A new look at the doubling algorithm for a structured palindromic quadratic eigenvalue problem

Linzhang Lu1,2, Fei Yuan1 and Ren-Cang Li3,*,†

1School of Mathematical Science, Xiamen University, Xiamen, 361005, China
2School of Mathematics and Computer Science, Guizhou Normal University, Guiyang, 550001, China
3Department of Mathematics, University of Texas at Arlington, Arlington, TX 76019, USA

SUMMARY

Recently, Guo and Lin [SIAM J. Matrix Anal. Appl., 31 (2010), 2784–2801] proposed an efficient numerical method to solve the palindromic quadratic eigenvalue problem (PQEP) \( \langle \lambda^2 A^T + \lambda Q + A \rangle z = 0 \) arising from the vibration analysis of high speed trains, where \( A, Q \in \mathbb{C}^{n \times n} \) have special structures: both \( Q \) and \( A \) are, among others, \( m \times m \) block matrices with each block being \( k \times k \) (thus, \( n = mk \)), and moreover, \( Q \) is block tridiagonal, and \( A \) has only one nonzero block in the \((1, m)\)th block position. The key intermediate step of the method is the computation of the so-called stabilizing solution to the \( n \times n \) nonlinear matrix equation \( X + A^T X^{-1} A = Q \) via the doubling algorithm. The aim of this article is to propose an improvement to this key step through solving a new nonlinear matrix equation having the same form but of only \( k \times k \) in size. This new and much smaller matrix equation can also be solved by the doubling algorithm. For the same accuracy, it takes the same number of doubling iterations to solve both the larger and the new smaller matrix equations, but each doubling iterative step on the larger equation takes about 4.8 as many flops than the step on the smaller equation. Replacing Guo’s and Lin’s key intermediate step by our modified one leads to an alternative method for the PQEP. This alternative method is faster, but the improvement in speed is not as dramatic as just for solving the respective nonlinear matrix equations and levels off as \( m \) increases. Numerical examples are presented to show the effectiveness of the new method. Copyright © 2014 John Wiley & Sons, Ltd.

Received 4 February 2014; Revised 6 September 2014; Accepted 5 October 2014

KEY WORDS: palindromic quadratic eigenvalue problem; nonlinear matrix equation; solvent approach; doubling algorithm; vibration analysis of high speed train

1. INTRODUCTION

The palindromic quadratic eigenvalue problem (PQEP) \([1–3]\) is to find scalars \( \lambda \) and nonzero vectors \( z \) such that

\[
P(\lambda)z \equiv (\lambda^2 A^T + \lambda Q + A)z = 0,
\]

where \( A \) and \( Q \) are \( n \times n \) (real or complex) matrices and \( Q^T = Q \) (complex symmetric). When (1.1) holds for a scalar \( \lambda \) and a vector \( z \neq 0 \), we call \( \lambda \) a quadratic eigenvalue and \( z \) a corresponding quadratic eigenvector. The word ‘quadratic’ before eigenvalue and eigenvector is often dropped when no confusion arises. Counting multiplicities, it has \( 2n \) eigenvalues, some of which may be infinities.‡ It can be seen that the eigenvalues of PQEP (1.1) come in reciprocal pairs \( \{\lambda, 1/\lambda\} \).

*Correspondence to: Ren-Cang Li, Department of Mathematics, University of Texas at Arlington, PO Box 19408, Arlington, TX 76019, USA.
1E-mail: rcli@uta.edu
2Infinite eigenvalues are defined through the quadratic eigenvalue \( 0 \) of \( \lambda^2 P(1/\lambda) = A^T + \lambda Q + \lambda^2 A \).

Copyright © 2014 John Wiley & Sons, Ltd.
Our focus in this paper is on the PQEP arising from the vibration analysis of high speed trains [4–6], where

\[
\begin{align*}
Q & = K_t + \tau \omega D_t - \omega^2 M_t \in \mathbb{C}^{n \times n}, \\
A & = K_c + \tau \omega D_c - \omega^2 M_c \in \mathbb{C}^{n \times n},
\end{align*}
\]  

\( \tau \) is the imaginary unit, \( \omega > 0 \) is the frequency of the external excitation force, and \( K_t, D_t, M_t, K_c, D_c, M_c \) are real \( m \times m \) block matrices with each block being \( k \times k \):

\[
K_t = \begin{bmatrix} k & & & \cdots & & k \\ k & K_0 & & & & k \\ & K_1 & K_0 & & & \\ & & \ddots & K_1 & & \\ & & & \cdots & \ddots & k \\ k & & & & & k \end{bmatrix}, \quad M_t = \begin{bmatrix} k & & & \cdots & & k \\ k & M_0 & & & & k \\ & M_1 & M_0 & & & \\ & & \ddots & M_1 & & \\ & & & \cdots & \ddots & k \\ k & & & & & k \end{bmatrix},
\]

\[
K_c = \begin{bmatrix} k & & & \cdots & & k \\ k & 0 & & & & k \\ & k & K_0 & & & k \\ & & \ddots & k & & \\ & & & \cdots & \ddots & k \\ k & & & & & k \end{bmatrix} \times \begin{bmatrix} k & & & \cdots & & k \\ k & 0 & & & & k \\ & k & K_0 & & & k \\ & & \ddots & k & & \\ & & & \cdots & \ddots & k \\ k & & & & & k \end{bmatrix}, \
\]

\[
D_t = c_1 M_t + c_2 K_t, \quad D_c = c_1 M_c + c_2 K_c \quad \text{with} \quad c_1, c_2 > 0,
\]

and \( K_0 = K_0^T \). From (1.2a)–(1.3c), we see that \( Q \) is a complex symmetric, Toeplitz block tridiagonal matrix and \( A \) has only one nonzero block in its \((1, m)\)th block position.

This PQEP (1.1) has been providing much motivation for studying palindromic polynomial eigenvalue problems [3, 5, 7]. Several numerical methods were proposed [1, 2, 4, 5, 8], and some of them are not limited to PQEP of this form. For the case here, \( A \) is of very low rank, and thus, PQEP (1.1) has many infinite eigenvalues, creating numerical challenges. Most of the existing methods start by deflating out those infinite eigenvalues.

In [4, 5], the so-called solvent approach was explored for numerically solving PQEP (1.1). The approach consists of two steps:

1. Compute the stabilizing solution \( \Phi \) of the matrix equation

\[
X + A^T X^{-1} A = Q
\]

by the doubling algorithm [4], where \( A \) and \( Q \) are given by (1.2a)–(1.3c). By the stabilizing solution, we mean the spectral radius \( \rho(\Phi^{-1} A) < 1 \). Guo and Lin [5] showed that \( P(\lambda) \) with \( Q \) and \( A \) given by (1.2a)–(1.3c) has no eigenvalues on the unit circle and, thus, the stabilizing solution \( \Phi \) exists. \( \Phi \) is one of many possible solutions of (1.4), called the solvent matrices.

Any solvent matrix \( X \) gives rise to the following factorization for \( P(\lambda) \):

\[
P(\lambda) = \lambda^2 A^T + \lambda Q + A = (\lambda A^T + X) X^{-1} (\lambda X + A).
\]

2. Solve the (linear) eigenvalue problems for matrix pencils \( \lambda A^T + \Phi \) and \( \lambda \Phi + A \). Note that the eigenvalues of \( \lambda A^T + \Phi \) and those of \( \lambda \Phi + A \) enjoy the reciprocal relation: if \( \mu \) is an eigenvalue of one, then \( 1/\mu \) is an eigenvalue of the other.

Finally, the quadratic eigenvalues of \( P(\lambda) \) are the multiset union of the eigenvalues of the two matrix pencils \( \lambda A^T + \Phi \) and \( \lambda \Phi + A \). The eigenvectors of \( \lambda \Phi + A \) are also the quadratic eigenvectors of \( P(\lambda) \), but those of \( \lambda A^T + \Phi \) need to be processed to yield the corresponding quadratic eigenvectors of \( P(\lambda) \) [5]. This solvent approach works on the whole PQEP directly without deflating out the infinite eigenvalues, and numerical tests suggest that it delivers more accurate numerical solutions than other existing methods [5].
The complete Guo–Lin algorithm [5] is essentially this solvent approach with clever exploitations of the structures in \( A \) and \( Q \) to dramatically reduce the cost in solving the \( mk \times mk \) matrix equation (1.4) by the doubling algorithm. In this paper, we will exploit the structures even further by proposing a new and more efficient implementation for the part in computing the stabilizing solution \( \Phi \). It is made possible by an observation in the structure of any solution \( X \) of (1.4): it differs from \( Q \) only in the \((m, m)\)th block position. Thus, it may be unnecessary to solve (1.4) but potentially some matrix equation for determining the \((m, m)\)th block alone. We prove that this is indeed the case and devise a new matrix equation

\[
\tilde{X} + \tilde{A}^T \tilde{X}^{-1} \tilde{A} = \tilde{Q}
\]

(1.6)
of \( k \times k \) in size only just for determining that block. The new and much smaller matrix equation can also be solved by the doubling algorithm, and because of its much smaller size, it can be solved much faster.

We point out that the solvent approach for more general quadratic eigenvalue problems was explored before [9–13].

It is worth mentioning that both Guo–Lin algorithm [5] and our proposed one here are not limited to \( A \) and \( Q \) given by (1.2a)–(1.3c). In principle, they can be easily adapted for PQEP with \( Q \) being just block tridiagonal and \( A \) having just one nonzero block in its \((1, m)\)th block position, but the convergence analysis in [5] and here may no longer be applicable.

The rest of this paper is organized as follows. In Section 2, we review briefly the doubling algorithm that used in [5] for computing the stabilizing solution \( \Phi \) of (1.4). In Section 3, we exploit the structure of the stabilizing solution and devise a \( k \times k \) matrix equation (1.6) whose solution can be used to recover \( \Phi \). In Section 4, we outline implementation details about our modified method. Section 6 presents our numerical results comparing the method here with the Guo–Lin method in [5]. Finally, concluding remarks are given in Section 7.

Notation. \( \mathbb{C}^{n \times m} \) is the set of all \( n \times m \) complex matrices, \( \mathbb{C}^n = \mathbb{C}^{n\times 1} \), and \( \mathbb{C} = \mathbb{C}^1 \). \( I_n \) (or simply \( I \) if its dimension is clear from the context) is the \( n \times n \) identity matrix, and \( e_j \) is its \( j \)th column. The superscripts ‘\(^T\)’ and ‘\(^H\)’ take the transpose and complex conjugate transpose of a matrix or vector, respectively. We shall also adopt MATLAB-like convention to access the entries of vectors and matrices. Let \( i : j \) be the set of integers from \( i \) to \( j \) inclusive. For a vector \( u \) and a matrix \( X \), \( u(j) \) is \( u \)'s \( j \)th entry, and \( X(i, j) \) is \( X \)'s \((i, j)\)th entry; \( X \)'s submatrices \( X(k, l;i, j) \), \( X(k, l;i, :) \), and \( X(:, i; j) \) consist of intersections of row \( k \) to row \( \ell \) and column \( i \) to column \( j \), row \( k \) to row \( \ell \), and column \( i \) to column \( j \), respectively.

2. THE DOUBLING ALGORITHM

The key and novel step of the Guo–Lin solvent approach is to use the stabilizing solution \( X = \Phi \) of (1.4) to factorize \( P(\lambda) \) as in (1.5). Once (1.5) with \( X = \Phi \) is obtained, the QZ algorithm implemented in LAPACK [14] and in MATLAB as \texttt{eig(\ldots)} is indirectly applied to solve the eigenvalue problems\(^8\) for \( \lambda A^T + \Phi \) and \( \lambda \Phi + A \) whose combined eigensolutions gives the full spectral information for PQEP (1.1).

The doubling algorithm [15], Algorithm 1, was used to solve the matrix equation (1.4) for its stabilizing solution [4, 5].

We mentioned in section 1 that for the PQEP arising from the vibration analysis of high speed trains, (1.4) has the stabilizing solution \( \Phi \), which is unique [5]. Once \( \Phi \) is computed, the \( n \) eigenvalues of \( P(\lambda) \) inside the unique circle can be computed by solving the eigenvalues of \( \lambda \Phi + A \), and the other \( n \) eigenvalues, which are outside the unique circle, are their reciprocals.

\(^8\)By exploiting the sparsity structure of \( A \), Guo and Lin [5] showed how the \( mk \times mk \) eigenvalue problems can be solved via two eigenvalue problems of only \( k \times k \) in size.
Algorithm 1 The doubling algorithm for solving (1.4)

Given $A, Q = Q^T \in \mathbb{C}^{n \times n}$, this algorithm computes a solution of (1.4).

1: $A_0 = A, X_0 = Q, Y_0 = 0.$
2: for $i = 1, 2, \ldots$, until convergence do
3: $A_{i+1} = A_i (X_i - Y_i)^{-1} A_i$;
4: $X_{i+1} = X_i - A_i^T (X_i - Y_i)^{-1} A_i$;
5: $Y_{i+1} = Y_i + A_i (X_i - Y_i)^{-1} A_i^T$;
6: end for
7: return $X_i$ as the computed solution at convergence.

It is shown in [5] that $X_i$ generated by Algorithm 1 converges to the stabilizing solution $\Phi$ quadratically, and

$$\lim_{i \to \infty} \sqrt[i]{\|X_i - \Phi\|} \leq [\rho(\Phi^{-1} A)]^2, \quad (2.1)$$

where $\rho(\cdot)$ is the spectral radius of a matrix and $\| \cdot \|$ is any matrix norm. As a by-product, $Q - Y_i$ converges to the stabilizing solution $\Psi$ of the complementary matrix equation

$$Y + AY^{-1}A^T = Q \quad (2.2)$$

of (1.4), and

$$\lim_{i \to \infty} \sqrt[i]{\|Q - Y_i - \Psi\|} \leq [\rho(\Phi^{-1} A)]^2.$$  

Because of the quadratic convergence, $X_{i+1} - X_i$ is usually a good indicator of error in $X_i$ as an approximation to $\Phi$. This can be seen as follows. Let $\epsilon = \|X_{i+1} - X_i\|$. For $i$ sufficiently large such that the quadratic convergent behavior shows, that is,

$$\|X_{j+2} - X_{j+1}\| \leq \alpha \|X_{j+1} - X_j\|^2 \quad \text{for } j \geq i,$$

where $\alpha$ is some constant, then

$$\Phi - X_i = \sum_{j=i}^{\infty} (X_{j+1} - X_j) \Rightarrow \|\Phi - X_i\| \leq \epsilon + \sum_{j=1}^{\infty} \alpha \epsilon^{2j} = \epsilon + \frac{\alpha \epsilon^2}{1 - \epsilon^2} \approx \epsilon.$$

Therefore, a reasonable stopping criterion for Algorithm 1 to use at its line 2 is

$$\frac{\|X_{i+1} - X_i\|}{\|X_i\|} \leq \texttt{tol}, \quad (2.3)$$

where \texttt{tol} is a given relative tolerance, which, in our numerical tests, was set to a modest multiple of $\epsilon_u = 2^{-52}$, the machine unit roundoff of the IEEE double precision because our tests were carried out within MATLAB.

### 3. STRUCTURE OF THE STABILIZING SOLUTION

It is evident that the success of this solvent approach lies critically in whether the stabilizing solution $\Phi$ can be efficiently computed by the doubling algorithm. In [5], it was shown that, by taking advantage of the special sparse structures of the coefficient matrices $Q$ and $A$, each iterative step in Algorithm 1 costs about $(154/3)k^3$, despite that all involved matrices are $n \times n$, where $n = mk$.

In this section, we will show that the complexity of computing the solvent can be further reduced to solving a matrix equation having the same form as (1.4) but of only $k \times k$ instead of $n \times n$, the size of original (1.4). This is made possible by the observation that any solvent matrix of (1.4) is the same as $Q$ in all its blocks except the one in the $(m, m)$th block position.
For the ease of presentation, we partition \( A \) and \( Q \) in (1.2a) and (1.2b) as

\[
A = \begin{bmatrix}
 I_k \\
 0 \\
 0 \\
 0 \\
 A_{13}
\end{bmatrix}, \quad \begin{bmatrix}
 Q_{11} \\
 Q_{12} \\
 Q_{13} \\
 Q_{22} \\
 Q_{23}
\end{bmatrix}, \quad \begin{bmatrix}
 Q_{11} \\
 Q_{12} \\
 Q_{13} \\
 Q_{22} \\
 Q_{23}
\end{bmatrix}. \tag{3.1}
\]

Although not explicitly stated, it is known to Guo and Lin [5] that if \( X \), a solution to (1.4), is partitioned in the same way, then it differs from \( Q \) only in the bottom-right \( k \times k \) block as detailed in the following lemma.

\textbf{Lemma 3.1 ([5])}

Let \( X \) be a solution to (1.4), and partition \( X \) in the same way as in (3.1) for \( A \) and \( Q \). Then,

\[
X = \begin{bmatrix}
 Q_{11} \\
 Q_{12} \\
 Q_{13} \\
 Q_{22} \\
 Q_{23}
\end{bmatrix}, \tag{3.2}
\]

that is, \( X_{ij} = Q_{ij} \) for \( i, j = 1, 2, 3 \), except for \( i = j = 3 \).

The following two theorems are the main theoretical results of this paper.

\textbf{Theorem 3.1}

In Lemma 3.1, also partition \( X^{-1} \) in the same way as in (3.1) and (3.2) for \( A, Q \), and \( X \), respectively. Then, \( X_{33} \) and \( (X^{-1})_{11} \) satisfy

\[
X_{33} + A_{13}^T (X^{-1})_{11} A_{13} = Q_{33}, \tag{3.3}
\]

where \( (X^{-1})_{11} \in \mathbb{C}^{k \times k} \) is the \((1, 1)\)st block of \( X^{-1} \).

\textbf{Proof}

Because

\[
A = \begin{bmatrix}
 I_k \\
 0 \\
 0 \\
 0 \\
 A_{13}
\end{bmatrix},
\]

the matrix equation (1.4) becomes

\[
X + \begin{bmatrix}
 0 \\
 0 \\
 A_{13}
\end{bmatrix} (X^{-1})_{11} \begin{bmatrix}
 0 \\
 0 \\
 A_{13}
\end{bmatrix} = Q. \tag{3.4}
\]

Equate the corresponding blocks in the two sides of (3.4) to obtain (3.3) (and (3.2) as well). \( \square \)

Despite being \( n \times n \), Theorem 3.1 says that only the \( k \times k \) submatrix of \( \Phi \) in the lower-right corner needs to be computed; the rest of \( \Phi \) is known for free. But (3.3) does not have the same form as (1.4), and thus, the doubling algorithm is not readily applicable. The next theorem transforms (3.3) into a \( k \times k \) matrix equation, which does have the same form as (1.4) and thus makes the doubling algorithm applicable.

\textbf{Theorem 3.2}

Under the conditions of \textbf{Theorem 3.1}, let

\[
C_{22} = \begin{bmatrix}
 Q_{11} \\
 Q_{12} \\
 Q_{22}
\end{bmatrix}, \tag{3.5}
\]

\[C_{22} = \begin{bmatrix}
 Q_{11} \\
 Q_{12} \\
 Q_{22}
\end{bmatrix}. \tag{3.5}
\]
Suppose $Q_{11}$ and $C_{22}$ are nonsingular, and view $C_{22}^{-1} \in \mathbb{C}^{(m-1)k \times (m-1)k}$ as an $(m-1) \times (m-1)$ block matrix with each block being $k \times k$, and denote by $(C_{22}^{-1})_{ij}$ its $(i, j)$th block. Then, the matrix equation (3.3) can be transformed into

$$\tilde{X} + \tilde{A}^T \tilde{X}^{-1} \tilde{A} = \tilde{Q},$$  \hspace{1cm} (3.6)

where

$$\tilde{X} = X_{33} - A_{13}^T (C_{22}^{-1})_{m-1,m-1} A_{13},$$ \hspace{1cm} (3.7a)

$$\tilde{A} = A_{13} (C_{22}^{-1})_{1,1} A_{13},$$ \hspace{1cm} (3.7b)

$$\tilde{Q} = Q_{33} - A_{13}^T (C_{22}^{-1})_{1,1} A_{13} - A_{13}^T (C_{22}^{-1})_{m-1,m-1} A_{13}.$$ \hspace{1cm} (3.7c)

**Proof**

By Theorem 3.1,

$$X = \begin{pmatrix} \mathbf{C} & \mathbf{D} \\ \mathbf{D}^T & \mathbf{X}_{33} \end{pmatrix}$$

with

$$\mathbf{C}_{23} = \begin{pmatrix} 0 \\ \mathbf{Q}_{23} \end{pmatrix}.$$

It can be seen that $Q_{23} = A_{13}$ by (1.2a)–(1.3c). Recall\(^\dagger\)

$$X^{-1} = \begin{bmatrix} C_{22} & C_{23} \\ C_{23}^T & X_{33} \end{bmatrix}^{-1} = \begin{bmatrix} C_{22}^{-1} + C_{22}^{-1} C_{23} \tilde{C}_{22}^{-1} C_{22}^T C_{22}^{-1} - C_{22}^{-1} C_{23} \tilde{C}_{22}^{-1} \\ - \tilde{C}_{22}^{-1} C_{22}^T C_{22}^{-1} & \tilde{C}_{22}^{-1} \end{bmatrix}.$$

where

$$\tilde{C}_{22} = X_{33} - C_{23}^T C_{22}^{-1} C_{23} = X_{33} - \begin{bmatrix} 0 & A_{13} \end{bmatrix} C_{22}^{-1} \begin{bmatrix} 0 \\ A_{13}^T \end{bmatrix} = X_{33} - A_{13}^T (C_{22}^{-1})_{m-1,m-1} A_{13}.$$ \hspace{1cm} (3.10)

Noticing that $C_{22}^{-1}$ is symmetric because $C_{22}$ is symmetric, we have

$$(X^{-1})_{1,1} = \begin{bmatrix} C_{22}^{-1} + C_{22}^{-1} C_{23} \tilde{C}_{22}^{-1} C_{22}^T C_{22}^{-1} \\ - \tilde{C}_{22}^{-1} C_{22}^T C_{22}^{-1} \end{bmatrix}_{1,1} = (C_{22}^{-1})_{1,1} + \begin{bmatrix} C_{22}^{-1} C_{23} \tilde{C}_{22}^{-1} C_{22}^T C_{22}^{-1} \end{bmatrix}_{1,1} = (C_{22}^{-1})_{1,1} + \begin{bmatrix} 0 \\ A_{13}^T \end{bmatrix}_{1,1} \tilde{C}_{22}^{-1} [ \begin{bmatrix} 0 & A_{13} \end{bmatrix} C_{22}^{-1} ]_{1,1} = (C_{22}^{-1})_{1,1} + (C_{22}^{-1})_{1,m-1} A_{13}^T \tilde{C}_{22}^{-1} A_{13} (C_{22}^{-1})_{m-1,1}. \hspace{1cm} (3.11)$$

Plug in the expression for $\tilde{C}_{22}^{-1}$ in (3.10) into (3.11), and then plug in the resulting expression for $(X^{-1})_{1,1}$ into (3.3) to obtain (3.6). \(\Box\)

By Theorem 3.2, it suffices to solve the $k \times k$ matrix equation (3.6), instead of the $n \times n$ matrix equation (1.4) that is solved in [5], in order to compute the stabilizing solution $\Phi$. This is where we improve the Guo–Lin solvent approach [5].

\(^\dagger\)This is well known. See, for example, [16, pp. 102–103], [17, p. 23].
4. SOLVING THE MUCH SMALLER NONLINEAR MATRIX EQUATION

By Theorems 3.1 and 3.2, to compute the solvent matrix $\tilde{\Phi}$, we need to calculate the matrices $\tilde{A}$ and $\tilde{Q}$ in (3.7b) and (3.7c) and then solve the matrix equation (3.6) to recover $X_{33}$ by (3.7a) and, thus, $\Phi$.

Both $\tilde{A}$ and $\tilde{Q}$ are not readily available because of their dependency on some blocks $(C_{22}^{-1})_{ij}$. In what follows, we will first show how to set up equation (3.6) and then discuss its solution via the doubling algorithm.

4.1. Set up (3.6)

For this purpose, we have to compute

$$\left( C_{22}^{-1} \right)_{1,1}, \; \left( C_{22}^{-1} \right)_{1,m-1}, \; \text{and} \; \left( C_{22}^{-1} \right)_{m-1,m-1}$$

(4.1)

because $A_{13}$ and $Q_{33}$ are known.

First, we note that $C_{22}$ defined in (3.5) is the $(m - 1)k \times (m - 1)k$ principal submatrix of $Q$ in (1.2a). Recall (1.2a)–(1.3c), and let

$$H_0 = K_0 + i\omega D_0 - \omega^2 M_0 \in \mathbb{C}^{k \times k}, \quad H_1 = K_1 + i\omega D_1 - \omega^2 M_1 \in \mathbb{C}^{k \times k}.$$  

(4.2)

It follows from (1.2a) that

$$C_{22} = \begin{bmatrix} k & k & k & \cdots & k \\ k & H_0 & H_1^T & \cdots \\ k & H_1 & H_0 & H_1^T & \cdots \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ k & \cdots & \cdots & H_1^T & \cdots \\ k & H_1 & H_0 \end{bmatrix},$$

(4.3)

an $(m - 1) \times (m - 1)$ block tridiagonal matrix. To compute the three blocks in (4.1), we use the QR decomposition of $C_{22}$ with the Q-factor stored in its factor form, as in [5]. It goes as follows.

1. Initially (step $j = 1$), perform the QR decomposition

$$\begin{bmatrix} H_0 \\ H_1 \end{bmatrix} = \hat{U}_1 \times_k \begin{bmatrix} R_{11} \\ 0 \end{bmatrix},$$

where $\hat{U}_1 \in \mathbb{C}^{2k \times 2k}$ is unitary and $R_{11}$ is upper triangular, and set

$$\begin{bmatrix} R_{12} & R_{13} \\ R_{22} & R_{23} \end{bmatrix} = \hat{U}_1^H \begin{bmatrix} H_1^T \\ H_0 \end{bmatrix}.$$  

(4.4)

It can be verified that

$$U_1^H C_{22} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{22} & R_{23} \\ H_1 & \cdots & \cdots \\ \cdots & \cdots & H_1^T \\ H_1 & H_0 \end{bmatrix}$$

with $U_1 = \text{diag} \left( \hat{U}_1, I_{(m-3)k} \right)$.
2. At the beginning of step \( j(j \geq 2) \), we have computed

\[
U_{j-1}^H \cdots U_2^H U_1^H C_{22} = \begin{bmatrix}
R_{11} & R_{12} & R_{13} & R_{24} \\
R_{22} & R_{23} & \cdots & \cdot \\
\vdots & \vdots & \ddots & \cdot \\
\cdot & \cdot & \cdots & \cdot \\
R_{j,j+1} & \tilde{R}_{j,j+1} & \cdots & \cdot \\
\cdot & \cdot & \cdots & \cdot \\
H_1 & \cdot & \cdots & \cdot \\
\cdot & \cdot & \cdots & \cdot \\
H_1 & \cdot & \cdots & H_1^T \\
H_1 & H_0
\end{bmatrix},
\]

(4.5)

where for \( 1 \leq i \leq j - 1 \)

\[
U_i = \text{diag} \left( I_{(i-1)k}, \tilde{U}_i, I_{(m-i-2)k} \right)
\]

(4.6)

with \( \tilde{U}_j \in \mathbb{C}^{2k \times 2k} \) being unitary. Now perform the QR decomposition

\[
\begin{bmatrix}
\tilde{R}_{jj} \\
H_1
\end{bmatrix} = \hat{U}_j \times_k \begin{bmatrix}
R_{jj} \\
0
\end{bmatrix},
\]

where \( \hat{U}_j \in \mathbb{C}^{2k \times 2k} \) is unitary and \( R_{jj} \) is upper triangular, and set

\[
k \begin{bmatrix}
R_{j,j+1} & R_{j,j+2} \\
\tilde{R}_{j+1,j+1} & \tilde{R}_{j+1,j+2}
\end{bmatrix} = \hat{U}_j^H \begin{bmatrix}
\tilde{R}_{j,j+1} & 0 \\
H_0 & H_1^T
\end{bmatrix},
\]

\[
k \begin{bmatrix}
R_{j,j+1} \\
\tilde{R}_{j+1,j+1}
\end{bmatrix} = \hat{U}_j^H \begin{bmatrix}
\tilde{R}_{j,j+1} \\
H_0
\end{bmatrix},
\]

(4.7c)

for \( j \leq m - 3, j = m - 2 \).

It can be verified that (4.5) remains valid with \( j - 1 \) replaced by \( j \).

At the end of this process, we have computed

\[
C_{22} = UR,
\]

(4.7a)

where

\[
U = U_1 U_2 \cdots U_{m-2},
\]

(4.7b)

\[
R = \begin{bmatrix}
R_{11} & R_{12} & R_{13} & R_{24} \\
R_{22} & R_{23} & \cdots & \cdot \\
\vdots & \vdots & \ddots & \cdot \\
\cdot & \cdot & \cdots & \cdot \\
R_{m-3,m-3} & R_{m-3,m-2} & R_{m-3,m-1} & R_{m-2,m-2} \\
R_{m-3,m-2} & R_{m-3,m-1} & R_{m-2,m-1} & R_{m-1,m-1}
\end{bmatrix},
\]

(4.7c)

\( U_i \) is as in (4.6); \( R_{ii} \) are upper triangular, except that \( \tilde{R}_{m-1,m-1} \) is a \( k \times k \) dense matrix. We point out that in actual implementation, \( U \) should not be explicitly computed but stored in its factor form by simply storing all \( \hat{U}_j \) into a 2\( k \)-by-2\( k \) \((m - 2)\) array, for example.

With (4.7), the three blocks in (4.1) can now be readily computed. Let

\[
E_1 = \begin{bmatrix} I_k \\ 0_{(m-2)k \times k} \end{bmatrix} \in \mathbb{C}^{(m-1)q}, \quad E_{m-1} = \begin{bmatrix} 0_{(m-2)k \times k} \\ I_k \end{bmatrix} \in \mathbb{C}^{(m-1)k}.
\]

Note \( C_{22} = UR = R^T U^T \) because \( C_{22} \) is complex symmetric, and thus,

\[
C_{22}^{-1} = R^{-1} U^H = \hat{U} R^{-T}.
\]
where $\hat{U}$ is its complex conjugate. We have
\begin{equation}
(C_{22}^{-1})_{1,1} = E_1^T C_{22}^{-1} E_1 = E_1^T R^{-1} U^H E_1,
\end{equation}
(4.8)
\begin{equation}
(C_{22}^{-1})_{1,m-1} = E_1^T C_{22}^{-1} E_{m-1} = E_1^T \hat{U} R^{-T} E_{m-1} = E_1^T \hat{U} \left[ \begin{array}{c} 0_{k \times k} \\ \vdots \\ 0_{k \times k} \\ \hat{R}_{m-1,m-1}^{-T} \end{array} \right],
\end{equation}
(4.9)
\begin{equation}
(C_{22}^{-1})_{m-1,m-1} = E_{m-1}^T C_{22}^{-1} E_{m-1} = E_{m-1}^T \hat{U} R^{-T} E_{m-1} = E_{m-1}^T \hat{U} \left[ \begin{array}{c} 0_{k \times k} \\ \vdots \\ 0_{k \times k} \\ \hat{R}_{m-1,m-1}^{-T} \end{array} \right].
\end{equation}
(4.10)

So we need

1. $E_1^T U$, the first block row of $U$. It is the following piece of pseudo-code in MATLAB-like notation illustrating how it can be computed.

\begin{verbatim}
   compute Z = E_1^T U :
   1   Z = \begin{bmatrix} (\hat{G}_1^H)_{(1:k,:)} , 0_{k \times (m-2)k} \end{bmatrix};
   2   for j = 2 : m - 2
   3     Z(:,(j-1)k + 1:(j+1)k) = Z(:,(j-1)k + 1:jk) \times (\hat{G}_j^H)_{(1:k,:)} ;
   4   end for.
\end{verbatim}

2. $\hat{R}_{m-1,m-1}^{-T}$. It is also needed in computing $E_{m-1}^T R^{-1}$ next.

3. $E_1^T R^{-1}$, the first block row of $R^{-1}$. This can be implemented through solving a block triangular system $E_1 Z = U R$, where $Z$ is $k \times (m - 1) k$.

4. the last $k \times k$ block of $E_{m-1}^T U$. Because

\begin{equation}
E_{m-1}^T U = E_{m-1}^T U_1^H I_2^H \cdots U_{m-2}^H = E_{m-1}^T U_{m-2}^H.
\end{equation}

the last $k \times k$ block of $E_{m-1}^T U$ is just the bottom-right $k \times k$ submatrix of $\hat{U}_{m-2}^H$, that is, $\left(\hat{U}_{m-2}^H\right)_{(k+1:2k,k+1:2k)}$.

4.2. Solve (3.6) by the doubling algorithm

Having computed the three $k \times k$ blocks in (4.1), the coefficient matrices for (3.6) are readily available. We propose to solve (3.6) by applying Algorithm 1 in a straightforward way. For future notation reference, we restate Algorithm 1 for (3.6) into Algorithm 2.

---

**Algorithm 2** The doubling algorithm for solving (3.6)

Given $\tilde{A}, \tilde{Q} = \tilde{Q}_1^T \in \mathbb{C}^{k \times k}$, this algorithm computes a solution of (3.6).

1. $A_0 = \tilde{A}, X_0 = \tilde{Q}, Y_0 = 0$.
2. for $i = 1, 2, \ldots$, until convergence do
3. $\tilde{A}_{i+1} = \tilde{A}_i (\tilde{X}_i - \tilde{Y}_i)^{-1} \tilde{A}_i$;
4. $\tilde{X}_{i+1} = \tilde{X}_i - \tilde{A}_i^T (\tilde{X}_i - \tilde{Y}_i)^{-1} \tilde{A}_i$;
5. $\tilde{Y}_{i+1} = \tilde{Y}_i - \tilde{A}_i (\tilde{X}_i - \tilde{Y}_i)^{-1} \tilde{A}_i^T$;
6. end for
7. return $\tilde{X}_i$ as the computed solution at convergence.
In [5], it is showed that Algorithm 1 produces a convergent sequence \( X_i \) that converges to the stabilizing solution \( \Phi \) at the rate of \([\rho(\Phi^{-1}A)]^2\) as in (2.1). Naturally, we would expect that Algorithm 2 also produces a convergent sequence \( \tilde{X}_i \) that converges to the solution

\[
\Phi := \Phi_{33} - A_{13}^T (C_{22}^{-1})_{m-1,m-1} A_{13}
\]  

(4.11)

at likely the same rate, where \( \Phi_{33} \) is the bottom-right \( k \times k \) block of \( \Phi \), that is,

\[
\Phi_{33} = \Phi_{((m-1)k+1:mk,(m-1)k+1:mk)}.
\]

**Theorem 4.1**

\( \Phi \) defined by (4.11) is the stabilizing solution of (3.6), and moreover,

\[
\rho(\Phi^{-1}A) = \rho(\Phi^{-1}A) < 1.
\]  

(4.12)

**Proof**

Recall, by Theorem 3.1,

\[
\Phi = \frac{(m-1)k}{k} \begin{bmatrix} C_{22} & C_{23} \\ C_{23}^T & \Phi_{33} \end{bmatrix}.
\]  

(4.13)

We note that all blocks of \( A \) as an \( m \times m \) block matrix are zeros, except its \((1,m)\)th block \( A_{1:(1:k,(m-1)k+1:mk)} =: A_{13} \), where \( A_{13} \) is as defined in (3.1). Thus,

\[
\Phi^{-1}A = \frac{(m-1)k}{k} \begin{bmatrix} 0 & \cdots & 0 \\ 0 & (\Phi^{-1})_{((m-1)k+1:mk,1:k)}A_{13} \end{bmatrix},
\]

where \((\Phi^{-1})_{((m-1)k+1:mk,1:k)} \) is the last \( k \times k \) block in the first block column of \( \Phi^{-1} \). This implies

\[
\rho(\Phi^{-1}A) = \rho((\Phi^{-1})_{((m-1)k+1:mk,1:k)}A_{13}).
\]  

(4.14)

For the same reason as for (3.9), we have

\[
\Phi^{-1} = \begin{bmatrix} C_{22}^{-1} + C_{22}^{-1} C_{23} \tilde{C}_{22}^{-1} & C_{22}^{-1} C_{23}^T \tilde{C}_{22}^{-1} \\ -C_{22}^{-1} C_{23}^T \tilde{C}_{22}^{-1} \end{bmatrix}
\]

\[
\Phi_{33} = A_{13} (C_{22}^{-1})_{m-1,m-1} A_{13}^T
\]

by (3.10). Therefore,

\[
(\Phi^{-1})_{((m-1)k+1:mk,1:k)} = -\tilde{C}_{22}^{-1} (C_{23}^T \tilde{C}_{22}^{-1})_{1,1}
\]

\[
(\Phi^{-1})_{((m-1)k+1:mk,1:k)}A_{13} = -[\Phi_{33} - A_{13} (C_{22}^{-1})_{m-1,m-1} A_{13}^T]^{-1} A_{13} (C_{22}^{-1})_{m-1,1} A_{13}
\]

\[
(\Phi^{-1})_{((m-1)k+1:mk,1:k)}A_{13} = -[\Phi_{33} - A_{13} (C_{22}^{-1})_{m-1,m-1} A_{13}^T]^{-1} A_{13} (C_{22}^{-1})_{m-1,1} A_{13}
\]

\[
= -\tilde{\Phi}^{-1} \tilde{A}.
\]

Therefore, \( \rho(\tilde{\Phi}^{-1} \tilde{A}) = \rho((\Phi^{-1})_{((m-1)k+1:mk,1:k)}A_{13}) = \rho(\Phi^{-1}A) < 1 \) by (4.14); that is, \( \tilde{\Phi} \) is the stabilizing solution of (3.6).
We now show that Algorithm 2 will not break down, and \( \bar{X}_i \) converges quadratically to the stabilizing solution \( \bar{\Phi} \) of (3.6) under certain nonsingularity assumption. Let \( \mathcal{A}(X, Z, Y) \) be the \( i \times i \) block tridiagonal Toeplitz matrix

\[
\mathcal{A}(X, Z, Y) := \begin{bmatrix}
Z & X & & \\
Y & Z & X & \\
& \ddots & \ddots & \ddots \\
& & Y & Z & X \\
& & & Y & Z
\end{bmatrix} \in \mathbb{C}^{ik \times ik},
\]

defined for any given \( X, Y, Z \in \mathbb{C}^{k \times k} \). In particular, \( \mathcal{A}(X, Z, Y) = Z \).

The complementary equation of (3.6) is

\[
\bar{Y} + \bar{A} \bar{Y}^{-1} \bar{A}^T = \bar{Q}.
\]

(4.15)

**Theorem 4.2**

Let \( \bar{A} \) and \( \bar{Q} \) be given by (3.7b) and (3.7c). Let \( \bar{\Phi} \) be the stabilizing solution of (3.6) and \( \bar{\Psi} \) be the stabilizing solution of the complementary equation (4.15). Suppose \( \mathcal{A}_{2\ell-1}(-A^T, \bar{Q}, -\bar{A}) \) is invertible for all \( \ell \geq 1 \). Then,

(a) the sequences \( \{\bar{A}_i\}, \{\bar{X}_i\}, \{\bar{Y}_i\} \) in Algorithm 2 are well defined, and \( \bar{X}_i \) and \( \bar{Y}_i \) are complex symmetric;

(b) \( \bar{X}_i \) converges to \( \bar{\Phi} \) quadratically, \( \bar{A}_i \) converges to 0 quadratically, and \( \bar{Q} - \bar{Y}_i \) converges to \( \bar{\Psi} \) quadratically. Moreover,

\[
\lim_{i \to \infty} \sup_{1 \leq i} \frac{1}{\sqrt{2i}} \| \bar{X}_i - \bar{\Phi} \| \leq [\rho(\bar{\Phi}^{-1}A)]^2, \quad \lim_{i \to \infty} \sup_{1 \leq i} \frac{1}{\sqrt{2i}} \| \bar{A}_i \| \leq \rho(\bar{\Phi}^{-1}A),
\]

\[
\lim_{i \to \infty} \sup_{1 \leq i} \frac{1}{\sqrt{2i}} \| \bar{Q} - \bar{Y}_i - \bar{\Psi} \| \leq [\rho(\bar{\Phi}^{-1}A)]^2,
\]

where \( \| \cdot \| \) is any matrix norm.

**Proof**

Let \( \bar{Z}_i = \bar{X}_i - \bar{Y}_i \). Then, the sequence \( \{\bar{Z}_i\} \) satisfies

\[
\bar{Z}_{i+1} = \bar{Z}_i - \bar{A}_i^T \bar{Z}_i^{-1} \bar{A}_i - \bar{A}_i \bar{Z}_i^{-1} \bar{A}_i^T
\]

(4.16)

with \( \bar{Z}_0 = \bar{Q} \). The following proof about the nonsingularity of \( \bar{Z}_i \) for all \( i \geq 0 \) is similar to [18, Theorem 13 and also (9)] and is presented here for completeness. In fact, we will prove a stronger statement:

\[
\mathcal{A}_{2\ell-1}(-\bar{A}_i^T, \bar{Z}_i, -\bar{A}_i) \text{ for all } i \geq 0 \text{ and } \ell \geq 1 \text{ are nonsingular.}
\]

(4.17)

It is stronger than \( \bar{Z}_i \) being nonsingular for all \( i \) because \( \bar{Z}_i = \mathcal{A}_{2\ell-1}(-\bar{A}_i^T, \bar{Z}_i, -\bar{A}_i) \) for \( \ell = 1 \). We proceed by induction on \( i \). By the assumption of the theorem,

\[
\mathcal{A}_{2\ell-1}(-\bar{A}_0^T, \bar{Z}_0, -\bar{A}_0) = \mathcal{A}_{2\ell-1}(-\bar{A}^T, \bar{Q}, -\bar{A}) \text{ for all } \ell \geq 0
\]

are nonsingular. The statement (4.17) holds for \( i = 0 \). Suppose it holds for \( i \). We now prove that it must hold for \( i + 1 \); that is,

\[
\mathcal{A}_{2\ell-1}(-\bar{A}_i^T, \bar{Z}_{i+1}, -\bar{A}_{i+1}) \text{ for all } \ell \geq 1 \text{ are nonsingular.}
\]

(4.18)
To this end, for any given matrices $X$ and $Y$, we introduce the following notations: $\mathcal{D}_j(X) = \text{diag}(X, \ldots, X)$, $\mathcal{L}_j(X, Y)$ is the $(j + 1) \times j$ block lower bidiagonal matrix with $X$ on the main diagonal and $Y$ on the sub-diagonal, and $\mathcal{U}_j(X, Y)$ is the $j \times (j + 1)$ block upper bidiagonal matrix with $X$ on the main diagonal and $Y$ on the sup-diagonal.

Applying the even-odd block row-and-column permutation

$$
\begin{bmatrix}
1, 3, \ldots, 2^\ell + 1 - 1, 2, 4, \ldots, 2^\ell + 1 - 2
\end{bmatrix}
$$

to $\mathcal{F}_{2\ell+1}(-\widetilde{A}_i^T, Z_i, -\widetilde{A}_i)$ leads to

$$
\begin{bmatrix}
\mathcal{D}_{2\ell}(Z_i) & \mathcal{L}_{2\ell-1}(-\widetilde{A}_i^T, -\widetilde{A}_i) \\
\mathcal{U}_{2\ell-1}(-\widetilde{A}_i, -\widetilde{A}_i^T) & \mathcal{D}_{2\ell-1}(Z_i)
\end{bmatrix}.
$$

The Schur complement of $\mathcal{D}_{2\ell}(Z_i)$, obtained by one step of block Gaussian elimination on this matrix, is

$$
\mathcal{D}_{2\ell-1}(Z_i) - \mathcal{U}_{2\ell-1}(-\widetilde{A}_i, -\widetilde{A}_i^T) \mathcal{D}_{2\ell}(Z_i^{-1}) \mathcal{U}_{2\ell-1}(-\widetilde{A}_i^T, -\widetilde{A}_i)
$$

$$
= \begin{bmatrix}
Z_i \\
\vdots \\
Z_i
\end{bmatrix} - \begin{bmatrix}
-\widetilde{A}_i & -\widetilde{A}_i^T \\
\vdots & \ddots & \ddots \\
-\widetilde{A}_i & -\widetilde{A}_i^T
\end{bmatrix} \begin{bmatrix}
Z_i^{-1} \\
\vdots \\
Z_i^{-1}
\end{bmatrix}
$$

$$
= \begin{bmatrix}
Z_{i+1} & -\widetilde{A}_i^T Z_i^{-1} \widetilde{A}_i \\
-\widetilde{A}_i Z_i^{-1} \widetilde{A}_i & Z_{i+1} & -\widetilde{A}_i^T Z_i^{-1} \widetilde{A}_i \\
\vdots & \ddots & \ddots & \ddots \\
-\widetilde{A}_i Z_i^{-1} \widetilde{A}_i & Z_{i+1} & -\widetilde{A}_i^T Z_i^{-1} \widetilde{A}_i
\end{bmatrix},
$$

which is $\mathcal{F}_{2\ell-1}(-\widetilde{A}_i^T, Z_{i+1}, -\widetilde{A}_i)$ and must be nonsingular because, by the inductive assumption, both $\mathcal{F}_{2\ell+1}(-\widetilde{A}_i^T, Z_i, -\widetilde{A}_i)$ and $\mathcal{D}_{2\ell}(Z_i)$ are nonsingular. So (4.18) holds; that is, (4.17) holds for $i + 1$. This completes the inductive proof.

It is evident that $\{\tilde{X}_i\}$ and $\{\tilde{Y}_i\}$ are complex symmetric because $\widetilde{Q}$ is complex symmetric. This proves item (a).

For item (b), following the proof in [5, Theorem 4.1(b)], we can obtain (see also [19])

$$
\begin{align*}
\limsup_{i \to \infty} 2^i \sqrt{\|\tilde{X}_i - \tilde{\Phi}\|} & \leq \left[\rho \left(\tilde{\Phi}^{-1} \tilde{A}\right)\right]^2, \\
\limsup_{i \to \infty} 2^i \sqrt{\|\tilde{A}_i\|} & \leq \rho \left(\tilde{\Phi}^{-1} \tilde{A}\right), \\
\limsup_{i \to \infty} 2^i \sqrt{\|\tilde{Q} - \tilde{Y}_i - \tilde{\Psi}\|} & \leq \left[\rho \left(\tilde{\Phi}^{-1} \tilde{A}\right)\right]^2.
\end{align*}
$$

Now use (4.12) to conclude the proof.

Because of the quadratic convergence claim in this theorem, we can use (2.3) with $X_i$ and $X_{i+1}$ replaced by $\tilde{X}_i$ and $\tilde{X}_{i+1}$, respectively, as the stopping criteria for Algorithm 2 to use at its line 2. For our numerical test in the next section, the spectral norm $\| \cdot \|_2$ is used.
Remark 4.1
Theorem 4.2 for (3.6) solved by the doubling algorithm is similar to [5, Theorem 4.1] for (1.4) also solved by the doubling algorithm. In [5], Guo and Lin proved that \( \mathcal{F}_i (A^T, Q, \hat{A}) \) for all \( i \) are nonsingular for \( Q \) and \( A \) as given by (1.2) and (1.3). It is not clear whether \( \mathcal{F}_i (A^T, Q, \hat{A}) \) for all \( i \) are also nonsingular for \( \hat{Q} \) and \( \hat{A} \) as given by (3.7) in terms of such \( Q \) and \( A \). It is certainly reasonable to expect such a conclusion, but we do not have a proof or counterexample. Note that \( \mathcal{F}_i (A^T, \hat{Q}, \hat{A}) \) and \( \mathcal{F}_i (-\hat{A}^T, \hat{Q}, -\hat{A}) \) are singular or nonsingular at the same time because

\[
\mathcal{F}_i (-\hat{A}^T, \hat{Q}, -\hat{A}) = D \mathcal{F}_i (\hat{A}^T, \hat{Q}, \hat{A}) D,
\]

where \( D = \text{diag}(I, -I, I, -I, \cdots) \) of apt size.

5. SOLVING THE EIGENVALUE PROBLEM FOR \( P(\lambda) \)

After solving (3.6) by Algorithm 2, we recover the stabilizing solution \( \Phi \) for (1.4) by (3.7):

\[
\Phi_{33} = \Phi + A_{13}^T (C_{22}^{-1})_{m-1,m-1} A_{13}.
\]

The PQEP (1.1) arising from the vibration analysis of high speed trains is now transformed to the eigenvalue problems for \( \lambda A^T + \Phi \) and for \( \lambda \Phi + A \) by (1.5).

Recall (4.13) and \( C_{22} = U R \) in (4.7). We have

\[
\begin{bmatrix}
U \\
I_k
\end{bmatrix}^H \Phi = \begin{bmatrix}
R \\
0_{k \times (m-2)k}
\end{bmatrix} \begin{bmatrix}
H_1 & U^H C_{23} \\
\Phi_{33}
\end{bmatrix}
\]

and notice \( U^H C_{23} = U_{m-2}^H C_{23} \). Now similar to what we did in subsection 4.1, we find a unitary matrix \( U_{m-1} = \text{diag} \left( (I_{m-2})_k, \tilde{U}_{m-1} \right) \) so that \( U_{m-1}^H \text{diag}(U^H, I_k) \Phi \) is block upper triangular, where \( \tilde{U}_{m-1} \in \mathbb{C}^{2k \times 2k} \). On the other hand, only the last \( k \) columns of \( U_{m-1}^H \text{diag}(U^H, I_k) A \) are possibly nonzero. Thus, the eigenvalue problem for \( \lambda \Phi + A \) is reduced to the one for \( \lambda Z_{1} + Z_2 \), where \( Z_1 \) and \( Z_2 \) are the right-bottom \( k \times k \) blocks of \( U_{m-1}^H \text{diag}(U^H, I_k) \Phi \) and \( U_{m-1}^H \text{diag}(U^H, I_k) A \), respectively. The rest, that is, recovering the eigenvectors for (1.1) from those of the \( k \times k \) eigenvalue problem for \( \lambda Z_{1} + Z_2 \), is the same as in [5]. This gives all the stable eigenvalues (those within the unit circle) and associated eigenvectors.

The unstable eigenvalues can be obtained by simply taking the reciprocals of all the stable ones. However, additional work and care are needed to compute the eigenvectors associated with the unstable eigenvalues. The reader is referred to Guo and Lin [5] for crucial details.

6. NUMERICAL EXPERIMENTS

For comparison purpose, we will identify our implementation as SDA_LYL and the one in [5] as SDA_GL. Because SDA_GL has been shown in [5] to have much better accuracy than earlier existing methods, we will only compare ours with SDA_GL.

We begin by estimating the flop counts for SDA_LYL. The QR decomposition of \( C_{22} \) requires about \( \frac{86}{3} (m - 1) k^3 \) flops. Computing the three blocks in (4.1) requires \( 4k^3 \) and \( (9m - 5) k^3 \). Each iteration of the doubling algorithm for (3.6) requires about \( 8k^3 \) flops. Recovering \( \Phi \) and finding all eigenvectors require about \( 14mk^3 + \frac{86}{3} k^3 \) flops. Table I summarizes the flop counts for SDA_LYL as well as those for SDA_GL taken from [5]. It is evident that the save in our new implementation comes from solving (3.6), which is of \( k \times k \) instead of (1.4), which is of \( mk \times mk \). In fact, as far as solving (3.6) and (1.4) by the doubling algorithm is concerned, ours on (3.6) is \( 154/32 = 4.8 \) times faster than theirs on (1.4). But overall, while SDA_LYL is always faster, its speed potential gradually drops as \( m \) increases. In fact, according to Table I, the flops ratio is

\[
\frac{\text{SDA_GL}}{\text{SDA_LYL}} = \frac{165m + 154\ell}{165m + 32\ell - 27}.
\]
Table I. Comparison of cost \((m \geq 2)\).

<table>
<thead>
<tr>
<th></th>
<th>Before DA iteration</th>
<th>Each DA iteration</th>
<th>Finding eigenpairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDA_GL [5]</td>
<td>(\frac{113}{3}mk^3)</td>
<td>(\frac{154}{3}k^3)</td>
<td>(14mk^3)</td>
</tr>
<tr>
<td>SDA_LYL</td>
<td>(\frac{113}{3}(m - 1)k^3)</td>
<td>(\frac{32}{3}k^3)</td>
<td>(14mk^3 + \frac{86}{3}k^3)</td>
</tr>
</tbody>
</table>

DA, doubling algorithm.

Table II. \(\rho = \rho(\Phi^{-1}A) = \rho(\tilde{\Phi}^{-1}\tilde{A})\).

<table>
<thead>
<tr>
<th>((k, m)) (\omega)</th>
<th>100</th>
<th>1000</th>
<th>3000</th>
<th>5000</th>
</tr>
</thead>
<tbody>
<tr>
<td>(159,11)</td>
<td>0.9593</td>
<td>0.8745</td>
<td>0.7925</td>
<td>0.7406</td>
</tr>
<tr>
<td>(303,19)</td>
<td>0.9307</td>
<td>0.7933</td>
<td>0.6692</td>
<td>0.5953</td>
</tr>
<tr>
<td>(705,51)</td>
<td>0.9622</td>
<td>0.8831</td>
<td>0.8060</td>
<td>0.7569</td>
</tr>
</tbody>
</table>

assuming that the doubling algorithm takes \(\ell\) iterative steps to finish. It decreases as \(m\) increases. For example, with \(\ell = 10\), the ratio (6.1) is 3.00 at \(m = 2\) and decreases to 1.14 at \(m = 50\).

Numerically, we tested SDA\_GL and SDA\_LYL on three sets of test data, \(^1\) generated by a finite element method, with

\[(k, m) = (159, 11), \quad (303, 19), \quad (705, 51), \tag{6.2}\]

respectively. The matrices \(Q = K_t + \omega D_t - \omega^2 M_t\) and \(A = K_c + \omega D_c - \omega^2 M_c\), where \(M_t, M_c, K_t,\) and \(K_c\) are as in (1.3a)–(1.3c), and \(\omega > 0\). All numerical experiments are carried out within MATLAB 7.0 with machine unit roundoff \(\epsilon_u = 2^{-52} \approx 2.22 \times 10^{-16}\).

In all cases, the doubling algorithm on (3.6) shows rapid convergence, as expected, because in theory, the doubling algorithm on (1.4) and on the transformed (3.6) has the same rate of convergence. Table II displays the spectral radii

\(\rho = \rho(\Phi^{-1}A) = \rho(\tilde{\Phi}^{-1}\tilde{A})\)

for the three pairs of \((k, m)\) in (6.2) and for

\(\omega = 100, \quad 1000, \quad 3000, \quad 5000, \)

respectively. For illustrating the convergence history of the doubling algorithm, Figure 1 plots the ratio

\[\frac{\|\tilde{X}_{i+1} - \tilde{X}_i\|_2}{\|\tilde{X}_i\|_2}, \tag{6.3}\]

where \(\tilde{X}_i\) is computed by Algorithm 2 for the first and last pair of \((k, m)\) in (6.2). What we can see from the figure is that in no more than 10 doubling iterations, this ratio reaches \(O(\epsilon_u) = O(10^{-16})\) or much less. There is no significant difference in the numbers of doubling iterations for different values of \((k, m)\), but for the larger \(\omega\), \(\rho\) becomes much smaller, and consequently, fewer doubling iterations are recorded.

Usually, the accuracy of an approximate eigenpair \((\lambda, z)\) of \(P(\lambda)\) is measured by the relative residual

\[\text{RRes} := \frac{\|\lambda^2A^Tz + \lambda Qz + Az\|_2}{(\|\lambda\|_2\|A\|_F + \|\lambda\|_2\|Q\|_F + \|A\|_F\|z\|_2)}, \tag{6.4}\]

\(^1\)We thank Prof. Wen-Wei Lin of National Chiao Tung University, Taiwan, and Prof. Tiexiang Li of Southeast University, China, for generously providing us with the data and their code.
Figure 1. \( \| \bar{X}_{i+1} - \bar{X}_i \|_2 / \| \bar{X}_i \|_2 \) for \((k, m) = (159, 11)\) (left plot) and \((705, 51)\) (right plot).

Figure 2. RRes and RRes_{new} for approximate eigenpairs when \((k, m) = (159, 11)\) (left plot) and \((705, 51)\) (right plot).

where \( \| \cdot \|_F \) is taken to be the Frobenius norm (for easy computation) but could be any other matrix norm. This is a common practice, and usually, the best one could hope for is that the relative residual is reduced to about \( O(\varepsilon_w) = O(10^{-16}) \). But as shown in [5], this RRes for the current problem can reach up to \( O(10^{-35}) \) or smaller (see also Figure 2). What is happening here? We shall now explain.

The RRes defined by (6.4) is a generic one; that is, it works for \( P(\lambda) \) without any special 0-block structure that could result in special roundoff error accumulations in evaluating residual \( \lambda^2 A^\top z + \lambda Q z + A z \). But this is not the case here, chiefly because \( A \) has only one nonzero block \( A_{13} \). Partition \( z \) into an \( m \)-block vector: \( z = [z_1^\top, z_2^\top, \ldots, z_m^\top]^\top \), where \( z_i \in \mathbb{C}^K \). It is not difficult to see that

\[
\text{fl}(\lambda^2 A^\top z + \lambda Q z + A z) = \lambda^2 A^\top z + \lambda Q z + A z + O \left( \varepsilon_w \left[ |\lambda|^2 \| A_{13} \|_F \| z_1 \|_2 + |\lambda| \| Q \|_F \| z \|_2 + \| A_{13} \|_F \| z_m \|_2 \right] \right),
\]

where \( \text{fl}(\cdot) \) denotes the computed result of an expression. In the ideal situation that \((\lambda, z)\) is an exact eigenpair, that is, \( \lambda^2 A^\top z + \lambda Q z + A z = 0 \) exactly, the best we can hope for the computed residual is

\[
\text{fl}(\lambda^2 A^\top z + \lambda Q z + A z) = O \left( \varepsilon_w \left[ |\lambda|^2 \| A_{13} \|_F \| z_1 \|_2 + |\lambda| \| Q \|_F \| z \|_2 + \| A_{13} \|_F \| z_m \|_2 \right] \right).
\]
Therefore, a correct relative residual for the current quadratic eigenvalue problem should be

\[
RRes_{\text{new}} := \frac{\| \lambda^2 A^T z + \lambda Qz + Az \|_2}{|\lambda|^2 \| A_{13} \|_F \| z_1 \|_2 + \| Q \|_F \| z_2 \|_2 + \| A_{13} \|_F \| z_m \|_2}.
\] (6.5)

This new \(RRes_{\text{new}}\) differs from the one in (6.4) in their denominators.

As in Figure 1, we use the first and last pairs of \((k, m)\) in (6.2) as examples and also for \(\omega = 1000\) only. We plot \(RRes\) and \(RRes_{\text{new}}\) for all approximate eigenpairs of \(P(\lambda)\) in Figure 2. As it shows, \(RRes_{\text{new}}\) is always about \(O(\epsilon_\lambda)\) for the computed eigenpairs, regardless of the magnitude of \(|\lambda|\). The reason behind it is that for tiny \(|\lambda|\), \(\|z_m\|_2/\|z\|_2\) is also tiny, while for huge \(|\lambda|\), \(\|z_1\|_2/\|z\|_2\) is tiny. For SDA_LYL and SDA_GL, corresponding \(RRes\) and \(RRes_{\text{new}}\) are indistinguishable, but \(RRes_{\text{new}}\) are always about \(O(\epsilon_\lambda)\) while \(RRes\) for \(\lambda\) with tiny or huge magnitude are skewed up to \(O(10^{-35})\).

7. CONCLUDING REMARKS

We have presented a numerical method to solve the PQEP (1.1) arising from the vibration analysis of high speed trains [4–6]. The method is a modification of the Guo–Lin method [5] in the part of the application of the doubling algorithm: we solve a \(k \times k\) nonlinear matrix equation (3.6) while Guo and Lin solved an \(mk \times mk\) nonlinear matrix equation (1.4). Despite their clever effort in exploiting the structures in \(A\) and \(Q\), each doubling iteration in [5] on the \(mk \times mk\) nonlinear matrix equation (1.4) is about 4.8 times as expensive as the one here on the \(k \times k\) nonlinear matrix equation (3.6) for large \(k\). But because of the cost in the other parts of the methods, our saving in the application of the doubling algorithm translates into a saving not so dramatic as just for solving the respective nonlinear equations and the saving levels off as \(m\) increases.

Numerical tests suggest that the quality of computed eigenpairs of \(P(\lambda)\) by either the original Guo–Lin method and the new version here is about the same. We also propose a new relative residual measure (6.5), which takes into consideration the uncharacteristic roundoff pattern as a result of the special 0-block structure in the coefficient matrices. The new relative residual measure always stays at about the order of machine unit roundoff, as it should, while the usual one (6.4) can be highly skewed to a level uncharacteristically tiny.

So far, we have focused on PQEP (1.1) from the vibration analysis of high speed trains. The idea here, as well as the one in [5], is easily carried over to the case that \(Q\) is block tridiagonal but not necessarily block Toeplitz, provided that \(P(\lambda)\) has no eigenvalues on the unit circle and Algorithm 2 does not breakdown; that is, all inverses exist. Other extensions are conceivably possible. We omit the detail.

ACKNOWLEDGEMENTS

Lu is supported in part by the National Natural Science Foundation of China grants 10261012 and 11428104. Li is supported in part by NSF grants DMS-1115834 and DMS-1317330, a Research Gift Grant from Intel Corporation, and NSFC grant 11428104.

REFERENCES