J. Math. Study
doi: 10.4208/jms.v55n2.22.05

Vol. **55**, No. 2, pp. 180-194
June 2022

Highly Accurate Latouche-Ramaswami Logarithmic Reduction Algorithm for Quasi-Birth-and-Death Process

Guiding Gu[1], Wang Li[2] and Ren-Cang Li[3,*]

[1] *School of Mathematics, Shanghai University of Finance and Economics, 777 Guoding Lu, Shanghai 200433, China.*
[2] *Department of Mathematics, University of Texas at Arlington, P.O. Box 19408, Arlington, TX 76019, USA.*
[3] *Department of Mathematics, University of Texas at Arlington, P.O. Box 19408, Arlington, TX 76019, USA.*

Received October 20, 2020; Accepted August 7, 2021;
Published online February 17, 2022.

Abstract. This paper is concerned with the quadratic matrix equation $A_0+A_1X+A_2X^2=X$ where $I-A_0-A_1-A_2$ is a regular M-matrix, i.e., there exists an entrywise positive vector $\boldsymbol{u}$ such that $(I-A_0-A_1-A_2)\boldsymbol{u}\geq 0$ entrywise. It broadly includes those originally arising from the quasi-birth-and-death (QBD) process as a special case where $I-A_0-A_1-A_2$ is an irreducible singular M-matrix and $(A_0+A_1+A_2)\mathbf{1}=\mathbf{1}$ with $\mathbf{1}$ being the vector of all ones. A highly accurate implementation of Latouche-Ramaswami logarithmic reduction algorithm [*Journal of Applied Probability*, 30(3):650–674, 1993] is proposed to compute the unique minimal nonnegative solution of the matrix equation with high entrywise relative accuracy as it deserves. Numerical examples are presented to demonstrate the effectiveness of the proposed implementation.

AMS subject classifications: 15A24, 65F30, 65H10

Key words: Quadratic matrix equation, M-matrix, quasi-birth-and-death process, minimal nonnegative solution, entrywise relative accuracy.

1 Introduction

In the quasi-birth-and-death (QBD) process, the following quadratic matrix equation [17, 25]

$$A_0+A_1X+A_2X^2=X, \tag{1.1}$$

*Corresponding author. *Email addresses:* guiding@mail.shufe.edu.cn (Gu G), li.wang@uta.edu (Wang L), rcli@uta.edu (Li R C)

plays a vital role in analyzing the process, where A_i for $i=0,1,2$ are $n\times n$ nonnegative matrices, sitting as blocks in an infinite block-tridiagonal transition matrix. In the QBD process, $I-A_0-A_1-A_2$ is irreducible and singular, and, furthermore,

$$(A_0+A_1+A_2)\mathbf{1}_n=\mathbf{1}_n, \tag{1.2}$$

where $\mathbf{1}_n$ (often simply $\mathbf{1}$ when its dimension is clear from the context) is the column vector of all ones of dimension n. Under these conditions, Eq. (1.1) admits a unique minimal nonnegative solution [23] (see also [6]), denoted by Φ hereafter, in the sense that

$$\Phi\leq X \quad \text{for any other nonnegative solution } X \text{ of equation (1.1).}$$

Existing numerical methods for computing the solution Φ include fixed point iterations (e.g., [8, 10] and references therein), Newton's method, the Latouche-Ramaswami logarithmic reduction algorithm (called the LR algorithm hereafter), the method of cyclic reduction, and doubling algorithms (see, e.g., [5, 6, 11, 13, 15, 16, 20, 30] and references therein). The fixed point iterations are usually linearly convergent and can suffer very slow convergence when $\rho(\Phi)$ is almost 1, or no convergence when it is 1. Newton's method needs to solve a Sylvester equation in each of its iterative steps, which, unfortunately, can be as expensive as solving equation (1.1) itself by other methods and thus is not competitive. It has been observed that the method of cyclic reduction is equivalent to the LR algorithm [11,16] which turns out to be a very efficient method nowadays. The application of doubling algorithms [14] to solve equation (1.1) from the QBD process and beyond is more recent [6] and their efficiency is about the same as the LR algorithm.

Because of (1.2) and that $A_0+A_1+A_2$ is nonnegative and irreducible, there exists a positive vector $\boldsymbol{z}$ [22, p.673] such that $\boldsymbol{z}^{\mathrm{T}}(A_0+A_1+A_2)=\boldsymbol{z}^{\mathrm{T}}$. The associated QBD process is further classified into three categories [17]: positive recurrent if $\boldsymbol{z}^{\mathrm{T}}(A_0-A_2)\mathbf{1}>0$, null recurrent if $\boldsymbol{z}^{\mathrm{T}}(A_0-A_2)\mathbf{1}=0$, and transient if $\boldsymbol{z}^{\mathrm{T}}(A_0-A_2)\mathbf{1}<0$. Except for the fixed point iterations, all other methods mentioned in the previous paragraph are quadratically convergent unless the QBD process is null recurrent [11, 30].

Before the work of Ye [30], it was noted that computed Φ by the LR algorithm can suffer heavy accuracy loss, especially when the QBD process is nearly null recurrent. It turns out that inaccurate numerical matrix inversions during the LR iterative process are to blame because the involved matrices are increasingly singular and thus increasingly ill-conditioned for inversions as the iteration progresses. It turns out those matrices are all nonsingular M-matrices, and Ye [30] came up with a new implementation by using the GTH-like algorithm for inverting all nonsingular M-matrices instead of the plain Gaussian elimination. The GTH-like algorithm, due to Alfa, Xue, and Ye [2] (see also [14, p.87]), is a variant of Gaussian elimination, and can guarantee to invert a nonsingular M-matrix, albeit how nearly singular it may be, with high entrywise relative accuracy. Ye's implementation, as a result, was a resounding success – able to compute Φ with high entrywise relative accuracy. The same can be said about the doubling algorithms [6]. The GTH-like algorithm has since been successfully employed in highly accurate solutions of M-matrix Riccati equations [14, 18, 19, 24, 26–29], among others.

Ye's implementation is for a stochastic QBD process, i.e., (1.2) holds. The LR algorithm, however, actually works broadly for equation (1.1) with $(A_0+A_1+A_2)\mathbf{1}_n \le \mathbf{1}_n$ [1]. Naturally, we may ask if there is an implementation, similar to Ye's, that works in this more general setting. The purpose of this paper is to present such an implementation under a more general condition that $I-A_0-A_1-A_2$ is a regular M-matrix.

We call (1.1) the QBD equation. The rest of this paper is organized as follows. In Section 2, we will set the stage by introducing the notation of regular M-matrix due to [12]. The Latouche-Ramaswami logarithmic reduction algorithm is stated in Section 3 and its highly accurate implementation is explained in Section 4. A couple of numerical examples are presented to demonstrate the superiority of our implementation in Section 5. Finally, we draw our conclusions in Section 6.

Notation. $\mathbb{R}^{m\times n}$ is the set of all $m\times n$ real matrices, $\mathbb{R}^n=\mathbb{R}^{n\times 1}$, and $\mathbb{R}=\mathbb{R}^1$. I_n (or simply I if its dimension is clear from the context) is the $n\times n$ identity matrix. The superscript "$\cdot^{\mathrm{T}}$" takes transpose. For $X\in\mathbb{R}^{m\times n}$, $X_{(i,j)}$ refers to its (i,j)th entry. Inequality $X\le Y$ means $X_{(i,j)}\le Y_{(i,j)}$ for all (i,j), and similarly for $X<Y$, $X\ge Y$, and $X>Y$. In particular, $X\ge 0$ means that X is entrywise nonnegative. For a square matrix X, denote by $\rho(X)$ its spectral radius.

2 The setting

A matrix $A\in\mathbb{R}^{n\times n}$ is called a Z-matrix if $A_{(i,j)}\le 0$ for all $i\ne j$. Any Z-matrix A can be written as $sI-B$ with $B\ge 0$, and it is called an M-matrix if $s\ge\rho(B)$. Specifically, it is a singular M-matrix if $s=\rho(B)$, and a nonsingular M-matrix if $s>\rho(B)$. An important characteristic of a nonsingular M-matrix A is that $A^{-1}\ge 0$.

It is well-known that a Z-matrix A is a nonsingular M-matrix if and only if $A\boldsymbol{u}>0$ for some positive vector $\boldsymbol{u}>0$ [4, chapter 6]. On the other hand, for an irreducible singular M-matrix A, there exists a positive vector $\boldsymbol{u}>0$ such that[†] $A\boldsymbol{u}=0$ [4]. Both are special cases of a more broad class of M-matrices, the so-called regular M-matrices introduced by Guo [12].

Definition 2.1 ([12])**.** An M-matrix A is said *regular* if there is positive vector $\boldsymbol{u}$ such that $A\boldsymbol{u}\ge 0$. Such a matrix A is called a *regular* M-matrix.

Throughout the rest of this paper, we will consider more broadly Eq. (1.1) under the following assumptions: $A_i\ge 0$ for $0\le i\le 2$ and either

$$I-A_0-A_1-A_2 \text{ is a nonsingular } M\text{-matrix}, \tag{2.1a}$$

or

$$\boxed{\begin{array}{l} I-A_0-A_1-A_2 \text{ is a singular regular } M\text{-matrix and} \\ \text{one of } A_0 \text{ and } A_2 \text{ has no zero rows.}\end{array}} \tag{2.1b}$$

[†]It is a well-known fact and can be easily proved as follows. Let $A=sI-B$ with irreducible $B\ge 0$ and $s=\rho(B)$. Applying the classical Perron-Frobenius theorem [4, Theorem 1.4 on p.27] to B, we conclude that there exists $\boldsymbol{u}>0$ such that $B\boldsymbol{u}=\rho(B)\boldsymbol{u}$. Hence $A\boldsymbol{u}=[s-\rho(B)]\boldsymbol{u}=0$.

As a consequence of (2.1), there exists a positive vector $\boldsymbol{u}>0$ in $\mathbb{R}^n$ such that

$$\boldsymbol{v}=(I-A_0-A_1-A_2)\boldsymbol{u}\begin{cases}>0, & \text{in the case of (2.1a)},\\ \geq 0, & \text{in the case of (2.1b)}.\end{cases} \tag{2.2}$$

In particular, if $I-A_0-A_1-A_2$ is irreducible, then we can take $\boldsymbol{v}=0$ in the case of (2.1b). In any case, we have

$$(I-A_1)\boldsymbol{u}=\boldsymbol{v}+(A_0+A_2)\boldsymbol{u}>0$$

because either $\boldsymbol{v}>0$ for (2.1a) or $(A_0+A_2)\boldsymbol{u}>0$ for (2.1b). Hence $I-A_1$ is a nonsingular M-matrix.

Theorem 2.1. *Assume that* (2.1) *holds. Then the* QBD *equation* (1.1) *has a unique minimal nonnegative solution* Φ.

Only under the case of (2.1b) this theorem seems to be new. Even for that, the same proof of [6, Theorem 3.2] works. In fact, Theorem 2.1 for the case when $I-A_0-A_1-A_2$ is an irreducible singular M-matrix satisfying (1.2) is well-known in the QBD application. Part of the theorem is also implied by [21, Theorem 2.3].

3 Latouche-Ramaswami algorithm

The Latouche-Ramaswami (LR) algorithm [16,30] is stated here as Algorithm 3.1. Matrix inversions appear at Lines 1 and 5. Whether we can compute the inverses accurately or not determines the accuracy of eventually computed approximation to Φ. We observe the strong similarity between Lines 1 and 5: they perform the same operations on (A_0,A_1,A_2) and $(A_{k;0},A_{k;1},A_{k;2})$, respectively. This was made apparent in the original statement of the algorithm [16, Figure 2], where notationally, (A_0^*,A_1^*,A_2^*) was used instead of $(A_{k;0},A_{k;1},A_{k;2})$ here. We add in the subscript k for the convenience of our analysis later. Although the LR algorithm was first proposed for the case $(A_0+A_1+A_2)\mathbf{1}_n=\mathbf{1}_n$, it was found later to actually work more broadly for the case $(A_0+A_1+A_2)\mathbf{1}_n\leq\mathbf{1}_n$ as well [1].

Algorithm 3.1. Latouche-Ramaswami (LR) Algorithm (LRQBD) [16]

Require: nonnegative $A_0,A_1,A_2\in\mathbb{R}^{n\times n}$ satisfying certain conditions;
Ensure: an approximation of Φ, minimal nonnegative solution to (1.1).
1: $L_0=(I-A_1)^{-1}A_0$, $H_0=(I-A_1)^{-1}A_2$;
2: $X_0=L_0$, $T_0=H_0$;
3: **for** $k=0,1,\ldots$, until convergence **do**
4: $\quad A_{k;0}=L_k^2$, $A_{k;1}=L_kH_k+H_kL_k$, $A_{k;2}=H_k^2$;
5: $\quad L_{k+1}=(I-A_{k;1})^{-1}A_{k;0}$, $H_{k+1}=(I-A_{k;1})^{-1}A_{k;2}$;
6: $\quad X_{k+1}=X_k+T_kL_{k+1}$; $T_{k+1}=T_kH_{k+1}$;
7: **end for**
8: **return** last X_k as the computed minimal nonnegative solution.

Ye [30] observed that the matrices $I-A_1$ at Line 1 and $I-A_{k;1}$ at Line 5 are all nonsingular M-matrices. The key ingredient in his implementation is the GTH-like algorithm
[2] for performing these inversions. It is made possible by alternatively representing a nonsingular M-matrix in the form of the so-called triplet representation. A triplet representation $(\text{offdiag}(A),\boldsymbol{u},\boldsymbol{v})$ of M-matrix $A\in\mathbb{R}^{n\times n}$ consists of

$$\text{offdiag}(A)=\text{diag}(A)-A\geq 0,\ 0<\boldsymbol{u}\in\mathbb{R}^n,\ \text{and}\ \boldsymbol{v}=A\boldsymbol{u}\geq 0,$$

where $\text{diag}(A)$ is the diagonal matrix obtained from extracting the diagonal part of A. In other words, $\text{offdiag}(A)$ is obtained from A by zeroing out its diagonal entries and reversing the sign of its off-diagonal entries. In what follows, we will not distinguish A from its triplet representation and write $A=(\text{offdiag}(A),\boldsymbol{u},\boldsymbol{v})$. A nonsingular M-matrix A can have infinitely many triplet representations, but for the purpose of inversion, any one is just as good as any other.

The main theoretical contribution in [3] is that if all entries of $\text{offdiag}(A)$, $\boldsymbol{u}$ and $\boldsymbol{v}$ are known to high entrywise relative accuracy, then all entries of A^{-1} are determined to a comparable high entrywise relative accuracy, or equivalently the solution $\boldsymbol{x}$ to $A\boldsymbol{x}=\boldsymbol{b}$ is determined to a comparable high entrywise relative accuracy for any given $\boldsymbol{b}\geq 0$. Numerically, using the trick of [9], Alfa, Xue, and Ye [2] (see also [14, p.87]) presented the GTH-like algorithm to compute the LU decomposition of $A=(\text{offdiag}(A),\boldsymbol{u},\boldsymbol{v})$, via the Gaussian elimination without pivoting, without any cancellation and consequently compute $\boldsymbol{x}$ to the claimed accuracy.

4 Highly accurate implementation

Throughout the rest of this paper, we assume that all entries of A_0, A_1, A_2, $\boldsymbol{u}$ and $\boldsymbol{v}$ of (2.2) are known with high entrywise relative accuracy, except for the diagonal entries of A_1 which will not be accessed at all during entire computations. Indeed for the stochastic QBD setting, $\boldsymbol{u}=\mathbf{1}_n$ and $\boldsymbol{v}=0$ are known exactly.

In what follows, we will simply explain how to construct entrywise accurate triplet representations of $I-A_1$ and $I-A_{k;1}$ for all k, but refer the reader to, e.g., [14, p.87], for the detail of the GTH-like algorithm for accurately solving associated linear systems.

In Section 2, we have already argued that $I-A_1$ is a nonsingular M-matrix under (2.1). With (2.2), a triplet representation for $I-A_1$ is readily available:

$$I-A_1=(\text{offdiag}(A_1),\boldsymbol{u},\hat{\boldsymbol{v}})\quad \text{with}\,\hat{\boldsymbol{v}}=\boldsymbol{v}+(A_0+A_2)\boldsymbol{u}>0, \tag{4.1}$$

where $\text{offdiag}(A_1)$ and $\boldsymbol{u}$ are assumed to have entrywise accuracy to begin with, and $\hat{\boldsymbol{v}}$ can be computed without any cancellation and thus has entrywise relative accuracy comparable to $A_0, A_2, \boldsymbol{u}$ and $\boldsymbol{v}$, too. Note that the diagonal entries of A_1 are not needed at all in constructing the triplet representation (4.1).

In the stochastic QBD setting where (1.2) holds, (4.1) becomes

$$I-A_1=(\text{offdiag}(A_1),\mathbf{1},\hat{\boldsymbol{v}}) \quad \text{with} \qquad \hat{\boldsymbol{v}}=(A_0+A_2)\mathbf{1}.$$

For Algorithm 3.1, it is known $(H_k+L_k)\mathbf{1}=\mathbf{1}$ [11, Lemma 3.1], [16]. Thus $(H_k+L_k)^2\mathbf{1}=\mathbf{1}$ for all k and hence

$$(A_{k;0}+A_{k;1}+A_{k;2})\mathbf{1}=\mathbf{1}.$$

Consequently,

$$I-A_{k;1}=(\text{offdiag}(A_{k;1}),\mathbf{1},\hat{\boldsymbol{v}}_k) \quad \text{with} \quad \hat{\boldsymbol{v}}=(A_{k;0}+A_{k;2})\mathbf{1}$$

gives a triplet representation needed at Line 5 for an accurate inversion by the GTH-like algorithm.

In Section 2, we argued that $I-A_1$ is a nonsingular M-matrix. But it is not clear if $I-A_{k;1}$ is a nonsingular M-matrix. Evidently, this issue of whether $I-A_{k;1}$ can be rigorously justified nonsingular for $k\geq 0$ didn't appear to attract any attention. Later, we will show that their nonsingularity is guaranteed under (2.1).

For convenience, we let

$$A_{-1;j}=A_j \quad \text{for } 0\leq j\leq 2, \tag{4.2}$$

and define $\boldsymbol{v}_k$ for $k\geq -1$

$$\boldsymbol{v}_k=(I-A_{k;0}-A_{k;1}-A_{k;2})\boldsymbol{u}. \tag{4.3}$$

Evidently, $\boldsymbol{v}_k$ for $k\geq 0$ is well-defined only if $A_{k;j}$ for $0\leq j\leq 2$ in Algorithm 3.1 exist up to iteration $k-1$. At this point, we only know $\boldsymbol{v}_k$ for $k=-1,0$ are well-defined under (2.1), thanks to our discussion so far, and $\boldsymbol{v}_{-1}=\boldsymbol{v}\geq 0$ due to (2.2). In what follows, we will show that all $\boldsymbol{v}_k$ are well-defined under (2.1), and, moreover, $\boldsymbol{v}_k>0$ for all k in the case of (2.1a) or $\boldsymbol{v}_k\geq 0$ for all k in the case of (2.1b).

Lemma 4.1. *Assume* (2.2) *holds. In* Algorithm 3.1, *if the loop is successfully executed up to iteration ℓ, then we have for $\ell\geq k\geq -1$*

$$\boldsymbol{v}_{k+1}=(I-A_{k;1})^{-1}\boldsymbol{v}_k+(H_{k+1}+L_{k+1})(I-A_{k;1})^{-1}\boldsymbol{v}_k. \tag{4.4}$$

In particular, if $\boldsymbol{v}=0$, then $\boldsymbol{v}_k=0$ for all $\ell+1\geq k\geq -1$.

Proof. Pre-multiply (4.3) by $(I-A_{k;1})^{-1}$ to get

$$\begin{aligned}
&(I-A_{k;1})^{-1}\boldsymbol{v}_k=-(H_{k+1}+L_{k+1})\boldsymbol{u}+\boldsymbol{u},\\
&(H_{k+1}+L_{k+1})\boldsymbol{u}=\boldsymbol{u}-(I-A_{k;1})^{-1}\boldsymbol{v}_k,\\
&(H_{k+1}+L_{k+1})^2\boldsymbol{u}=(H_{k+1}+L_{k+1})\boldsymbol{u}-(H_{k+1}+L_{k+1})(I-A_{k;1})^{-1}\boldsymbol{v}_k\\
&\qquad\qquad=\boldsymbol{u}-(I-A_{k;1})^{-1}\boldsymbol{v}_k-(H_{k+1}+L_{k+1})(I-A_{k;1})^{-1}\boldsymbol{v}_k.
\end{aligned} \tag{4.5}$$

Noting $(H_{k+1}+L_{k+1})^2=A_{k+1;0}+A_{k+1;1}+A_{k+1;2}$, by definition (4.3) we infer from (4.5) that

$$\begin{aligned}\boldsymbol{v}_{k+1}&=\boldsymbol{u}-(A_{k+1;0}+A_{k+1;1}+A_{k+1;2})\boldsymbol{u}\\&=(I-A_{k;1})^{-1}\boldsymbol{v}_k+(H_{k+1}+L_{k+1})(I-A_{k;1})^{-1}\boldsymbol{v}_k,\end{aligned}$$

yielding (4.4). If $\boldsymbol{v}_{-1}=\boldsymbol{v}=0$, then $\boldsymbol{v}_k=0$ for all $\ell+1\geq k\geq -1$ by the recursive formula (4.4). □

Theorem 4.1. *Assume* (2.1a), *i.e., $I-A_0-A_1-A_2$ is a nonsingular M-matrix and thus $\boldsymbol{v}>0$ in* (2.2). *Then* Algorithm 3.1 *runs without any breakdown, i.e., all inverses exist, and for $k\geq -1$,*

$$\boldsymbol{v}_k>0,\quad A_{k;0}\geq 0,\quad A_{k;1}\geq 0,\quad A_{k;2}\geq 0,$$

and $I-A_{k;1}$ is a nonsingular M-matrix.

Proof. A straightforward induction based on the recursive formula (4.4) will do, starting with $k=-1$ for which the conclusions hold because of the assumption (2.1a). □

In the case of (2.1b), during executing Line 4 of Algorithm 3.1 for $k=0$, it is not clear if $I-A_{0;1}$ is nonsingular, even though we have $A_{0;j}\geq 0$ for $j\in\{0,1,2\}$ and $(I-A_{0;0}-A_{0;1}-A_{0;2})\boldsymbol{u}=\boldsymbol{v}_0\geq 0$ by Lemma 4.1.

Theorem 4.2. *Assume* (2.1b), *i.e., $I-A_0-A_1-A_2$ is a singular regular M-matrix and one of A_0 and A_2 has no zero rows. Then* Algorithm 3.1 *runs without any breakdown, i.e., all inverses exist, and, for $k\geq -1$,*

$$\boldsymbol{v}_k\geq 0,\quad A_{k;0}\geq 0,\quad A_{k;1}\geq 0,\quad A_{k;2}\geq 0, \tag{4.6a}$$

$$\text{either } A_{k;0}\boldsymbol{u}>0 \text{ or } A_{k;2}\boldsymbol{u}>0. \tag{4.6b}$$

As a result, $I-A_{k;1}$ is a nonsingular M-matrix.

Proof. That $I-A_{k;1}$ for a fixed k is nonsingular is a corollary of (4.4) and (4.6). To see this, we note that (4.3) yields

$$(I-A_{k;1})\boldsymbol{u}=\boldsymbol{v}_k+(A_{k;0}+A_{k;2})\boldsymbol{u}>0.$$

Thus $I-A_{k;1}$ is a nonsingular M-matrix because it is evidently a Z-matrix.

It remains to show (4.6). To that end, we will do induction on k. By the assumption,

$$\begin{aligned}&\boldsymbol{v}_{-1}=\boldsymbol{v}\geq 0,\\&A_{-1;0}=A_0\geq 0,\quad A_{-1;1}=A_1\geq 0,\quad A_{-1;2}=A_2\geq 0,\\&\text{either } A_{-1;0}\boldsymbol{u}=A_0\boldsymbol{u}>0\quad\text{or}\quad A_{-1;2}\quad\boldsymbol{u}=A_2\boldsymbol{u}>0,\end{aligned}$$

i.e., (4.6) holds for $k=-1$. Suppose that (4.6) holds for k. Then $I-A_{k;1}$ is a nonsingular M-matrix, and thus $(I-A_{k;1})^{-1}\geq 0$ and the diagonal entries of $(I-A_{k;1})^{-1}$ is positive. Therefore, $H_{k+1}\geq 0$, $L_{k+1}\geq 0$, implying $A_{k+1;j}\geq 0$ for $j\in\{0,1,2\}$. It remains to show that either $A_{k+1;0}\boldsymbol{u}>0$ or $A_{k+1;2}\boldsymbol{u}>0$. In fact, in the case of $A_{k;0}\boldsymbol{u}>0$, i.e., $A_{k;0}$ has no zero row, we know $(I-A_{k;1})^{-1}A_{k;0}\boldsymbol{u}>0$ and thus

$$A_{k+1;0}\boldsymbol{u}=H_{k+1}^2\boldsymbol{u}=(I-A_{k;1})^{-1}A_{k;0}\big[(I-A_{k;1})^{-1}A_{k;0}\boldsymbol{u}\big]>0.$$

Similarly, in the case of $A_{k;2}\boldsymbol{u}>0$, we will have $A_{k+1;2}\boldsymbol{u}>0$, as expected. It follows from (4.4) that $\boldsymbol{v}_{k+1}\geq 0$, as well. In summary, (4.6) holds with k replaced by $k+1$. This completes the inductive proof. □

Algorithm 4.1. Highly Accurate LatoucheRamaswami (LR) Algorithm (accLRQBD)

Require: $A_0, A_1, A_2\in\mathbb{R}^{n\times n}$, $\boldsymbol{u}$ and $\boldsymbol{v}$ as in (2.2);
Ensure: an approximation of Φ, minimal nonnegative solution to (1.1).
1: compute the triplet representation (4.1) of $I-A_1$;
2: use the GTH-like algorithm to compute $L_0=(I-A_1)^{-1}A_0$, $H_0=(I-A_1)^{-1}A_2$,
and $\boldsymbol{w}_0=(I-A_1)^{-1}\boldsymbol{v}$;
3: $\boldsymbol{v}_0=\boldsymbol{w}_0+(H_0+L_0)\boldsymbol{w}_0$;
4: $X_0=L_0$, $T_0=H_0$;
5: **for** $k=0,1,\ldots,$ until convergence **do**
6: $A_{k;0}=L_k^2$, $A_{k;1}=H_kL_k+L_kH_k$, $A_{k;2}=H_k^2$;
7: compute the triplet representation (4.7) of $I-A_{k;1}$;
8: use the GTH-like algorithm to compute $L_{k+1}=(I-A_{k;1})^{-1}A_{k;0}$, $H_{k+1}=(I-A_{k;1})^{-1}A_{k;2}$, and $\boldsymbol{w}_{k+1}=(I-A_{k;1})^{-1}\boldsymbol{v}_k$;
9: $\boldsymbol{v}_{k+1}=\boldsymbol{w}_{k+1}+(H_{k+1}+L_{k+1})\boldsymbol{w}_{k+1}$;
10: $X_{k+1}=X_k+T_kL_{k+1}$; $T_{k+1}=T_kH_{k+1}$;
11: **end for**
12: **return** last X_k as the computed minimal nonnegative solution.

Eq. (4.3) immediately yields a triplet representation for $I-A_{k;1}$ as

$$I-A_{k;1}=(\text{offdiag}(A_{k;1}),\boldsymbol{u},\hat{\boldsymbol{v}}_k)\quad \text{with}\,\hat{\boldsymbol{v}}_k=\boldsymbol{v}_k+(A_{k;0}+A_{k;2})\boldsymbol{u}. \tag{4.7}$$

With it, we present a highly accurate implementation of the LR algorithm in Algorithm 4.1 under the more general setting of (2.1), extending Ye's implementation [30]. As to when to stop the iteration at Line 5, we follow the discussion in [6, section 8]. Specif-

ically, there are three viable options for use as stopping criteria:

$$|X_{k+1}-X_k| \le \varepsilon \cdot X_{k+1}, \tag{4.8a}$$

$$\texttt{ERRes}(X_{k+1}) \le \varepsilon, \tag{4.8b}$$

$$\frac{(X_{k+1}-X_k)^2_{(i,j)}}{(X_k-X_{k-1})_{(i,j)}-(X_{k+1}-X_k)_{(i,j)}} \le \varepsilon \cdot (X_{k+1})_{(i,j)} \quad \text{for all } i \text{ and } j, \tag{4.8c}$$

where ε is a pre-selected tolerance. The first one (4.8a) is the simplest and also cheapest one to use, the second one (4.8b) is based on the entrywise relative residual [6, (7.3)]:

$$\texttt{ERRes}(X_{k+1}) := \max_{i,j} \frac{|(A_0+A_1X_{k+1}+A_2X_{k+1}^2)-X_{k+1}|_{(i,j)}}{(X_{k+1})_{(i,j)}}, \tag{4.9}$$

and the third one (4.8c) is Kahan's stopping criterion, previously used in [26, 28, 29]. Both the simple (4.8a) and Kahan's stopping criterion (4.8c) can be too conservative in the case of a monotonically quadratically convergent sequence in the sense that they stop iterations unnecessarily late, wasting the last one or two iterations. Another shortcoming for both is a possible pitfall: false-convergence in the sense that the iteration may be stopped due to a period of very slowly moving X_k. The second stopping criterion (4.8b) is most expensive to use among the three, especially $\texttt{ERRes}(X_{k+1})$ is not needed in the iteration kernel. But it does not have the pitfall mentioned above.

As in [6], we will use Kahan's stopping criterion (4.8c) (ε=10^{-15} in our test) with a safeguard, in the sense that when Kahan's stopping criterion is satisfied we check if (4.8b) is also satisfied to avoid possible false-convergence.

Algorithm 4.1 for $\boldsymbol{v}=0$ and $\boldsymbol{u}=\mathbf{1}_n$ is the same as in Ye [30, Algorithm 3].

5 Numerical examples

In this section, we will present two numerical examples to illustrate the performance of Algorithm 4.1 in delivering entrywise accuracy in computed approximations to the minimal nonnegative solution Φ.

For illustration, we will report, besides ERRes in (4.9), also three other different error measures. The commonly used legacy error measure is the following normalized residual (NRes):

$$\texttt{NRes}(X_{k+1}) = \frac{\|A_0+A_1X_{k+1}+A_2X_{k+1}^2-X_{k+1}\|_{\mathrm{F}}}{\|X_{k+1}\|_{\mathrm{F}}(\|A_2\|_1\|X_{k+1}\|_1+\|A_1\|_1+1)+\|A_0\|_{\mathrm{F}}}, \tag{5.1}$$

where $\|\cdot\|_{\mathrm{F}}$ and $\|\cdot\|_1$ are the matrix Frobenius norm and the ℓ_1 operator norm, respectively. The use of the ℓ_1-operator norm, instead of the spectral norm $\|\cdot\|_2$, is inconsequential but for computational convenience. For testing purpose, we will also show the

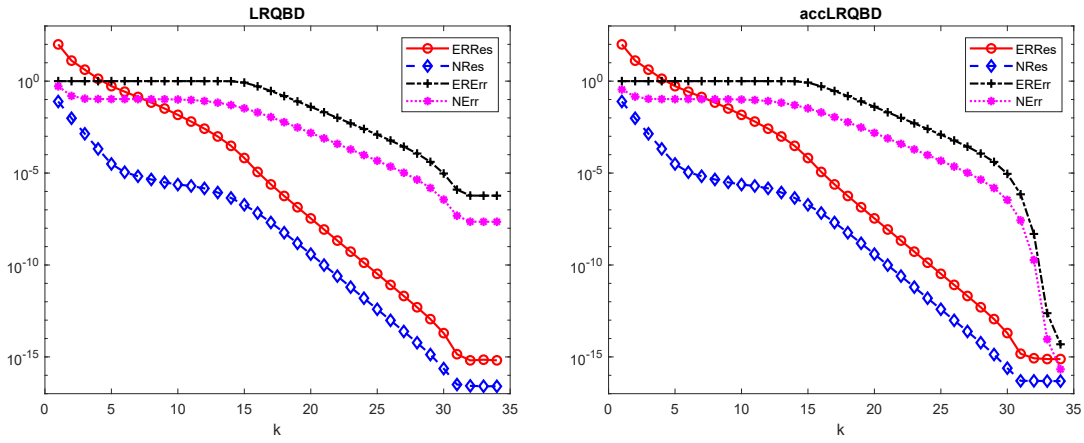

Figure 1: Example 5.1: convergence history. *Left:* by LRQBD; *Right:* by accLRQBD. Both ERRes and NRes move down to about $O(10^{-15})$ or tinier, but LRQBD can only achieve entrywise relative accuracy ERErr $= 5.9\times10^{-7}$, losing 10 significant decimal digits in some of the entries, while accLRQBD delivers a computed solution with each entry having at least 15 significant decimal digits.

normalized error (NErr) in X_{k+1}:

$$\texttt{NErr}(X_{k+1}) := \frac{\|X_{k+1}-\Phi\|_{\mathrm{F}}}{\|\Phi\|_{\mathrm{F}}}, \tag{5.2}$$

and the entrywise relative error (ERErr),

$$\texttt{ERErr}(X_{k+1}) = \max_{i,j}\frac{|(X_{k+1}-\Phi)_{(i,j)}|}{\Phi_{(i,j)}}. \tag{5.3}$$

Tiny ERErr guarantees that all entries of Φ, large or small in magnitude, are well approximated, but NErr is good only in telling relative errors in the large entries of X_{k+1}, i.e., those of $O(\|X_{k+1}\|_1)$ in magnitude.

Both NErr and ERErr are not available in actual computations because Φ is unknown in the first place. In our tests, we compute an "exact" Φ using MATLAB's variable-precision floating-point arithmetic vpa with digits(100) so that we can report both ERErr and NErr for testing purpose.

For convenience, we will denote Algorithm 3.1 by LRQBD whose line 5 is simply carried out by MATLAB's linear system solver A\B, and Algorithm 4.1 by accLRQBD.

Example 5.1 ([13])**.** For this example, n=24. It starts by lettiing A_0', A_1' and A_2' be giv-

en by for $1\le i,j\le n$

$$(A_0')_{(i,j)}=\begin{cases}192[1-(i-1)/24], & \text{for } i=j,\\ 0, & \text{for } i\neq j,\end{cases}\qquad (A_2')_{(i,j)}=\begin{cases}192\rho_d, & \text{for } i=j,\\ 0, & \text{for } i\neq j,\end{cases}$$

$$(A_1')_{(i,j)}=\begin{cases}\alpha r(\beta-i+1)/\beta, & \text{for } i-j=-1,\\ (i-1)r, & \text{for } i-j=1,\\ \xi_i, & \text{for } i-j=0,\\ 0, & \text{elsewhere,}\end{cases}$$

where α,r,β and ρ_d are parameters, and ξ_i for $1\le i\le n$ are determined by $(A_0'+A_1'+A_2')\mathbf{1}_{24}=0$. Finally,

$$A_0=-(A_1')^{-1}A_0',\quad A_1=0,\quad A_2=-(A_1')^{-1}A_2'.$$

For the parameters, we take $r=1/300,\alpha=18.244$, $\beta=65536$ and $\rho_d=0.280$. The "exact solution" shows that

$$5.2533\cdot 10^{-57}\le\Phi_{(i,j)}\le 9.9956\cdot 10^{-1}.$$

Figure 1 plots the convergence history for the four error measures. It can be observed that both LRQBD and accLRQBD can reduce `ERRes` and `NRes` to about $O(10^{-15})$ or tinier, which is near the best possible one could hope for in general. However, the computed solution by LRQBD has an entrywise relative accuracy `ERErr`$=5.9\times 10^{-7}$, meaning some entries have only about 7 correct significant decimal digits. On the other hand, the computed solution by accLRQBD has an entrywise relative accuracy `ERErr`$=4.9\times 10^{-15}$, meaning all entries are guaranteed to have about 15 correct significant decimal digits, 8 more than by LRQBD. Besides what we have observed from Figure 1, there are a couple of more interesting convergence behaviors observable from Figure 1 that warrant comments:

(1) In theory, tiny `ERRes` implies tiny `ERErr` for QBD equation (1.1) that is either positive recurrent or transient [6, Theorem 7.2]. But we observed from the left plot in Figure 1 that this is not case. In fact, we see that `ERRes` is about 10^{-15} in the end while `ERErr` about 10^{-6}. The reason behind this is that this example is a nearly null recurrent case: $\mathbf{z}^{\mathrm{T}}(A_0-A_2)\mathbf{1}$ is about 3.5×10^{-6}, and as a result, [6, Theorem 7.2] is no longer applicable within the double precision. This also explains that `NErr` can only get down to 10^{-7}, about the square root of the double precision roundoff unit $2^{-53}\approx 10^{-17}$.

(2) Despite that this example is a nearly null recurrent case, it is still not null recurrent, nonetheless. For that reason, we should expect eventually quadratically convergent behavior. This is clearly visible from the left plot in Figure 1, where we notice a precipitated drops in `ERErr` and `NErr` starting after iteration $k=30$, indicating quadratic convergence. However, we do not see such sudden drops to

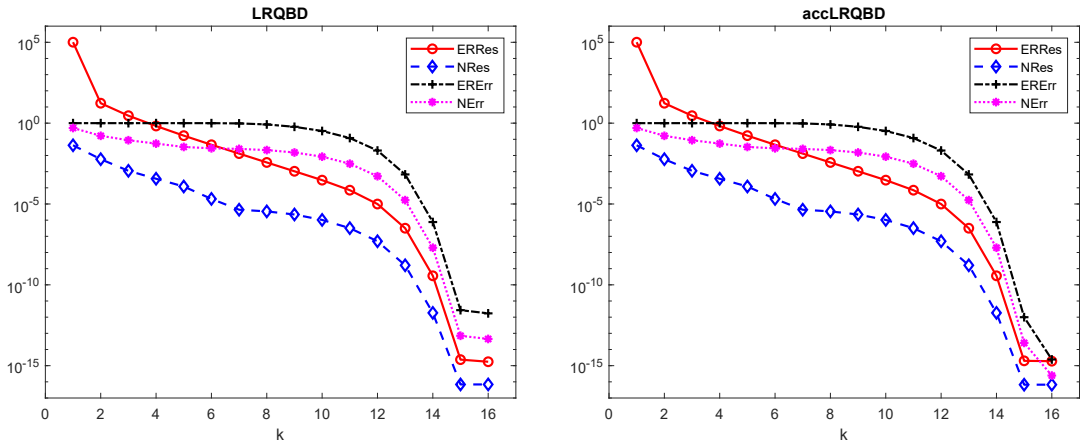

Figure 2: Example 5.2: convergence history. *Left:* by LRQBD; *Right:* by accLRQBD. Both ERRes and NRes move down to about $O(10^{-15})$ or tinier, but LRQBD can only achieve entrywise relative accuracy ERErr $= 1.7\times10^{-12}$, losing 5 significant decimal digits in some of the entries, while accLRQBD delivers a computed solution with each entry having at least 15 significant decimal digits.

ERRes and NRes. That is because both have already reached to the point at iteration 31 to the point where further improvement is not possible within the double precision.

Example 5.2 ([13])**.** This is a random prblem, generated by the following piece of MATLAB code:

```
n=100;
A0=triu(tril(rand(n),1),-1); % a nonnegative tridiagonal matrix
A1=triu(tril(rand(n),1),-1);
A2=triu(tril(rand(n),1),-1);

u=ones(n,1); b=(A0+A1+A2)*u;
eta=1.0-1.e-8; t=eta*ones(n,1)./b;
A0=(t*ones(1,n)).*A0; A1=(t*ones(1,n)).*A1; A2=(t*ones(1,n)).*A2;
v=u-t.*b;
```

For a typical problem so generated, we find $\max_{i,j}\Phi_{(i,j)}=O(1)$ while $\min_{i,j}\Phi_{(i,j)}=O(10^{-9})$, which suggests that in the worst case LRQBD could lose up to 7 significant decimal digits, although actual results are better because straightforwardly inverting nonsingular M matrices is still often much more accurate than the generic existing error analysis [7] for the Gaussian elimination indicates. Figure 2 plots the convergence history for the four measures. It can be observed that both LRQBD and accLRQBD can reduce ERRes and NRes to about $O(10^{-15})$ or tinier, which is again near the best possible one could

hope for in general. However, the computed solution by LRQBD has an entrywise relative accuracy $\texttt{ERErr}=1.7\times10^{-12}$, meaning some entries have only about 12 correct significant decimal digits. On the other hand, the computed solution by accLRQBD has an entrywise relative accuracy $\texttt{ERErr}=2.4\times10^{-15}$, meaning all entries are guaranteed to have about 15 correct significant decimal digits, 3 more than by LRQBD.

6 Conclusions

The quadratic matrix equation of the form $A_0+A_1X+A_2X^2=X$ has to be solved repeatedly in the quasi-birth-and-death (QBD) process. It is a nonlinear equation and thus Newton's method is a natural choice. But because of its special form, it is known today that most effective methods are the Latouche-Ramaswami logarithmic reduction (L-R) algorithm [17], the cyclic reduction (equivalent to the LR algorithm as shown in [11]), and the doubling algorithms [6]. It was well-known earlier the LR algorithm suffers from accuracy loss in the computed solution. It was Ye [30] who solved this issue of accuracy loss. Later in [6], the authors proposed a highly accurate doubling algorithm that is also free of this inaccuracy drawback.

The original QBD quadratic equation has an additional property (1.2): $(A_0+A_1+A_2)\mathbf{1}_n=\mathbf{1}_n$, among others. The contribution of this paper is to extend Ye's work in [30] to broadly include the quadratic equation for the case when $I-A_0-A_1-A_2$ is either a nonsingular M-matrix or a singular regular M-matrix. Numerical examples are presented to illustrate the effectiveness of the proposed extension.

Acknowledgments

Gu was supported in part by National Natural Science Foundation of China (Grant Nos. 11771100 and 12071332). Wang was supported in part by NSF grant DMS-2009689, U-TA's Reserach Enhancement grant REP-270075, and USDA grant 58-3070-0-006. Li is supported in part by NSF grants DMS-1719620 and DMS-2009689. The authors are grateful to two anonymous reviewers for their useful comments and suggestions that improve the presentation of the paper.

References

[1] Ahn S, Ramaswami V. Efficient algorithms for transient analysis of stochastic fluid flow models. J. Appl. Probab., 2005, 42(2): 531–549.
[2] Alfa A S, Xue J, Ye Q. Accurate computation of the smallest eigenvalue of a diagonally dominant M-matrix. Math. Comput., 2002, 71: 217–236.
[3] Alfa A S, Xue J, Ye Q. Entrywise perturbation theory for diagonally dominant M-matrices with applications. Numer. Math., 2002, 90(3): 401–414.
[4] Berman A, Plemmons R J. Nonnegative Matrices in the Mathematical Sciences. SIAM, Philadelphia, 1994.

[5] Bini D A, Meini B. Improved cyclic reduction for solving queueing problems. Numer. Alg., 1997, 15: 57–74.
[6] Chen C, Li R C, Ma C. Highly accurate doubling algorithm for quadratic matrix equation from quasi-birth-and-death process. Linear Algebra Appl., 2019, 583: 1–45.
[7] Demmel J. Applied Numerical Linear Algebra SIAM. Philadelphia, PA, 1997.
[8] Favati P, Meini B. Relaxed functional iteration techniques for the numerical solution of M/G/1 type Markov chains. BIT Numer. Math., 1998, 38(3): 510–526.
[9] Grassmann W K, Taksar M J, Heyman D P. Regenerative analysis and steady-state distributions for Markov chains. Oper. Res., 1985, 33: 1107–1116.
[10] Guo C H. On the numerical solution of a nonlinear matrix equation in Markov chains. Linear Algebra Appl., 1999, 288: 175–186.
[11] Guo C-H. Convergence analysis of the Latouche-Ramaswami algorithm for null recurrent quasi-birth-death process. SIAM J. Matrix Anal. Appl., 2002, 23: 744–760.
[12] Guo C H. On algebraic Riccati equations associated with *M*-matrices. Linear Algebra Appl., 2013, 439(10): 2800–2814.
[13] He C Y, Meini B, Rhee N H. A shifted cyclic reduction algorithm for quasi-birth-death problems. SIAM J. Matrix Anal. Appl., 2002, 23(3): 679–691.
[14] Huang T M, Li R C, Lin W W. Structure-preserving doubling algorithms For nonlinear matrix equations. Chapter 3: General Theory of Doubling Algorithms, SIAM, Philadelphia, 2018, 14: 11-51.
[15] Latouche G. Newton's iteration for nonlinear equations in Markov chains. IMA J. Numer. Anal., 1994, 14: 583–598.
[16] Latouche G, Ramaswami V. A logarithmic reduction algorithm for quasi-birth-death processes. J. Appl. Probab., 1993, 30(3): 650–674.
[17] Latouche G, Ramaswami V. Introduction to Matrix Analytic Methods in Stochastic Modeling. SIAM, Philadelphia, 1999.
[18] Liu C, Xue J, Li R C. Accurate numerical solution for shifted *M*-matrix algebraic Riccati equations. J. Sci. Comput., 2020 ,84(1).
[19] Liu C, Wang W G, Xue J, *et al.* Accurate numerical solution for structured *M*-matrix algebraic Riccati equations. J. Comput. Appl. Math., 2021, 396: 113614.
[20] Meini B. Solving QBD problems: the cyclic reduction algorithm versus the invariant subspace method. Advances in Performance Analysis, 1998, 1: 215–225.
[21] Meng J, Seo S-H, Kim H-M. Condition numbers and backward error of a matrix polynomial equation arising in stochastic models. J. Sci. Comput., 2018, 76(2): 759–776.
[22] Meyer C D. Matrix Analysis and Applied Linear Algebra. SIAM, Philadelphia, 2000.
[23] Neuts M F. Matrix Geometric Solutions in Stochastic Models: an Algorithmic Approach. The Johns Hopkins University Press, Baltimore, 1981.
[24] Nguyen G-T, Poloni F. Componentwise accurate fluid queue computations using doubling algorithms. Numer Math, 2015, 130(4): 763–792.
[25] Ramaswami V. Nonlinear matrix equations in applied probability – solution techniques and open problems. SIAM Rev., 1988, 30(2): 256–263.
[26] Wang W G, Wang W C, Li R C. Alternating-directional doubling algorithm for *M*-matrix algebraic Riccati equations. SIAM J. Matrix Anal. Appl., 2012, 33(1): 170–194.
[27] Xue J, Li R C. Highly accurate doubling algorithms for *M*-matrix algebraic Riccati equations. Numer. Math., 2017, 135(3):733–767.
[28] Xue J, Xu S, Li R C. Accurate solutions of *M*-matrix algebraic Riccati equations.

Numer. Math., 2012, 120(4): 671–700.
[29] Xue J, Xu S, Li R C. Accurate solutions of *M*-matrix Sylvester equations. Numer. Math., 2012, 120(4): 639–670.
[30] Ye Q. On Latouche-Ramaswami's logarithmic reduction algorithm for quasi-birth-and-death processes. Stoch. Models, 2002, 18: 449–467.