-preserving doubling algorithms for Riccati-type matrix equations, SIAM J. Matrix Anal. Appl. 28 (2006) 26–39.
[25] B. Meini, Efficient computation of the extreme solutions of $X + A^*X^{-1}A = Q$ and $X - A^*X^{-1}A = Q$, Math. Comp. 71 (2002) 1189–1204.
[26] C.-H. Guo, W.-W. Lin, Solving a structured quadratic eigenvalue problem by a structure-preserving doubling algorithm, SIAM J. Matrix Anal. Appl. 31 (2010) 2784–2801.
[27] J.-G. Sun, S.-F. Xu, Perturbation analysis of the maximal solution of the matrix equation $X + A^*X^{-1}A = P$, Linear Algebra Appl. 362 (2003) 211–228.
[28] C.-Y. Weng, T. Li, E.K.-W. Chu, W.-W. Lin, Solving large-scale nonsymmetric algebraic Riccati equations by doubling, Technical Report, NCTS Preprints in Mathematics, 2012-5-004, National Tsing Hua University, Hsinchu, Taiwan. Available from: <http://www.math.cts.nthu.edu.tw/publish/publish.php?class=102>.
[29] S.-F. Xu, Perturbation analysis of the maximal solution of the matrix equation $X + A^*X^{-1}A = P$, Linear Algebra Appl. 336 (2001) 61–70.